UNIVERSIDADE DE BRASÍLIA
DEPARTAMENTO DE ADMINISTRAÇÃO
FACULDADE DE ECONOMIA, ADMINISTRAÇÃO E CONTABILIDADE (FACE)
Programa de Pós-Graduação em Administração
Finanças e Métodos Quantitativos

SARAH SABINO DE FREITAS MARCELINO

Formação de Portfólio por meio de Máquinas de Suporte Vetorial e Redes de Camadas Profundas

Brasília – DF
2016

SARAH SABINO DE FREITAS MARCELINO

Formação de Portfólio por meio de Máquinas de Suporte Vetorial e Redes de Camadas Profundas

Dissertação apresentada ao Programa de Pós-Graduação em Administração da Universidade de Brasília para obtenção do Título de Mestre em Administração.

Brasília – DF

2016

SARAH SABINO DE FREITAS MARCELINO

Formação de Portfólio por meio de Máquinas de Suporte Vetorial e Redes de Camadas Profundas

Dissertação apresentada ao Programa de Pós-Graduação em Administração da Universidade de Brasília para obtenção do Título de Mestre em Administração.

Orientador:
Prof. Dr. Pedro Henrique Melo Albuquerque

Brasília – DF

2016

Dissertação de Mestrado sob o título Formação de Portfólio por meio de Máquinas de Suporte Vetorial e Redes de Camadas Profundas, defendida por Sarah Sabino de Freitas Marcelino e aprovada em de novembro de 2016, em Brasília, Distrito Federal, pela banca examinadora constituída pelos doutores:

Doutor, Pedro Henrique Melo Albuquerque

Professor-Orientador

Doutor, Vinicius Amorim Sobreiro

Professor-Examinador

Doutor, Daniel Cajueiro

Professor-Examinador

AGRADECIMENTOS

Ao meu Professor Orientador, Pedro Henrique Melo Albuquerque, por ser meu principal suporte vetorial nessa caminhada. Obrigada Pedro, por todo engajamento, apoio, incentivo, paciência e disponibilidade. Me sinto honrada de tê-lo como mentor e como amigo. Agradeço também aos meus colegas do PPGA/UnB, em especial: Pedro Alexandre Moura Barros Henrique, Peng Yahao, Mariana Montenegro e Leonardo Bosque.

Agradeço também à minha família pela compreensão nos momentos de ausência e por todos incentivos que me permitiram concluir mais uma etapa da minha vida profissional.

Agradeço aos professores Daniel Cajueiro, Donald Matthew Pianto, José Carneiro da Cunha Oliveira Neto e Vinicius Amorim Sobreiro pelas contribuições para o aprimoramento deste trabalho.

RESUMO

Este estudo teve como objetivo verificar se o uso de Máquinas de Suporte Vetorial contribui para que o retorno do portfólio seja superior ao Mercado. A amostra desta pesquisa foram as ações do S&P 100. Na primeira aplicação, utilizou-se o *Support Vector Regression* com 15 tipos de *Kernels* diferentes. Na segunda aplicação, utilizou-se o SVM de classificação para formar portfólios e estes também foram comparados com o *benchmark* de mercado. Aqui utilizou-se apenas o *Kerne*l Gaussiano e foram analisados o impacto das variáveis inseridas como *inputs* e o impacto do uso de custos de erro diferentes para cada classe. Na terceira aplicação desta dissertação, o SVR foi utilizado com 9 diferentes tipos de *Kernel* Arco-Cosseno Compostos que imitam o processo computacional em grandes redes neurais de múltiplas camadas e apresentam algumas das vantagens do *Deep Learning.* Na primeira e na segunda aplicação da pesquisa, o SVM apresentou resultados de acurácia satisfatórios aliados com valores de retorno e risco bastante promissores para os portfólios trimestrais. Porém, na terceira aplicação, os portfólios não mostraram-se superiores em termos de retorno e nem em termos de risco. Os resultados desta pesquisa corroboram a hipótese de superioridade do método inovador das Máquinas de Suporte Vetorial na formação de portfólios, caracterizado pela construção de um hiperplano pela implementação do Princípio da Minimização do Risco Estrutural, o qual procura minimizar o limite superior do erro de generalização, em vez de minimizar apenas o erro do processo de estimação.

Palavras-chave: Máquinas de Suporte Vetorial. SVM. SVR. *Deep Learning*. Formação de Portfólio.

ABSTRACT

This study aimed to verify whether the use of Support Vector Machines makes the portfolio return exceeds the market. The sample of this research consisted on the S&P 100 stocks. In the first application, Support Vector Regression was used with 15 different Kernels to select the bests stocks to form a portfolio. In the second application, the traditional SVM was used with Gaussian Kernel to form portfolios. The impact of more variables as inputs and the impact of using different error costs for each class were analyzed. In the third application, the SVR was used with 9 different types of Arc-Cosine Kernels that mimic the computational process in large neural networks of multiple layers and have some of the advantages of Deep Learning. In the first and second application of the research, the SVM presented satisfactory and very promising results for the quarterly portfolios. However, in the third application, the portfolios were not superior in terms of return neither in terms of risk. The results of this study corroborate the hypothesis of the superior and innovative method of Support Vector Machine in the selection of portfolios, characterized by the construction of a hyperplane for implementing the principle of minimization of structural risk, which seeks to minimize the upper limit of the error generalization, rather than just minimizing the error estimation process.

Key-Words: Support Vector Machine. Support Vector Regression. Deep Learning. Portfolio Selection.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ABBS - *Accrual Based on Balance Sheet*

ABCF - *Accrual Based on Cash Flow*

AG – Algoritmo Genético

AR - *Accounts Receivable*

ATR - *Average True Range*

AUD- *Australian Dollar*

BPN - *Back-Propagation Neural Network*

BV - *Book Value*

CAC40 - Índice *Cotation Assistée en Continu*

CBOT - *Chicago Board Of Trade*

CBR - *Case-based reasoning*

CCI - *Commodity Channel Index*

CE - *Cash and Equivalents*

CEx - *Capital Expenditures*

CFFA - *Cash From Financing Activities*

CFIA - *Cash From Investing Activities*

CFOA - *Cash From Operating Activities*

CL - *Total Current Liabilities*

CMF - *Chaikin Money Flow Indicator*

D - *Dividends*

DNEPS - *Diluted Normalized Earnings Per Share*

DP - *Depreciation*

DPR - *Dividend Payout Ratio*

EG - *Equity Growth*

EMA - Erro Médio Absoluto

EMA - *Exponential Moving Average*

EQM - Erro Quadrático Médio

EQMN - Erro Quadrático Médio Normalizado

EUR – Euro

EUREX-Bund – Contrato de câmbio alemão

FH - *Financial Health*

GA - *Growth in Assets*

GBP – Libra

GNP - *Growth in Net Profit*

GP - *Gross Profit*

IAT - *Income After Tax*

IBT - *Income Before Tax*

JPY – Yen

LSTM – *Long Short Term Memory Network*

MACD - *Moving Average Convergence-Divergence Trading Method*

MASS - *Mass Index*

MATIF – Empresa francesa chamada *Marché à Terme International de France*

MFI - *Money Flow Index*

MKM - *Multilayer Kernel Machine*

Mo - *Momentum*

NCIC - *Net Change In Cash*

NI - *Net Income*

NIBEI - *Net Income Before Extraordinary Items*

NSE - *National Stock Exchange*

NZD - *New Zealand Dollar*

OI - *Operating Income*

OSCP - *Price Oscilator*

RCGA - *Real Coded Genetic Algorithm*

ROA - *Return on Assets*

ROC - *Price Rate of Change*

ROE - *Return on Equity*

RSI - *Relative Strength Index*

RUB - Rublo Russo

SAC - *Snapshot Accrual*

SD - Simetria Direcional

STI - *Short Term Investments*

STL - *Short Term Liabilities*

SVM – *Support Vector Machine*

SVR – *Support Vector Regression*

TA - *Total Assets*

TCA - *Total Current Assets*

TCA-CL - *Working Capital*

TD - *Total Debt*

TE - *Total Equity*

TI - *Total Inventory*

TL - *Total Liabilities*

TLTD - *Total Long Term Debt*

TR - *Total Revenue*

TRIX - *Triple Exponential Smoothing of the Log of Closing Price*

TS - *Total Shares*

US – *United States*

USD – Dólar Americano

SUMÁRIO

1 INTRODUÇÃO

A seleção de ações é uma parte desafiadora e crucial do processo de decisão dos investidores. Considerando o enorme montante de opções de ativos disponíveis no mercado financeiro, segundo Fan e Palaniswami (2001), o desafio da seleção de ações está na identificação dos ativos com potencial de superar o mercado no próximo ano.

Para Tay e Cao (2001), “a previsão de séries temporais financeiras é considerada uma das aplicações mais desafiadoras da previsão de séries temporais” (p.309, tradução nossa). Abu-Mostafa e Atiya (1996) apontam que os especuladores, investidores e empresas, em sua busca para prever o comportamento dos mercados, assumem que as ocorrências futuras se baseiam, pelo menos em parte, em eventos e dados presentes e passados. No entanto, as séries financeiras são permeadas por ruídos, não-estacionariedade e caos determinístico. A primeira característica refere-se à indisponibilidade de informações completas do comportamento passado dos mercados financeiros que poderiam contribuir para a análise da dependência dos preços futuros e passados, sendo assim, ruídos são as informações não incluídas no modelo ou erros de medidas. A não-estacionariedade implica que a distribuição das séries temporais financeiras muda ao longo do tempo e padrões que representam dados passados podem não ser mais aplicáveis. A presença de caos determinístico, por sua vez, significa que no curto prazo, o comportamento das séries temporais financeiras é aleatório, mas no longo prazo apresenta um padrão determinístico.

Este contexto levou muitos economistas a adotarem a Hipótese do Mercado Eficiente que considera que as mudanças nos preços das ações são independentes do passado e seguem um padrão aleatório (Abu-Mostafa e Atiya, 1996). As mudanças de preços seriam então imprevisíveis e qualquer alteração no preço representaria uma reação imediata a um novo evento ou a uma mudança inesperada de oferta ou demanda. Se houvesse qualquer oportunidade inesperada de lucro, por exemplo, os investidores a explorariam imediatamente de forma que o preço voltaria ao patamar que estava quando essa oportunidade não existia. Ainda segundo essa teoria, quaisquer padrões úteis deveriam refletir no preço corrente,

porém, se o mercado é eficiente a ponto de todos os preços das ações refletirem plenamente todas as informações públicas disponíveis, não se pode esperar que essa forma de análise consiga identificar com antecedência os investimentos com retornos superiores ao mercado. Apesar de existirem vários debates sobre a Hipótese do Mercado Eficiente, é difícil refutá-la (Abu-Mostafa e Atiya, 1996).

Na perspectiva de mineração de dados, os retornos futuros das ações são considerados, em alguma medida, previsíveis. De acordo com Fan e Palaniswami (2001), o problema de predição envolve a descoberta de padrões de relacionamentos úteis nos dados e aplicação dessa informação para classificar as ações. Uma abordagem que tem se mostrado promissora para esse problema são as Máquinas de Suporte Vetorial (*Support Vector Machine* – SVM), propostas inicialmente por Boser, Guyon, e Vapnik (1992).

O aprendizado de máquina tem por objetivo criar e implementar algoritmos que permitem que máquina aprenda a partir de exemplos. Esses algoritmos permitem o reconhecimento de padrões nos dados, extração de informações e predição para exemplos novos e desconhecidos. Segundo Albuquerque (2014), "por meio desse reconhecimento é possível realizar um processo de inferência indutiva, o qual seria capaz de realizar previsões para um conjunto de dados observados posteriormente à estimação dos parâmetros do modelo" (p.11).

A literatura mostra que algumas técnicas de aprendizado de máquina, especialmente as de classificação e regressão, são aplicáveis em análises quantitativas na indústria financeira, assim como no reconhecimento de padrões implícitos em grandes quantidades de dados financeiros (Gerlein, McGinnity, Belatreche, e Coleman, 2016).

Os modelos de aprendizado de máquina possuem parâmetros que modelam os dados, por isso muitas vezes são formulados como problemas de otimização de parâmetros. Nessa otimização, deve-se evitar tanto o *overfitting* como o *underfitting* do modelo. O primeiro ocorre quando o modelo é desnecessariamente complexo e apesar de conseguir modelar detalhes irrelevantes dos exemplos de treinamento, falha em generalizar para os outros exemplos. O *underfitting,* por sua vez, ocorre quando o modelo não é capaz de cobrir a variabilidade dos exemplos no treinamento e também falha ao generalizar.

No aprendizado de máquina supervisionado, os modelos recebem como *inputs* exemplos para treinamentos e as respectivas categorias de classificação

destes exemplos. Segundo Cho e Saul (2011), no aprendizado supervisionado se provê um conjunto de treinamento de N exemplos, onde se tem observações (x_n) e suas respectivas classificações (y_n). y_n é a categoria na qual a observação se encaixa e pode ser tanto um número real, como acontece em contextos de regressão, como um valor categorizado, quando o contexto é de classificação.

Sendo assim, o objetivo do aprendizado de máquina supervisionado é aprender um modelo baseado em exemplos já categorizados para posteriormente generalizar este modelo para exemplos desconhecidos e prever a classificação dessas novas observações de forma correta.

No aprendizado de máquina não supervisionado, o foco é a captação de importantes padrões nos dados independentemente da categorização ou classificação deles. Essa captação é geralmente realizada por meio da redução de dimensionalidade ou extração de características e estes métodos são benéficos para modelos de aprendizado supervisionado (Cho e Saul, 2010). Este tipo de aprendizado torna possível reduzir a dimensionalidade dos dados sem a perda informações essenciais, e aliviar a Maldição da Dimensionalidade, melhor descrita no Capítulo 2 desta dissertação. Dessa forma, os modelos de aprendizado supervisionado podem concentrar a modelagem em um espaço de *input* reduzido sem se ajustar a ruídos. Além das vantagens apresentadas, como é muito mais fácil encontrar exemplos de observações para treinamento não classificadas, o aprendizado não supervisionado pode ser ainda mais útil na prática.

Originalmente, as Máquinas de Suporte Vetorial foram desenvolvidas dentro do campo de aprendizado supervisionado para esse propósito de reconhecimento de padrões em um conjunto de dados. Desde sua criação em 1992, marcaram o início de uma nova era na inteligência artificial, representando uma quebra de pensamento na Teoria de Aprendizado Estatístico (Soman, Loganathan, e Ajay, 2009). Com a implementação do Princípio de Minimização do Risco Estrutural, que minimiza o limite superior do erro de generalização ao invés de minimizar apenas o erro empírico, o SVM abre um novo panorama para modelagem de algoritmos de aprendizagem de máquina com maior capacidade de generalização, superando a maioria das dificuldades enfrentadas pelos algoritmos tradicionais, como o *overfitting* e a alta dimensionalidade de dados. Segundo Soman et al. (2009), com o SVM, a solução ótima, na grande maioria da vezes, é encontrada.

Visando aprimorar a forma com que os ativos de um portfólio são selecionados, esta pesquisa utilizou as Máquinas de Suporte Vetorial como ferramenta de classificação de ativos. Foram feitas três aplicações do método em uma mesma base de dados para o mesmo período.

1.1 Formulação do problema

A seleção de ações para formação de portfólios envolve a obtenção de proporções ideais entre os ativos, para construir uma carteira que respeite as preferências dos investidores (Gupta, Mehlawat, e Mittal, 2012). As Máquinas de Suporte Vetorial surgem como método alternativo para seleção de ações, aplicando simultaneamente a minimização do erro de classificação e a maximização da margem geométrica. Portanto, pode-se indagar se um portfólio formado por meio de Máquinas de Suporte Vetorial possui um retorno superior ao do mercado.

1.2 Objetivo Geral

Os objetivos deste estudo foram: a) verificar se o uso de Máquinas de Suporte Vetorial, em suas variações de regressão e classificação, contribui para que o retorno do portfólio seja superior a um *benchmark* de mercado; b) analisar os impactos do uso de Redes de Camadas Profundas por meio de *Kernels* Arco-Cosseno Compostos com o SVR na formação de portfólios.

1.3 Objetivos Específicos

Para alcançar os objetivos gerais apresentados foram feitas três aplicações das Máquinas de Suporte Vetorial para a formação de portfólios trimestrais com as ações do S&P 100.

Na primeira aplicação, o SVR, uma variação do SVM para contextos de regressão, foi usado para selecionar portfólios trimestrais visando não apenas uma

simples separação das observações em classes, mas também o retorno de valores previstos pela máquina e a coerência destes com o modelo que descreve as observações estudadas. Além disso, com o SVR foram testados 15 *Kernels* diferentes a fim de observar o impacto da escolha do núcleo da função de decisão.

Na segunda aplicação, utilizou-se o SVM de classificação com o mesmo propósito de formar portfólios trimestrais, mas tendo com base apenas a classificação dos ativos pela máquina em bons ou ruins e o uso do *Kernel* Gaussiano.

Na terceira aplicação, o SVR foi usado novamente, mas com 9 diferentes *Kernels* Arco-Cosseno Compostos (Cho e Saul, 2011*)* a fim de incorporar redes de camadas profundas no método de formação de portfólio construído nas aplicações anteriores.

Nas três aplicações, os objetivos específicos foram: a construção da Máquina de Suporte Vetorial; formação de portfólios com ações selecionadas pela máquina; análise dos retornos obtidos e comparação do retorno do portfólio selecionado pelo SVM com um *benchmark*.

1.4 Justificativa

Com a publicação do estudo pioneiro de Markowitz (1952), o modelo média-variância revolucionou a forma como as pessoas pensam sobre portfólios de ativos e ganhou ampla aceitação como uma ferramenta prática para otimização de portfólios (Lai, Yu, Wang, e Zhou, 2006). Entretanto, a Teoria de Markowitz fornece solução apenas para alocação de ativos previamente selecionados. Como no Mercado Financeiro, centenas de diferentes ativos - ações, títulos, opções, *commodities*, contratos futuros - estão disponíveis para negociação e a qualidade destes pode variar bastante, a escolha dos ativos para investimento é um passo crítico, segundo Lai et al. (2006). Os autores ainda defendem que a qualidade dos ativos escolhidos para investimento é essencial para o bom desempenho do portfólio, mesmo que estejam alocados de forma a minimizar o risco, e isso é muitas vezes negligenciado pela Teoria de Markowitz (1952).

A aplicação de Máquinas de Suporte Vetorial é um tema bastante recente, visto que o modelo foi proposto inicialmente por Boser et al. (1992) e aplicado em finanças pela primeira vez no estudo de Fan e Palaniswami (2001). Segundo Marcelino (2015), apesar de ainda existirem poucos estudos da aplicação do SVM em previsão de séries temporais financeiras, o número de investigações acerca do tema está em constante crescimento.

As Máquinas de Suporte Vetorial têm se mostrado eficientes na superação dos ruídos que permeiam as Séries Temporais Financeiras. Fan e Palaniswami (2001) e Yu, Lu e Chang (2008) mostram que essa abordagem supera o Mercado consistentemente na classificação de ações, abrindo assim espaço para mais pesquisa sobre esse tema. Huerta, Corbacho, e Elkan (2013) apontam que SVMs não lineares conseguem identificar sistematicamente ações com alto ou baixo retorno. No estudo de Kim (2003), as Máquinas de Suporte Vetorial superaram outras técnicas de mineração de dados na previsão do preço de ações do Mercado da Coréia. Yu, Lu e Chang (2008) utilizaram o SVM para prever as melhores ações do Mercado de Taiwan e os resultados mostram que a performance desse modelo supera a de outros métodos do mercado em termos de risco, acurácia e menor erro. Portanto, apesar de algumas limitações, vários trabalhos publicados utilizando as Máquinas de Suporte Vetorial em séries temporais apresentaram bons resultados, justificando assim a utilização dessa abordagem em Séries Temporais Financeiras.

A relevância teórica deste projeto de pesquisa relaciona-se à geração de conhecimento científico que abrange o estado da arte atual sobre aplicação de Máquinas de Suporte Vetorial em conjunto com métodos de aprendizado profundo. Esta dissertação também tem grande valor para o Mercado Financeiro, especialmente no que tange a métodos de formação de portfólios e à integração do SVR com Redes de Camdas Profundas para formação de portfólios.

Do ponto de vista aplicado, esta pesquisa é relevante na medida em que contribuiu para a implementação em *software* do modelo SVR para construção de portfólios com 24 *Kernels* distintos. Os modelos aqui formulados são ferramentas de grande utilidade em contextos de tomada de decisão em investimentos no Mercado Financeiro.

1.5 Organização e Descrição Geral da Pesquisa

Para o desafio de seleção de ações, apenas a acurácia de previsão do modelo não é um indicador suficiente da capacidade de classificação do modelo. Por este motivo, nesta dissertação, para comprovar o desempenho do SVM, o retorno do portfólio ponderado escolhido pela máquina foi comparado com o de todas as ações do S&P 100. Assim como no estudo de Marcelino, Henrique e Albuquerque (2015), o SVM foi utilizado para selecionar portfólios a cada trimestre e não a cada ano, como na maioria das aplicações da literatura.

Foram feitas três aplicações das Máquinas de Suporte Vetorial para a formação de portfólios trimestrais com as ações do S&P 100. Na primeira aplicação, utilizou-se o SVM de regressão, chamado também de SVR (*Support Vector Regression*), com 15 *Kernels* diferentes. Na segunda aplicação, o SVM de classificação foi utilizado com apenas um tipo de *Kernel.* Na terceira aplicação conjugou-se o SVR com Redes de Camadas Profundas para formação de portfólios.

A base de dados de todas as aplicações da pesquisa consistiu em indicadores financeiros e cotações no período de 29/06/1990 a 30/06/2014 das ações que compuseram o índice S&P 100 (*Standard & Poors 100*) em junho de 2015. O S&P 100 é composto pelas 100 ações de maior tamanho e estabilidade selecionadas do índice S&P 500 e que referenciam contratos de opções. Para análise dos dados utilizou-se o *software* livre R.

A variável dependente é o retorno futuro da ação, sendo esta uma variável discreta binária $y = \pm 1$, onde $+1$ representa ações com retornos futuros excepcionais e -1 representa as ações consideradas normais. As variáveis independentes são os preços das ações e indicadores financeiros fundamentalistas.

Esta dissertação está organizada em 8 capítulos. O Capítulo 2 trata do SVM de Regressão (SVR) e inicia-se o com uma revisão da literatura. Em seguida apresenta-se a construção do algoritmo do SVR e seu processo de estimação.

O Capítulo 3, complementarmente, traz a primeira aplicação empírica feita nesta dissertação, isto é, o uso do SVR com 15 *Kernels* distintos como estratégia de formação de portfólio. Neste capítulo são apresentados os procedimentos de coleta e análise de dados, assim como os resultados obtidos.

O Capítulo 4 por sua vez, apresenta o SVM de Classificação por meio de uma revisão da literatura e demonstração de sua metodologia clássica. Neste capítulo, são abordados também alguns pontos essenciais no estudo das máquinas de suporte vetorial, como: as variações lineares, não lineares e de margem suave; a importância dos núcleos *Kernels*; os tipos de *Kernels* e uma breve discussão sobre parâmetros do SVM.

O Capítulo 5 apresenta a segunda aplicação empírica desta dissertação. Nesta etapa, o SVM de classificação com o *Kernel* Gaussiano foi utilizado também como método para formação de portfólio. Neste capítulo são apresentados os procedimentos de coleta e análise de dados, assim como os resultados obtidos.

O Capítulo 6 introduz os principais conceitos e estudos relevantes acerca da abordagem *Deep Learning*. O intuito deste capítulo é mostrar como o *Deep Learning* está inserido no contexto de aprendizado de máquina e qual a utilidade desta abordagem na formação de portfólios por meio do SVR.

O Capítulo 7 traz a terceira aplicação empírica que consiste no uso de *Kernels* Arco-Cosseno Compostos e SVR para formação de portfólios.

O Capítulo 8 apresenta as principais conclusões desta dissertação, limitações, recomendações e ainda propostas de estudos futuros.

2 MÁQUINAS DE SUPORTE VETORIAL DE REGRESSÃO

Para compreensão do contexto no qual as Máquinas de Suporte Vetorial de regressão utilizadas nesta pesquisa estão inseridas, tornam-se necessárias a revisão da literatura sobre as aplicações mais significativas do SVR e a demonstração de sua formulação matemática.

2.1 Referencial Teórico para Máquinas de Suporte Vetorial de Regressão em Finanças

Drucker, Burges, Kaufman, Smola, e Vapnik (1997) introduziram o *Support Vector Regression* (SVR), outra derivação do SVM, sendo aplicável em casos de estudos de séries temporais e momentos onde se busca uma função de regressão.

Os trabalhos de Beltrami et al. (2011) e Ferreira (2011) mostram que o SVR, por ser um modelo sofisticado e ao mesmo tempo flexível, é uma ferramenta de ampla aplicabilidade em finanças, especialmente em séries temporais. Beltrami et al. (2011) mostrou em seu trabalho a eficiência do SVR comparando-o com as Redes Neurais e concluiu que em geral o SVR apresenta resultados superiores para a precificação de opções. Neste trabalho ambas metodologias também foram mais eficientes que o método *Black & Scholes*. Os estudos de Tay e Cao (2001), Pires e Marwala (2005) e Trafalis e Ince (2000) também utilizaram o SVR para averiguar a sua eficiência em finanças e alcançaram resultados promissores.

Huang (2012) propôs um método híbrido envolvendo o SVR e Algoritmos Genéticos para a seleção ótima de ativos e posterior formação de portfólio. A proposta de Huang (2012) foi utilizar o SVR para gerar um *ranking* confiável de ações das 200 maiores empresas da Bolsa de Valores de *Taiwan,* tendo como critério o seu retorno. As ações pertencentes ao topo desse *ranking* foram selecionadas para formar o portfólio e nelas o Algorítimo Genético (AG) foi aplicado a fim de otimizar os parâmetros e selecionar variáveis.

Esse modelo é similar ao proposto por Huang (2012), mas possibilita a otimização dos parâmetros do SVR também por meio do AG. Os resultados mostraram que quando se utilizou apenas o SVR obteve-se um retorno acumulado

de aproximadamente 275%, para uma carteira de 20 ativos. Já ao combinar o SVR e o GA, em uma carteira de 10 ativos, o retorno foi de aproximadamente 350%. O *benchmark* por sua vez teve seu maior retorno acumulado igual a 175%.

Tay e Cao (2001) aplicaram o SVR para previsão do retorno de ativos a fim de comparar a acurácia desse modelo com a apresentada pelo modelo não-paramétrico de Redes Neurais. Os dados utilizados são oriundos de cinco contratos futuros do mercado *Chicago Mercantile* que referiam-se a: índices das 500 ações da *Standard & Poor*, um título do governo americano com vencimento em 30 anos (CBOT-US), um título da dívida pública americana com vencimento em 10 anos (CBOT-BO), um título público alemão com vencimento em 10 anos (EUREX-Bund) e ao índice de ações do governo francês (MATIF-CAC40). Um diferencial deste estudo foram as transformações feitas nos dados visando aproximar os dados da distribuição normal.

Primeiramente transformou-se os dados para a diferença relativa de cinco dias em porcentagem do preço, como sugerido por Thomason (1999). Segundo Tay e Cao (2001), essa transformação suaviza a cauda pesada dos dados e o alto grau de curtose. Depois, os valores faltantes foram substituídos por valores superiores ou inferiores a 2 desvios-padrão. Por fim, os dados foram padronizados no intervalo $[-0.9,0.9]$. Para escolha do modelo mais ajustado foram utilizados como critérios: Erro Quadrático Médio Normalizado (EQMN), Erro Médio Absoluto (EMA), Simetria Direcional (SD) e a Simetria Direcional Ponderada (SDP).

Os resultados da pesquisa deixaram evidente que o SVR superou o modelo de Redes Neurais em quase todos os critérios e os autores concluem que o SVR é aplicável em Séries Temporais Financeiras, principalmente pelo fato de não minimizar apenas o erro empírico, mas também o Risco Estrutural. Além disso, foram apontadas como vantagens o fato do SVR sempre convergir para um máximo global e ter menos parâmetros livres que as Redes Neurais. Porém, o SVR ainda carece de aprimoramentos na escolha dos parâmetros para previsão valores e na escolha do núcleo *Kernel.*

Huang, Chuang, Wu e Lai (2010) implementaram um modelo que considera o caos intrínseco às séries temporais a fim de prever taxas de câmbio. Eles incorporaram o SVR ao modelo para prever, no período de 3 de Janeiro de 2005 a 31 de Dezembro de 2007, 6 taxas de câmbio, sendo elas: Euro/Dólar (EUR/USD), Libras/Dólar (GBP/USD), NZD/Dólar (NZD/USD), AUD/Dólar (AUD/USD), Yen/Dólar

(JPY/USD) e RUB/Dólar (RUB/USD).

Os autores compararam os resultados de 4 modelos: SVR puro, SVR com a abordagem de caos determinístico, Redes Neurais puras e Redes Neurais com a abordagem de caos determinístico. Para medir o desempenho de cada um deles utilizaram o Erro Quadrático Médio, a Raiz do Erro Quadrático Médio e Erro Médio Absoluto. O modelo que apresentou melhor desempenho consistente em todas as métricas foi o SVR com abordagem de caos determinístico proposta pelos autores.

2.2 Metodologia Máquinas de Suporte Vetorial de Regressão

O SVR (*Support Vector Machine Regression*) tem por objetivo encontrar uma função generalista não-linear que descreva o modelo sugerido com mínimo erro. Isto é, o SVR busca minimizar a distância entre a função e a variável dependente a menos de um erro estipulado. Essa versão das Máquinas de Suporte Vetorial ainda nos permite escolher com maior liberdade os parâmetros de penalização e o grau de achatamento da função, o que permite melhor adequamento da função aos dados.

O problema básico de regressão é encontrar uma função que aproxima um determinado conjunto de dados e determinar o seu grau de perda que é dado por uma Função de Perda, conhecida como *Loss Function*, descrita por:

$$|y - f(x)| \leq \varepsilon \tag{1}$$

ε é entendido como um erro permitido e à medida que o ponto se distancia de ε, imputa-se uma penalização numérica no modelo. Dessa forma, além de considerar a função de perda, permitindo a inclusão de erros, o SVR tenta ainda minimizar o recíproco da margem e permite a inclusão de variáveis de folga, evitando que a função resultante seja extremamente ajustada ao treinamento, como ilustrado de forma linear na Figura 1 e de forma não linear na Figura 2.

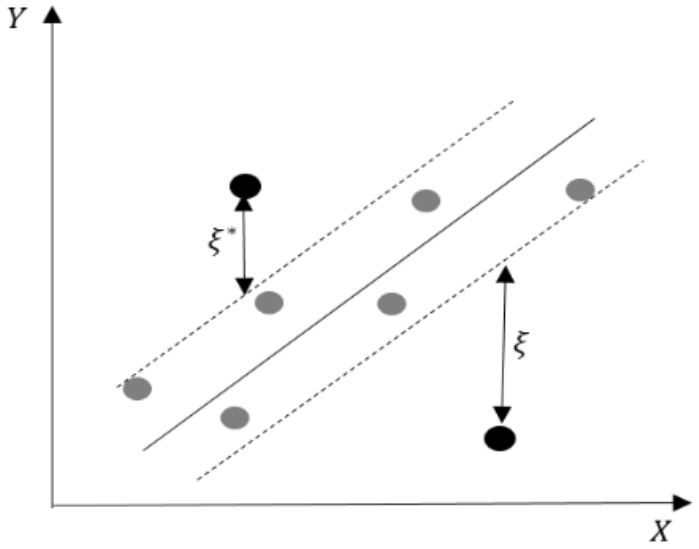

Figura 1: Função de regressão linear
Fonte: Albuquerque (2014)

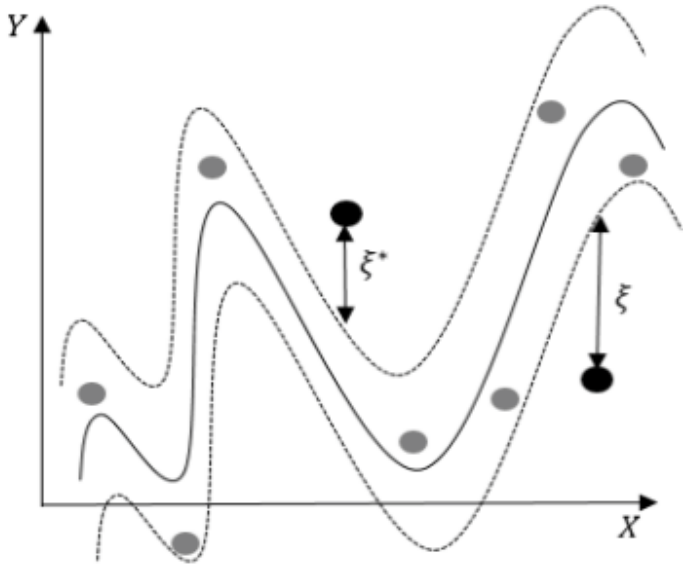

Figura 2: Função de regressão não linear
Fonte: Albuquerque (2014)

Assim como nas Máquinas de Suporte Vetorial de classificação, o SVR lida com a não linearidade por meio do mapeamento dos dados em alta dimensão através da função *Kernel* para que nesse espaço de característica de alta dimensão os dados sejam linearmente separáveis. A Figura 3 ilustra esse processo.

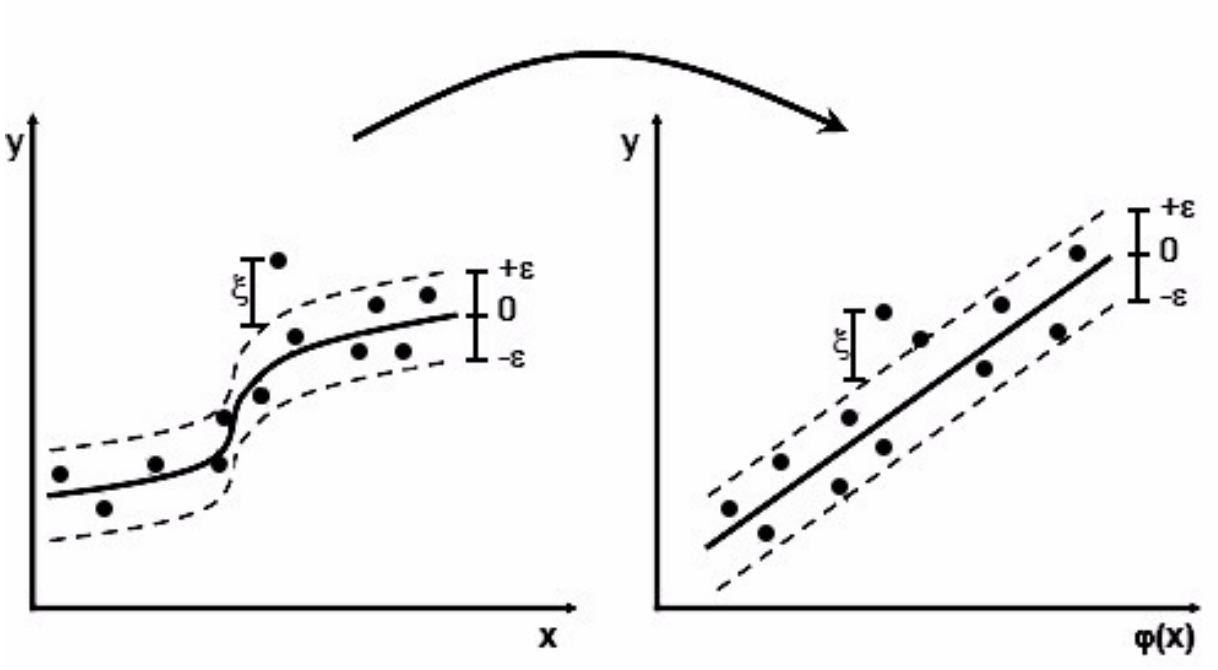

Figura 3: Processo de mapeamento SVR.
Fonte: Elaborado pela autora.

Segundo Soman et al. (2009), a formulação primal do SVR com variáveis de folga pode ser descrita como:

(2)

$$\text{Min} \quad \text{W} = \frac{1}{2}\mathbf{w}^{\mathbf{T}}\mathbf{w} + \text{C}\mathbf{1}^{\mathbf{T}}\xi + \text{C}\mathbf{1}^{\mathbf{T}}\xi^{*}$$

Sujeito a:

$$-\boldsymbol{\phi}(\text{x})\mathbf{w} + \xi \geq -\varepsilon\mathbf{1} - y$$

$$\boldsymbol{\phi}(\text{x})\mathbf{w} + \xi^{*} \geq -\varepsilon\mathbf{1} + y$$

$$\xi \geq 0$$

$$\xi^{*} \geq 0$$

onde,

(3)

$\boldsymbol{\Phi} = \begin{pmatrix} \boldsymbol{\phi}(\text{x}_1)^{\text{T}} \\ \boldsymbol{\phi}(\text{x}_2)^{\text{T}} \\ \vdots \\ \boldsymbol{\phi}(\text{x}_\text{n})^{\text{T}} \end{pmatrix}$, $\mathbf{w} = \begin{pmatrix} \text{w}_1 \\ \text{w}_2 \\ \vdots \\ \text{w}_\text{n} \end{pmatrix}$ e ξ_i, ξ_i^{*} são variáveis de folga.

Na forma Lagrangeana, o SVR é:

(4)

$$L(w,\xi,\xi^*) = \frac{1}{2}\boldsymbol{w^T w} + C[\mathbf{1}^T\xi + \mathbf{1}^T\xi^*] - \boldsymbol{\lambda^T}[-\boldsymbol{\phi}(x)\boldsymbol{w} + \xi + \varepsilon\mathbf{1} + y]$$
$$- \boldsymbol{\lambda^{*T}}[+\boldsymbol{\phi}(x)\boldsymbol{w} + \xi^* + \varepsilon\mathbf{1} - y] - \boldsymbol{\mu^T}\xi - \boldsymbol{\mu^{*T}}\xi^*$$

sendo $\mu, \mu^*, \lambda, \lambda^*$os multiplicadores de Langrange. Resolvendo as questões de primeira ordem obtém-se:

(5)

$$\frac{\partial}{\partial w}L(w,\xi,\xi^*) = \boldsymbol{w^T} - \boldsymbol{\lambda^T\Phi} - \boldsymbol{\lambda^{*T}\Phi} = 0 \rightarrow \boldsymbol{w^T} = (\boldsymbol{\lambda^{*T}} - \boldsymbol{\lambda^T})\boldsymbol{\Phi}$$

(6)

$$\frac{\partial}{\partial w}L(w,\xi,\xi^*) = C\mathbf{1^T} - \boldsymbol{\lambda^T} - \boldsymbol{\mu^T} = 0$$

(7)

$$\frac{\partial}{\partial w}\boldsymbol{L}(w,\xi,\xi^*) = C\mathbf{1^T} - \boldsymbol{\lambda^{*T}} - \boldsymbol{\mu^{*T}} = 0$$

Substituindo na equação 4, o problema pode ser escrito na sua forma dual, dada pelo Dual de Wolfe (1961):

(8)

$$\text{Max} \quad L(\lambda,\lambda^*) = \ -\tfrac{1}{2}(\boldsymbol{\lambda^{*T}} - \boldsymbol{\lambda})\boldsymbol{\Phi\Phi^T} - \varepsilon(\boldsymbol{\lambda^T}\mathbf{1} - \boldsymbol{\lambda^{*T}}\mathbf{1}) + (\boldsymbol{\lambda^{*T}} - \boldsymbol{\lambda^T})y$$

Sujeito a:

$0 \leq \ \lambda, \lambda^* \leq \mathrm{C}1$

onde $\boldsymbol{\Phi\Phi^T}$ é a matriz *Kernel* composta pelos elementos $\mathbf{K}(\mathbf{x_i}, \mathbf{x_j}) = \boldsymbol{\phi}(x_i)^T\boldsymbol{\phi}(x_j)$ para $i = 1, \ldots, n$ e $j = 1, \ldots, n$

Sendo assim, a função de decisão do SVR é dada por:

(9)

$$f(x_i) = \sum_{j=1}^{n} \mathbf{K}(\mathbf{x_i}, \mathbf{x_j})(\boldsymbol{\lambda^*}_j - \boldsymbol{\lambda_j})$$

ou

(10)

$$f(x_i) = \textstyle\sum_{j=1}^{n} \mathbf{K}(\mathbf{x_i}, \mathbf{x_j})(\boldsymbol{\lambda^*}_j - \boldsymbol{\lambda_j}) - b$$

quando se trabalha com um termo de viés.

2.2.1 Funções Kernel

O mapeamento dos dados em alta dimensão é definido pela função *Kernel* $\mathrm{K}(\boldsymbol{x_i}, \boldsymbol{x_j})$ que pode ser descrita sinteticamente como:

(11)

$$\mathbf{K}(\boldsymbol{x_i}, \boldsymbol{x_j}) = \boldsymbol{\phi}(x_i)^{\boldsymbol{T}} \boldsymbol{\phi}(x_j)$$

O *Kernel* corresponde a um produto interno em um espaço de característica, usualmente de alta dimensão (Hofmann, Schölkopf, e Smola, 2008). Segundo Hofmann et al. (2008), os *Kernels* abordam as principais questões em aprendizado de máquina e inferência, pois formalizam a noção de similaridade dos dados e caracterizam a função da classe, usada para estimação.

A interpretação da função *Kernel* como medida de similaridade fica evidente quando observa-se que ela define a distância d no espaço de *input* por Rüping (2001):

(12)

$$\begin{aligned} d^2(x_i, x_j) &= \left(\boldsymbol{\phi}(x_i) - \boldsymbol{\phi}(x_j) \right)^2 \\ &= \boldsymbol{k}(\boldsymbol{x}, \boldsymbol{x}) - 2\boldsymbol{k}(\boldsymbol{x}, \boldsymbol{y}) + \boldsymbol{k}(\boldsymbol{y}, \boldsymbol{y}) \end{aligned}$$

O *Kernel* leva cada ponto x em um mapa $\boldsymbol{\phi}(x)$**.** Dessa forma, ao invés de se trabalhar com a matriz $\mathbf{A}$ - de dimensão *n* x *p*, onde cada linha representa uma observação de uma população e cada coluna, uma característica, isto é, uma variável dessa população - trabalha-se com $\mathbf{F}$, construída a partir de $\boldsymbol{\phi}(x)$**.** Como $\mathbf{F}$ é de dimensão muito grande, o cálculo da matriz $\mathbf{FF^T}$ exige uma quantidade enorme de operações, o que torna o processo bastante oneroso. Essa explosão de dimensionalidade pode ser evitada com o uso do *Kernel Trick*. A função $\mathbf{K}(\boldsymbol{x_i}, \boldsymbol{x_j})$, por ser uma função do espaço de *input,* não requer o mapeamento, mas sim o resultado escalar do produto $\boldsymbol{\phi}(x_i)^{\boldsymbol{T}} \boldsymbol{\phi}(x_j)$ no espaço de característica. Os produtos escalares são encontrados diretamente com o cálculo dos *Kernels* $\mathbf{K}(\boldsymbol{x_i}, \boldsymbol{x_j})$ para

determinado conjunto de vetores de dados de treinamento em um espaço de *input*. Então, basta que a matriz $\mathbf{FF^T}$ seja conhecida para construir um SVM que opere em um espaço de dimensão extremamente alta, até mesmo infinita. Por meio dessa artimanha o algoritmo do treinamento do SVM não linear se torna o mesmo que o do SVM linear, tornando as operações e a abordagem do problema da Maldição da Dimensionalidade bem mais simples. A Maldição da Dimensionalidade surge no aprendizado de máquina porque a generalização correta se torna exponencialmente mais difícil à medida que o número de exemplos de treinamento cresce. Quanto maior a dimensão, mais complexo se torna o cálculo do produto interno $\boldsymbol{\phi}(x_i)^{\boldsymbol{T}}\boldsymbol{\phi}(x_j)$ e uma das saídas é trabalhar diretamente com a matriz K.

Considere a seguinte matriz de dados e que em sua dimensão original eles não são linearmente separáveis:

(13)

$$\mathbf{A} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix}$$

Sabe-se que:

(14)

$$\mathbf{F} = \begin{pmatrix} \boldsymbol{\phi}^{\boldsymbol{T}}(x_i) \\ \vdots \\ \boldsymbol{\phi}^{\boldsymbol{T}}(x_n) \end{pmatrix}$$

(15)

$$\mathbf{F^T} = \left(\boldsymbol{\phi}(x_i), \dots, \boldsymbol{\phi}(x_n)\right)$$

Logo,

(16)

$$\mathbf{FF^T} = \begin{pmatrix} \boldsymbol{\phi}^{\boldsymbol{T}}(x_1)\boldsymbol{\phi}(x_1) & \cdots & \boldsymbol{\phi}^{\boldsymbol{T}}(x_1)\boldsymbol{\Phi}(x_n) \\ \vdots & \ddots & \vdots \\ \boldsymbol{\phi}^{\boldsymbol{T}}(x_n)\boldsymbol{\phi}(x_1) & \cdots & \boldsymbol{\phi}^{\boldsymbol{T}}(x_n)\boldsymbol{\Phi}(x_n) \end{pmatrix}$$

A partir do *Kernel* Quadrático, dado por $\boldsymbol{\phi}(x) = \begin{pmatrix} x^2 \\ \sqrt{2}x \\ 1 \end{pmatrix}$, é possível levar cada ponto em $\mathbb{R}^2$ para $\mathbb{R}^3$ e construir a matriz $\mathbf{F}$. Como $\mathbf{K} = \mathbf{F}\mathbf{F}^{\mathbf{T}} = \boldsymbol{\phi}^{\boldsymbol{T}}(x_i)\boldsymbol{\phi}(x_j)$, a matriz *Kernel* pode ser escrita como:

(17)

$$\mathbf{K} = \begin{bmatrix} \boldsymbol{\phi}(x_1)^{\boldsymbol{T}}\boldsymbol{\phi}(x_1) & \boldsymbol{\phi}(x_1)^{\boldsymbol{T}}\boldsymbol{\phi}(x_2) & \boldsymbol{\phi}(x_1)^{\boldsymbol{T}}\boldsymbol{\phi}(x_3) \\ \boldsymbol{\phi}(x_2)^{\boldsymbol{T}}\boldsymbol{\phi}(x_1) & \boldsymbol{\phi}(x_2)^{\boldsymbol{T}}\boldsymbol{\phi}(x_2) & \boldsymbol{\phi}(x_2)^{\boldsymbol{T}}\boldsymbol{\phi}(x_3) \\ \boldsymbol{\phi}(x_3)^{\boldsymbol{T}}\boldsymbol{\phi}(x_1) & \boldsymbol{\phi}(x_3)^{\boldsymbol{T}}\boldsymbol{\phi}(x_2) & \boldsymbol{\phi}(x_3)^{\boldsymbol{T}}\boldsymbol{\phi}(x_3) \end{bmatrix}$$

(18)

$$\mathbf{K} = \begin{bmatrix} (\boldsymbol{x_1}^{\boldsymbol{T}}\boldsymbol{x_1} + 1)^2 & (\boldsymbol{x_1}^{\boldsymbol{T}}\boldsymbol{x_2} + 1)^2 & (\boldsymbol{x_1}^{\boldsymbol{T}}\boldsymbol{x_3} + 1)^2 \\ (\boldsymbol{x_2}^{\boldsymbol{T}}\boldsymbol{x_1} + 1)^2 & (\boldsymbol{x_2}^{\boldsymbol{T}}\boldsymbol{x_2} + 1)^2 & (\boldsymbol{x_2}^{\boldsymbol{T}}\boldsymbol{x_3} + 1)^2 \\ (\boldsymbol{x_3}^{\boldsymbol{T}}\boldsymbol{x_1} + 1)^2 & (\boldsymbol{x_3}^{\boldsymbol{T}}\boldsymbol{x_2} + 1)^2 & (\boldsymbol{x_3}^{\boldsymbol{T}}\boldsymbol{x_3} + 1)^2 \end{bmatrix}$$

Dado que o *Kernel* em questão é $\mathbf{K}(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i^T x_j} + 1)^2$ e $\mathbf{x_1} = \binom{1}{2}$, $\mathbf{x_2} = \binom{3}{4}$, $\mathbf{x_3} = \binom{5}{6}$, tem-se que:

(19)

$$\mathbf{K} = \begin{pmatrix} 36 & 144 & 324 \\ 144 & 676 & 1600 \\ 324 & 1600 & 3844 \end{pmatrix}$$

Condiderando um problema a ser solucionado por um SVM de classificação, cuja metodologia está detalhada no Capítulo 4, pode-se citar um exemplo simples de uma situação na qual há a necessidade da construção de modelos de classificação não lineares. Na Figura 4, onde o limite de separação é quadrático, fica evidente que nenhum hiperplano linear de separação com erro zero pode ser encontrado nessa situação. A melhor função linear de separação geraria cinco classificações erradas, sendo três na classe negativa e duas na classe positiva. Entretanto, se a função não linear de separação for utilizada, as classes podem ser separadas sem nenhum erro.

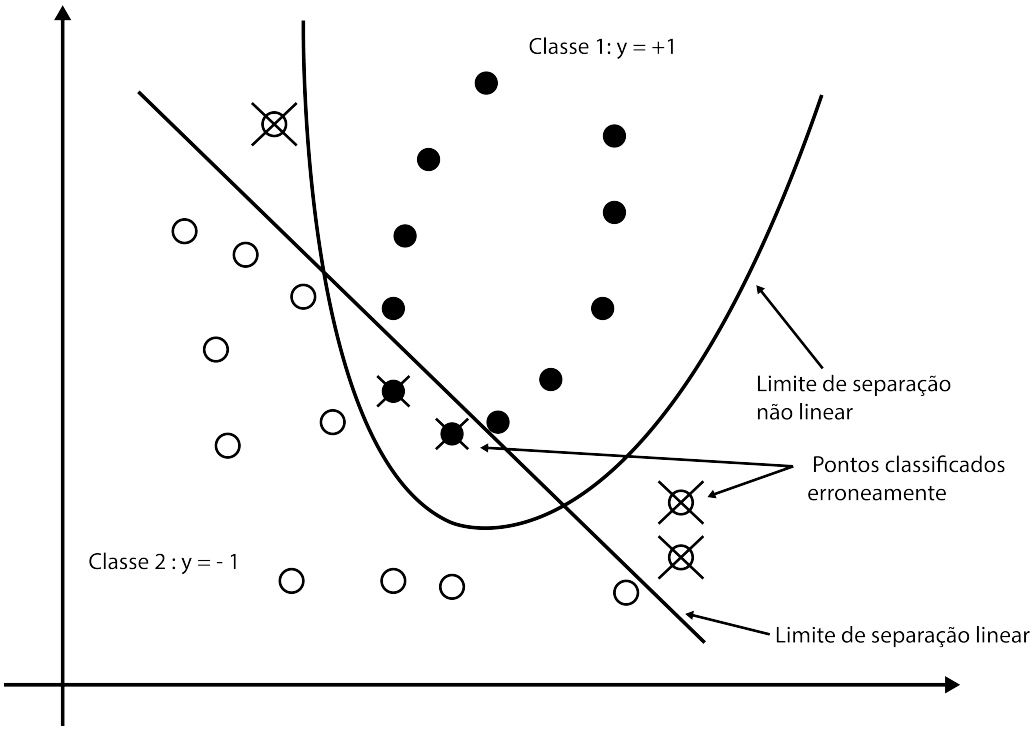

Figura 4: Separador linear *versus* separador não linear
Fonte: Traduzido de Soman et al. (2009).

Agora suponha $\mathbf{x} \in \Re^2$, isto é, $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$. Se o mapeamento escolhido for $\boldsymbol{\phi} = \begin{bmatrix} x_1^2 \\ \sqrt{x_1 x_2} \\ x_2^2 \end{bmatrix}$, ou seja, $\Re^2 \longrightarrow \Re^3$, então o produto interno será:

(20)

$$\boldsymbol{\phi}(x_i)^{\mathbf{T}}\boldsymbol{\phi}(x_j) = \begin{bmatrix} x_{i1}^2 & \sqrt{2}x_{i1}^2 x_{i2}^2 & x_{i2}^2 \end{bmatrix} \begin{bmatrix} x_{j1}^2 \\ \sqrt{2}x_{j1}^2 x_{j2}^2 \\ x_{j2}^2 \end{bmatrix}$$

$$= \begin{bmatrix} x_{i1}^2 x_{j1}^2 & 2x_{i1}^2 x_{i2}^2 x_{j1}^2 x_{j2}^2 & x_{i2}^2 x_{j2}^2 \end{bmatrix} = (\mathbf{x_i}^{\mathbf{T}}\mathbf{x_j})^{\mathbf{2}} = \mathbf{K}(\mathbf{x_i}, \mathbf{x_j})$$

É possível notar novamente que para calcular o produto escalar no espaço de característica $\boldsymbol{\phi}(x_i)^{\mathbf{T}}\boldsymbol{\phi}(x_j)$ não há necessidade de se aplicar de fato o mapeamento $\boldsymbol{\phi} = \begin{bmatrix} x_1^2 \\ \sqrt{x_1 x_2} \\ x_2^2 \end{bmatrix}$, já que o produto pode ser encontrado diretamente calculando $(\mathbf{x_i}^{\mathbf{T}}\mathbf{x_j})^{\mathbf{2}}$.

Para exemplificar ainda mais, assuma agora o seguinte mapeamento:

(21)

$$\boldsymbol{\phi} = \begin{bmatrix} 1 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \\ \sqrt{2}x_1x_2 \\ x_1^2 \\ x_2^2 \end{bmatrix}$$

Ou seja, nesse caso há um mapeamento $\Re^2 \longrightarrow \Re^5$ e um termo de viés como o valor da sexta dimensão, sendo este uma constante. Então, o produto interno no espaço de característica $\mathbf{F}$ é dado como:

(22)

$$\boldsymbol{\phi}(x_i)^{\mathbf{T}}\boldsymbol{\phi}(x_j) = \left[1 + 2x_{i1}x_{j1} + 2x_{i2}x_{j2} + 2x_{i1}x_{i2}x_{j1}x_{j2} + x_{i1}^2x_{j1}^2 + x_{i2}^2x_{j2}^2\right]$$
$$= 1 + 2\mathbf{x_i}^{\mathbf{T}}\mathbf{x_j} + (\mathbf{x_i}^{\mathbf{T}}\mathbf{x_j})^{\mathbf{2}} = (\mathbf{x_i}^{\mathbf{T}}\mathbf{x_j} + \mathbf{1})^{\mathbf{2}} = \mathbf{K}(\mathbf{x_i}, \mathbf{x_j})$$

Ou

(23)

$$\mathbf{K}(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i}^{\mathbf{T}}\mathbf{x_j} + 1)^{\mathbf{2}} = \boldsymbol{\phi}(x_i)^{\mathbf{T}}\boldsymbol{\phi}(x_j)$$

Sintetizando, um *Kernel* polinomial de grau p é calculado da seguinte forma:

(24)

$$\mathbf{K} = \begin{bmatrix} (\mathbf{x_1}^{\mathbf{T}}\mathbf{x_1} + 1)^p & (\mathbf{x_1}^{\mathbf{T}}\mathbf{x_2} + 1)^p & . & (\mathbf{x_1}^{\mathbf{T}}\mathbf{x_m} + 1)^p \\ . & . & (x_i{}^{T}x_j + 1)^p & . \\ . & . & . & . \\ (\mathbf{x_m}^{\mathbf{T}}\mathbf{x_1} + 1)^p & (\mathbf{x_m}^{\mathbf{T}}\mathbf{x_2} + 1)^p & . & (\mathbf{x_m}^{\mathbf{T}}\mathbf{x_m} + 1)^p \end{bmatrix}$$

Nota-se que o produto interno é calculado na sua dimensão original.

Soman et al. (2009) apresentam o seguinte exemplo com o *Kernel* Gaussiano para provar a operação do SVM em dimensões infinitas. Considerando sua função que é dada por:

(25)

$$\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\sigma \parallel \boldsymbol{x} - \boldsymbol{y} \parallel^2\right)$$

onde σ é um parâmetro positivo para controlar o raio, pode-se expandi-la, o que resulta em:

(26)

$$exp(-\sigma \parallel \boldsymbol{x} - \boldsymbol{y} \parallel^2) = exp(\parallel \boldsymbol{x} \parallel^2)\, exp(-\sigma \parallel \boldsymbol{y} \parallel^2)\, exp(2\sigma \boldsymbol{x^T y})$$

Como,

(27)

$$exp(2\sigma \boldsymbol{x^T y}) = 1 + 2\sigma \boldsymbol{x^T y} + \frac{(2\sigma)^2}{2!}(\boldsymbol{x^T y})^2 + \frac{(2\sigma)^3}{3!}(\boldsymbol{x^T y})^3 + \cdots$$

$exp(2\sigma \boldsymbol{x^T y})$ é um somatório infinito de polinômios, ou seja, o *Kernel* Gaussiano é um *Kernel* que mapeia um ponto em um espaço de dimensões infinitas. Vale ressaltar também que $\exp(-\sigma \parallel \boldsymbol{x} \parallel^2)$ e $\exp(-\sigma \parallel \boldsymbol{y} \parallel^2)$ também são *Kernels* assim como o produto deles, $K(\boldsymbol{x}, \boldsymbol{y}) = \exp\,(-\sigma \parallel \boldsymbol{x} - \boldsymbol{y} \parallel^2)$.

Os métodos *Kernel* possuem várias vantagens que os tornam atrativos. Eles são considerados universalmente aplicáveis, pois à medida que a amostra de treinamento m se torna arbitrariamente grande, $m \to \infty$, a estimativa do *Kernel* se aproxima da função alvo ideal, com apenas leves restrições. Outra vantagem dos métodos *Kernel* é que não se faz necessário nenhum treinamento para construir o modelo, o próprio conjunto de treinamento é o modelo. Os procedimentos são também conceitualmente simples e fáceis de serem explicados.

Entretanto, os métodos *Kernel* também possuem algumas desvantagens que têm impedido várias aplicações, principalmente no campo da mineração de dados. Como não há um modelo concreto para construção dos *Kernels*, não é possível discernir como a função depende das respectivas variáveis preditoras x. Então, pode-se dizer que a predição dos métodos *Kernel* é uma "caixa preta". Além disso, para fazer uma previsão, os *Kernels* precisam examinar toda a base de dados, o que requer uma memória computacional suficiente para armazenar todo o conjunto de dados, e a computação necessária para fazer cada previsão é proporcional ao tamanho da amostra de formação m. Então, no caso de grandes conjuntos de dados, esse método se torna mais lento do que métodos concorrentes. Talvez a maior limitação dos métodos *Kernel* seja estatística já que para qualquer m finito, a acurácia da previsão depende criticamente da escolha da função *Kernel* e ainda há o risco do desencadeamento da Maldição da Dimensionalidade.

2.2.2 *Tipos de Kernel*

A seleção adequada da função *Kernel* tem um papel importante na classificação do SVM já que o *Kernel* define o espaço de característica no qual os dados serão classificados (Wei, 2012).

Para que um *Kernel* seja válido, ele deve obedecer às Condições de Mercer (1909). Soman et al. (2009) definem o Teorema de Mercer como uma representação de uma função simétrica, positiva definida em um quadrado integrável cuja soma converge para o produto de funções. Em outras palavras, existe um mapeamento $\boldsymbol{\Phi}(x)$ e uma expansão $\boldsymbol{K}(\boldsymbol{x_i}, \boldsymbol{x_j}) = \boldsymbol{\phi}^{\boldsymbol{T}}(x_i)\boldsymbol{\phi}(x_j)$ se e somente se:

Teorema 1. *Seja χ o domínio de uma função, considere uma função real $K(.,.)$ bivariada, simétrica e contínua definida em $\chi.\chi$. Considere $\oint$ um espaço de característica. Então, existe uma transformação $\phi: \chi \rightarrow \oint$ tal que $K(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{\Phi}^{\boldsymbol{T}}(x)\boldsymbol{\Phi}(y)$ se e somente se, **K** satisfaz a condição de Mercer.*

Portanto, o *Kernel* é admissível quando é uma função bivariada, simétrica e positiva definida em um quadrado integrável, ou seja, o *Kernel* é admissível quando para toda função $g(x) \in L^2(\mathbb{R}^2)\left[i, e \int |g(x)|^2 dx < \infty\right]$. Então, se $\int K(x_i, x_j) g(x_i) g(x_j) d(x_i) d(x_j) \geq 0$, o *Kernel* é dito admissível.

Na maioria dos casos, o uso de um K*ernel* positivamente definido assegura que o problema de otimização seja convexo e consequentemente a solução obtida seja única (Boughorbel, Tarel, e Boujemaa, 2005). Entretanto, alguns *Kernels* com funções que não são estritamente positivas definidas também têm apresentado bons desempenhos na prática. Um exemplo é o *Sigmoid Kernel*, que apesar do seu vasto uso, não é positivo definido para certos valores dos seus parâmetros.

Várias funções podem ser aplicadas como função *Kernel* no SVM e cada uma delas constrói uma superfície de separação diferente no espaço de *input*. Como achar eficientemente o *Kernel* ótimo para um determinado aprendizado é ainda um problema sem solução (Rüping, 2001). O Quadro 1 apresenta os 15 diferentes tipos de *Kernel* utilizados neste estudo.

Kernel	**Fórmula**	**Referências**
Linear Kernel	$k(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^T \boldsymbol{y} + d$	Rüping (2001) Hsu, Chang, e Lin (2003)
Gaussian Kernel	$k(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{\| x - y \|^2}{2\sigma^2}\right)$	Fan e Palaniswami (2001) Tay e Cao (2001) Rüping (2001) Ali e Smith (2003) Hofmann et al. (2008) Wei (2012) Genton (2002) Rahimi e Recht (2007) Boughorbel et al. (2005) Hsu et al. (2003)
Exponential Kernel	$k(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{\| x - y \|}{2\sigma^2}\right)$	Genton (2002)
Laplacian Kernel	$k(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{\| x - y \|}{\sigma}\right)$	Rahimi e Recht (2007) Boughorbel et al. (2005)
Hyperbolic Tangent (Sigmoid) Kernel	$k(\boldsymbol{x}, \boldsymbol{y}) = \tanh(\alpha \boldsymbol{x}^T \boldsymbol{y} + d)$	Hsu et al. (2003)
Rational Quadratic Kernel	$k(\boldsymbol{x}, \boldsymbol{y}) = 1 - \frac{\| x - y \|^2}{\| x - y \|^2 + d}$	Genton (2002)
Multiquadratic Kernel	$k(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\| x - y \|^2 + d^2}$	Ali e Smith (2003)
Inverse Multiquadratic Kernel	$k(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{\sqrt{\| x - y \|^2 + d^2}}$	Micchelli (1986)
Power Kernel	$k(\boldsymbol{x}, \boldsymbol{y}) = - \| x - y \|^d$	Boughorbel et al. (2005)
Log Kernel	$k(\boldsymbol{x}, \boldsymbol{y}) = -\log(\| x - y \|^d + 1)$	Boughorbel et al. (2005)
Cauchy Kernel	$k(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{1 + \frac{\|x-y\|^2}{\sigma^2}}$	Rahimi e Recht (2007)
Chi-Square Kernel	$k(\boldsymbol{x}, \boldsymbol{y}) = 1 - \sum_{i=1}^{n} \frac{(x_i - y_i)^2}{\frac{1}{2}(x_i + y_i)}$	Vedaldi e Zisserman (2012)
Histogram Intersection Kernel	$k(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{n} \min(x_i, y_i)$	Vedaldi e Zisserman (2012)

Generalized T-Student Kernel	$k(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{1+\| x - y \|^d}$	Kamath, Shehu, e De Jong (2010)
Wavelet Kernel	$k(\boldsymbol{x}, \boldsymbol{y}) = \prod_{i=1}^{N} h\left(\frac{x_i - d}{a}\right) h\left(\frac{y_i - d}{a}\right)$	Zhang e Zhao (2010) Wei (2012)

Quadro 1: Funções *Kernel* utilizadas.
Fonte: Elaborado pela autora.

Os modelos de SVM com *Kernel* Gaussiano apresentam boas previsões, mas fornecem pouca informação sobre as propriedades fundamentais do problema, por exemplo, quais atributos são importantes (Rüping, 2001).

O estudo de Rüping (2001) comprova que diferentes *Kernels* podem ser usados para análises de séries temporais univariadas ou multivariadas com o SVM, entretanto, cada um desses *Kernels* modela suposições diferentes no processo de geração de séries temporais. Segundo Ali e Smith (2003), antes de escolher o *Kernel* apropriado é importante analisar a natureza dos dados, isto é, a distribuição deles, pois a escolha do *Kernel* depende em grande parte do problema em questão. Portanto, a motivação para essa escolha pode ser bastante intuitiva e depende diretamente do que se está tentando modelar, ou seja, depende do tipo de informação que se espera extrair dos dados.

3 PRIMEIRA APLICAÇÃO: MÁQUINAS DE SUPORTE VETORIAL DE REGRESSÃO PARA FORMAÇÃO DE PORTFÓLIOS

Este capítulo relata a primeira aplicação empírica desta dissertação. Nesta aplicação, o SVR foi usado para prever o retorno de cada ativo e foram formados portfólios trimestrais ponderados com as melhores ações.

3.1 Procedimentos de Coleta e Análise de Dados: Máquinas de Suporte Vetorial de Regressão

Primeiramente coletou-se os dados de indicadores financeiros trimestrais a serem utilizados como insumos para o SVR. Não há consenso na literatura sobre quais variáveis são mais significativas para melhor desempenho das Máquinas de Suporte Vetorial. Portanto, na primeira aplicação desta dissertação foram escolhidos *a priori* 24 indicadores financeiros fundamentalistas, considerando sugestões da literatura e a disponibilidade dos dados no sistema de coleta da *Bloomberg* para o período delimitado. O Quadro 2 elenca essas variáveis.

Variável	Tradução	Autores que utilizaram
Current Assets	Ativo Circulante	Fan e Palaniswami (2001) Huerta et al. (2013)
Current Liabilities	Passivo Circulante	Fan e Palaniswami (2001) Huerta et al. (2013)
Long Term Borrowings	Financiamentos de Longo Prazo	Fan e Palaniswami (2001) Huerta et al. (2013)
Total Assets	Ativo Total	Fan e Palaniswami (2001) Huerta et al. (2013)
Total Capital	Capital Total	Fan e Palaniswami (2001)
Total Debt	Dívida Total	Fan e Palaniswami (2001) Huerta et al. (2013)
Total Liabilities and Equity	Passivo Total e Patrimônio Líquido	Fan e Palaniswami (2001)
Total Market Value	Valor de Mercado Total	Fan e Palaniswami (2001) Emir, Dinçer, e Timor (2012)
Closing Price	Preço de Fechamento	Lai et al. (2006) Emir et al. (2012)
Total Liabilities	Passivo Total	Yu, Lu e Chang (2008) Fan e Palaniswami (2001)

		Huerta et al. (2013)
Operating Cash Flow	Fluxo de Caixa Operacional	Yu, Lu e Chang (2008)
Other Long Term Liabilities	Outras Obrigações de Longo Prazo	Yu, Lu e Chang (2008) Fan e Palaniswami (2001) Huerta et al. (2013)
Total Equity	Patrimônio Líquido Total	Huerta et al. (2013) Emir et al. (2012)
Book Value per Share	Valor Contábil da Ação	Huang (2012)
Accounts Receivable	Contas a Receber	Huerta et al. (2013)
Depreciation	Depreciação	Huerta et al. (2013)
Comparable sales per shares	Venda Comparáveis s por Ação	Huerta et al. (2013) Huang (2012)
Diluted Earnings per Share	Lucro por Ação Diluído	Huerta et al. (2013)
Net Income before extraordinary items	Lucro Líquido Anterior a Item Extraordinários	Huerta et al. (2013)
Operating Income	Resultado Operacional	Huerta et al. (2013)
Income before Tax	Lucro antes dos impostos	Huerta et al. (2013)
Volume	Volume	Huerta et al. (2013)
Revenue per shares	Receita por Ação	Huang (2012).
Sales per shares	Vendas por ação	Huerta et al. (2013) Huang (2012).

Quadro 2: Indicadores financeiros utilizados na pesquisa.
Fonte: Elaborado pela autora.

A próxima etapa foi a coleta do histórico de cotações trimestrais das ações das carteiras em questão para o cálculo do retorno.

O retorno trimestral para cada uma das ações foi calculado da seguinte forma, utilizando os respectivos preços de fechamento ajustados a *splits* e dividendos:

(28)

$$R_t = \frac{P_t}{P_{t-1}} - 1$$

Sendo P_t o preço de fechamento da ação no fim do trimestre e P_{t-1}, o preço de fechamento no início do trimestre. O retorno acumulado ao longo do período foi calculado como:

(29)

$$\mathrm{R_{acumulado}} = \prod_{t=1}^{T} \left(\frac{\mathrm{P_t}}{\mathrm{P_{t-k}}} + 1 \right)$$

Observações com dados incompletos foram imputadas pela média. Foi formada então uma base de dados com indicadores e retornos trimestrais.

As ações foram então ranqueadas em função do retorno apresentado e classificadas em duas classes. A Classe 1 ($y_i = +1$) foi composta pelos 25% das ações com maiores retornos e a Classe 2 ($y_i = -1$) foi composta pelas demais ações que representam 75%.

A fim de evitar *overtraining* utilizou-se o método de Validação Cruzada, sendo a bases de dados dividida em três conjuntos. Há tendências de que o cenário macroeconômico gere um viés significativo nos resultados, então, para minimizar tal efeito, a divisão da base nos conjuntos de treinamento e validação foi feita de forma aleatória e não pelo critério cronológico como usualmente é feita. A seleção do conjunto de teste, por sua vez, foi selecionada de modo temporal para possibilitar o cálculo do retorno acumulado e avaliar o desempenho da máquina.

Inicialmente a base foi dividida aletoriamente em dois conjuntos. O primeiro foi constituído por 25% das observações e foi usado para teste. O segundo composto por 75% das observações e foi usado para treinamento e validação. Dentro deste segundo conjunto, 70% foi usado para treinamento e 30% foi usado para validação. O grande diferencial desta primeira estratégia de formação de portfólios desenvolvida com o SVR foi a construção de 15 máquinas diferentes, cada uma com um *Kernel* distinto. Sendo assim, foi necessário realizar 15 vezes o procedimento de validação cruzada, uma para cada máquina, a fim de encontrar os parâmetros ótimos para cada *Kernel*.

Para definição dos parâmetros C e σ, criou-se uma sequência para o parâmetro C variando de 1 a 1000 e outra para o parâmetro σ, variando de 1^{-19} a 4. Um *grid* foi construído com estas duas sequências para que a combinação ótima de parâmetros fosse encontrada, isto é, para que o par de parâmetros que gerasse o menor erro na etapa de validação fosse encontrado.

A próxima etapa foi a construção das Máquinas de Suporte Vetorial em sua variação de regressão (SVR) para relacionar os indicadores financeiros com o retorno obtido no trimestre seguinte. Foram utilizados os parâmetros ótimos encontrados na etapa anterior de *grid search*.

Para medir o desempenho do SVR, utilizou-se uma medida de acurácia que explicita quantos ativos bons foram classificados corretamente em relação ao total

de ativos selecionados pelo SVM para formar o portfólio, isto é, a porcentagem de acerto dentro de todos ativos escolhidos pela máquina e pode ser descrita por:

(30)

$$\frac{\boldsymbol{Total\ de\ ativos\ da\ Classe\ 1\ classificados\ corretamente}}{\boldsymbol{Total\ de\ ativos\ selecionados}}$$

O SVR foi usado para prever o retorno de cada ativo e esta previsão resultou em um novo *ranking* trimestral dos ativos. Os 25% de maior retorno esperado foram classificados como 1 e os 75% restantes, como -1. Os ativos classificados como 1 foram selecionados para compor os portfólios trimestrais ponderados. O peso de cada ativo nos portfólios se deu pelo seu próprio retorno esperado, isto é, quanto maior o retorno esperado, maior o peso no portfólio.

Posteriormente, os retornos destes portfólios foram comparados trimestralmente aos retornos de um *benchmark* de mercado, cujo retorno foi calculado por uma média igualmente ponderada de todos os do S&P 100. Como medida de risco, utilizou-se o VaR com 5% de chance de erro.

Para avaliar a significância estatística da estratégia e considerar os efeitos de *Data Snooping*, aplicou-se o Teste de Realidade de White (2000) (*White's Reality Check)*, cujas hipóteses são:

$H_0 = t$odos os portfólios formados apresentam retorno acumulado menor ou igual ao retorno acumulado do $benchmark$ de mercado

$H_1 =$ pelo menos 1 dos portfólios formados apresenta retorno acumulado maior que o retorno acumulado do $benchmark$ de mercado

Sendo assim, para que a estratégia de seleção de portfólios seja estatisticamente relevante, os dados devem corroborar para a rejeição da hipótese nula dentro do intervalo de confiança definido como 95%, isto é, o p-valor deve ser menor que 0,5.

3.2 Resultados e Discussão: Máquinas de Suporte Vetorial de Regressão

Na primeira aplicação desta pesquisa o SVR foi construído tendo como insumos os resultados trimestrais dos 24 indicadores financeiros escolhidos. A previsão foi feita então 15 vezes, sendo uma para cada *Kernel* e usando sempre os parâmetros ótimos e específicos de cada *Kernel*, isto é, aqueles parâmetros que no momento do treinamento apresentaram melhor performance de previsão para o respectivo *Kernel*. De acordo com a Tabela 1, a maior acurácia alcançada foi 37.81% com o *Linear Kernel.*

Tabela 1: *Kernels*, parâmetros ótimos e acurácias.

Kernel	**Parâmetros ótimos**	**Acurácia**
Linear Kernel	$d = 50$ $C = 215{,}07$	37,81%
Inverse Multiquadric Kernel	$d = 5{,}94$ $C = 1$	36,98%
Generalized T-Student Kernel	$d = 0{,}0256$ $C = 72{,}36$	36,78%
Hyperbolic Tangent (Sigmoid) Kernel	$d = 6{,}4286$ $C = 357{,}79$	36,16%
Gaussian Kernel	$\sigma = 1^{-15}$ $C = 579{,}37$	36,16%
Power Kernel	$d = 0{,}0345$ $C = 72{,}36$	35,33%
Cauchy Kernel	$\sigma = 3{,}5294$ $C = 1$	35,12%
Exponential Kernel	$\sigma = 6{,}3530$ $C = 1$	35,12%
Log Kernel	$d = 1^{-5}$ $C = 72{,}36$	34,50%
Laplacian Kernel	$\sigma = 3{,}0769$ $C = 72{,}36$	34,09%
Rational Quadratic Kernel	$d = 1^{-5}$ $C = 1$	33,47%
Wavelet Kernel	$a = 15$ $C = 500{,}5$	33,06%
Histogram Intersection Kernel	$C = 123{,}33$	28,72%
Chi-Square Kernel	$C = 11{,}09$	26,65%

Multiquadratic Kernel	$d = 0{,}9677$ $C = 1$	23,35%

Fonte: Elaborado pela autora,

As medidas de acurácia encontradas na aplicação do SVR não se mostraram satisfatórias quantitativamente. Pode-se atribuir isso a alguns fatores que necessitam de aprimoramento para o aperfeiçoamento dessa estratégia, como por exemplo, a hipótese de algumas variáveis utilizadas não terem sido adequadas, possivelmente por alta variabilidade ou por simplesmente não representarem características relevantes para o aprendizado do padrão do ativos e posterior classificação destes. Outra hipótese que justifica este resultado é a possibilidade da medida de acurácia não ter sido justa. Gerlein et al. (2016) argumenta que a maximização da acurácia pode não ser a melhor métrica para avaliar a performance de métodos de aprendizado de máquina que visam classificação em um contexto de mercado financeiro. Segundo este autor, a métrica mais importante para avaliar as estratégias é a lucratividade, refletida no retorno acumulado ao fim do período de investimento.

Buscando mais explicações para este resultado foi feita uma análise qualitativa das ações que compuseram os portfólios selecionados pelo SVR. Nos 15 portfólios formados nessa etapa, em média, 30% das ações que foram selecionadas apresentaram um performance excepcional, isto é, pertenciam de fato ao primeiro quartil do *ranking* de retornos. As demais selecionadas podem ser consideradas boas ou medianas, pois pertenciam ao segundo quartil do *ranking* de retornos. Ou seja, constatou-se que os resultados foram favoráveis para os *Kernels* aqui utilizados, porém, sabe-se que existem outros *Kernels* que não foram utilizados e que poderiam apresentar um desempenho ruim selecionando a maioria das ações no terceiro e quarto quartil.

Com a seleção das melhores ações pelo SVR, foram formados 15 portfólios ponderados pelo retorno previsto. A Tabela 2 apresenta os retornos acumulados e trimestrais médios de cada portfólio, assim como o VaR e a relação retorno/risco de cada um deles.

Tabela 2: Resultados do SVM de Regressão

Opção de	Retorno	Retorno Trimestral	VaR	Retorno/Risco

Investimento	Acumulado	Médio		
Portfólio SVR com Inverse *Multiquadratic Kernel*	374,40%	6,49%	-6,87%	0,9
Portfólio SVR com *Exponential Kernel*	358,81%	6,56%	-5,99%	1,1
Portfólio SVR com *Power Kernel*	330,75%	6,30%	-6,87%	0,92
Portfólio SVR com *Laplacian Kernel*	327,49%	5,30%	-8,02%	0,66
Portfólio SVR com *Generalized T-Student*	315,58%	6,16%	-6,94%	0,89
Portfólio SVR com *Cauchy Kernel*	288,77%	5,61%	-7,74%	0,73
Portfólio SVR com *Hyperbolic Tangent (Sigmoid) Kernel*	285,56%	5,64%	-7,72%	0,73
Portfólio SVR com *Gaussian Kernel*	272,02%	5,48%	-7,80%	0,70
Portfólio SVR com *Linear Kernel*	244,62%	4,12%	-9,40%	0,44
Portfólio SVR com *Rational Quadratic Kernel*	241,53%	5,39%	-7,16%	0,75
Portfólio SVR com *Log Kernel*	240,01%	5,20%	-7,44%	0,70
Portfólio SVR com *Wavelet Kernel*	222,82%	4,12%	-12,72%	0,32
Portfólio SVR com *Histogram Intersection Kernel*	214,54%	4,52%	-9,01%	0,50
Benchmark de Mercado	192,65%	4,03%	-11,74%	0,44
Portfólio SVR com *Chi-Square Kernel*	138,27%	3,27%	-9,49%	0,34
Portfólio SVR com *Multiquadratic Kernel*	130,62%	3,11%	-11,95%	0,26

Fonte: Elaborado pela autora.

O maior retorno acumulado foi o resultante do portfólio selecionado pelo SVR com o *Inverse Multiquadratic Kernel*. Este portfólio apresentou um retorno acumulado em 22 trimestres de $374{,}40\%$ e um VaR de $-6{,}87\%$. Já o portfólio de menor retorno foi aquele formado pelas ações apontadas pelo SVR com o

Multiquadratic Kernel, sendo seu retorno acumulado de $130{,}62\%$ e VaR de $-11{,}95\%$. A menor medida de risco apresentada foi $-5{,}99\%$ e refere-se ao portfólio formado pelo *Exponential Kernel* que também apresentou um retorno acumulado bastante interessante de $358{,}81\%$. O *Wavelet Kernel,* por sua vez, selecionou um portfólio com o maior VaR, $-12{,}72\%$, e retorno de $222{,}82\%$. O portfólio que apresentou melhor relação retorno/risco, foi o portfólio formado pelo *Exponential Kernel* com o valor de $1{,}1$ *,* enquanto o *benchmark* apresentou o valor de $0{,}45$ para esta medida. Isso significa que para cada 1% de risco no portfólio formado pelo *Exponential Kernel*, há $1{,}10\%$ de retorno e no *benchmark*, para cada 1% de risco, há $0{,}45\%$ de retorno. Como essa razão relaciona a maximização do retorno e a minimização do risco, quanto maior o valor do quociente, melhor é a opção de investimento. Ressalta-se aqui que essa relação retorno/risco vale apenas para 5% dos trimestres, visto que o VaR representa justamente a maior perda que se pode ter em um trimestre com 95% de confiança e que, em média, os retornos dos portfólios são os explicitados na Tabela 2. Apesar dessa restrição, a relação retorno/risco foi explicitada aqui porque auxilia na comparação entre os portfólios.

Considerando apenas o retorno acumulado, o SVR superou os resultados do *benchmark* de mercado 13 vezes, visto que apenas 2 dos 15 portfólios apresentaram valores de retornos acumulados menores, sendo eles, portfólio com *Chi-Square Kernel* e portfólio com *Multiquadratic Kernel.*

Em termos risco, 4 portfólios tiveram um VaR maior que o apresentado pelo *benchmark*, sendo eles: *Linear Kernel, Chi-Squared Kernel, Multiquadratic Kernel* e *Wavelet Kernel*. Portanto, em termos de risco, o SVR superou o *benchmark* 11 vezes.

Na relação retorno/risco, o SVR foi superior ao *benchmark* de mercado também 11 vezes. Os mesmos portfólios citados anteriormente, *Linear, Chi-Square, Wavelet e Multiquadratic,* apresentaram uma relação retorno/risco inferior a 0,45.

Figura 5 ilustra comparação entre o retorno acumulado e o risco de cada opção de investimento. A Figura 6, por sua vez, ilustra o desempenho do retorno acumulado dos portfólios ao longo do período de teste, mais especificamente, de 31/03/2009 a 30/6/2014.

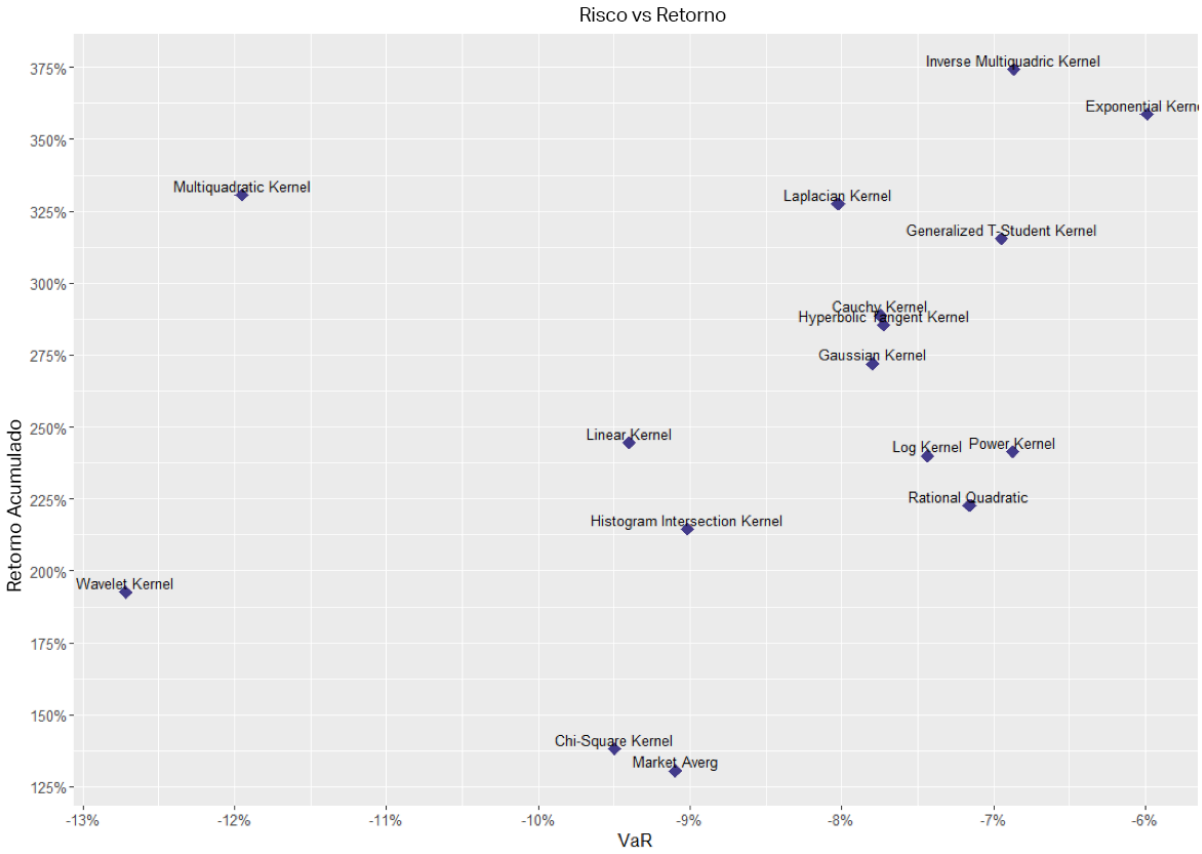

Figura 5: Retorno acumulado e risco dos portfólios.
Fonte: Elaborado pela autora.

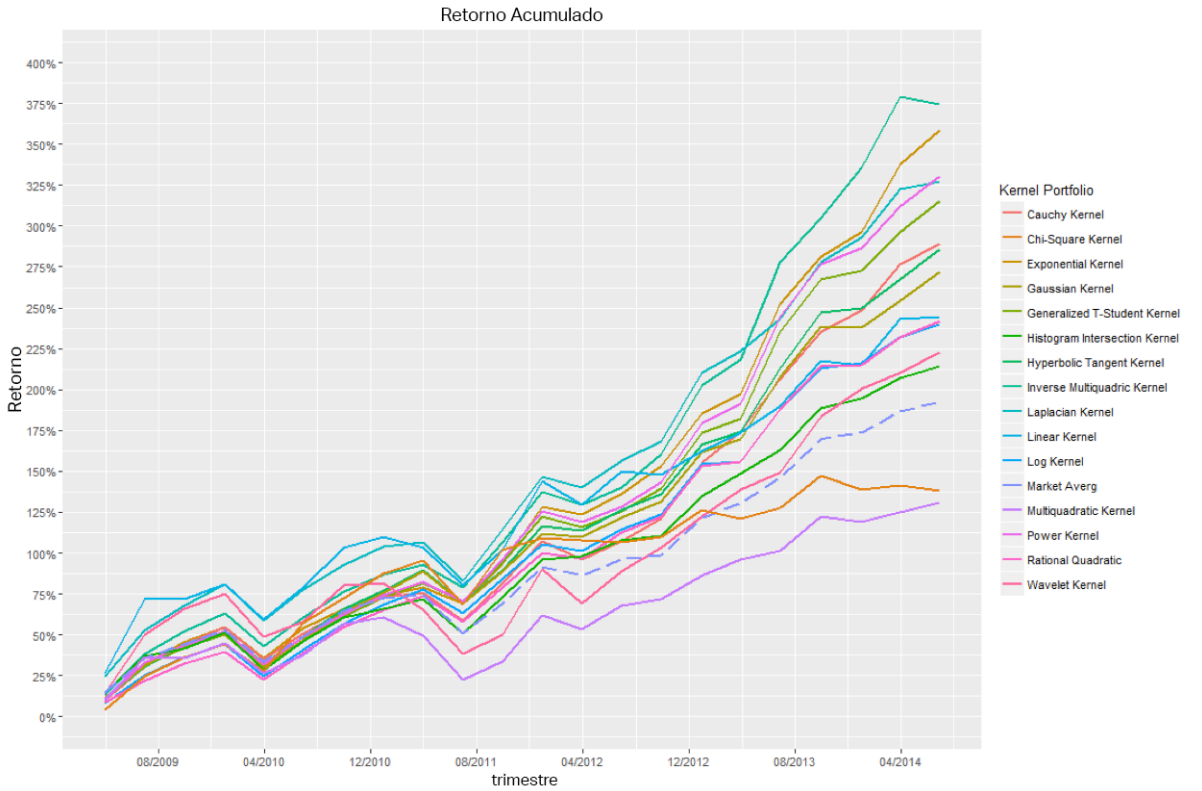

Figura 6: Retorno acumulado dos portfólios no período de teste.
Fonte: Elaborado pela autora.

Na aplicação do SVR, o teste de White (2000) foi utilizado para comparação estatística dos excessos de retornos dos portfólios em relação ao retorno do mercado, visando compreender se existe algum portfólio significativamente superior ao *benchmark* de mercado. O resultado das 100.000 amostragens realizadas no conjunto de teste refutou a hipótese nula de que todos os portfólios formados apresentam retorno acumulado menor ou igual ao retorno acumulado do *benchmark* de mercado, com p-valor de 0,3. Dessa forma, pode-se concluir que pelo menos um dos 15 portfólios formados apresenta, de forma consistente, um desempenho superior ao mercado.

4 MÁQUINAS DE SUPORTE VETORIAL DE CLASSIFICAÇÃO

Para compreensão do contexto no qual as Máquinas de Suporte Vetorial de Classificação utilizadas nesta pesquisa estão inseridas, este capítulo apresenta uma revisão da literatura sobre as aplicações mais significativas do SVM em finanças e sua formulação matemática.

4.1 Referencial Teórico para Máquinas de Suporte Vetorial de Classificação em Finanças

A primeira aplicação direta das Máquinas de Suporte Vetorial em finanças refere-se à aplicação do modelo para classificação de ações e formação de portfólio, abordagem proposta por Fan e Palaniswami (2001). A utilidade do SVM foi testada com informações contábeis das ações negociadas na *Australian Stock Exchange* para o período de 1992 a 2000. O Quadro 3, mostra todos os indicadores calculados pelos autores com os relatórios financeiros e dados de preços disponíveis.

Retorno sobre Capital	**Investimento**
Profit Before Tax / Total Assets	*Price-Earnings Ratio*
Profit Before Tax / Total Capital	*Net Tangible Assets per Share*
Net Income / Total Capital	*Dividend Yield*
Cash Flow / Total Assets	*Earning Yield*
Cash Flow / Total Capital	*Shareholders' Equity / Total Market Value*
Rentabilidade	**Crescimento**
Profit Before Tax / Sales	*Sales Growth*
Profit After Tax / Sales	*Earning Before Tax Growth*
Net Income / Sales	*Earning After Tax Growth*
Cash Flow / Sales	*Net Recurring Profit Growth*
Profit After Tax / Equity	*Operating Profit Growth*
Cash Flow / Total Market Value	*Shareholders' Fund Growth*
Profit After Tax / Cash Flow	*Total Assets Growth*

Alavancagem	**Liquidez a Curto Prazo**
Debt / Equity	*Current Assets / Current Liabilities*
Total Liabilities / Total Capital	*Current Liabilities / Total Assets*
Total Liabilities / Shareholders' Equity	*Current Liabilities / Equity*
Total Assets / Shareholders' Equity	*Long Term Debt / Total Debt*
Total Assets / Total Market Value	**Retorno sobre Investimentos**
Risco	*Return on Assets*
Profit Before Tax / Current Liabilities	
Profit After Tax / Current Liabilities	
Cash Flow / Current Liabilities	

Quadro 3: Indicadores financeiros utilizados por Fan e Palaniswami (2001).
Fonte: Fan e Palaniswami (2001)

Os resultados foram comparados com um modelo de *benchmark* que foi determinado pelos autores como uma carteira de investimentos uniformemente ponderada composta por todas as ações disponíveis para a classificação.

Neste estudo, para reduzir o nível de ruído e manter a consistência, apenas relatórios anuais foram considerados e relatórios com mais de uma variável faltando foram descartados. Posteriormente, pela Análise dos Componentes Principais, Fan e Palaniswami (2001) agruparam os indicadores financeiros similares em oito categorias: Retorno sobre Capital, Lucratividade, Alavancagem, Investimento, Liquidez a Curto Prazo, Retorno sobre Investimento, Risco. Os dados foram convertidos em vetores de oito elementos, cada elemento representando um único componente principal de cada grupo. Como as empresas possuem ciclos de relatórios diferentes, os retornos das ações foram calculados individualmente usando dados de preços para cada 12 meses a partir da data de publicação.

O retorno esperado das ações foi definido como a variável dependente binária, podendo assumir dois valores: $+1$ que representa ações com retorno excepcional e -1 que representa as ações consideradas normais. Dessa forma, as ações que estavam no primeiro quartil empírico da distribuição de retornos das empresas da Bolsa de Valores Australiana foram classificadas como pertencentes à Classe 1, das melhores ações. Já aquelas que apresentaram retorno entre o segundo e quarto quartil empírico, constituíram a Classe 2, classe das piores ações.

Fan e Palaniswami (2001) utilizaram o método de Validação Cruzada com três anos de dados para prever o retorno futuro da ação no ano seguinte. O primeiro e segundo ano de dados foram usados para treinamento, o terceiro ano para validação, e com os dados do quarto ano o modelo foi testado.

Quando o SVM foi usado para selecionar 25% das ações de cada ano, o portfólio igualmente ponderado obteve um retorno total de 208% durante um período de 5 anos, superando o desempenho *benchmark* que gerou um retorno de 71%. Portanto, o SVM mostrou-se bastante útil para seleção de ações e este resultado é corroborado também por outros estudos.

Recentemente, Huerta et al. (2013) desenvolveram um estudo similar ao de Fan e Palaniswami (2001) e ressaltam que escolheram o SVM para identificar ações com alto ou baixo retorno esperado devido a sua simplicidade e eficácia. Dois diferenciais da abordagem são o fato do SVM ter sido aplicado mensalmente para se ajustar às mudanças do mercado e a seleção dos dados que foram usados para treinar o SVM. Não foram utilizados todos os dados disponíveis, mas um conjunto de dados presentes nos quantis mais altos e baixos da distribuição histórica, também chamados de dados de cauda. Segundo os autores, a porcentagem de ações dentro do quantil escolhido é suficiente para que o SVM aprenda as correlações entre as características da ação e a classe à qual ela pertence. Foi determinado um quantil de 5%, então 5% das ações de mais alto retorno e 5% das ações de mais baixo retorno foram escolhidas. Segundo essa abordagem, esses 10% dos dados são suficientes para o treinamento do modelo e a omissão das ações que estão no meio da distribuição alavanca o desempenho, pois é possível treinar o classificador mais rapidamente. Os dados coletados estão compreendidos no período de 1981 a 2010, sendo retirados de uma base de dados comum CRSP/Compustat.

Três filtros foram aplicados para formar a base de dados com ativos negociáveis. O primeiro é uma *proxy* para liquidez (LIQ), o segundo, uma *proxy* para o volume de dólar negociado (VDN) e por último, o preço da ação apenas. Para um determinado ativo, o cálculo da liquidez envolveu regredir os retornos de mercado sobre o volume de dólares levando em consideração o sinal do fluxo de pedidos. Já os retornos diários $r(t)$ foram regredidos nos preços $p(t)$ e volume $v(t)$ de acordo com a seguinte equação:

(1)

$$r(t) = c + \lambda \,\mathrm{sinal}(t)\, \log v(t) p(t)$$

Onde $\mathrm{sinal}(t)$ assume os valores $+1$ se $r(t) \geq 0$, e -1, caso contrário. O coeficiente de regressão λ foi usado como uma *proxy* inversa da liquidez e estimado usando todas as negociações em uma janela de 91 dias. Segundo os autores, o inverso do fator de liquidez quantifica o impacto do volume de dólar negociado nas alterações de preço daquele dia. O VDN foi calculado multiplicando o volume diário pelo preço ação $v(t)p(t)$. Esse filtro elimina as ações que não possuem capacidade suficiente para serem negociadas em fundos mútuos. Os valores diários do VDN foram suavizados por uma média diária exponencial expressa por:

(2)

$$e(t) = \alpha p(t)v(t) + (1 - \alpha)e(t - 1)$$

com $\alpha = 2/(91 + 1)$. Visando simular as condições reais de negociação, os autores aplicaram os filtros todos os dias em que houve negociação antes da abertura das posições. Os limites de corte dos filtros são 50% inferiores para o VDN e preço, e 50% superiores para λ no filtro LIQ.

Se uma ação que pertencia à lista de negociáveis caiu abaixo da marca de corte durante o período de realização da carteira, a ação foi mantida até que as posições fossem fechadas. Se fosse feito o inverso, se introduziria um viés ao longo da carteira por manter na lista de ações negociáveis apenas aquelas que estavam melhores que a média. Resumindo, os autores aplicam os filtros VDN e LIQ para o modelo não aprender correlações de ações que são difíceis de serem negociadas.

Como cada setor da economia possui características únicas, os autores construíram um modelo para cada um dos seguintes setores: Energia, Materiais, Indústria, Consumo de Luxo, *Staples* do Consumidor, Saúde, Finanças, Tecnologia da Informação, Serviços de Telecomunicações e Utilitários. Os setores Serviços de Telecomunicações e Utilitários não apresentaram um número de ações suficientes para se construir o modelo, portanto, foram descartados.

As características técnicas de cada ativo foram calculadas pelo CRSP e as fundamentais foram obtidas do Compustat. A seleção das características foi feita de acordo com a popularidade destas na literatura. O Quadro 4 elenca os indicadores fundamentais utilizados:

Indicador	Fórmula ou Variável
Total Revenue	TR
Gross Profit	GP
Operating Income	OI
Income Before Tax	IBT
Income After Tax	IAT
Net Income Before Extraordinary Items	NIBEI
Net Income	NI
Dividends	D
Diluted Normalized Earnings Per Share	DNEPS
Cash and Equivalents	CE
Short Term Investiments	STI
Accounts Receivable	AR
Total Inventory	TI
Total Current Assets	TCA
Total Assets	TA
Short Term Liabilities	STL
Total Current Liabilities	CL
Total Long Term Debt	TLTD
Total Debt	TD
Total Liabilities	TL
Total Equity	TE
Total Shares	TS
Depreciation	DP
Cash From Operating Activities	CFOA
Capital Expenditures	CEx
Cash From Investing Activities	CFIA
Cash From Financing Activities	CFFA
Net Change In Cash	NCIC
Snapshot Accrual	$\mathrm{SAC} = \mathrm{TCA} - \mathrm{CE} - \mathrm{CL} + \mathrm{TD}$
Accrual Based on Balance Sheet	$\mathrm{ABBS} = \mathrm{SAC}(quarter) - \mathrm{SAC}(quarter - 4)$
Accrual Based on Cash Flow	ABCF
Financial Health	FH

Working Capital	$\mathrm{TCA} - \mathrm{CL}$
Quick Ratio	$(\mathrm{TCA} - \mathrm{TI})/\mathrm{CL}$
Dividend Payout Ratio	$\mathrm{DPR} = \mathrm{D}/\mathrm{NI}$
Book Value	BV
Book Value - Total Debt	$\mathrm{BV} - \mathrm{TD}$
Receivables to Sales	AR/TR
Debt to Assets	TD/TA
Debt to Equity	TD/TE
Cash to Assets	CE/TA
Liabilities to Income	T/NI
Return on Equity	$\mathrm{ROE} = \mathrm{NI}/\mathrm{TE}$
Sales per Share	TR/TS

Quadro 4: Indicadores fundamentais utilizados por Huerta et al. (2013)
Fonte: Huerta et al. (2013)

Os portfólios foram formados com a classificação dos *outputs* do SVM, sendo as ações classificadas nas posições mais altas no *ranking* utilizadas para vendas de longo prazo na carteira, e as ações em posições mais baixas, usadas para vendas de curto prazo. Usando as ações pertencentes aos 25% de maior retorno para posições de longo prazo e as 25% de menor retorno para posições curto prazo, foram formadas carteiras de 10 posições de longo prazo igualmente ponderadas e 10 posições de curto prazo também igualmente ponderadas. O estudo chegou a um retorno anual de 15% com volatilidade próxima a 8% para o portfólio formado.

O estudo Lai et al. (2006) teve como objetivo desenvolver um Algoritmo Genético de Otimização de Dois Estágios para formação de portfólios, sendo os dados oriundos dos preços de fechamento diários de 100 ações selecionadas aleatoriamente da Bolsa de Valores de Shangai, *Shanghai Stock Exchange (SSE),* para o período de Janeiro de 2001 a Dezembro de 2004.

No primeiro estágio, o algoritmo genético é usado como uma ferramenta de classificação das ações tendo como insumos indicadores financeiros das ações listadas. O objetivo desse estágio é permitir que os investidores selecionem apenas ações de boa qualidade. No segundo estágio, a alocação dos ativos de boa qualidade é otimizada usando um algoritmo genético baseado na Teoria de Markowitz (1952). Os autores ressaltam que no primeiro estágio algumas ações

podem ser consideradas de boa qualidade tendo como base apenas o *ranking* de retornos, entretanto, como ressaltado anteriormente, o gerenciamento de um portfólio não deve focar apenas no retorno, mas também na minimização do risco. Sendo assim, a melhor solução para alocação de ativos na carteira é uma quantidade de ações que minimize o risco de um determinado nível de retorno esperado e definido pelo investidor.

Os resultados mostraram que o retorno líquido acumulado do portfólio igualmente ponderado foi pior do que o portfólio otimizado pelo algoritmo genético. Isso implica que se um investidor sem experiência escolhe aleatoriamente ações para seu portfólio, o retorno esperado da carteira vai ser aproximadamente igual ao seu valor. Mesmo que o investidor não perca dinheiro com essa seleção de ações, devido ao custo de capital ele já perde algum recurso. Os resultados do estudo de Lai et al. (2006) também mostram que o desempenho do portfólio diminui à medida que o número de ações aumenta. Segundo os autores, um maior número de ações gera maior flexibilidade para composições de minimização de risco, mas selecionar ações de boa qualidade é um pré-requisito para se obter um bom portfólio. Ações de má qualidade, mesmo se incluídas em composições que minimizem o risco, podem influenciar negativamente o desempenho do portfólio.

Os resultados do estudo também mostraram que se um investidor selecionar apenas ações de boa qualidade, um portfólio mais volumoso não necessariamente supera um portfólio com poucas ações. Dessa forma, segundo Lai et al. (2006) é sensato que os investidores selecionem um número limite de ações e que todas sejam de boa qualidade.

O estudo de Emir et al. (2012) teve como objetivo construir um modelo financeiro ótimo que permitisse a classificação das melhores ações do Mercado Turco. Para este propósito, anualmente, as ações que apresentaram os 10 retornos mais altos foram classificadas como “1” e as demais, classificadas como “0”. Segundo os autores, a aplicação de redução de dimensionalidade dos dados antes do processamento destes para classificação, melhora o resultado final.

Os dados foram coletados para cada ação que compunha o Índice *Istanbul Stock Exchange (ISE)* no período de 2002 a 2010 e este estudo foi inovador ao utilizar tanto parâmetros técnicos como fundamentalistas para a análise. Os dados técnicos foram 13 indicadores do Índice *Istanbul Stock Exchange (ISE)* listados no Quadro 5.

Atributo	Fórmula ou Variável
Growth in Assets	GA
Growth in Net Profit	GNP
Equity Growth	EG
Current Assets / Assets	$\mathrm{CA/A}$
Fixed Assets / Assets	$\mathrm{FA/TA}$
Equity / Assets	$\mathrm{E/TA}$
Equity / Tangible Assets	$\mathrm{E/TGA}$
Return on Assets	$\mathrm{ROA} = \mathrm{NI/TA}$
Net Profit / Current Assets	$\mathrm{NP/CA}$
Return on Equity	$\mathrm{ROE} = \mathrm{NI/TE}$
Earnings per Share	$\mathrm{NI} - \mathrm{DPS/AOS}$
Price-Earnings Ratio	$\mathrm{MVPS/EPS}$
Market to Book Value	$\mathrm{MV/BV}$

Quadro 5: Indicadores técnicos utilizados por Emir et al. (2012).
Fonte: Emir et al. (2012).

Já a análise fundamentalista foi feita com 14 indicadores considerados essenciais para representar as empresas do ISE como um todo, apesar de pertencerem a diferentes setores. Estes estão elencados no Quadro 6.

Atributo	Fórmula ou Variável
Mass Index (MASS)	$\sum_{1}^{25} \frac{9-\text{day EMA of (High} - \text{Low)}}{9-\text{day EMA of a } 9-\text{day EMA of (High} - \text{Low)}}$
Average True Range (ATR)	$\mathrm{ATR}_t = \frac{\mathrm{ATR}_{t-1} \times (n-1) + \mathrm{TR}_t}{n}$ O primeiro ATR é calculado usando a seguinte média aritmética: $\mathrm{ATR}_t = \frac{1}{n}\sum_{i=1}^{n} TR_i$
Momentum (Mo)	$C_t - C_{t-4}$
Chaikin Money Flow Indicator (CMF)	$CMF = \frac{\sum_{t=20}^{t} CLV_t \times volume_t}{\sum_{t=20}^{t}(vol_t)}$ onde, $CLV = \frac{(close_1 - low_1) - (high_1 - close_1)}{(high_1 - low_1)}$
Commodity Channel Index (CCI)	$\frac{(M_t - SM_t)}{(0.015\ D_t)}$ onde: $M_t = (\frac{H_t + L_t + C_t}{3})$ $SM_t = \frac{\sum_{i=1}^{n} M_{t-i+1}}{n}$

Moving Average Convergence-Divergence Trading Method (MACD)	$MCD = 2(DIF - DEA)$ $DIF = EMA(12) - EMA(26)$ $DEA_t = \frac{2}{10} dif + \frac{8}{10} DEA_{t-1}$ $EMA_t(n) = \frac{2}{N+1} C_t + \frac{N-1}{N} + 1MA_{t-1}(n)$
Exponential Moving Average (EMA)	$EMA_{todau} = EMA_{yesterday} + \alpha(price_{todau} - EMA_{yesterday})$
Relative Strength Index (RSI)	$100 - \frac{100}{1 + (\sum_{i=0}^{n-1} Up_{t-i}/n) \,/\, (\sum_{i=0}^{n-1} Dw_{t-i}/n)}$
Money Flow Index (MFI)	$100 - \frac{100}{(1 + \text{Money Flow Ratio})}$
Stochastics %K	$\frac{C_t - LL_{t-n}}{HH_{t-n} - LL_{t-n}} \times 100$
Triple Exponential Smoothing of the Log of Closing Price (TRIX)	$TripleEMA_0 = (1-f)^3(p_0 + 3fp_1 + 6f^2p_2 + 10f^3p_3 + \cdots)$
William's %R	$\frac{H_n - C_t}{H_n - L_n} \times 100$

Quadro 6: Indicadores fundamentalistas utilizados por Emir et al. (2012).
Fonte: Elaborado pela autora.

Para fins de comparação, um modelo de Rede Neural foi aplicado nas mesmas circunstâncias e os resultados mostraram que as Máquinas de Suporte Vetorial apresentaram desempenho superior na acurácia da previsão. Portanto, os resultados empíricos do estudo de Emir et al. (2012) também corroboram para o sucesso do SVM como modelo para previsão em séries temporais financeiras.

No mesmo âmbito de abordagens para construção de portfólios, Gupta et al. (2012) desenvolveram uma abordagem híbrida para facilitar as tomadas de decisão dos investidores. Primeiramente, utilizaram as Máquinas de Suporte Vetorial para classificar as ações em três classes pré-definidas de acordo com o desempenho delas em três indicadores financeiros: liquidez, retorno e risco.

Como retorno do portfólio considerou-se o retorno a curto prazo, equivalente ao desempenho médio dos ativos no período de 12 meses, e o retorno a longo prazo, também equivalente ao desempenho médio dos ativos, mas para um período de 36 meses. O risco da carteira foi definido como o desvio semi-absoluto de rentabilidade do portfólio abaixo do retorno esperado. Já a liquidez, foi considerada

como a probabilidade de conversão de um investimento em dinheiro, sem qualquer perda significativa de valor e medida através da taxa de *turnover*.

Ativos da Classe 1 foram classificados como ativos líquidos, já que o indicador de liquidez foi o mais alto nesta classe. Ativos da Classe 2 foram classificados como de alto rendimento, uma vez que apresentaram altos retornos. Ativos da Classe 3 foram classificados como ativos de menor risco, visto que em comparação com as demais classes, esses ativos apresentaram o menor desvio padrão, mas retorno e liquidez médios.

A base de dados foi composta por 150 ativos listados no *National Stock Exchange (NSE*), o principal mercado de ativos financeiros da Índia. O conjunto de treinamento foi composto por 60% do total dos dados e o conjunto de teste por 40%. O segundo passo do estudo foi a aplicação de um algoritmo genético, mais especificamente, *Real Coded Genetic Algorithm* (RCGA), em cada uma das três classes para formação de portfólios ótimos. O portfólio formado a partir das ações da Classe 1 apresentou maior liquidez, mas um nível de risco médio. O portfólio formado a partir da Classe 2 apresentou maior nível de retorno e maior nível de risco. Já o portfólio da Classe 3 apresentou o menor nível de risco comparado aos demais portfólios, e como esperado, um nível de retorno médio. Sendo assim, os autores concluem que investidores à procura de maior liquidez deveriam investir em ativos da Classe 1. Já os investidores à procura de maiores retornos deveriam optar pela Classe 2 e àqueles à procura de investimentos mais seguros deveriam investir em ativos da Classe 3. Estes resultados indicam que a abordagem desenvolvida é capaz de classificar os ativos com boa acurácia e ainda mais, é capaz gerar portfólios otimizados para cada classe de ativos de acordo com as preferências dos consumidores.

As Máquinas de Suporte Vetorial também podem ser aplicadas na previsão da direção do mercado. Kim (2003) analisou a aplicabilidade das Máquinas de Suporte Vetorial na previsão da direção das alterações diárias nos preços das ações em comparação com dois modelos: BPN (*Back-Propagation Neural Network)* e CBR (*Case-based reasoning*). Os dados utilizados foram as observações diárias dos preços das ações que compõem o Índice de Mercado da Coreia (KOSPI) e 12 indicadores técnicos para período de Janeiro de 1989 a Dezembro de 1998. Os indicadores selecionados como *inputs* estão descritos no Quadro 7.

Nome do Atributo	Fórmula
Stochastics %K	$\frac{C_t - LL_{t-n}}{HH_{t-n} - LL_{t-n}} \times 100$
%D (3-period moving average of %K)	$\frac{\sum_{i=0}^{\%n-1} \%K_{t-i}}{n}$
Slow %D	$\frac{\sum_{i=0}^{\%n-1} \%D_{t-i}}{n}$
Momentum	$C_t - C_{t-4}$
Price Rate of Change (ROC)	$\frac{C_t}{C_{t-n}} \times 100$
William's %R	$\frac{H_n - C_t}{H_n - L_n} \times 100$
A/D Oscilador	$\frac{H_t - C_{t-1}}{H_t - L_t}$
Disparity5	$\frac{C_t}{MA_5} \times 100$
Disparity10	$\frac{C_t}{MA_{10}} \times 100$
Price Oscilator (OSCP)	$\frac{M_5 - M_{10}}{MA_5}$
Commodity Channel Index (CCI)	$\frac{(M_t - SM_t)}{(0.015\ D_t)}$ onde: $M_t = (\frac{H_t + L_t + C_t}{3})$ $SM_t = \frac{\sum_{i=1}^{n} M_{t-i+1}}{n}$ $D_t = \frac{\sum_{i=1}^{n} \lvert M_{t-i+1} - SM_t \rvert}{n}$
Relative Strength Index (RSI)	$100 - \frac{100}{1 + (\sum_{i=0}^{n-1} Up_{t-i}/n)\ /\ (\sum_{i=0}^{n-1} Dw_{t-i}/n)}$

Quadro 7: Atributos selecionados por Kim (2003)
Fonte: Kim (2003)

O autor classificou as alterações diárias dos preços em duas classes: "0" ou "1". A primeira classe foi composta por ações cujo preço do dia posterior foi menor que do dia anterior. Já a segunda classe foi composta por ações cujo índice no dia posterior foi mais alto se comparado ao dia anterior. 80% dos dados foram usados para treinamento e estimação dos parâmetros e os 20% restantes foram utilizados para validação do modelo. Para desenvolver os experimentos, o software LIBSVM foi utilizado.

O desempenho na predição das séries temporais P, foi avaliada usando a seguinte equação:

(31)

$$\mathrm{P} = \frac{1}{m}\sum_{i=1}^{m} R_i \ (i = 1,2, \dots, m)$$

onde R_i é o resultado da predição para o *i*-ésimo dia de negociação e é definido por:

(32)

$$R_i = \begin{cases} 1 \ se \ PO_i = AO_i \\ 0 \ caso \ contrário, \end{cases}$$

PO_i é o valor previsto do *output* para o *i*-ésimo dia de negociação, AO_i é o *output* atual para o *i*-ésimo dia de negociação e *m* é a quantidade de exemplos de teste.

Os resultados empíricos mostram que no conjunto de validação, o SVM obteve um desempenho na previsão de 57,83% contra 54,73% e 51,97% dos modelos BPN e CBR, respectivamente. Fica evidente então que as Máquinas de Suporte Vetorial superaram os dois modelos no nível de acurácia da previsão e isso pode ser atribuído ao fato de que o SVM implementa o Princípio da Minimização do Risco Estrutural, permitindo uma melhor generalização. Este estudo concluiu que o SVM é uma alternativa promissora em previsão de séries temporais financeiras.

Zhang e Zhao (2010) por sua vez, aplicaram o SVM no mercado de câmbio para prever mudanças nas taxas de câmbio euro/dólar. Neste estudo, os *inputs* para o modelo foram indicadores técnicos, sendo os dados oriundos do sistema *Bloomberg* no intervalo de 10 de julho de 2007 a 9 de julho de 2009. Os indicadores utilizados assim como suas fórmulas estão representados no Quadro 8.

Indicador	**Fórmula**
Moving Average Line	$MA_t(n) = \frac{1}{N}C_t + \frac{N-1}{N}MA_{t-1}(n)$
Moving Average Convergence and Divergence Line	$MCD = 2(DIF - DEA)$ $DIF = EMA(12) - EMA(26)$ $DEA_t = \frac{2}{10}dif + \frac{8}{10}DEA_{t-1}$ $EMA_t(n) = \frac{2}{N+1}C_t + \frac{N-1}{N} + 1MA_{t-1}(n)$
Random Index	$K_t = \frac{2}{3}K_{t-1} + \frac{1}{3}RSV_t$

	$D_t = \frac{2}{3} D_{t-1} + \frac{1}{3} K_t$ $J = 3D - 2K$ $RSV_t = \frac{C_t - L_n}{H_n - L_t} \times 100\%$
Relative Strength Index	$RSI(n) = \frac{A}{A + B} \times 100\%$
BIAS	$BIAS(n) = \frac{C_t - MA(n)}{MA(n)} \times 100\%$

Quadro 8: Técnicos selecionados por Zhang e Zhao (2010)
Fonte: Zhang e Zhao (2010)

Os autores ressaltam que as mudanças na taxa de câmbio dependem dos ajustes políticos do governo, portanto, os preços das ações estão intimamente relacionados aos resultados de suas análises e, genericamente, pode-se dizer que a mudança de uma taxa de câmbio está mais próxima de um processo estocástico. Para os operadores do Mercado determinarem se o preço de uma taxa de câmbio subirá ou cairá, eles precisam de muitas experiências e um vasto conhecimento sobre indicadores e sobre seu comportamento recente para somente depois tomarem uma decisão. As Máquinas de Suporte Vetorial por sua vez, precisam estudar dados históricos das taxas de câmbio e estabelecer um modelo de classificação, sendo este um processo bem menos oneroso.

Segundo os autores, análises de indicadores técnicos de câmbio podem ser descritos como um problema de aprendizagem geral. Primeiramente é preciso reconhecer se a análise técnica é válida, isto é, se os indicadores técnicos e a tendência da taxa de câmbio têm alguma ligação intrínseca. Se houver esse tipo de relacionamento, a chave do problema é achar uma função que minimize o Risco Esperado e seja aplicável em uma grande amostra. Nesse contexto, os *inputs* são os indicadores técnicos e os *outputs,* que indicam a mudança no preço futuro, derivam do relacionamento entre os indicadores e a tendência da taxa de câmbio. Entretanto, a escolha dos indicadores não é uma tarefa fácil.

Como na maioria dos estudos que utilizam Máquinas de Suporte Vetorial, Zhang e Zhao (2010) classificam o *output* do modelo em duas classes: Classe 1, composta pelas observações em que houve aumento no preço, isto é, $y_i = +1$, e Classe 2, formada pelas observações em que houve queda no preço, ou seja, $y_i = -1$. Os dias em que não houve variação no preço foram ignorados. Os resultados empíricos mostraram que a precisão da previsibilidade do SVM é maior que 60%.

Sendo assim, Zhang e Zhao (2010) chegaram à conclusão de que com o SVM, é possível fazer previsões independente da complexidade do Mercado Financeiro.

4.2 Metodologia Máquinas de Suporte Vetorial de Classificação

Em 1992, Boser et al. (1992) descreveram um algoritmo que automaticamente regula a capacidade de classificação da função por meio da maximização da margem entre os dados do conjunto de treinamento e o limite da classe. As Máquinas de Suporte Vetorial são originárias deste estudo de Boser et al. (1992) e o seu maior diferencial é justamente a construção de um hiperplano que separa os dados em duas classes ou mais, para atingir a separação máxima entre elas. A Figura 7, mostra os diversos hiperplanos possíveis para a separação dos dados.

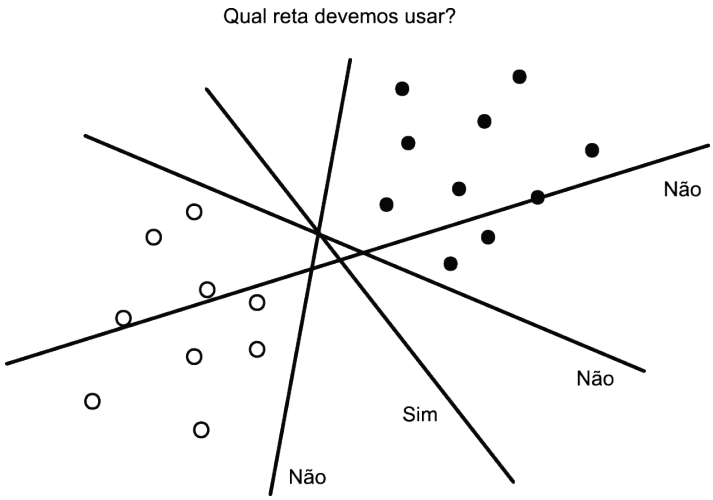

Figura 7: Escolhendo o melhor hiperplano para classificação.
Fonte: Traduzido de Soman et al. (2009)

De acordo Fan e Palaniswami (2001), a aplicação do Princípio de Minimização do Erro Empírico aplicado em métodos bastante difundidos como Redes Neurais, não garante um menor erro real. O SVM resolve essa questão com a implementação do Princípio da Minimização do Risco Estrutural, o qual procura minimizar o limite superior do erro de generalização, em vez de minimizar apenas o erro do processo de estimação. Isso significa que na classificação de novas observações de classes desconhecidas, a chance de haver um erro na predição, baseado na aprendizagem do classificador, será mínima. Além disso, análises da

Teoria de Aprendizado Computacional provam que a maximização da margem é um modo efetivo de minimizar o risco de *overfitting* (Vapnik e Vapnik, 1998).

Sendo assim, a aprendizagem do SVM pode ser entendida como a descoberta do hiperplano central que maximiza a margem de forma que as observações da Classe 1 $(y_i = +1)$ fiquem o mais separadas possível das observações da Classe 2 $(y_i = -1)$. A Figura 8 ilustra o conceito de máxima margem para um conjunto de dados que podem ser separados de maneira linear e direta.

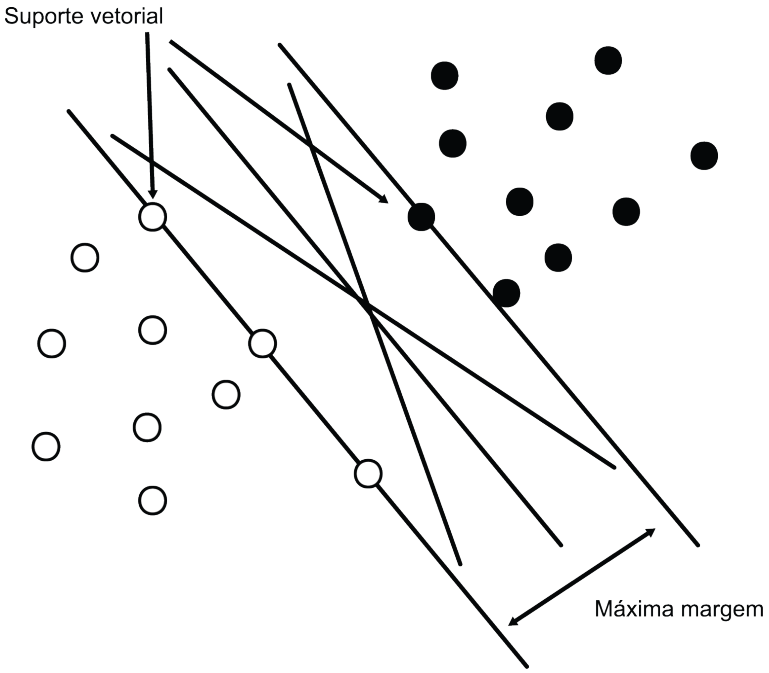

Figura 8: Classificador de máxima margem.
Fonte: Traduzido de Soman et al. (2009)

Os pontos localizados em cima das retas paralelas ao classificador são chamados de Suportes Vetoriais. Estes pontos possuem um papel crucial na teoria, justificando o nome do método, sendo que o termo “Máquinas” refere-se ao algoritmo.

4.2.1 Formulação do SVM Linear

O modelo clássico de Máquinas de Suporte Vetorial é o modelo de classificação linear dicotômica, que tem como objetivo encontrar uma função de decisão com a seguinte forma:

(33)

$$f(x) = sinal(\boldsymbol{w^T x} - \gamma)$$

Onde $\boldsymbol{x}$ é um vetor de dimensão *p* x *1* representando o vetor de uma observação arbitrária com *p* variáveis, $\boldsymbol{w}$ é o vetor de parâmetros de dimensão *p* × 1 e γ é um parâmetro escalar, denominado termo de viés.

A formulação do problema de separação linear tem como insumo para a estimação, uma matriz $\mathbf{A}$ de dimensão *n* x *p*, onde cada linha representa uma observação de uma população e cada coluna, uma característica, isto é, uma variável dessa população. Além disso, para fins de estimação deve-se considerar também um vetor $\boldsymbol{y}$ que representa o grupo no qual cada observação se encontra, sendo ele de dimensão *n* x 1 e contendo somente os valores $+1$ ou -1.

Considere o conjunto de dados abaixo:

(34)

$$\begin{bmatrix} \boldsymbol{i} & \boldsymbol{x_1} & \boldsymbol{x_2} & y_i \\ 1 & 1 & 1 & -1 \\ 2 & 2 & 1 & -1 \\ 3 & 1 & 2 & -1 \\ 4 & 2 & 2 & -1 \\ 5 & 4 & 4 & +1 \\ 6 & 4 & 5 & +1 \\ 7 & 5 & 4 & +1 \\ 8 & 5 & 5 & +1 \end{bmatrix}$$

$\boldsymbol{x_1}$ e $\boldsymbol{x_2}$ representam duas variáveis do conjunto de dados, i representa as observações e y_i a classe à qual cada observação pertence. A plotagem desses dados é dada pela Figura 9.

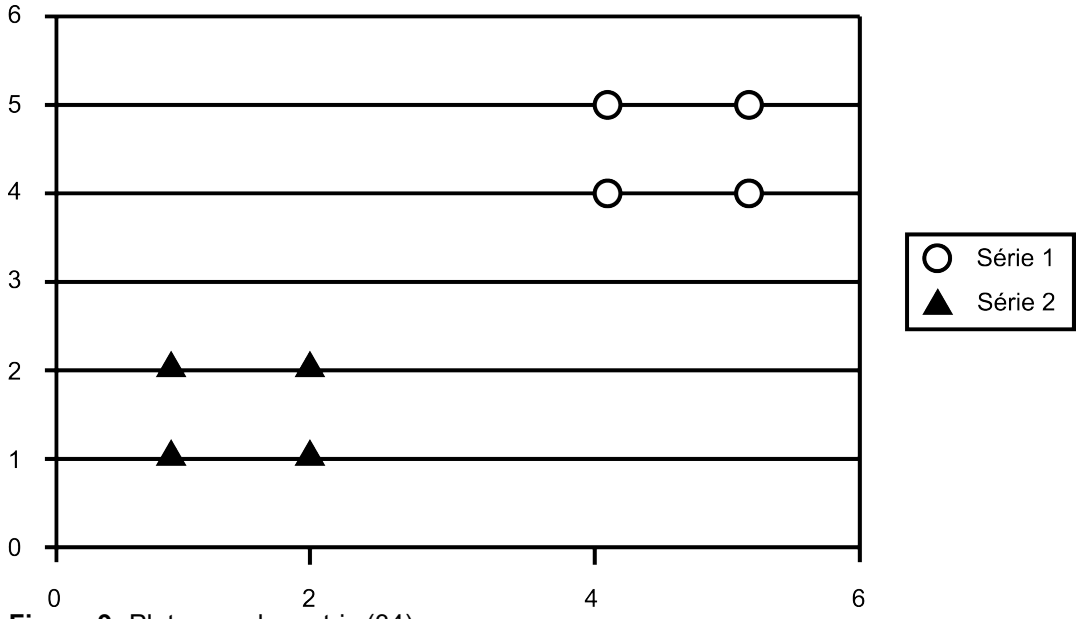

Figura 9: Plotagem da matriz (34)
Fonte: Traduzido de Soman et al. (2009)

Aplicando o SVM a esse conjunto de dados, o objetivo é achar o hiperplano de máxima margem, da forma $w_1x_1 + w_2x_2 - \gamma = 0$, e dois outros planos que limitarão cada uma das classes, assumindo a forma $w_1x_1 + w_2x_2 - \gamma \geq +1$ e $w_1x_1 + w_2x_2 - \gamma \leq -1$. Em outras palavras, o SVM visa encontrar dois planos tal que os pontos com $d = -1$ satisfaçam a restrição $w_1x_1 + w_2x_2 - \gamma \leq -1$ e os pontos com $d = +1$ satisfaçam $w_1x_1 + w_2x_2 - \gamma \geq +1$.

As restrições podem ser escritas da seguinte forma:

(35)

$$1w_1 + 1w_2 - \gamma \leq -1$$
$$2w_1 + 1w_2 - \gamma \leq -1$$
$$1w_1 + 2w_2 - \gamma \leq -1$$
$$2w_1 + 2w_2 - \gamma \leq -1$$
$$4w_1 + 4w_2 - \gamma \geq +1$$
$$4w_1 + 5w_2 - \gamma \geq +1$$
$$5w_1 + 4w_2 - \gamma \geq +1$$
$$5w_1 + 5w_2 - \gamma \geq +1$$

Isso equivale à:

(36)

$$(-1)\ (1w_1 + 1w_2 - \gamma) \geq +1$$
$$(-1)\ (2w_1 + 1w_2 - \gamma) \geq +1$$

$$(-1)\,(1w_1 + 2w_2 - \gamma) \geq +1$$
$$(-1)\,(2w_1 + 2w_2 - \gamma) \geq +1$$
$$(+1)\,(4w_1 + 4w_2 - \gamma) \geq +1$$
$$(+1)\,(4w_1 + 5w_2 - \gamma) \geq +1$$
$$(+1)\,(5w_1 + 4w_2 - \gamma) \geq +1$$
$$(+1)\,(5w_1 + 5w_2 - \gamma) \geq +1$$

A forma matricial resultante é:

(37)

$$\begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 1 & 2 \\ 2 & 2 \\ 4 & 4 \\ 4 & 5 \\ 5 & 4 \\ 5 & 5 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} - \gamma \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \geq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Ou:

(38)

$$\mathbf{D}(\mathbf{A}\boldsymbol{w} - \gamma\mathbf{1}) \geq 1$$

Onde,

(39)

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 1 & 2 \\ 2 & 2 \\ 4 & 4 \\ 4 & 5 \\ 5 & 4 \\ 5 & 5 \end{bmatrix} = \begin{bmatrix} \boldsymbol{x_1^T} \\ \boldsymbol{x_2^T} \\ \boldsymbol{x_3^T} \\ \boldsymbol{x_4^T} \\ \boldsymbol{x_5^T} \\ \boldsymbol{x_6^T} \\ \boldsymbol{x_7^T} \\ \boldsymbol{x_8^T} \end{bmatrix};\ \mathbf{D} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

Sabe-se que a distância do hiperplano $w_1x_1 + w_2x_2 - \gamma = +1$ até a origem é $\frac{|-\gamma-1|}{\sqrt{w_1^2+w_2^2}}$ e que a distância até o hiperplano $w_1x_1 + w_2x_2 - \gamma = -1$ é $\frac{|-\gamma+1|}{\sqrt{w_1^2+w_2^2}}$. Portanto, a distância entre esses dois planos é $\frac{2}{\sqrt{w_1^2+w_2^2}}$. O objetivo então é achar o

w e o γ que maximizem a distância e ao mesmo tempo satisfaçam a matriz de restrições.

Maximizar a margem $\frac{2}{\sqrt{w_1^2+w_2^2}}$ equivale a minimizar o seu recíproco $\frac{w_1^2+w_2^2}{2} = \frac{1}{2}\boldsymbol{w^T w}$, então o problema de programação quadrática pode ser escrito de duas formas:

(40)

Maximize: $\varsigma = \frac{2}{||\boldsymbol{w}||}$

Sujeito à

$\mathbf{D}(\mathbf{A}\boldsymbol{w} - \gamma\mathbf{1}) \geq 1$

Para $\boldsymbol{w} \in \mathbb{R}^p$, $\gamma \in \mathbb{R}$.

Ou,

(41)

Minimize : $\varsigma^* = \frac{1}{2}\ \boldsymbol{w^T w}$

Sujeito à

$\mathbf{D}(\mathbf{A}\boldsymbol{w} - \gamma\mathbf{1}) \geq 1$

Para $\boldsymbol{w} \in \mathbb{R}^p$, $\gamma \in \mathbb{R}$.

Sintetizando, o treinamento do SVM consiste em achar $\boldsymbol{w} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$ e γ dada uma matriz de dados $\mathbf{A}$ e o vetor de classes $\boldsymbol{y}$. Como apresentado anteriormente, a função de decisão do SVM linear é dada por:

(42)

$$f(x) = sinal(\boldsymbol{w^T x} - \gamma)$$

Isso significa que para uma nova observação, o sinal de $\boldsymbol{w^T x} - \gamma$ a classificará na Classe 1 ou na Classe 2.

O problema primal do SVM também pode ser escrito na sua forma dual, dada pelo Dual de Wolfe (1961):

(43)

$$Max_{\lambda \geq 0}\left[Min_{w,\gamma} \mathrm{L}(\mathrm{w}, \gamma, \lambda)\right]$$

Para determinado λ devemos encontrar w e γ que minimizem $\mathrm{L}(\mathrm{w},\gamma,\lambda)$ e depois substituir o resultado na função Lagrangeana para maximizá-la em função de λ. Note que $\mathrm{L}(\mathrm{w},\gamma,\lambda)$ é um escalar e a função Lagrangeana é dada por:

(44)

$$\mathrm{L}(w,\gamma,\lambda) = \frac{1}{2}\boldsymbol{w^T w} - \boldsymbol{\lambda^T}\{\boldsymbol{D}[(\boldsymbol{Aw} - \gamma\mathbf{1}) - \mathbf{1}]\}$$

Resolvendo as condições de primeira ordem, se tem:

(45)

$$\frac{\partial}{\partial\gamma}\mathrm{L}(\mathrm{w},\gamma,\lambda) = 0 \rightarrow \frac{\mathrm{d}}{\mathrm{d}\gamma}\left(\frac{1}{2}\boldsymbol{w^T w} - \boldsymbol{\lambda^T}\mathbf{D}\mathbf{A}\boldsymbol{w} + \boldsymbol{\lambda^T}\mathbf{D}\gamma\mathbf{1} - \boldsymbol{\lambda^T}\mathbf{D}\mathbf{1}\right) = 0 \rightarrow \boldsymbol{\lambda^T}\mathbf{D}\mathbf{1} = 0$$

(46)

$$\frac{\partial}{\partial w}\mathrm{L}(\mathrm{w},\gamma,\lambda) = 0 \rightarrow \frac{\mathrm{d}}{\mathrm{d}w}\left(\frac{1}{2}\boldsymbol{w^T w} - \boldsymbol{\lambda^T}\mathbf{D}\mathbf{A}\boldsymbol{w} + \boldsymbol{\lambda^T}\mathbf{D}\gamma 1 - \boldsymbol{\lambda^T}\mathbf{D}\mathbf{1}\right) = 0 \rightarrow$$

$$\rightarrow \frac{1}{2}2\boldsymbol{w} - \boldsymbol{\lambda^T}\mathbf{D}\mathbf{A}\boldsymbol{w} = 0 \rightarrow \boldsymbol{w^T} - \boldsymbol{\lambda^T}\mathbf{D}\mathbf{A} = 0$$

Isso significa que $\boldsymbol{w^T} = \boldsymbol{\lambda^T}\mathbf{D}\mathbf{A}$ **e** $\boldsymbol{w} = \mathbf{A^T}\mathbf{D}\boldsymbol{\lambda}$**.** Substituindo os resultados na função Lagrangeana, o problema Dual é:

(47)

$$\mathrm{L}(\lambda) = -\frac{1}{2}\boldsymbol{\lambda^T}\mathbf{D}\mathbf{A}\mathbf{A^T}\mathbf{D}\boldsymbol{\lambda} + \boldsymbol{\lambda^T}\mathbf{1}$$

Sujeito à

$$\mathbf{1^T}\mathbf{D}\boldsymbol{\lambda} = 0$$

$$\lambda \geq 0$$

Para definição algébrica do problema, considere:

(48)

$$\boldsymbol{\lambda^T}\mathbf{D}\mathbf{A}\mathbf{A^T}\mathbf{D}\boldsymbol{\lambda} = (\lambda_1 \dots \lambda_n)\begin{pmatrix} y_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & y_n \end{pmatrix}\begin{pmatrix} \boldsymbol{x_1^T x_1} & \cdots & \boldsymbol{x_1^T x_n} \\ \vdots & \ddots & \vdots \\ \boldsymbol{x_n^T x_1} & \cdots & \boldsymbol{x_n^T x_n} \end{pmatrix}\begin{pmatrix} y_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & y_n \end{pmatrix}\begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_n \end{pmatrix}$$

$$= (\lambda_1 y_1 \dots \lambda_n y_n)\begin{pmatrix} \boldsymbol{x_1^T x_1} & \cdots & \boldsymbol{x_1^T x_n} \\ \vdots & \ddots & \vdots \\ \boldsymbol{x_n^T x_1} & \cdots & \boldsymbol{x_n^T x_n} \end{pmatrix}\begin{pmatrix} \lambda_1 y_1 \\ \vdots \\ \lambda_n y \end{pmatrix}$$

$$= (\lambda_1 y_1 \boldsymbol{x_1^T x_1} + \lambda_2 y \boldsymbol{x_2^T x_1} + \lambda_3 y_3 \boldsymbol{x_3^T x_1} + \cdots + \lambda_n y_n \boldsymbol{x_n^T x_1} + \cdots + \lambda_1 y_1 \boldsymbol{x_1^T x_n} + \cdots + \lambda_n y_n \boldsymbol{x_n^T x_n})\begin{pmatrix} \lambda_1 y_1 \\ \vdots \\ \lambda_n y_n \end{pmatrix}$$

$$= (\sum_{i=1}^{n} \lambda_i y_i \boldsymbol{x_i^T x_1}, \dots, \sum_{i=1}^{n} \lambda_i y_i \boldsymbol{x_i^T x_n})\begin{pmatrix} \lambda_1 y_1 \\ \vdots \\ \lambda_n y_n \end{pmatrix}$$

$$= (\sum_{i=1}^{n} \lambda_i y_i \boldsymbol{x_i^T x_1})\lambda_1 y_1 + \cdots + (\sum_{i=1}^{n} \lambda_i y_i \boldsymbol{x_i^T x_n})\lambda_n y_n$$

$$= \sum_{j=1}^{n}\sum_{i=1}^{n} \lambda_i y_i \boldsymbol{x_i^T x_j} \lambda_j y_j$$

Então,

(49)

$$\mathrm{L}(\mathrm{w}, \gamma, \lambda) = \frac{1}{2}\boldsymbol{w^T w} - \sum_{i=1}^{n} \lambda_i [y_i(\boldsymbol{w^T} x_i - \gamma) - 1]$$

$$\mathrm{L}(\mathrm{w}, \gamma, \lambda) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j \left(\boldsymbol{x_i^T x_j}\right) - \sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \lambda_j y_i y \left(\boldsymbol{x_i^T x_j}\right) + \sum_{i=j}^{n} \lambda_i$$

$$\mathrm{L}(\mathrm{w}, \gamma, \lambda) = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j \left(\boldsymbol{x_i^T x_j}\right) + \sum_{i=j}^{n} \lambda_i$$

Assim, o Lagrangeano em função somente de λ é:

(50)

$$\mathrm{Max}\ \mathrm{L}(\lambda) = \sum_{i=j}^{n} \lambda_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j \left(\boldsymbol{x_i^T x_j}\right)$$

Sujeito à

$$\sum_{i=j}^{n} \lambda_i\, y_i = 0$$

$$\lambda \geq 0, i = 1,2, \dots, n.$$

4.2.2 L1-SVM com Margem Suave: Kernel Linear

Agora, considere um conjunto de dados que não são totalmente separáveis por um hiperplano. A solução neste caso é achar o hiperplano que “melhor” separa esses dados, ou seja, aquele de máxima margem e que permite que apenas um pequeno número de pontos caia na classe errada. O desvio desses pontos em relação à sua classe original é denominado “erro”. Desejamos, portanto, encontrar o hiperplano com o mínimo de pontos que contribuem para o erro. A condição de máxima margem e a de minimização do número de pontos que contribuem para o erro são contraditórias, pois uma margem maior gerará mais pontos com erros. Por isso, o parâmetro C é introduzido e ele representa o custo do erro, isto é, o peso de se fazer uma classificação errada.

A Figura 10 ilustra o seguinte conjunto de dados:

(51)

$$\begin{bmatrix} \boldsymbol{y} & \boldsymbol{x_1} & \boldsymbol{x_2} \\ +1 & 1.0 & 0.8 \\ +1 & 3.0 & 2.5 \\ +1 & 2.5 & 1.0 \\ -1 & 1.0 & 1.8 \\ -1 & 3.0 & 4.5 \\ -1 & 2.5 & 2.8 \end{bmatrix}$$

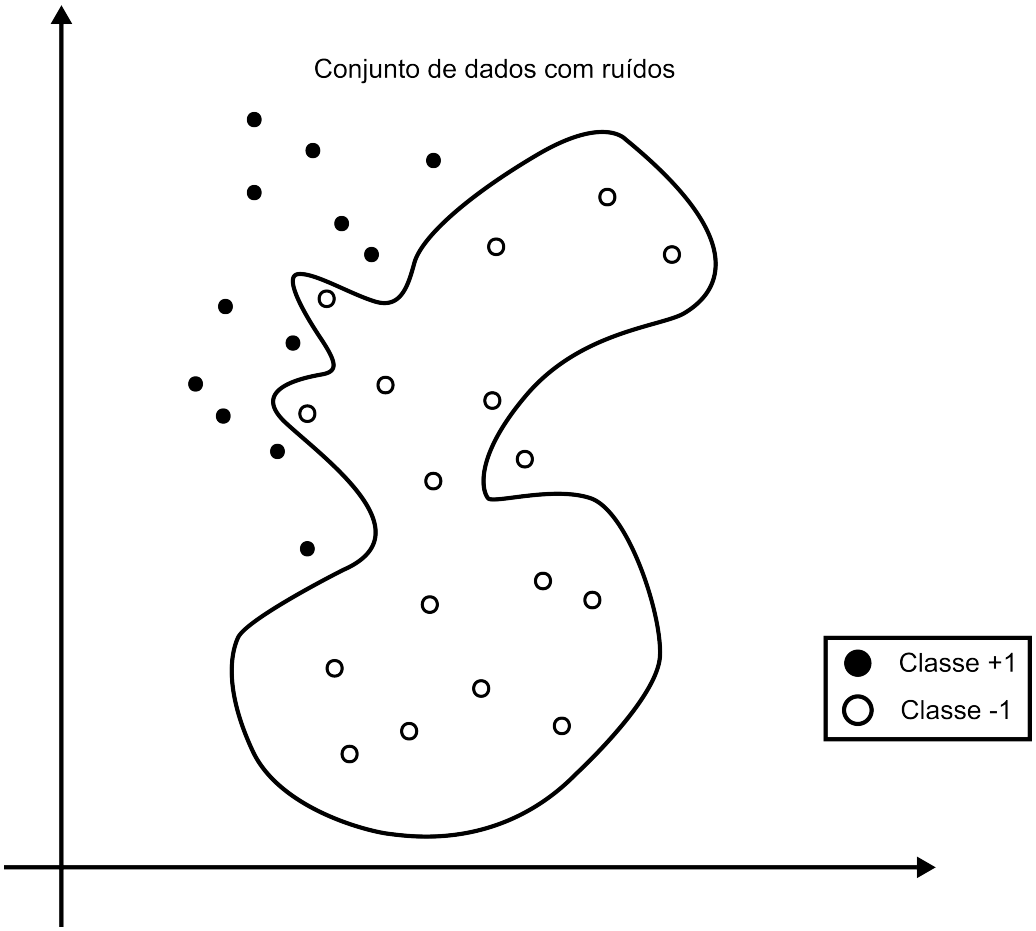

Figura 10: Conjunto de dados que requer o SVM linear com margem suave.
Fonte: Traduzido de Soman et al. (2009)

Neste caso, segundo Soman et al. (2009), não recomenda-se o uso de um SVM não linear porque o risco de *overfitting* com um modelo mais robusto é grande. Considere também as seguintes restrições:

(52)

$$1w_1 + 0.8w_2 - \gamma \geq +1$$
$$3w_1 + 2.5w_2 - \gamma \geq +1$$
$$2.5w_1 + 1.0w_2 - \gamma \geq +1$$
$$1w_1 + 1.8w_2 - \gamma \leq -1$$
$$3w_1 + 4.5w_2 - \gamma \leq -1$$
$$2.5w_1 + 2.8w_2 - \gamma \leq -1$$

Como permite-se o erro de treinamento ξ_i e os pontos não são linearmente separáveis, as restrições passam a ser:

(53)

$$1w_1 + 0.8w_2 - \gamma + \xi_1 \geq +1$$
$$3w_1 + 2.5w_2 - \gamma + \xi_2 \geq +1$$
$$2.5w_1 + 1w_2 - \gamma + \xi_3 \geq +1$$
$$1w_1 + 1.8w_2 - \gamma + \xi_4 \leq -1$$
$$3w_1 + 4.5w_2 - \gamma + \xi_5 \leq -1$$
$$2.5w_1 + 2.8w_2 - \gamma + \xi_6 \leq -1$$

Isso equivale à:

(54)

$$\begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \left\{ \begin{bmatrix} 1.0 & 0.8 \\ 3.0 & 2.5 \\ 2.5 & 1.0 \\ 1.0 & 1.0 \\ 3.0 & 4.5 \\ 2.5 & 2.8 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} - \gamma \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \right\} + \geq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Logo, a forma matricial primal do L1-SVM é:

(55)

Minimize : $\varsigma^* = \frac{1}{2}\, \boldsymbol{w^T w} + \mathrm{C}\mathbf{1}^{\boldsymbol{T}}\xi$

Sujeito à

$\mathbf{D}(\mathbf{A}\boldsymbol{w} - \gamma\mathbf{1}) + \xi \geq 1$

$\xi \geq 0$

Para achar a forma primal, basta seguir os mesmos passos descritos para o SVM Linear, atentando para o fato de que agora se terá mais um multiplicador de Lagrange.

(56)

$$\mathrm{Max}_{\substack{\lambda \geq 0 \\ \mu \geq 0}} \left[\mathrm{Min}_{\mathrm{w},\gamma,\xi} \mathrm{L}(\mathrm{w}, \gamma, \lambda, \mu, \xi) \right]$$

Seguindo a mesma lógica, para determinado λ e μ devemos encontrar w, γ e ξ que minimizem $\mathrm{L}(\mathrm{w}, \gamma, \lambda, \mu, \xi)$ e depois substituir o resultado na função Lagrangeana para maximizá-la em função de λ e μ. Lembrando que $\mathrm{L}(\mathrm{w}, \gamma, \lambda, \mu, \xi)$ é um escalar e a função Lagrangeana é dada por:

(57)

$$L(w,\gamma,\lambda,\mu,\xi) = \frac{1}{2}\boldsymbol{w^T w} + C\mathbf{1}^{\boldsymbol{T}}\xi - \boldsymbol{\lambda^T}[\boldsymbol{D}(\boldsymbol{Aw} - \gamma 1) + \xi - 1] - \boldsymbol{\mu^T}\xi$$

Resolvendo as condições de primeira ordem, se tem:

(58)

$$\frac{\partial}{\partial w}\mathrm{L}(\mathrm{w},\gamma,\lambda,\mu) = 0$$

$$\rightarrow \frac{\mathrm{d}}{\mathrm{d}w}\left(\frac{1}{2}\boldsymbol{w^T w} + \mathrm{C}\mathbf{1}^{\boldsymbol{T}}\xi - \boldsymbol{\lambda^T DAw} + \boldsymbol{\lambda^T}\mathbf{D}\gamma\mathbf{1} - \boldsymbol{\lambda^T}\xi + \boldsymbol{\lambda^T} - \boldsymbol{\mu^T}\xi\right) = 0$$

$$\rightarrow \boldsymbol{w^T} - \boldsymbol{\lambda^T}\mathbf{DA} = 0 \rightarrow\ \boldsymbol{w^T} = \boldsymbol{\lambda^T}\mathbf{DA}$$

(59)

$$\frac{\partial}{\partial\gamma}\mathrm{L}(\mathrm{w},\gamma,\lambda,\mu) = 0$$

$$\rightarrow \frac{\mathrm{d}}{\mathrm{d}\gamma}\left(\frac{1}{2}\boldsymbol{w^T w} + \mathrm{C}\mathbf{1}^{\boldsymbol{T}}\xi - \boldsymbol{\lambda^T}\mathbf{DA}\boldsymbol{w} + \boldsymbol{\lambda^T}\mathbf{D}\gamma\mathbf{1} - \boldsymbol{\lambda^T}\xi + \boldsymbol{\lambda^T} - \boldsymbol{\mu^T}\xi\right) = 0$$

$$\rightarrow\ \boldsymbol{\lambda^T}\mathbf{D1} = 0$$

(60)

$$\frac{\partial}{\partial\xi}\mathrm{L}(\mathrm{w},\gamma,\lambda,\mu) = 0 \rightarrow \frac{\mathrm{d}}{\mathrm{d}\xi}\left(\frac{1}{2}\boldsymbol{w^T w} + \mathrm{C}\mathbf{1}^{\boldsymbol{T}}\xi - \lambda^T\mathbf{DA}\boldsymbol{w} + \boldsymbol{\lambda^T}\mathbf{D}\gamma\mathbf{1} - \boldsymbol{\lambda^T}\xi + \boldsymbol{\lambda^T} - \boldsymbol{\mu^T}\xi\right)$$

$$= 0 \rightarrow\ \mathrm{C}\mathbf{1}^{\boldsymbol{T}} - \boldsymbol{\lambda^T} - \boldsymbol{\mu^T} = 0$$

Vale ressaltar que como $\lambda \geq 0$, $\mu \geq 0$ e $\mathbf{1}^{\boldsymbol{T}} - \boldsymbol{\lambda^T} - \boldsymbol{\mu^T}$, o maior valor de μ é C e λ_i é máximo quando $\mu_i = 0$. Portanto, os multiplicadores de Langrange possuem valores truncados. Essa é maior diferença entre o SVM de Margem Suave e o SVM Linear Clássico.

Substituindo os resultados na função Lagrangeana, o problema Dual é:

(61)

$$L(\lambda) = -\frac{1}{2}\boldsymbol{\lambda^T}\mathbf{DAA^T}\mathbf{D}\boldsymbol{\lambda} + \boldsymbol{\lambda^T}\mathbf{1}$$

Sujeito a

$\boldsymbol{1^T}\mathbf{D1} = 0$

$0 \leq \lambda \leq \mathrm{C}1$

Existem outras variações do SVM Linear, como o SVM L2-Norm SVM , onde ao invés de se minimizar apenas o erro, opta-se por minimizar a soma dos quadrados dos erros. A formulação do L2-Norm SVM é:

(62)

$$\text{Minimize} : \varsigma^* = \frac{1}{2}\,\boldsymbol{w^T w} + \frac{\mathrm{C}}{2}\sum_{i=1}^{m}\xi_i^2$$

Sujeito a

$$\mathbf{D}(\boldsymbol{w^T x} - \gamma) + \xi_i - 1 \geq 0$$

$$\xi_i \geq 0$$

4.2.3 SVM Não Linear

Apesar dos modelos lineares serem eficientes e fáceis de aplicar, um problema de separação linear é apenas um caso particular e na prática eles apresentam muitas limitações, por isso, torna-se necessária uma formulação capaz de lidar com problemas mais complexos. Esse contexto levou Vapnik e Vapnik (1998) a formular métodos baseados em funções *Kernel* para aprendizado em dados de alta dimensão.

O SVM é facilmente estendido para modelos não lineares pela incorporação implícita de características não lineares na representação dos dados através de *Kernels*. O De acordo com Soman et al. (2009), o problema de aprendizagem para SVMs é definido como uma relação de dependência desconhecida e não linear entre dados de alta dimensão, representados por um vetor ou matriz $\mathbf{A}$, e uma variável alvo ou variável *target*, representada por um escalar y ou vetor $\boldsymbol{y}$, no caso de SVMs de múltiplas classes. Essa relação é descrita por um mapeamento ou função $y = f(\mathrm{x})$, onde a matriz ou vetor $\boldsymbol{x}$ são os *inputs* e a variável y, é o *output.* A única informação disponível é o conjunto de treinamento $T_D = \{(x_i, y_i)\},\ i = 1,2,\ldots,m$, onde m representa o número de pares do conjunto de treinamento e é, portanto, igual à dimensão do conjunto de treinamento T_D.

Em situações como a da Figura 11, se há uma tentativa de separação linear dos dados, o grau de tolerância para erros de classificação deve ser bem alto. Neste caso, é preferível construir um mapeamento dos dados em algum espaço de

dimensão maior, em um “espaço de característica” no qual eles serão linearmente separáveis.

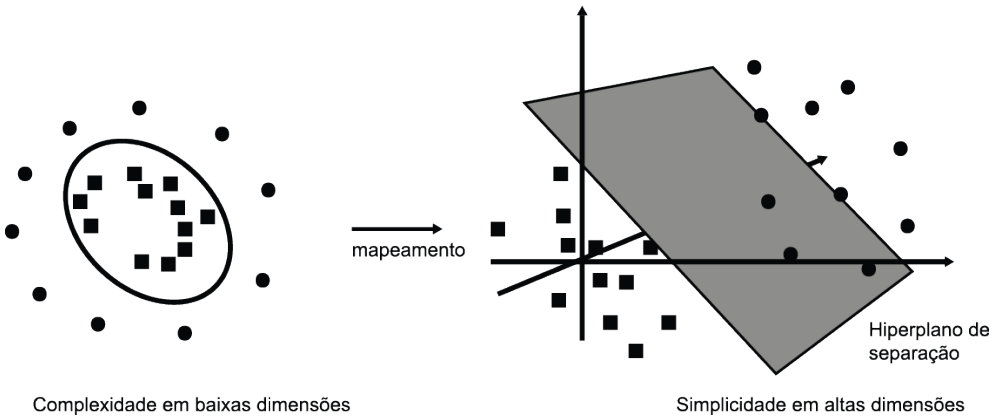

Figura 11: Classificador não linear.
Fonte: Traduzido de Soman et al. (2009).

Para distinguir a dimensão original dos dados e o espaço de característica, Soman et al. (2009) denominam o primeiro de “espaço de *input”*. A Figura 12 ilustra o processo de mapeamento não linear no espaço de característica.

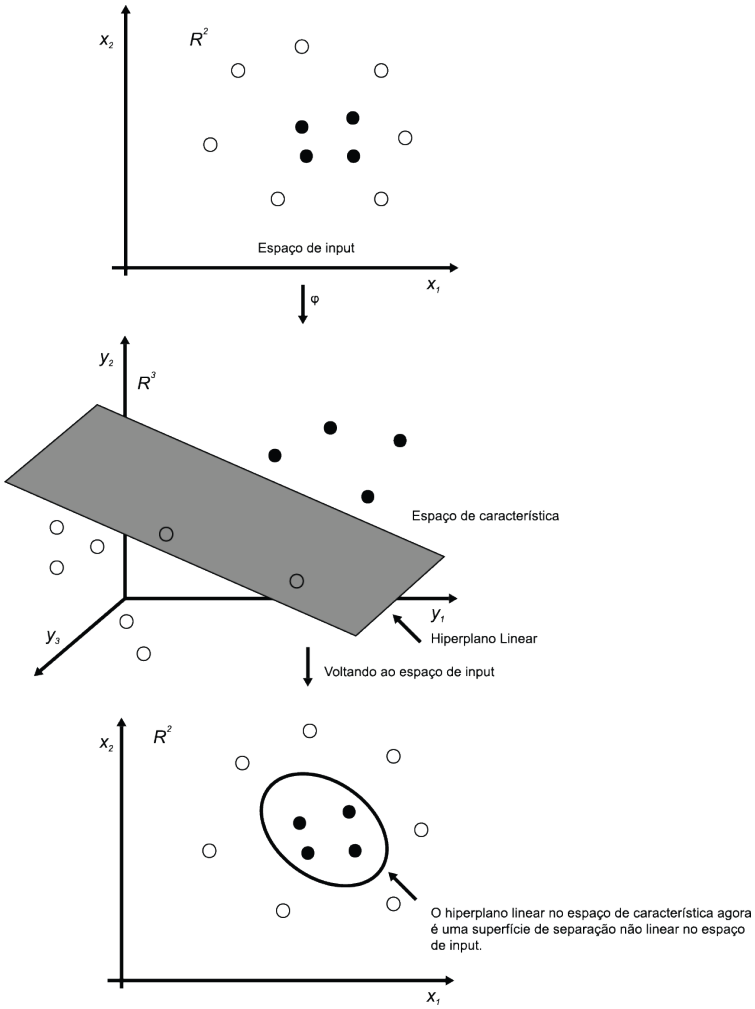

Figura 12: Processo de mapeamento.
Fonte: Traduzido de Soman et al. (2009).

Fica evidente então que a estratégia para se trabalhar com a não linearidade de dados é criar novas dimensões por meio do processo de mapeamento e este é descrito da seguinte forma:

(63)

$$x \rightarrow \phi(x)$$
$$\mathbb{R}^p \rightarrow \mathbb{R}^q \quad \text{tal que } q \gg p.$$

Na situação ilustrada na Figura 12, por exemplo, partiu-se de um espaço bidimensional (x_1, x_2) para um espaço tridimensional $(t_1, t_2, t_3) = \left(x_1^2, x_2^2, \sqrt{2}x_1x_2\right)$. Porém, existem diversos mapeamentos que podem levar a diferentes espaços de características e o desafio que surge é justamente identificar qual o melhor para determinado problema de classificação de forma que este minimize o erro de generalização.

4.2.4 Formulação do SVM Não Linear com Margem Suave

Com as considerações apresentadas, fica evidente que a matriz *Kernel* substitui a matriz $\mathbf{A}$ na formulação do SVM e assim, o problema de separação não linear com margem suave, pode ser escrito como:

(64)

$$\text{Minimize} : \varsigma^* = \frac{1}{2}\,\boldsymbol{w^T w} + \mathrm{C}\mathbf{1}^{\boldsymbol{T}}\xi$$

Sujeito à

$$\mathbf{D}[\boldsymbol{\Phi}(\mathrm{x})\boldsymbol{w}] + \xi \geq 1$$

$$\xi \geq 0$$

Onde,

(65)

$$\boldsymbol{\Phi} = \begin{pmatrix} \boldsymbol{\phi}(\mathrm{x}_1)^{\boldsymbol{T}} \\ \boldsymbol{\phi}(\mathrm{x}_2)^{\boldsymbol{T}} \\ \vdots \\ \boldsymbol{\phi}(\mathrm{x}_{\boldsymbol{n}})^{\boldsymbol{T}} \end{pmatrix} \text{ e } \boldsymbol{w} = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix}$$

Ou seja, $\boldsymbol{\Phi}$ é uma matriz de dimensão $n \times q$ e nesse caso o vetor $\boldsymbol{w}$ possui dimensão $q \times 1$. Assim como no caso linear, para a resolução do problema (64) é mais interessante trabalhar com o Dual de Wolfe (1961). Então, primeiramente é preciso, encontrar a função Lagrangeana que pode ser escrita como:

(66)

$$\mathrm{L}(\mathrm{w}, \gamma, \lambda, \mu, \xi) = \frac{1}{2}\boldsymbol{w^T w} + \mathrm{C}\mathbf{1}^{\boldsymbol{T}}\xi - \boldsymbol{\lambda^T}[\mathbf{D}[\boldsymbol{\Phi}(\mathrm{x})w] + \xi - 1] - \boldsymbol{\mu^T}\xi$$

Resolvendo as condições de primeira ordem, se tem:

(67)

$$\frac{\partial}{\partial \mathrm{w}}\mathrm{L}(\mathrm{w}, \gamma, \lambda, \mu) = 0 \rightarrow \frac{\mathrm{d}}{\mathrm{dw}}\left(\frac{1}{2}\mathbf{w^T w} + \mathrm{C}\mathbf{1}^{\mathbf{T}}\xi - \boldsymbol{\lambda}^{\mathbf{T}}\mathbf{D}\boldsymbol{\Phi}(\mathrm{x})\mathbf{w} - \boldsymbol{\lambda}^{\mathbf{T}}\xi + \boldsymbol{\lambda}^{\mathbf{T}} - \boldsymbol{\mu}^{\mathbf{T}}\xi\right) = 0$$

$$\rightarrow \mathbf{w^T} - \boldsymbol{\lambda}^{\mathbf{T}}\mathbf{D}\boldsymbol{\Phi}(\mathrm{x}) = 0 \rightarrow \ \mathbf{w^T} = \boldsymbol{\lambda}^{\mathbf{T}}\mathbf{D}\boldsymbol{\Phi}(\mathrm{x})$$

$$\frac{\partial}{\partial \xi} \mathrm{L}(\mathrm{w}, \gamma, \lambda, \mu) = 0 \rightarrow \frac{\mathrm{d}}{\mathrm{d}\xi}\left(\frac{1}{2}\boldsymbol{w^T w} + \mathrm{C}\mathbf{1}^{\boldsymbol{T}}\xi - \boldsymbol{\lambda^T}\mathbf{D}\boldsymbol{\Phi}(\mathrm{x})\boldsymbol{w} - \boldsymbol{\lambda^T}\xi + \boldsymbol{\lambda^T} - \boldsymbol{\mu^T}\xi\right) = 0$$

$$\rightarrow \mathrm{C}\mathbf{1}^{\boldsymbol{T}} - \boldsymbol{\lambda^T} - \boldsymbol{\mu^T} = 0$$

Substituindo,

(68)

$$\mathrm{L}(\lambda) = \frac{1}{2}\lambda^T \mathbf{D}\boldsymbol{\Phi}(\mathrm{x})\boldsymbol{\Phi}(\mathrm{x})^{\mathrm{T}}\mathbf{D}\lambda - \lambda^T \mathbf{D}\boldsymbol{\Phi}(\mathrm{x})\boldsymbol{\Phi}(\mathrm{x})^{\mathrm{T}}\mathbf{D}\boldsymbol{\lambda} + \mathrm{C}\mathbf{1}^{\boldsymbol{T}}\xi - \boldsymbol{\lambda^T}\xi - \boldsymbol{\mu^T}\xi + \boldsymbol{\lambda^T}\mathbf{1}$$

(69)

$$\mathrm{L}(\lambda) = -\frac{1}{2}\boldsymbol{\lambda^T}\mathbf{D}\boldsymbol{\Phi}(\mathrm{x})\boldsymbol{\Phi}(\mathrm{x})^{\mathrm{T}}\mathbf{D}\boldsymbol{\lambda} + \xi(\mathrm{C}\mathbf{1}^{\boldsymbol{T}} - \boldsymbol{\lambda^T} - \boldsymbol{\mu^T}) + \boldsymbol{\lambda^T}\mathbf{1}$$

Sendo assim, o Dual de Wolfe (1961) do SVM Não Linear com Margem Suave é:

(70)

$$\mathrm{L}(\lambda) = -\frac{1}{2}\boldsymbol{\lambda^T}\mathbf{D}\boldsymbol{\Phi}(\mathrm{x})\boldsymbol{\Phi}(\mathrm{x})^{\boldsymbol{T}}\mathbf{D}\boldsymbol{\lambda} + \boldsymbol{\lambda^T}\mathbf{1}$$

Sujeito à

$0 \leq \lambda \leq \mathrm{C}1$

$\lambda \geq 0$

4.2.5 Parâmetros do SVM

O bom desempenho de um modelo classificador de padrões é alcançado quando a capacidade da função de classificação é compatível com o tamanho do conjunto de treinamento. Classificadores com um grande número de parâmetros ajustáveis geralmente apresentam grande capacidade de aprender os padrões do conjunto de treinamento sem erro, mas isso vem acompanhado por uma baixa capacidade de generalização. Sendo assim, deve existir um equilíbrio entre a capacidade de generalização do classificador e sua complexidade.

Segundo Kim (2003), uma das vantagens do SVM é o fato dele depender de um pequeno número de parâmetros, diferente da maioria modelos de previsão. Não há um valor definido para cada uma das constantes e embora sejam poucas, a

escolha destas é essencial para o bom desempenho do modelo, visto que o problema de predição relaciona-se diretamente com o *trade-off* entre capacidade de generalização do classificador e sua complexidade.

O valor do parâmetro C é importante porque indica o peso que se dá para uma classificação errada, ou seja, quanto custa o erro. Tay e Cao (2001) inferiram que no conjunto de treinamento, um valor muito pequeno para C causaria *underfitting* e um valor muito alto, *overfitting*. Por isso, defendem que o valor apropriado para o C é entre 1 e 100. No estudo de Kim (2003), os resultados corroboram essa hipótese, pois o desempenho do SVM no conjunto de validação aumenta quando C aumenta de 1 a 78, mas cai quando C é 100.

Na literatura existente, a maioria das aplicações utiliza a validação cruzada no processo de seleção dos parâmetros. Gupta et al. (2012) por exemplo, determinaram os valores de C e σ por meio de uma validação cruzada com 10 etapas e chegaram aos valores ótimos de 2^{25} e 2^{-3}, respectivamente. Fan e Palaniswami (2001) por sua vez, utilizaram um ano de dados para selecionar os parâmetros C e σ que oferecessem maior acurácia, por meio da validação cruzada. A Classe $+1$, composta pelos ativos que apresentaram retorno excepcional correspondeu a um terço da Classe -1, composta pelos ativos com retornos "normais". Assim, os dados de treinamento do estudo foram sempre não balanceados e por essa razão foram atribuídos valores diferentes de C para as diferentes classes.

Já Huerta et al. (2013) para escolher os parâmetros C e σ que maximizam o desempenho do modelo na etapa de validação, optaram por criar pares com valores de C e σ, usados para construir um portfólio no tempo t e estes pares foram chamados de a. Foi definido um conjunto de 16 pares com $\mathrm{C} = 0.5, 1, 2, 4$ e $\sigma = 0.5, 1, 2, 4$. A qualidade da escolha é definida como $Q(t, a)$, sendo atualizada com uma média exponencial móvel, descrita por:

(71)

$$Q(t, a) = (1 - \alpha)Q(t - 1, a) + \alpha R(t - 1, a)$$

Onde α é uma taxa de aprendizagem escolhida com a média de três anos e $R(t - 1, a)$ é a remuneração obtida com a escolha a no período imediatamente anterior. Para fins de aplicação, os autores consideraram o retorno puro como $R(t -$

$1, a)$ e os valores dos meta parâmetros usados para formar o portfólio no tempo t foram obtidos por meio de:

(72)

$$a_t = \arg\max_{\mathrm{a}} Q(t, a)$$

Para o parâmetro σ, Tay e Cao (2001) encontraram o intervalo de 1 a 100 como o mais adequado e sugerem o raciocínio contrário ao proposto para o C. Um valor muito pequeno de σ geraria *overfitting* nos dados de treinamento e um σ grande causaria *underfitting*. Os resultados de Kim (2003) mostram que o desempenho do SVM nos dados de treinamento diminuiu com o valor de σ e no conjunto de validação, o desempenho se manteve estável com a variação de parâmetro de 25 a 100.

Apesar de ser possível encontrar na literatura diversas sugestões de valor para os parâmetros do SVM, esta ainda é uma questão em aberto. Não há evidências que comprovem empiricamente que existem valores ótimos universais para C e σ. São encontrados diversos parâmetros ótimos para diferentes tipos de dados e por isso, na maioria dos casos, são escolhidos de forma arbitrária.

5 SEGUNDA APLICAÇÃO: MÁQUINAS DE SUPORTE VETORIAL DE CLASSIFICAÇÃO PARA FORMAÇÃO DE PORTFÓLIOS

Este capítulo relata a segunda aplicação empírica desta dissertação. Aqui, o SVM clássico foi usado para classificar os ativos em bons ou ruins e posteriormente formar portfólios trimestrais ponderados com as melhores ações.

5.1 Procedimentos de Coleta e Análise de Dados: Máquinas de Suporte Vetorial de Classificação

Na segunda aplicação deste estudo, primeiramente foram utilizados os mesmos dados dos 24 indicadores financeiros trimestrais utilizados como insumos para o SVM de regressão. Porém, depois de resultados não satisfatórios com essas variáveis escolhidas *a priori*, optou-se utilizar mais 103 indicadores financeiros disponíveis na base da *Bloomberg*, totalizando assim, 127 *inputs*.

No que tange ao histórico de retornos, utilizou-se o mesmo histórico de retornos trimestrais e acumulados já previamente calculados na primeira aplicação. Da mesma forma, observações com dados incompletos foram imputadas pela média.

As ações foram então ranqueadas em função do retorno apresentado e classificadas em duas classes. A Classe 1 $(y_i = +1)$ foi composta pelos 25% das ações com maiores retornos e a Classe 2 $(y_i = -1)$ foi composta pelas demais ações que representam 75%.

Nesta segunda aplicação do modelo, a Validação Cruzada também foi utilizada, mantendo-se a proporção de cada conjunto: o conjunto de validação foi constituído por 22,5% das observações, o de treinamento por 52,5% e o de teste por 25%, sendo que os dois primeiros conjuntos foram escolhidos de forma aleatória visando minimizar os efeitos macroeconômicos.

Desta vez, o *grid* para definição dos parâmetros ótimos foi constituído pelas sequências de C variando entre 1 e 1000 e σ variando de 1^{-3} a 10.

Como medida de acurácia do desempenho do SVM, utilizou-se a razão entre a quantidade de vezes que uma ação pertencente à Classe 1 foi classificada

pelo SVM como pertencente à Classe 2, pelo número total de ações selecionadas para o portfólio, ou seja, o quanto o SVM errou na classificação das boas ações.

(73)

$$\frac{\boldsymbol{Total\ de\ ativos\ da\ Classe\ 1\ classificados\ erroneamente}}{\boldsymbol{Total\ de\ ativos\ selecionados}}$$

Com o SVM de classificação, apenas uma máquina foi construída e o núcleo utilizado foi o *Kernel* Gaussiano, elencado no Quadro 1. Este é o mais popular nas aplicações do SVM em finanças, principalmente porque ele mapeia um ponto em um espaço de dimensões infinitas, permitindo uma busca mais generalizada e mais rápida da solução ótima. Por essa razão, foi utilizado nesta pesquisa, assim como nos estudos de Tay e Cao (2001), Huerta et al. (2013), Emir et al. (2012) e Kim (2003).

Nota-se que o conjunto de dados deste estudo é desbalanceado e por isso surgiu um viés em direção à Classe 2, visto que esta é três vezes maior que a Classe 1. Para resolver tal problema, Veropoulos, Campbell, e Cristianini (1999) sugerem uma ponderação diferente do parâmetro C em cada classe, o que implica em uma mudança na formulação do problema (70):

(74)

$$\mathrm{L}(\lambda) = -\frac{1}{2}\boldsymbol{\lambda^T D \Phi}(\mathrm{x})\boldsymbol{\Phi}(\mathrm{x})^{\boldsymbol{T}}\mathbf{D}\boldsymbol{\lambda} + \boldsymbol{\lambda^T 1}$$

Sujeito à

$$0 \leq \lambda \leq \mathrm{C}_{+},\ y_i = +1$$

$$0 \leq \lambda \leq \mathrm{C}_{-},\ y_i = -1$$

$$\lambda \geq 0$$

Nota-se que para formação de portfólio, é mais custoso errar a classificação de uma ação da Classe 1 do que classificar erroneamente uma ação da Classe 2. Assim, nesta pesquisa o SVM foi construído com pesos de erro diferentes para cada uma das classes, o que permitiu níveis diferentes de erro e, consequentemente, um controle mais efetivo sobre a sensibilidade do modelo. O critério de classificação do SVM de classificação é uma probabilidade, uma medida discreta. Já o critério do SVR é o retorno previsto, uma medida contínua. Dessa forma, com o SVR, a base de dados não ficou desbalanceada o que dispensou o uso de dois C diferentes na primeira aplicação desta dissertação.

Considerando que a distância de um ponto até o hiperplano que limita a classe na qual ele está inserido está relacionado à probabilidade de erro de classificação, é possível usar informações geométricas para interpretar as saídas do SVM como probabilidades. Dessa forma, após o treinamento da máquina com os parâmetros ótimos, a função de predição do SVM foi programada para retornar no período de teste, as classes de probabilidade e não apenas escalares $+1$ e -1.

Segundo Platt (1999), construir um classificador que produza probabilidades da relação classe-*input* é bastante útil em caso práticos de reconhecimento de padrões. Segundo este mesmo autor, *outputs* em forma de probabilidades são necessários quando um classificador direciona uma pequena parte de uma decisão mais geral e os *outputs* têm que ser combinados para se chegar à decisão final.

Sendo assim, as ações foram classificadas e ranqueadas de acordo com a probabilidade de pertencerem à Classe 1 ($y_i = +1$), ou seja, os 25% de ações com maior probabilidade de apresentarem um retorno excepcional foram selecionadas para a formação do portfólio final. Nota-se também que a classificação feita por meio de probabilidades permite o controle do tamanho do portfólio, pois garante-se que em cada período será selecionado um número mínimo ou máximo de ações, o que diminui o risco geral pela diversificação. O portfólio foi ponderado pela probabilidade do ativo pertencer à Classe 1 ($y_i = +1$).

O retorno do portfólio foi comparado trimestralmente aos retornos do *benchmark* de mercado definido como o retorno trimestral médio das ações do S&P 100. Por meio de uma análise *ex-post* da distribuição do retorno, o VaR indicou com 5% de chance de erro, o quanto poderia se perder em um trimestre na pior das hipóteses.

5.2 Resultados e Discussão: Máquinas de Suporte Vetorial de Classificação

Os resultados da segunda aplicação empírica desta dissertação serão apresentados em duas etapas. A primeira corresponde à aplicação do SVM feita com seleção de 24 variáveis de inputs *à priori*. A segunda refere-se à aplicação do mesmo método, mas com todas as 127 variáveis disponíveis na base de dados.

5.2.1 *Primeira etapa com seleção de variáveis a priori*

Na segunda aplicação desta dissertação, a Máquina de Suporte Vetorial foi construída primeiramente tendo como insumos os resultados trimestrais dos 24 indicadores financeiros escolhidos na primeira aplicação. Infere-se que estes 24 indicadores explicam o retorno trimestral líquido do período seguinte de cada ativo, e este por sua vez, determina a classificação da ação em uma das duas classes ($y_i = +1$ ou $y_i = -1$).

O modelo probabilístico da função *ksvm* foi utilizado para que o SVM interpretasse os *outputs* como a probabilidade dos ativos serem classificados como $+1$, isto é, pertencente à Classe das boas ações. A Classe 1 foi composta então pelos 25% de ações com maiores probabilidades e a Classe 2, pelo restante dos ativos. Assim como no estudo de Fan e Palaniswami (2001), o modelo probabilístico corroborou para maior retorno acumulado do portfólio.

Visando contornar o viés criado pela base de dados não balanceada, utilizou-se para a Classe 1 $\mathrm{C}_+ = 80\%$ e para a Classe 2, $\mathrm{C}_- = 20\%$.

Considerando a amostra em questão, chegou-se aos parâmetros ótimos de $\mathrm{C} = 857{,}29$ e $\sigma = 0{,}096$ com uma performance de $0{,}2854$, ou seja, o SVM classificação foi capaz de prever corretamente a classificação das ações em apenas 28,54% das vezes. Ressalta-se que foram encontrados 22 pares de parâmetros ótimos, ou seja, 22 pares retornaram a mesma acurácia máxima e de forma arbitrária o primeiro par foi escolhido. Sendo assim, pode-se inferir que a volatilidade do portfólio é influenciada não apenas pelos retornos obtidos, mas também pelos parâmetros utilizados no modelo. Estes parâmetros ainda podem ser melhorados em grande medida e mesmo que se trabalhe com a mesma acurácia, existe a possibilidade de alteração nos valores de risco e retorno caso outro par ótimo seja utilizado.

Apesar da baixa acurácia, formou-se um portfólio ponderado pela probabilidade do ativo de ser classificado como uma boa ação e este apresentou um retorno acumulado em 22 trimestres de $185{,}77\%$ e VaR de $-13{,}14\%$. Nesse mesmo período, o retorno acumulado do S&P 100 foi de $192{,}65\%$.

Considerando a relação retorno/risco, obteve-se 0,40 para o portfólio formado pelo SVM e 0,45 para o *benchmark* de mercado. Isso significa que para

cada 1% de risco no portfólio selecionado pelo SVM, há 0,40% de retorno. Já o *benchmark* de mercado apresentou 0,45% de retorno para cada 1% de risco. Como essa razão relaciona a maximização do retorno e a minimização do risco, quanto maior o valor do quociente, melhor é a opção de investimento. Ressalta-se novamente que essa relação retorno/risco vale apenas para 5% dos trimestres.

Tendo em vista os resultados encontrados, a máquina de suporte vetorial foi construída novamente, porém, com apenas uma medida C para ambas classes. Com essa nova máquina, chegou-se aos parâmetros ótimos de $C = 1$ e $\sigma = 0{,}0542$ com uma performance de $47{,}6\%$. O portfólio ponderado formado apresentou um retorno acumulado de $191{,}72\%$ e VaR de $-14{,}05\%$.

Fica evidente então que apesar das recomendações na literatura, em especial de Fan e Palaniswami (2001), a máquina de suporte vetorial de classificação construída nesta pesquisa apresentou melhor acurácia quando não foram utilizados custos de erros diferentes para cada uma das classes, porém, em nenhuma das aplicações a acurácia se mostrou satisfatória.

Apesar do retorno acumulado do portfólio ter aumentado um pouco, o risco também aumentou, culminando em uma relação retorno/risco um pouco menor que a encontrada com o portfólios selecionado pelo primeiro classificador. Para o novo portfólio selecionado pelo SVM obteve-se 0,39% de retorno para cada 1% de risco, como pode ser observado na Tabela 3.

Tabela 3: Resultados do SVM de Classificação com seleção de variáveis *a priori.*

	Portfólio SVM	**Portfólio SVM**	**Benchmark de Mercado**
Kernel	Gaussiano	Gaussiano	-
Quantidade *inputs*	24	24	-
Cs diferentes?	Sim	Não	-
Sigma	0,096	0,0542	-
C	857,29	1	-
Performance	28,50%	47,60%	-
Retorno acumulado	185,77%	191,72%	192,60%
Retorno Trimestral Médio	5,29%	5,43%	5,30%
VaR	-13,14%	-14,05%	11,74%
Retorno/Risco	0,4	0,39	0,45

Fonte: Elaborado pela autora.

5.2.1 Segunda etapa com todas as variáveis disponíveis

Visando aprimorar o classificador, opotou-se por aumentar a quantidade de variáveis de *inputs*. Dessa forma, foram utilizados 127 indicadores financeiros disponíveis na base de dados da Bloomberg no período da coleta de dados. Nessa abordagem com as novas variáveis chegou-se aos parâmetros ótimos de $C = 1$ e $\sigma = 0{,}004$ com uma performance de $81{,}82\%$. Novamente formou-se um portfólio ponderado pela probabilidade do ativo de ser classificado como uma boa ação e este apresentou um retorno acumulado em 22 trimestres de $215{,}42\%$ e VaR de $-12{,}69\%$, enquanto o *benchmark* de mercado apresentou $192{,}65\%$ de retorno acumulado e $-11{,}74\%$ de VaR.

Considerando a relação retorno/risco, obteve-se 0,46 para o portfólio formado pelo SVM e 0,45 para o *benchmark* de mercado. Isso significa que para cada 1% de risco no portfólio selecionado pelo SVM, há 0,46% de retorno. Já o *benchmark* de Mercado apresentou 0,45% de retorno para cada 1% de risco.

Essa mesma máquina que recebeu como *inputs* 127 variáveis foi modificada para inserir novamente as duas medidas diferentes de custo de erro para as classes. Com essa modificação, chegou-se os parâmetros ótimos de $\mathrm{C} = 286{,}43$ e $\sigma = 0{,}1$ com uma performance de $33{,}96\%$. O portfólio formado da mesma forma que os anteriores, apresentou retorno acumulado de $211{,}75\%$ e VaR de -7,99%. Tendo em vista novamente a relação retorno/risco, obteve-se 0,7 para o portfólio formado pelo SVM, ou seja, 0,7% de retorno para cada 1% de risco. Então, a ponderação do custo do erro entre as classes piorou a acurácia da máquina e o retorno do portfólio, porém, contribuiu para uma menor medida de risco, conforme pode ser observado na Tabela 4.

Tabela 4: Resultados do SVM de Classificação com 127 *inputs.*

	Portfólio SVM	**Portfólio SVM**	**Benchmark de Mercado**
Kernel	Gaussiano	Gaussiano	-
Quantidade *inputs*	127	127	-
Cs diferentes?	Não	Sim	-
Sigma	0,004	0,1	-
C	1	286,43	-
Performance	81,82%	33,96%	-
Retorno acumulado	215,42%	211,75%	192,60%
Retorno Trimestral	5,86%	5,59%	5,30%

Médio			
VaR	-12,69%	-7,99%	11,74%
Retorno/Risco	0,46	0,7	0,45

Fonte: Elaborado pela autora.

A Figura 13 ilustra o comportamento dos retornos trimestrais do melhor portfólio selecionado pelo SVM, aquele recebeu como *input* 127 variáveis e no qual utilizou-se a mesma medida de custo de erro para ambas as classes, assim como os retornos trimestrais do *benchmark* de mercado, ambos ao longo dos 22 trimestres de teste.

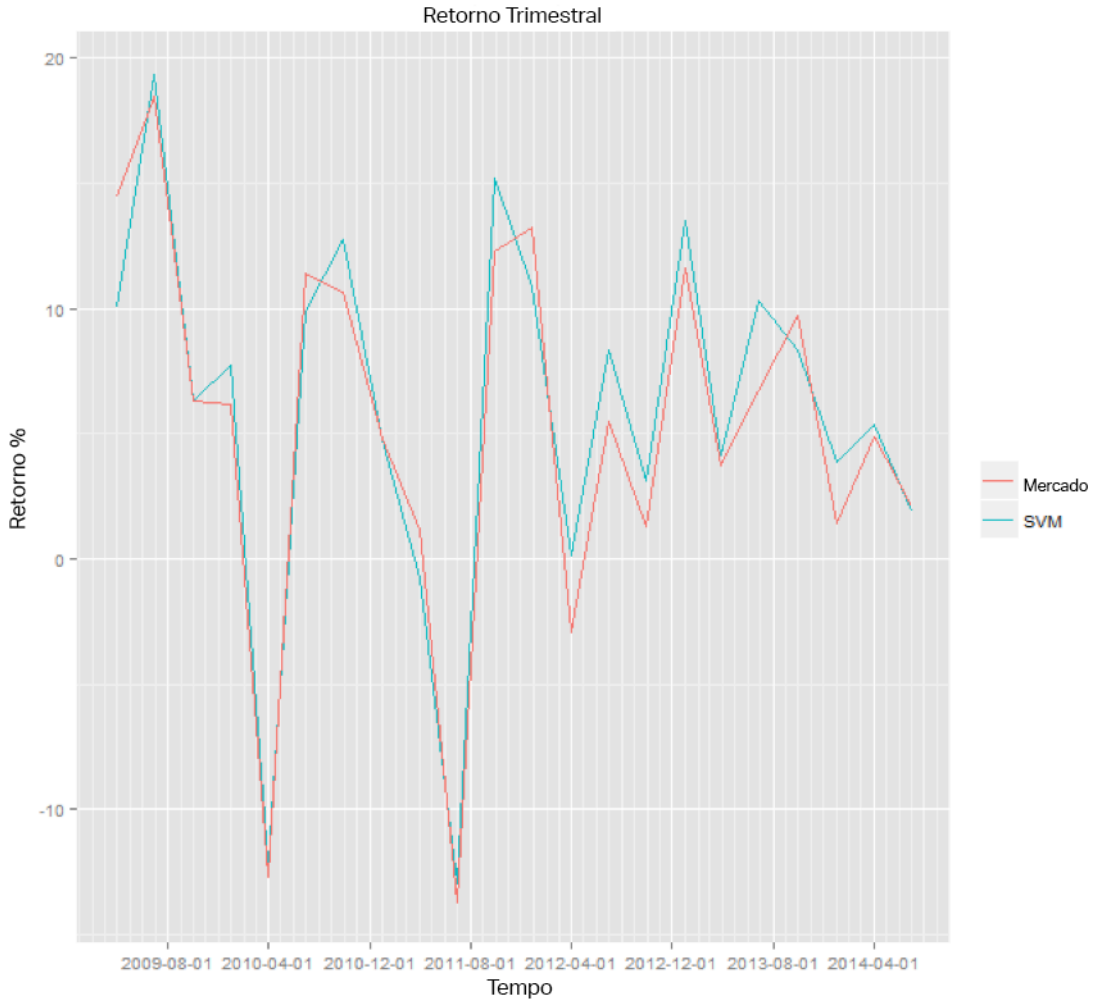

Figura 13: Comparação dos retornos trimestrais.
Fonte: Elaborado pela autora.

A Figura 14 compara o retorno acumulado de 22 trimestres desse portfólio e o retorno do *benchmark* de mercado.

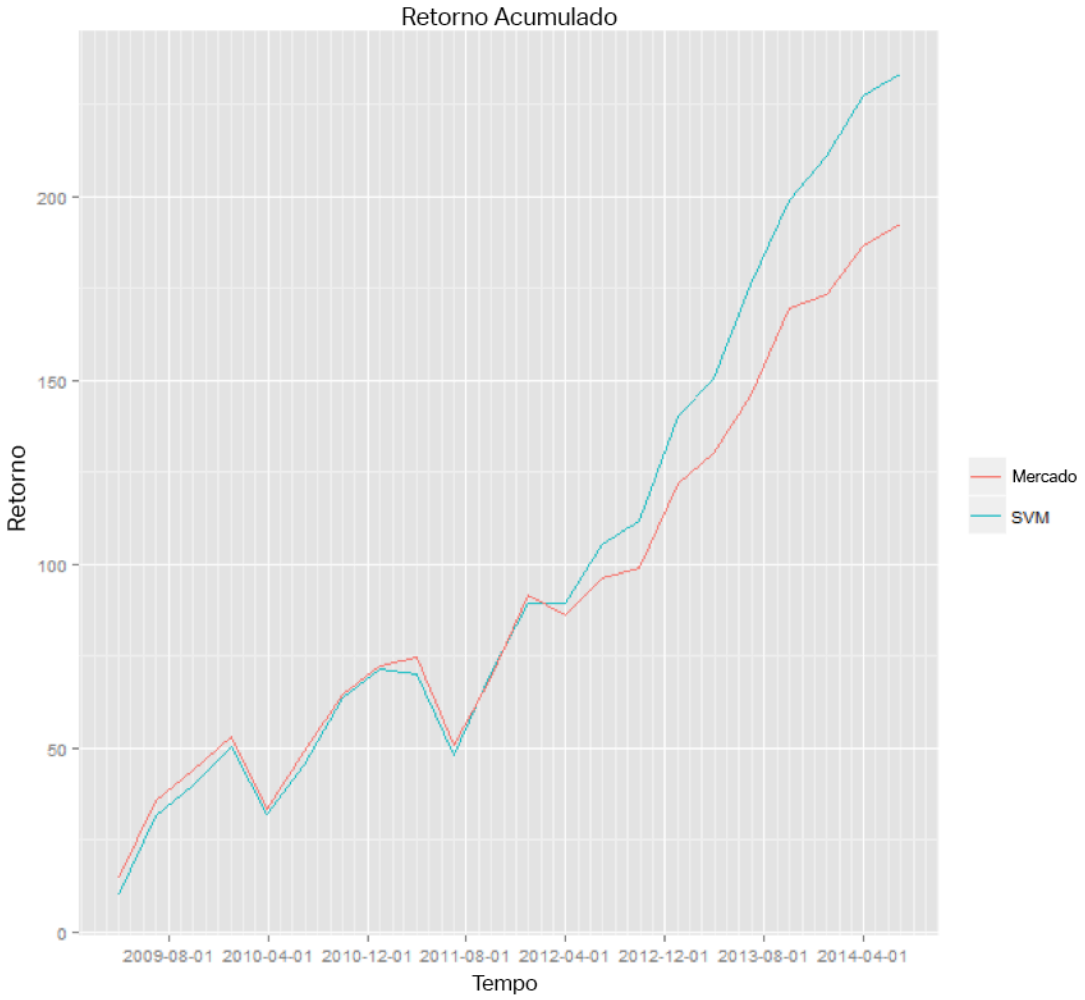

Figura 14: Comparação dos retornos acumulados.
Fonte: Elaborado pela autora.

A Tabela 5 traz a compilação de todos os resultados das máquinas construídas com o SVM de Classificação.

Tabela 5: Compilação dos resultados do SVM de Classificação

	Portfólio SVM	**Portfólio SVM**	**Portfólio SVM**	**Portfólio SVM**	***Benchmark* de Mercado**
Kernel	Gaussiano	Gaussiano	Gaussiano	Gaussiano	-
Quantidade de *inputs*	24	24	127	127	-
Cs diferentes?	Sim	Não	Não	Sim	-
Sigma	0,096	0,054	0,004	0,1	-
C	857,29	1	1	286,43	-
Performance	28,50%	47,60%	81,82%	33,96%	-
Retorno acumulado	185,77%	191,72%	215,42%	211,75%	192,60%
Retorno Trimestral	5,29%	5,43%	5,86%	5,59%	5,30%

Médio					
VaR	-13,14%	-14,05%	-12,69%	-7,99%	11,74%
Retorno/Risco	0,4	0,39	0,46	0,7	0,45

Fonte: Elaborado pela autora.

Observa-se que na primeira etapa, com 24 *inputs,* quando se deixou de usar o C ponderado, a acurácia da máquina aumentou consideravelmente. A máquina se tornou ainda mais confiável quando mais indicadores financeiros foram fornecidos como *inputs*, atingindo $81{,}82\%$ de acertabilidade.

No que tange ao retorno acumulado, nota-se que o portfólio selecionado pelo SVM superou o *benchmark* de Mercado apenas quando foram utilizados como *inputs* todos os indicadores disponíveis, isto é, os 127, o que reforça a importância da seleção dessas características para alimentar a máquina.

Considerando o risco, apenas um portfólio dentre os quatro formados, apresentou um VaR $(-7{,}99\%)$ menor que o apresentado pelo *benchmark*. Este mesmo portfólio apresentou o segundo maior retorno acumulado $(211{,}75\%)$ e uma relação retorno/risco significantemente maior $(0{,}7)$ maior que a do mercado. Porém, a acurácia do classificador não foi satisfatória.

6 ABORDAGEM *DEEP LEARNING* EM APRENDIZADO DE MÁQUINA

Apesar do sucesso dos métodos baseados em *Kernels* e das Máquinas de Suporte Vetorial, ambos considerados modelos de arquitetura rasa, trabalhos recentes em aprendizado de máquina têm evidenciado vários contextos que favorecem o uso de modelos com arquiteturas profundas (Bengio e LeCun, 2007).

A abordagem de arquitetura profunda com grande apelo comercial que surgiu nos últimos anos é o *Deep Learning*, convencionalmente usado dentro de Redes Neurais Artificiais.

Métodos de *Deep Learning* aprendem mapeamentos complexos por meio da transformação dos dados de *input* ao longo de múltiplas camadas de processamento não linear (Hinton, Osindero, e Teh, 2006), isto é, conseguem aprender representações dos dados com múltiplos níveis de abstração (LeCun, Bengio, e Hinton, 2015). Segundo LeCun et al. (2015), este aprendizado complexo é possível porque a abordagem *Deep Learning* consegue identificar estruturas intrínsecas em grandes conjuntos de dados por meio de um algorítimo que indica como a máquina deve alterar seus parâmetros internos que são usados para calcular a representação em cada camada com base na representação da camada anterior.

6.1 Referencial Teórico para Abordagem de *Deep Learning*

Segundo LeCun et al. (2015), os métodos *Deep Learning* permitem que a máquina seja alimentada com dados brutos e descubra automaticamente os *inputs* relevantes para identificação de padrões ou classificação. O aprendizado *Deep Learning* é feito por meio de múltiplos níveis de representações dos dados obtidos pela composição de módulos simples, mas não lineares, que transformam a representação de um nível, sendo a representação do primeiro nível os próprios dados brutos, em representações mais abstratas e pertencentes a um nível mais alto. Em outras palavras, a cada camada, *inputs* não lineares são transformados para aprimorar a seletividade e invariância da representação, e os *outputs* são os mapeamentos em dimensões mais altas. Com essas sucessivas transformações,

diversas funções complexas e padrões intrínsecos nos dados podem ser aprendidos. Para questões de classificação, as representações das camadas mais altas trazem à tona aspectos de *inputs* que são importantes para a discriminação e suprimem aquelas variações irrelevantes. O grande diferencial do *Deep Learning* é que essas camadas de características não são arquitetadas por um indíviduo. Elas são aprendidas pela própria máquina no conjunto de dados (LeCun et al., 2015).

Alguns desafios de aprendizado de máquina, como por exemplo, reconhecimento de imagens, requerem que a função de decisão seja sensível a variações irrelevantes nos *inputs*, como por exemplo, variações na posição, orientação e iluminação do objeto, mas também sensível a variações mínimas essenciais para reconhecimento do objeto. Por exemplo, quando se deseja que a máquina reconheça imagens de Samoyeds, uma raça de cachorro muito parecida com lobos brancos, é preciso que a máquina consiga distringuir quais características são essenciais ao Samoyed e quais são essenciais aos lobos brancos, suprimindo aquelas características irrelevantes do ambiente. Em nível de *pixels*, imagens de dois Samoyeds em poses e ambientes diferentes podem ser claramente diferenciadas uma da outra, porém, a imagem de um Samoyed e outra imagem de um lobo branco, na mesma posição e em ambientes parecidos, podem ser muito similares entre si, conforme ilustram as Figuras 15 e 16.

Figura 15: Imagem de um Samoyed.
Fonte: Kimballstock[1]

[1] Diponível em: <http://www.kimballstock.com/results.asp?image=DOG%2001%20JE0159%2001> Acesso em Set. 2016.

Figura 16: Imagem de um lobo branco.
Fonte: Pt.Anawalls[2]

Classificadores lineares ou outros modelos de arquiteturas rasas que recebam como *inputs* os dados brutos de *pixels* geralmente não são capazes de distinguir as duas imagens com o Samoyed e o lobo branco e acabam colocando-os em uma mesma categoria. Classificadores de arquiteturas rasas são acompanhados de um dilema entre a seletividade e invariância, pois esses modelos conseguem, na maioria das vezes, selecionar bem as características importantes para a discriminação da imagem, mas não são invariantes para aspectos irrelevantes como a pose do animal no ambiente.

Segundo Gerlein et al. (2016), seleção de variáveis de *input* é um aspecto crítico na construção de modelos de aprendizado de máquina. O principal risco de um grande grupo de variáveis de *input* é a presença de informação redundante que dificulta o aprendizado e a classificação. Para contornar a informação redudante, os classificadores de arquiteturas rasas necessitam de um bom extrator de características, porém, torná-lo eficiente é extremamente oneroso e requer habilidades de engenharia computacional.

Por outro lado, por meio da abordagem *Deep Learning*, a construção de um extrator pode ser evitada já que o aprendizado das características essenciais pode ser feito automaticamente por meio de um procedimento genérico de aprendizado. Com múltiplas camadas não lineares, o modelo *Deep Learning* consegue criar uma função intrínseca aos *inputs* que é simultaneamente sensível a detalhes essenciais para a discriminação, como as diferenças físicas entre Samoyed e lobos brancos, e insensível a grandes, mas irrelevantes, variações de certos *inputs*, como variações

[2] Diponível em: <http://pt.anawalls.com/imagens-de-lobo-papeis-de-parede-do-cao-neve-floresta-vetor-fundos-materiais-mentindo/5401221960/>Acesso em Set. 2016.

no ambiente, posição e luminosidade (LeCun et al., 2015).

Há um debate crescente na literatura atual acerca do *trade-off* entre arquiteturas profundas e arquiteturas rasas. Arquiteturas profundas são geralmente mais difíceis de serem treinadas se comparadas a arquiteturas rasas, pois envolvem uma otimização não linear em alta dimensão e muitas heurísticas para o aprendizado que não garantem uma solução ótima global devido a não-convexidade do problema de otimização. Esses desafios do *Deep Learning* explicam o recente e contínuo apelo na literatura para o uso de SVMs. Ao contrário das arquiteturas profundas, os SVMs são treinados por meio de um problema de otimização convexo que fornece um ótimo global. Por outro lado, segundo Cho e Saul (2011), o SVM parece não ser tão eficiente na descoberta de importantes representações internas nos dados como acontece nas Redes Neurais com múltiplas camadas. Segundo este mesmo autor, outra motivação a favor de arquiteturas profundas é o potencial que possuem para combinar métodos de aprendizado supervisionado e não supervisionado.

A abordagem *Deep Learning* tem avançado significativamente na resolução de problemas que resistiram por muitos anos às melhores tentativas das comunidades de inteligência artificial (LeCun et al., 2015). Nos últimos anos, as Redes Neurais Profundas, isto é, Redes Neurais que incoroporam o *Deep Learning*, têm atraído muita atenção principalmente pelo fato de superarem a performance de métodos de aprendizado de máquina alternativos, como por exemplo, o SVM baseado em *Kernels.* Segundo Schmidhuber (2015), desde 2009, as Redes Neurais Profundas vêm ganhando várias competições internacionais de reconhecimento de padrões, alcançando em alguns campos de conhecimento uma capacidade de reconhecimento visual de padrões chamada de super-humana, isto é, a máquina consegue superar a capacidade humana de reconhecer padrões visuais.

Deep Learning por meio de Redes Neurais é aclamado hoje como o estado da arte em uma série de aplicações de reconhecimento de padrões, como processamento natural de linguagem, reconhecimento de discursos (Dahl, Mohamed, e Hinton, 2010), e classificação de imagens (Ciresan, Meier, Masci, Maria Gambardella, e Schmidhuber, 2011; Jarrett, Kavukcuoglu, Ranzato, e LeCun, 2009; Krizhevsky, Sutskever, e Hinton, 2012).

Tang (2013) propôs a substituição de uma das camadas da Rede Neural por um SVM linear para 3 problemas de classificação: reconhecimento de expressões

faciais, reconhecimento de vocábulos escritos à mão e classificação de objetos por imagens. Nas três situações abordadas, o *Deep Learning* com SVM gerou resultados superiores àqueles obtidos com a função de ativação *Softmax*. Tang (2013) sugere para pesquisas futuras a exploração de outras formulações de SVM para problemas de classificação e melhor entendimento de onde e quanto de ganho é obtido com a combinação *Deep Learning* e SVM.

Strobl e Visweswaran (2013) ressaltam que apesar dos métodos de *Deep Learning* aplicados em Redes Neurais Artificiais serem consagrados hoje como estado da arte em performance de aprendizado, as Rede Neurais não apresentam um desempenho de generalização satisfatório em um conjunto de dados pequeno. Por isso, Strobl e Visweswaran (2013) adotaram uma abordagem diferente, permitindo o aprendizado por meio de múltiplas camadas de *Kernels*. Como explicitado anteriormente nesta dissertação, os métodos de aprendizado baseados em *Kernels* têm obtido sucesso em uma variedade de tamanhos de amostras porque eles permitem que um classificador aprenda em um contexto de decisões complexas com apenas poucos parâmetros por meio da projeção dos dados em um espaço de alta dimensão. Strobl e Visweswaran (2013) provaram que, no modelo criado, a conjugação de *Deep Learning* com múltiplos *Kernels* permite que o limite superior do erro de generalização seja significativamente menor que o limite superior para modelos de *Deep Learning* em Redes Neurais, aumentando a acurácia do modelo em testes com amostras pequenas.

O trabalho de Cho e Saul (2009) é bastante relevante na literatura *Deep Learning*. Inicialmente eles propuseram um *Kernel* Arco-Cosseno por meio do qual é possível codificar representações internas semelhantes às de redes neurais de uma única camada.

Em um segundo momento, Cho e Saul (2010) mostram que por meio da mistura de *Kernels* Arco-Cossenos é possível criar um *Kernel* Arco-Cosseno Composto que imita o processo computacional em grandes redes neurais de múltiplas camadas. Cho e Saul (2010) argumentam que os resultados de misturas de *Kernels* são mais interessantes para *Kernels* da família Arco-Cosseno do que para *Kernels* polinomiais ou gaussianos, por exemplo. A composição de dois *Kernels* polinomiais é simplesmente outro *Kernel* polinomial, mas de grau maior. Da mesma forma, a composição de dois *Kernels* gaussianos é apenas outro *Kernel* Gaussiano, mas com variância diferente. Quando essas operações são feitas com *Kernels* Arco-

Cossenos, surgem novos *Kernels* Arco-Cossenos Compostos que não fazem parte da família de *Kernels* original.

Cho e Saul (2011) testaram o SVM com *Kernels* Arco-Cossenos Compostos utilizando 6 bases de dados diferentes, sendo a primeira um conjunto de dígitos escritos à mão, tradicionalmente usado para problemas de classificações, e as demais, conjuntos de imagens utilizadas também em problemas de *Deep Learning*.

Com os resultados obtidos, Cho e Saul (2011) notaram que SVMs construídos com *Kernels* Arco-Cossenos Compostos são mais simples de serem treinados, pois diferentemente dos SVMs com *Kernel* Gaussiano, eles não precisam de um parâmetro de amplitude, e diferentemente das redes neurais, eles não precisam resolver um problema de otimização não linear complexo ou procurar diversas possíveis arquiteturas.

Os resultados dos experimentos revelaram uma tendência interessante: os SVMs com *Kernels* Arco-Cossenos Compostos apresentaram melhor desempenho do que aqueles SVMs com *Kernels* Arco-Cossenos originais. Cho e Saul (2010) inferem que o processamento ao longo de várias camadas da rede neural imitada pelo *Kernel* contribuiu para melhores resultados e que estes *Kernels* Arco-Cossenos Compostos possuem algumas das vantagens tipicamente associadas ao *Deep Learning*.

Segundo Erhan et al. (2010), as lacunas no método SVM são justamente a lógica por traz de modelos que usam *Deep Learning*. As arquiteturas com *Deep Learning* têm a etapa de aprendizado não supervisionado anterior ao treinamento como uma grande vantagem e isso tem gerado melhores resultados de eficiência no treinamento e no desempenho da máquina. Diversos outros autores exploraram as conexões entre SVMs e *Deep Learning*, entre os quais pode-se citar Bengio e LeCun (2007) e Zhang e Zhao (2010).

Devido ao grande sucesso do *Deep Learning* e considerando que *Kernels* Arco-Cossenos Compostos incorporam algumas das vantagens tipicamente associadas a arquiteturas profundas (Cho e Saul, 2010), a terceira aplicação desta dissertação consistiu no uso do SVR com *Kernels* Arco-Cossenos Compostos para analisar os impactos desse aprimoramento no retorno dos portfólios formados com os ativos do S&P 100 selecionados pela máquina.

7 TERCEIRA APLICAÇÃO: FORMAÇÃO DE PORTFÓLIO POR MEIO DE SVR E *KERNELS* ARCO-COSSENO COMPOSTOS.

A fim de superar as fraquezas encontradas nas duas primeiras aplicações das Máquinas de Suporte Vetorial desta dissertação, buscou-se inserir um aprendizado profundo no modelo SVR da primeira aplicação. Isso foi feito com o uso do SVR e 9 diferentes *Kernels* Arco-Cosseno Compostos para formação de portfólios trimestrais com as ações indicadas pela máquina.

Os *Kernels* Arco-Cosseno Compostos propostos por Cho e Saul (2011) foram utilizados exclusivamente para problemas de reconhecimento de imagens. Esta dissertação inovou ao utilizar tais *Kernels* para aplicações em Finanças.

7.1 Metodologia *Kernels* Arco-Cossenos

Uma função *Kernel* pode ser vista como o que induz um mapeamento não linear dos *inputs* em um espaço de alta dimensão. O *Kernel* calcula o produto interno nesse espaço de alta dimensão:

(75)

$$k(x, y) = \Phi(x) \cdot \Phi(y)$$

Cho e Saul (2011) propuseram os *Kernels* Arco-Cosseno Compostos como cálculo do produto interno após sucessivas aplicações do mapeamento não linear induzido pela função básica do *Kernel* Arco-Cosseno original:

(76)

$$k^l(x, y) = \Phi\left(\Phi\left(\dots \Phi(x)\right)\right) \cdot \Phi\left(\Phi\left(\dots \Phi(y)\right)\right)$$

onde l indica o número de sucessivas aplicações do mapeamento não linear $\Phi(\cdot)$. Esses *Kernels* Arco-Cosseno Compostos, considerados como *Kernels* mais profundos são descritos por:

(77)

$$k^{(l+1)}(x,y) = \frac{1}{\pi}\left[k_n^l(x,x)k_n^{(l)}(y,y)\right]^{\frac{n}{2}} J_n(\theta_n^{(l)})$$

onde $\theta_n^{(l)}$ é o ângulo entre x e y no espaço de alta dimensão induzido por uma composição de l camadas e um *Kernel* de grau n.

A Figura 17 ilustra o triângulo formado pelos vetores de *input* x, y e pelo vetor da diferença entre os dois primeiros, $x - y$. θ, ξ e ψ são os ângulos desse triângulo. Cho e Saul (2011) observaram que dados de *input* que possuem um pequeno valor para θ são mapeados em pontos distantes entre si no espaço de característica. Inferiram assim quefo o uso de *Kernels* com maior sensibilidade ao deslocamento angular no espaço de *input* aumenta a acurácia de classificação da máquina por induzir uma melhor separação das diferentes, mas facilmente confundíveis, classes.

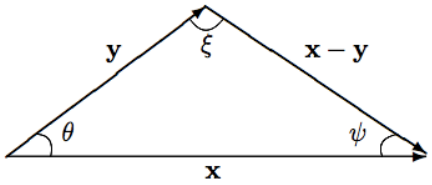

Figura 17: Triângulo formado pelos vetores de dados de *input* $\boldsymbol{x}$ e $\boldsymbol{y}$, e o vetor da diferença entre eles.
Fonte: Cho e Saul (2011).

Segundo Cho e Saul (2011), a função básica do *Kernel* Arco-Cosseno imita o procedimento computacional de uma rede neural de única camada e, portanto, a interação de vários *Kernels* Arco-Cossenos deve imitar o processo computacional de uma rede neural de múltiplas camadas cujos pesos seguem uma distribuição normal com $\mu = 0$ e $\sigma = 1$. Imitando diretamente o processo computacional em grandes redes neurais de múltiplas camadas, os métodos *Kernels* podem ser bastante úteis em *Deep Learning.* A Figura 18 ilustra uma rede neural de única camada onde $x \in \mathbb{R}^d$é transformado em $f \in \mathbb{R}^m$ por meio de uma mapeamento não linear. Já as Figuras 20 e 21 ilustram redes neurais de múltiplas camadas:

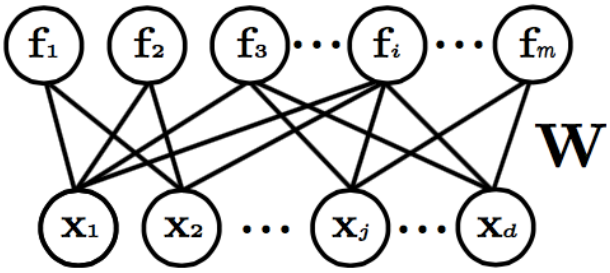

Figura 18: Rede neural de única camada.
Fonte: Cho e Saul (2011).

Sabe-se que as redes neurais dependem de uma função de ativação que atribui pesos e viéses em cada nó para que o classificador gere o *output*. Cho e Saul (2011) fizeram experimentos considerando 3 tipos diferentes de *Kernel* Arco-Cosseno Composto, cada um com um grau n diferente, sendo que este grau indica a função de ativação da rede neural reproduzida pelo *Kernel*.

Para $n = 0$, a função de ativação utilizada na camada é a *Step Function*. Já para $n = 1$, na camada utiliza-se a *Ramp Function*. Considerando $n = 2$, a função de ativação é a *Quarter-Pipe Function*. Nos experimentos, Cho e Saul (2011) também variaram o número de camadas da composição de *Kernels* com l variando de 1 a 6. A Figura 19 ilustra as diferentes funções de ativação.

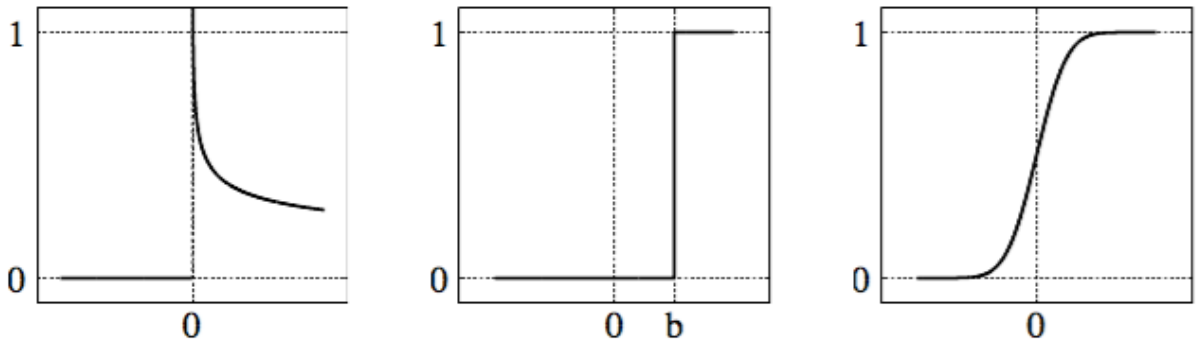

Figura 19: Funções de ativação.
Fonte: Cho e Saul (2011).

Na composição dos *Kernels* de Cho e Saul (2011) é possível variar o grau n em cada camada l e ela pode ser representada por:

(78)

$$k_{n_1,\ldots,n_l}(x,y) = \Phi_{n_l}\left(\ldots \Phi_{n_1}(x)\right) \cdot \Phi_{n_l}\left(\ldots \Phi_{n_1}(y)\right)$$

onde n denota a função de ativação na i-ésima camada da composição.

A Figura 20 mostra que diferentes mapeamentos não lineares, induzidos por diferentes *Kernels*, podem ser usados em diferentes camadas de uma rede neural de múltiplas camadas modelada pela composição de *Kernels* Arco-Cosseno Compostos.

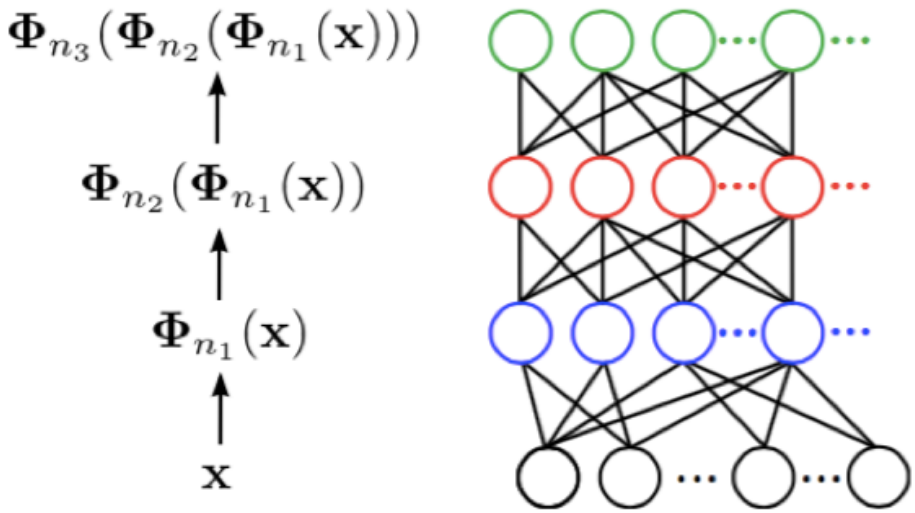

Figura 20: Rede neural de múltiplas camadas modelada pela composição de *Kernels* Arco-Cosseno Compostos.
Fonte: Cho e Saul (2011).

A Figura 21 ilustra uma rede neural de múltiplas camadas modelada pela multiplicação de *Kernels* Arco-Cossenos. $\Phi_{n,m}$ é a m-ésima característica induzida pelo *Kernel* Arco-Cosseno de grau n. As diferentes cores nas camadas do meio ilustram diferentes mapeamentos não lineares induzidos por tais *Kernels.*

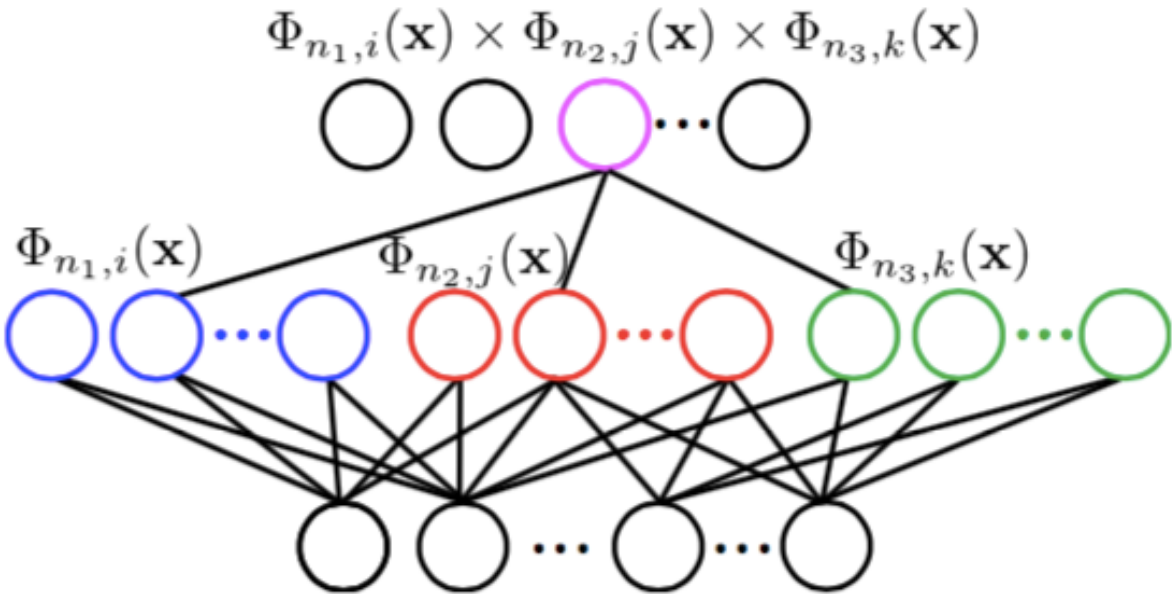

Figura 21: Rede neural de múltiplas camadas modelada pela multiplicação de *Kernels* Arco-Cossenos.
Fonte: Cho e Saul (2011).

Os experimentos realizados nesta dissertação limitaram-se aos *Kernels* Arco-Cosseno Compostos que imitam redes neurais de 2, 3 ou 4 camadas, isto é, os experimentos foram feitos para apenas 3 diferentes níveis de recursividade. Importante ressaltar também que foram testados apenas *Kernels* Arco-Cosseno Compostos com $n = 0$, $n = 1$ ou $n = 2$ na primeira camada, e $n = 1$ em todas as outras. Isto é, foram testados, *Kernels* que imitam redes com diferentes funções de ativação na primeira camada, mas *Ramp Function* constante em todas as camadas restantes. Essas limitações de número de camadas e funções de ativação foram feitas porque os resultados de Cho e Saul (2011) mostraram que quando o SVM possui $n = 1$ e $l > 1$, isto é, quando utiliza-se um *Kernel* Arco-Cosseno com a função de ativação *Ramp Function* e com um nível de recursividade maior que 1, ele supera os *benchmarks* em termos de acurácia.

7.2 Procedimentos de Coleta e Análise de Dados: SVR e Redes de Camadas Profundas

Assim como nas duas outras aplicações desta dissertação, utilizou-se a série histórica de retornos trimestrais e dados trimestrais de 127 indicadores financeiros referente às ações do S&P 100.

Para seleção dos parâmetros do modelo também se utilizou a Validação Cruzada para evitar *overtraining*. Assim como nas demais aplicações, a base de dados foi dividida em três conjuntos, sendo que os dois primeiros conjuntos foram escolhidos de forma aleatória, visando minimizar os efeitos macroeconômicos. O conjunto de treinamento foi constituído por 56,25% das observações, o de validação por 18,75% e o de teste por 25%.

O *grid* construído para definição dos parâmetros ótimos conteve uma sequência para o parâmetro C variando de 1 a 1000 e outra para o parâmetro σ, variando de 1^{-8} a 2.

A próxima etapa foi a construção das 9 Máquinas de Suporte Vetorial em sua variação de regressão (SVR). As máquinas distinguem-se pela função de

ativação na primeira camada e pelo número de camadas. Seguindo as sugestões de Cho e Saul (2011), foram testadas apenas máquinas com $l > 1$. Foram utilizados os parâmetros ótimos encontrados na etapa anterior de *grid search*. O Quadro 9 resume a estrutura das diferentes máquinas criadas.

Máquina	**Quantidade de Camadas** (l)	**Função de ativação da primeira camada**	**Função de ativação da segunda camada**	**Função de ativação da terceira camada**	**Função de ativação da quarta camada**
Máquina 1	2	*Step Function* $(n = 0)$	*Ramp Function* $(n = 1)$	-	-
Máquina 2	2	*Ramp Function* $(n = 1)$	*Ramp Function* $(n = 1)$	-	-
Máquina 3	2	*Quarter-Pipe Function* $(n = 2)$	*Ramp Function* $(n = 1)$	-	-
Máquina 4	3	*Step Function* $(n = 0)$	*Ramp Function* $(n = 1)$	*Ramp Function* $(n = 1)$	-
Máquina 5	3	*Ramp Function* $(n = 1)$	*Ramp Function* $(n = 1)$	*Ramp Function* $(n = 1)$	-
Máquina 6	3	*Quarter-Pipe Function* $(n = 2)$	*Ramp Function* $(n = 1)$	*Ramp Function* $(n = 1)$	-
Máquina 7	4	*Step Function* $(n = 0)$	*Ramp Function* $(n = 1)$	*Ramp Function* $(n = 1)$	*Ramp Function* $(n = 1)$
Máquina 8	4	*Ramp Function* $(n = 1)$	*Ramp Function* $(n = 1)$	*Ramp Function* $(n = 1)$	*Ramp Function* $(n = 1)$
Máquina 9	4	*Quarter-Pipe Function* $(n = 2)$	*Ramp Function* $(n = 1)$	*Ramp Function* $(n = 1)$	*Ramp Function* $(n = 1)$

Quadro 9: Característicacs das máquinas com *Kernels* Arco-Cosseno Compostos.
Fonte: Elaborado pela autora.

Os dados foram padronizados e para medir a qualidade das previsões utilizou-se Erro Quadrático Médio (EQM) descrito por:

(79)

$$EQM = \sum_{i=1}^{n} \frac{(y_i - \widehat{y_i})^2}{n}$$

onde, $(y_i - \widehat{y_i})^2$ representa os desvios quadrados entre os retornos observados e os retornos previstos pelo modelo, e n representa o tamanho da amostra.

As máquinas foram usadas para prever o retorno de cada ativo e esta previsão resultou em 9 novos *rankings* trimestrais de ativos. As ações foram então classificadas em duas classes. A Classe 1 $(y_i = +1)$ foi composta pelos 25% das ações com maiores retornos e a Classe 2 $(y_i = -1)$, pelas demais ações que representam 75%. Os ativos classificados como 1 foram selecionados para compor os portfólios trimestrais ponderados. O peso de cada ativo nos portfólios se deu pelo seu próprio retorno esperado, isto é, quanto maior o retorno esperado, maior o peso no portfólio.

Posteriormente, os retornos destes 9 portfólios foram comparados ao portfólio formado pelo SVR com o *Inverse Multiquadratic Kernel* que foi definido como *benchmark* porque, na primeira aplicação desta dissertação, quando foram testados 15 *Kernels* diferentes, ele apresentou o maior retorno e a segunda melhor relação retorno/risco.

7.3 Resultados e Discussão: SVR Deep Learning

Na terceira aplicação desta dissertação, o SVR foi construído tendo como insumos os resultados trimestrais de 127 indicadores financeiros e a previsão foi feita 9 vezes, variando o grau de cada *Kernel* e a quantidade de camadas da composição. O menor EQM encontrado foi $0{,}033281$ e refere-se ao *Kernel* Arco-Cosseno Composto 7 que tem $l = 4$, *Step Function* $(n = 0)$ como função de ativação na primeira camada e *Ramp Function* $(n = 1)$ em todas as demais. A Tabela 6 mostra os parâmetros ótimos e o EQM mínimo de cada *Kernel* Arco-Cosseno Composto.

Tabela 6: *Kernels Arco-Cosseno Compostos*, parâmetros ótimos e acurácias.

Máquina	**Quantidade de Camadas (l)**	**Função de ativação em cada camada**	**Parâmetros ótimos**	**EQM**
Kernel Arco-Cosseno Composto 1	2	$1°: (n = 0)$ $2°: (n = 1)$	$e = 0{,}3157895$ $C = 1$	0,0338888

Kernel Arco-Cosseno Composto 2	2	$1°: (n = 1)$ $2°: (n = 1)$	$e = 0{,}3157895$ $C = 1$	0,0600741
Kernel Arco-Cosseno Composto 3	2	$1°: (n = 2)$ $2°: (n = 1)$	$e = 1{,}157895$ $C = 1$	0,415232
Kernel Arco-Cosseno Composto 4	3	$1°: (n = 0)$ $2°: (n = 1)$ $3°: (n = 1)$	$e = 0{,}3157895$ $C = 1$	0,0335205
Kernel Arco-Cosseno Composto 5	3	$1°: (n = 1)$ $2°: (n = 1)$ $3°: (n = 1)$	$e = 0{,}3157895$ $C = 1$	0,0514889
Kernel Arco-Cosseno Composto 6	3	$1°: (n = 2)$ $2°: (n = 1)$ $3°: (n = 1)$	$e = 1{,}157895$ $C = 1$	0,403777
Kernel Arco-Cosseno Composto 7	4	$1°: (n = 0)$ $2°: (n = 1)$ $3°: (n = 1)$ $4°: (n = 1)$	$e = 0{,}3157895$ $C = 1$	0,033281
Kernel Arco-Cosseno Composto 8	4	$1°: (n = 1)$ $2°: (n = 1)$ $3°: (n = 1)$ $4°: (n = 1)$	$e = 0{,}3157895$ $C = 1$	0,0460738
Kernel Arco-Cosseno Composto 9	4	$1°: (n = 2)$ $2°: (n = 1)$ $3°: (n = 1)$ $4°: (n = 1)$	$e = 1{,}157895$ $C = 1$	0,402763

Fonte: Elaborado pela autora.

Formaram-se, com a seleção das melhores ações pelo SVR, 9 portfólios ponderados pelo retorno previsto. A Tabela 7 apresenta os retornos acumulados e trimestrais médios de cada portfólio, assim como o VaR e a relação retorno/risco de cada um deles.

Tabela 7: Resultados do SVM de Regressão com *Kernels* Arco-Cosseno Compostos.

Máquina	Retorno Acumulado	Retorno Trimestral Médio	VaR	Retorno/Risco
Portfólio SVR com *Kernel* Arco-Cosseno Composto 1	294,55%	6,89%	-12,90%	0,53

Portfólio SVR com *Kernel* Arco-Cosseno Composto 2	253,44%	6,44%	-13,54%	0,48
Portfólio SVR com *Kernel* Arco-Cosseno Composto 3	182,03%	5,16%	-14,58%	0,35
Portfólio SVR com *Kernel* Arco-Cosseno Composto 4	239,11%	6,06%	-12,49%	0,49
Portfólio SVR com *Kernel* Arco-Cosseno Composto 5	42,31%%	6,26%	-13,42%	0,47
Portfólio SVR com *Kernel* Arco-Cosseno Composto 6	173,68%	5,03%	-13,74%	0,37
Portfólio SVR com *Kernel* Arco-Cosseno Composto 7	258,65%%	6,33%	-12,06%	0,53
Portfólio SVR com *Kernel* Arco-Cosseno Composto 8	265,16%	6,61%	-14,49%	0,46
Portfólio SVR com *Kernel* Arco-Cosseno Composto 9	185,83%%	5,22%	-12,26%	0,43
Portfólio SVR com *Kernel Inverse Multiquadratic*	374,40%	6,49%	-6,87%	0,94

Fonte: Elaborado pela autora.

Dentre os 9 portfólios formados, o maior retorno acumulado foi o resultante daquele selecionado pelo SVR com o *Kernel* Arco-Cosseno Composto 1 de $l = 2$,

sendo na primeira camada, $n = 0$ e na segunda camada, $n = 1$. Este portfólio apresentou um retorno acumulado em 22 trimestres de $294{,}55\%$ e um VaR de $-12{,}90\%$. Já o portfólio de menor retorno foi aquele formado pelas ações apontadas pelo SVR com o *Kernel* Arco-Cosseno Composto 6, sendo seu retorno acumulado de $173{,}68\%$ e VaR de $-13{,}74\%$. O portfólio de *benchmark*, formado com as ações indicadas pelo SVR com o *Kernel Inverse Multiquadratic*, apresentou, por sua vez um retorno $374{,}40\%$.

Vale notar também que, ao contrário do que esperava-se, o aumento no número de camadas da rede neural imitada pelos *Kernel* Arco-Cosseno Compostos não aumentou o retorno do portfólio e também não contribuiu para minimização do risco. Também não foi observada nenhuma melhoria significativa com a alternância das funções de ativação da primeira camada da rede neural.

Por outro lado, analisando o EQM, observou-se uma tendência decrescente com o aumento do número de camadas, isto é, a máquina tornou-se mais acurada com redes neurais maiores. Em especial, aqueles *Kernel*s Arco-Cosseno Compostos que imitam redes redes neurais com a *Step Function* na primeira camada $(n = 0)$ apresentaram os três menores EQM.

Todos os 9 portfólios formados com *Kernels* Arco-Cosseno Compostos apresentaram medidas de risco elevadas, se comparadas àquelas obtidas na primeira aplicação desta dissertação. O *Kernel Inverse Multiquadratic* apresentou $-6{,}87\%$ de risco, enquanto a menor medida de risco apresentada entre os *Kernels* Arco-Cosseno Compostos foi $-12{,}06\%$ e refere-se ao portfólio formado pelo *Kernel* Arco-Cosseno Composto 7 que também apresentou o menor EQM e o terceiro maior retorno acumulado, de $258{,}65\%$.

Os portfólios formados pelo *Kernel* Arco-Cosseno Composto 1 e pelo *Kernel* Arco-Cosseno Composto 7 apresentaram a melhor relação retorno/risco com o valor de $0{,}53$, enquanto o *benchmark* apresentou o valor de $0{,}94$ para esta medida. Isso significa que para cada 1% de risco nos portfólios formados pelos *Kernel* Arco-Cosseno Composto 1 e pelo Arco-Cosseno Composto 7, há $0{,}53\%$ de retorno. Já no *benchmark*, para cada 1% de risco, há $0{,}94\%$ de retorno. Como essa razão relaciona a maximização do retorno e a minimização do risco, quanto maior o valor do quociente, melhor é a opção de investimento e fica evidente que o portfólio de *benchmark* foi bastante superior neste aspecto. Ressalta-se novamente que essa

relação retorno/risco vale apenas para 5% dos trimestres, visto que o VaR representa justamente a maior perda que se pode ter em um trimestre com 95% de confiança e que, em média, os retornos dos portfólios são os explicitados na Tabela 7. Apesar dessa restrição, a relação retorno/risco auxilia na comparação entre os portfólios.

Figura 22 ilustra comparação entre o retorno trimestral médio e o risco de cada portfólio aqui formado. Já a Figura 23 ilustra o desempenho do retorno acumulado dos portfólios ao longo do período de teste.

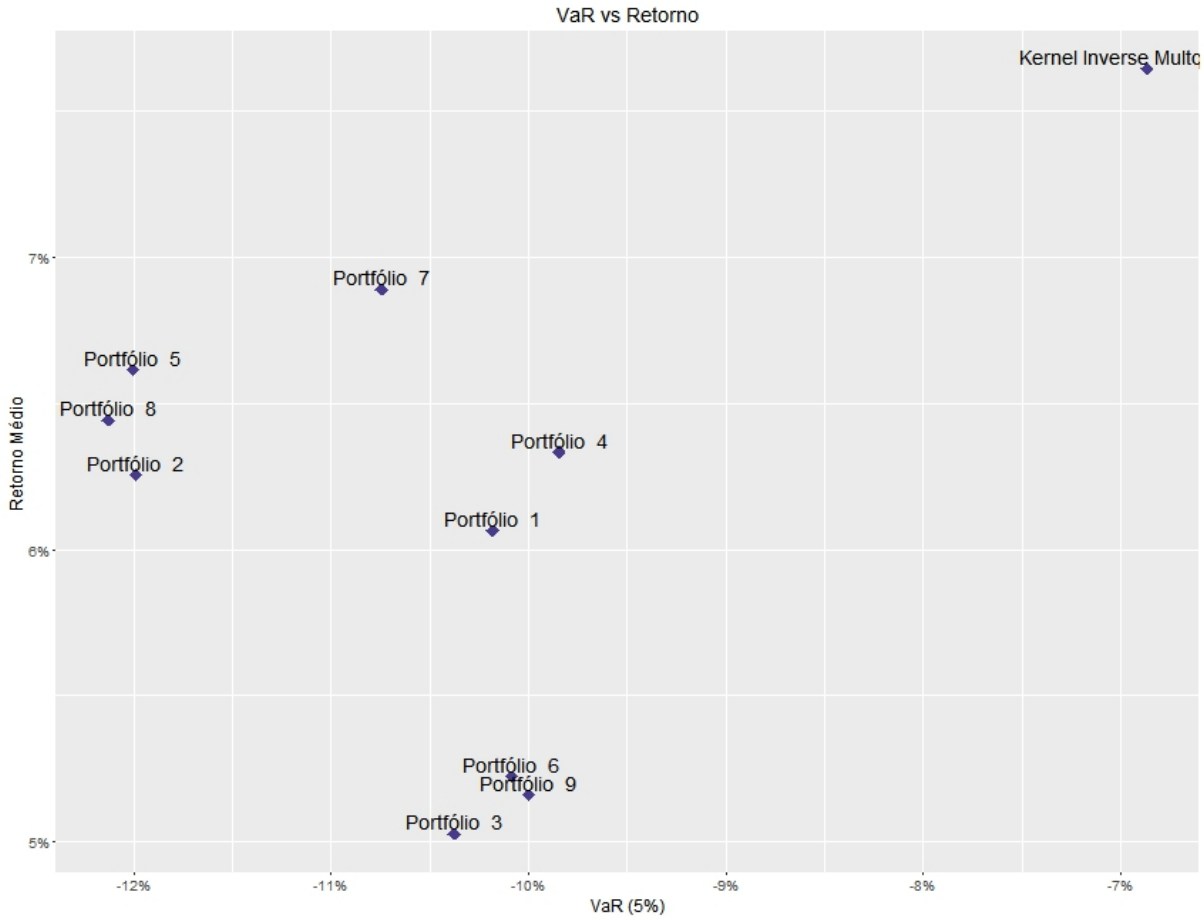

Figura 22: Comparação do retorno trimestral médio e risco de cada portfólio formado com *Kernel* Arco-Cosseno Composto e do *benchmark.*
Fonte: Elaborado pela autora.

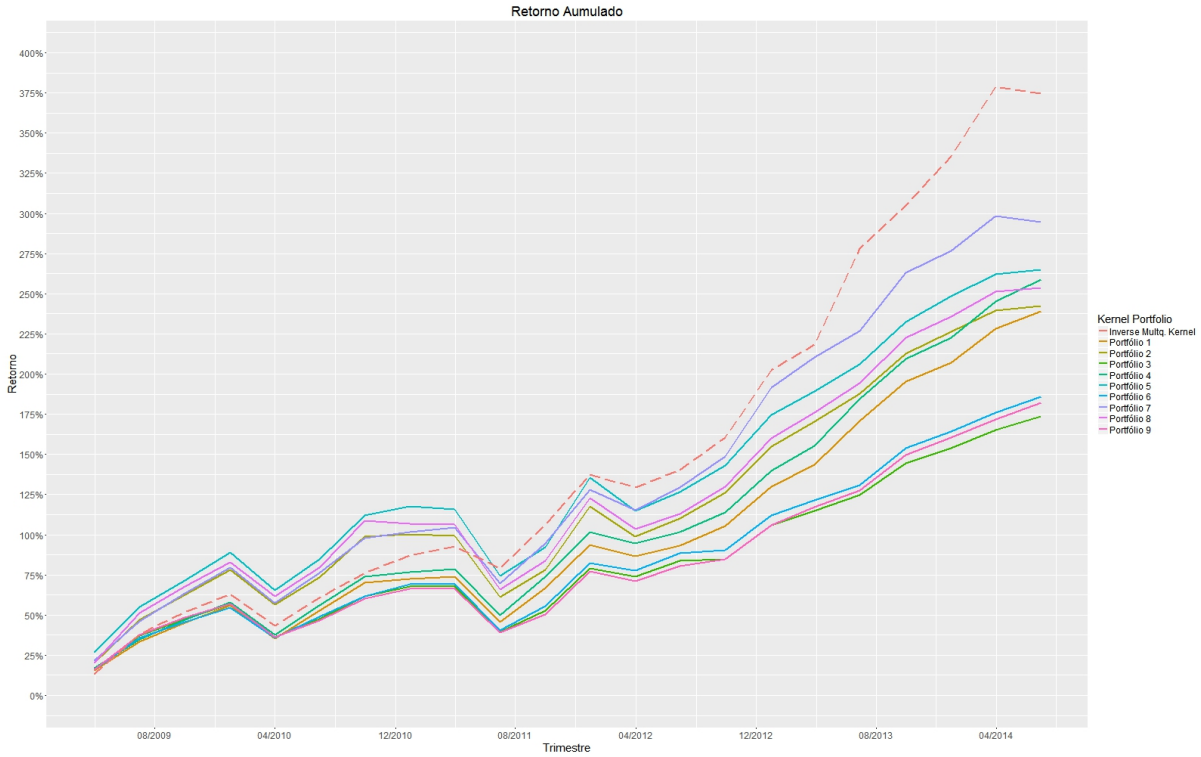

Figura 23: Desempenho dos retornos acumulados dos portfólios formados com *Kernels* Arco-Cosseno Compostos e do *benchmark.*
Fonte: Elaborado pela autora.

Com os resultados apresentados, fica evidente que os portfólios formados pelos *Kernels* Arco-Cosseno Compostos não superaram o *benchmark* em termos de retorno acumulado e nem em termos de risco do portfólio. O *Kernel Inverse Multiquadratic* possibilitou um portfólio com retorno acumulado 27,11% maior e risco 87,75% menor, se comparado ao portfólio obtido com o melhor *Kernel* Arco-Cosseno Composto. Com os resultados obtidos, o teste de White (2000) não foi aplicado, visto que na melhor das hipóteses, o retorno dos portfólios com os *Kernel* Arco-Cosseno Compostos e o retorno do *benchmark* seriam iguais.

8 CONCLUSÕES E RECOMENDAÇÕES

Esta pesquisa teve por objetivo verificar se o uso do Máquinas de Suporte Vetorial na formação de portfólios de fato contribui para que o retorno seja superior ao Mercado. Para tal, foram feitas três aplicações das Máquinas de Suporte Vetorial para a formação de portfólios trimestrais. Na primeira aplicação utilizou-se o SVM de Regressão (SVR) com 15 *Kernels* diferentes. Na segunda aplicação, o SVM de Classificação. Na terceira aplicação, usou-se o SVR novamente, mas com 9 *Kernels* Arco-Cosseno Compostos.

Na primeira aplicação da pesquisa, os dados do S&P 100 foram utilizados para construir 15 máquinas diferentes por meio do SVR, a fim de analisar os resultados com o uso de 15 *Kernels* distintos. Esta pesquisa inovou ao separar os conjuntos de treinamento e validação de forma aleatória para minimizar efeitos macroeconômicos que a divisão temporal poderia acarretar. O *Linear Kernel* foi o *Kernel* que apresentou maior acurácia com valor de $37{,}81\%$.

Observando os resultados desta aplicação fica evidente que 13 *Kernels,* dentre os 15 testados, superaram o *benchmark* de mercado em termos de retorno acumulado. Já em termos de risco, 11 deles apresentaram uma medida de VaR inferior ao do *benchmark.* O maior retorno acumulado ao longo dos 22 trimestres de teste foi o resultante do portfólio selecionado pelo SVR com o *Inverse Multiquadratic Kernel*. Este portfólio apresentou um retorno acumulado de $374{,}40\%$ e um VaR de $-6{,}87\%$. Já o portfólio de menor retorno foi aquele formado pelas ações apontadas pelo SVR com o *Multiquadratic Kernel,* sendo seu retorno acumulado de $130{,}62\%$ e VaR de $-11{,}95\%$. A menor medida de risco apresentada foi $-5{,}99\%$ e refere-se ao portfólio formado pelo *Exponential Kernel* que também apresentou um retorno acumulado bastante interessante de $374{,}40\%$. O *Wavelet Kernel,* por sua vez, selecionou um portfólio com o maior VaR, $-12{,}72\%$, e retorno de $222{,}82\%$.

Na segunda aplicação da pesquisa, o SVM de Classificação foi construído com o *Kernel* Gaussiano para classificação das ações por uma nova perspectiva. A classificação não foi feita com base no *ranking* dos retornos, mas sim, na probabilidade da ação pertencer à Classe 1, classe das boas ações. Além disso, para amenizar o desbalanceamento da base, o custo do erro foi ponderado de forma desigual em cada uma das classes. Com os parâmetros ótimos $\mathrm{C} = 1$ e $\sigma = 0{,}0042$,

o SVM apresentou uma acurácia de $81{,}82\%$, o que é um resultado bastante satisfatório.

As ações que ficaram dentro do conjunto dos 25% de maiores probabilidades foram selecionadas para formar portfólios trimestrais e o retornos destes foram comparados com o *benchmark* de mercado. O retorno acumulado do portfólio selecionado pelo SVM foi de $233{,}52\%$ em 22 trimestres com VaR de $-10{,}68\%$. Neste mesmo período, as ações do S&P 100 apresentaram uma rentabilidade média acumulada de $192{,}65\%$ e VaR igual a $-11{,}74\%$.

Na terceira aplicação desta dissertação, o SVR foi utilizado com 9 diferentes tipos de *Kernel* Arco-Cosseno Compostos. Esses *Kernels* Compostos imitam o processo computacional em grandes redes neurais de múltiplas camadas (Cho e Saul, 2011). Eles foram utilizados com o objetivo de aprimorar a previsão do retorno das ações e, posteriormente, sua classificação como boas ou ruins, por meio da inserção do aprendizado profundo no método SVR. Como *benchmark*, utilizou-se o portfólio formado pelo SVR com o *Kernel Inverse Multiquadratic.*

O portfólio de maior retorno acumulado foi aquele formado pelo SVR com o *Kernel* Arco-Cosseno Composto 1. Este *Kernel* imita uma rede neural de 2 camadas, sendo que na primeira a função de ativação foi a *Step Function* e na segunda, a *Ramp Function.* O retorno acumulado deste portfólio em 22 trimestres foi de $294{,}55\%$ e o VaR foi $-12{,}90\%$. Já o portfólio de menor retorno foi aquele formado pelas ações apontadas pelo SVR com o *Kernel* Arco-Cosseno Composto 6*,* sendo seu retorno acumulado de $173{,}68\%$ e VaR de $-13{,}74\%$. O portfólio de *benchmark*, apresentou*,* por sua vez um retorno $374{,}40\%$ e VaR de $-6{,}87\%$.

Nota-se que nem o melhor portfólio formado com *Kernels* Arco-Cosseno Compostos conseguiu superar o portfólio formado pelo SVR com o *Kernel Inverse Multiquadratic.* Este mostrou-se superior em termos de retorno, em termos de risco, e, consequentemente, também superior na relação retorno/risco*.*

Apesar das limitações, os resultados desta pesquisa corroboram a hipótese de superioridade do método inovador das Máquinas de Suporte Vetorial na formação de portfólios, caracterizado pela construção de um hiperplano pela implementação do Princípio da Minimização do Risco Estrutural, o qual procura minimizar o limite superior do erro de generalização, em vez de minimizar apenas o erro do processo de estimação.

Cho e Saul (2011) originalmente propuseram os *Kernels* Arco-Cosseno Compostos para problemas de reconhecimento de imagens e, nesta dissertação, eles foram utilizados para aplicações em Finanças. Uma das possíveis causas para os resultados não satisfatórios aqui encontrados é a ausência de uma etapa de seleção de características prévia ao uso do SVR. Indicadores não relevantes podem ter induzido um alto nível de ruído e consequentemente, atrapalhado a previsão. Portanto, para inserção de fato da abordagem de *Deep Learning* no método de seleção de ações proposto nesta dissertação, sugere-se como estudo futuro o uso de uma estrutura hierárquica profunda como extrator de características para que sejam colocados como *inputs* no SVM apenas aqueles indicadores de fato relevantes para identificação de boas ações. Cai, Hu, e Lin (2012) por exemplo, implementaram uma *Restricted Boltzmann Machine* (RBM) para extrair características de indicadores técnicos que foram usados como *inputs* em um SVR.

Atualmente, na literatura não há estudos sobre um *Kernel* que imite uma rede neural indicada especificamente para problemas de previsão de séries temporais. Dessa forma, uma proposta para estudos futuros é a construção de um *Kernel* que imite o processamento de uma *Long Short Term Memory Network* (LSTM) que é um tipo especial de *Recurrent Neural Network*, capaz de aprender dependências de longo prazo nos dados (Hochreiter e Schmidhuber, 1997) e relembrar informação por longos perídos de tempo.

Muitas lacunas na abordagem de formação de portfólios por meio das Máquinas de Suporte Vetorial ainda podem ser exploradas. Para estudo futuros, sugere-se também: aprimoramento da forma de definição dos parâmetros ótimos; definição dos *inputs* mais adequados para o modelo; uso de indicadores contábeis relativizados; uso de indicadores contábeis defasados em mais de um período; adequação da medida de acurácia utilizada; uso de outra estratégia para imputação de dados omissos; uso apenas dos dados de cauda; mudança do critério de classificação das classes para outro que não seja as 25% de maior retorno; aprofundamentos sobre a ponderação do custo do erro a fim de entender até que ponto ela representa uma vantagem real para a máquina; construção de diferentes máquinas de suporte vetorial de acordo com os setores industriais nos quais as ações estão inseridas; formação e comparação de portfólios com ações de bolsas de diferentes países; aprofundamentos sobre as características do *Inverse Multiquadratic Kernel* para justificar seu desempenho superior aos demais *Kernels*

na primeira aplicação desta pesquisa. Além disso, para tornar essa estratégia de seleção de ações mais aplicável às condições reais do mercado financeiro, sugere-se considerar custos de transação; definir uma janela temporal para operacionalizar as mudanças de um portfólio para o outro e testar o modelo retirando aquelas ações do S&P 100 já conhecidas no mercado pelo seu desempenho em termos de retorno. Aprofundamentos como estes poderão elevar em grande medida a acurácia de classificação e o retorno gerado pelo portfólio, contribuindo assim, para o aperfeiçoamento do método.

ANEXO A: Código de programação em R

Segue abaixo o código base de programação utilizado para análise empírica do método de formação de portfólios proposto.

```
memory.size(max = TRUE)

install.packages(c("kernlab","FSelector","mlbench","foreach","doParallel","doSNOW","rgl"))

{

Base<-read.csv("C:\\Users\\Sarah Sabino\\Dropbox\\SVM\\Base de Artigos e Estudos\\Artigo KERNELS\\Kernels\\Base.csv")

Base<-Base[,2:ncol(Base)]

##     Tratamento de dados perdidos

## filtro de dias que ouveram dados para cada empresa.

BBase<-c()

Tikers<-unique(Base$Tik)

for(i in 1:length(Tikers))

{

  t<-subset(Base,Base$Tik == Tikers[i])

  t<-subset(t,as.Date(t$Data)>0)

  t<-subset(t,t$PX_LAST>0)

  t<-subset(t,as.Date(t$Data)<"2014-09-30")

  BBase<-rbind(BBase,t)

}

#####         variaveis com porucas observações.

x<-1

coluna<-c()

for (i in 1:ncol(BBase))

{
```

```
  if(sum(is.na(BBase[,i]))>=(nrow(BBase)*0.85))
  {
    coluna[x]<-i
    x<-x+1
  }
}
coluna
BBase<-BBase[,-coluna]

################################        retirada de NA
BBase
for(i in 1:ncol(BBase))
{
  BBase[which(is.na(BBase[,i])==TRUE),i]<-mean(BBase[,i],na.rm=TRUE)
}

BaseTotal<-BBase
BaseTotal$Data<-as.Date(BaseTotal$Data)

#################   Clasificação por 25%
quantil<-by(BaseTotal$Retorno_t,BaseTotal$Data,quantile,scale=FALSE,na.rm=T)
quantil<-as.table(quantil)
quantil<-data.frame(do.call("rbind",quantil))
quantil$Data<-as.Date(rownames(quantil))
BaseTotal<-merge(BaseTotal,quantil,by="Data",all=T)

######            Regra de clasificação
```

```
BaseTotal$classe<-ifelse(BaseTotal$Retorno_t>=BaseTotal$X75.,1,0)

table(BaseTotal$classe)


BBaseTotal<-BaseTotal

########################  Filtro de variaveis irrelevantes. #######

BBaseTotal$Data<-as.Date(BaseTotal$Data)

#######                 Organisar ordem de data

BBaseTotal<-BBaseTotal[order(BBaseTotal$Data, decreasing=FALSE),]

unique(BBaseTotal$Data)

##############              Separação de bases

###           sequencia aleatoria para divisão da base de dados   ##################

set.seed(40)

linhasDEtreino<-sample(1:(nrow(BBaseTotal)*0.75),
size=(length((1:(nrow(BBaseTotal)*0.75)))*0.7), replace = FALSE)

treinamento<-BBaseTotal[linhasDEtreino,]

Validacao<-BBaseTotal[(1:(nrow(BBaseTotal)*0.75)),][-linhasDEtreino,]

teste<-BBaseTotal[((nrow(BBaseTotal)*0.75):nrow(BBaseTotal)),]


nrow(Validacao)+nrow(treinamento)+nrow(teste)


DDAta<-as.data.frame(unique(BBaseTotal$Data))


treinamento$Data<-as.Date(treinamento$Data)

Validacao$Data<-as.Date(Validacao$Data)

teste$Data<-as.Date(teste$Data)


#########################    NORMALIZAÇÃO DOS DADOS
#########################################################

##                    Ajuste para svm
```

```
##                    Aplicação de scala

treino<-(treinamento[1:(nrow(treinamento)-1),-c(1,97,102,101,100,99,98,103)])
valida<-(Validacao[,-c(1,97,102,101,100,99,98,103)])
testepredi<-(teste[,-c(1,97,102,101,100,99,98,103)])

colnames(treino[,])

treino[1:(ncol(treino))]<-scale(treino)
valida[1:(ncol(valida))]<-scale(valida)
testepredi[1:(ncol(testepredi))]<-scale(testepredi)
##

########          seleÃ§Ã£o de dados variaveis  ######################
BBaseTotal2<-BBaseTotal[,-c(1,97,102,101,100,99,98,103)]
colnames(BBaseTotal2)
##  base
library(FSelector)
library(mlbench)
weights <- chi.squared(Retorno_t ~., BBaseTotal2)
print(weights)
subset <- cutoff.k(weights, 25)
f <- as.simple.formula(subset, "Retorno_t")
print(f)
#########   Opções de metodos
#weights <- information.gain(Retorno_t~., BBaseTotal2)
#print(weights)
```

```
#subset <- cutoff.k.percent(weights, 0.70)

#f <- as.simple.formula(subset, "Retorno_t")

#print(f)

#########################################################################################################

#########################################      CRIAÇÃO DA SEQUENCIAS DO SVM
#########################################

library(kernlab)

#Carrega os pacotes necessários para realizar o paralelismo

library(foreach)

library(doParallel)

library(doSNOW)

####              ####       Configuração de nucleos para Otimização

#Checa quantos núcleos existem

ncl<-detectCores()

ncl

#Registra os clusters a serem utilizados

cl <- makeCluster(ncl)

registerDoParallel(cl)

}

#########################################################################################################

        sigma<- seq(0.000000000000001,2,length.out=30)

        C<-seq(1,1000,length.out=20)

        parametro<-as.data.frame(expand.grid(sigma=sigma,C=C))


performace<-as.data.frame(rep(0.0,nrow(parametro)))

#########################################################################################################

ptm <- proc.time()
```

```
acuracia <-foreach(i=(1:nrow(parametro)),.packages="kernlab", .combine='rbind') %dopar% {

##############################################################################
##################################################

        sigma<-parametro[i,1]

        C<-parametro[i,2]

        ## gaussiano

        Kernel<-function(x,y)

        {

          res<-exp(-sigma*(sum((x-y)^2)))

          return(res)

        }

        class(Kernel) <- "kernel"


   svm<-ksvm (f,data=treino,type="eps-svr",C=C,kernel=Kernel,scaled=T)

##############################################################################
##################################################

{

##       probabilidade

  Validacao<-Validacao[,1:103]

  Validacao$Probabilidade<-predict(svm,valida)

  ##              _____________

  quantil<-by(Validacao$Probabilidade,Validacao$Data,quantile,scale=false,na.rm=T)

  quantil<-as.table(quantil)

  quantil<-data.frame(do.call("rbind",quantil))

  quantil$Data<-as.Date(rownames(quantil))

  Validacao<-merge(Validacao,quantil,by="Data",all=T)

  quantil<-c()

  ###      cria coluna de classe
```

```
  Validacao$Seleciona<-ifelse(Validacao$Probabilidade>=Validacao[,(108)],1,0)

  table(Validacao$Seleciona)

  ypred<-Validacao$Seleciona

  ytest<-as.numeric(Validacao$classe) #isso eh a classificacao observada

  CXP<-as.data.frame(as.numeric(ypred))

  CXP$ytest<-ytest

  CXP$soma<-CXP[,1]+CXP[,2]

  CXP$dif<-CXP[,1]-CXP[,2]


  ## erro ceral

  performace[i,1]<-

    ((sum(CXP$soma==2))/sum(CXP[,1]))

   # ((sum(CXP$dif==0))/length(CXP[,1]))


  #sqrt(mean((valida$Retorno_t-Validacao$Probabilidade)^2))

}

}

proc.time() - ptm

parametro$performace<-unname(acuracia)

stopCluster(cl)

save.image("C:\\Users\\Sarah Sabino\\Dropbox\\SVM\\Base de Artigos e Estudos\\Artigo
KERNELS\\Kernels\\Gaussiano.RData")


{

##seleção de parametros

melhorparametro<-
parametro[which(parametro$performace==max(parametro$performace))[1],]

melhorparametro

}
```

```
##############################################################################
##################################################

        sigma<-as.numeric(melhorparametro[1])

        C<-as.numeric(melhorparametro[2])

        ## gaussiano

        Kernel<-function(x,y)

        {

          res<-exp(-sigma*(sum((x-y)^2)))

          return(res)

        }

        class(Kernel) <- "kernel"

        svm<-ksvm (f,data=treino,type="eps-svr",C=C,kernel=Kernel,scaled=T)

##############################################################################
##################################################

{

library(rgl)

y<-parametro$sigma

x<-parametro$C

z<-parametro$performace

plot3d(x, y, z, col="red", size=3,type = "p",xlab="C", ylab="Sigma")




##############################################################################
##################################################

##############################################################################
##################################################

########################################      SELEÇÃO DE ATIVOS EM TESTE
##########################################

teste<-teste[,1:103]
```

```
Probabilidade<-predict(svm,testepredi)#,type="probabilities")

teste$Seleciona<-c()

teste$Probabilidade<-as.numeric(Probabilidade)#[,2])

#_____#_____#_____#_____#_____#_____

##   escolha de quartil

quantil<-
by(teste$Probabilidade,teste$Data,quantile,scale=false,na.rm=T,prob=seq(0,1,length=11))

quantil<-by(teste$Probabilidade,teste$Data,quantile,scale=false,na.rm=T)


quantil<-as.table(quantil)

quantil<-data.frame(do.call("rbind",quantil))

quantil$Data<-as.Date(rownames(quantil))

teste<-merge(teste,quantil,by="Data",all=T)

quantil<-c()

colnames(teste)

#cria coluna de classe

teste$Seleciona<-ifelse(teste$Probabilidade>=teste[,108],1,0)

table(teste$Seleciona)

##########################################################################
##################################################

##########################################################################
##################################################

ypred<-teste$Seleciona

ytest<-as.numeric(teste$classe) #isso eh a classificacao observada

CXP<-as.data.frame(as.numeric(ypred))

CXP$ytest<-ytest

CXP$soma<-CXP[,1]+CXP[,2]

CXP$dif<-CXP[,1]-CXP[,2]
```

```
Acerto_prev<-((sum(CXP$soma==2))/sum(CXP[,1]))

positivos<-sum(teste[which(teste$Seleciona==1),"Retorno_t"]>0)/sum(teste$Seleciona==1)


#######################################      APURAÇÃO DOS RESULTADOS
##########################################

DData<-unique(as.Date(teste$Data))

length(DData)


RetornoP<-c()

conta<-1

ativosSele<-as.data.frame(c())

retox<-as.data.frame(c())

for(i in 1:length(DData))

{

  tri<-subset(teste, teste$Data==DData[i])

  tri

  table(tri$Seleciona)

  #RetornoP[conta]<-(mean(as.numeric(tri[which(tri$Seleciona==1),"Retorno_t"])))

  RetornoP[conta]<-
(sum(as.numeric(tri[which(tri$Seleciona==1),"Retorno_t"])*(tri[which(tri$Seleciona==1),"Prob
abilidade"])/sum(tri[which(tri$Seleciona==1),"Probabilidade"])))

  ativosSele[1:length(tri[which(tri[,"Seleciona"]==1),2]),conta]<-
c(as.character(tri[which(tri[,"Seleciona"]==1),"Tik"]))

  retox[1:length(tri[which(tri[,"Seleciona"]==1),2]),conta]<-
c((tri[which(tri[,"Seleciona"]==1),"Retorno_t"]))

  conta<-conta+1

}


##              quais ativos foram selecionados em cada semestre.
```

```
ativosSele<-as.data.frame(ativosSele)

retox<-as.data.frame(retox)

#Quantos ativos passam pela carteira

table(teste$Seleciona)

##       Retorno por trimestre do portifolio

RetornoP<-as.data.frame(RetornoP)

RetornoP$DData<-as.Date(DData)

###           Retorno acumlado

### Calculo do retorno acumulado da estrategia de seleção de ativos.

Rac<-c()

for(i in 1:nrow(RetornoP))

{

  Rac[i]<-(prod((RetornoP[1:i,1])+1))-1

}

RetornoP$Rac<-Rac

RetornoP

###

RetornoP_Gaussiano<-RetornoP

Acerto_prev_Gaussiano<-Acerto_prev

ativosSele_Gaussiano<-ativosSele

retox_Gaussiano<-retox

positivos_Gaussiano<-positivos


write.csv2(RetornoP_Gaussiano,"C:\\Users\\Sarah Sabino\\Google Drive\\Artigo 2\\Kernels
1\\concl\\RetornoP_Gaussiano.csv")

write.csv2(Acerto_prev_Gaussiano,"C:\\Users\\ Sarah Sabino \\Google Drive\\Artigo
2\\Kernels 1\\concl\\Acerto_prev_Gaussiano.csv")

write.csv2(ativosSele_Gaussiano,"C:\\Users\\ Sarah Sabino \\Google Drive\\Artigo 2\\Kernels
1\\concl\\ativosSele_Gaussiano.csv")
```

```
write.csv2(retox_Gaussiano,"C:\\Users\\ Sarah Sabino \\Google Drive\\Artigo 2\\Kernels 1\\concl\\retox_Gaussiano.csv")

write.csv2(positivos_Gaussiano,"C:\\Users\\ Sarah Sabino \\Google Drive\\Artigo 2\\Kernels 1\\concl\\positivos_Gaussiano.csv")


###   Volatilidade

sd(RetornoP[,1])

mean(RetornoP[,1])

##---------------------------------------  Retorno trimestral do mercado ----------------------------------------

### ----------------------------  Media das ações comparadas ------------------------------------------------------

RetornoMercado<-c()

conta<-1

retoxMercado<-as.data.frame(c())

for(i in 1:length(DData))

{

  tri<-subset(teste, teste$Data==DData[i])

  tri

  RetornoMercado[conta]<-(mean(as.numeric(tri[,"Retorno_t"])))

  retoxMercado[1:length(tri[,2]),conta]<-c(as.character(tri[,"Retorno_t"]))

  conta<-conta+1

}


##               quais ativos foram selecionados em cada semestre.

ativosSele<-as.data.frame(ativosSele)

retoxMercado<-as.data.frame(retoxMercado)

#Quantos ativos passam pela carteira

table(teste$Seleciona)

##   Retorno por trimestre do portifolio
```

```
RetornoMercado<-as.data.frame(RetornoMercado)

RetornoMercado$DData<-DData[1:length(DData)]

###           Retorno acumlado

### Calculo do retorno acumulado da estrategia de seleção de ativos.


RacMercado<-c()

for(i in 1:nrow(RetornoMercado))

{

  RacMercado[i]<-(prod((RetornoMercado[1:i,1])+1))-1

}

RetornoMercado$RacMercado<-RacMercado

RetornoMercado

###   Volatilidade

sd(RetornoMercado[,1])

mean(RetornoMercado[,1])

##-----------------------------------------------------------------------------------------------------------------------
----------------------


##-----------------------------------------------------------------------------------------------------------------------
----------------------

###-----------------------------------------  Comparação de resultados ----------------------------------
----------------------

Conclusao<-as.data.frame(DData)

Conclusao$Retorno_Portifolio<-as.numeric(RetornoP$RetornoP)

Conclusao$Retorno_Mercado<-as.numeric(RetornoMercado$RetornoMercado)


Conclusao$Retorno_AC_Portifolio<-as.numeric(RetornoP$Rac)

Conclusao$Retorno_AC_Mercado<-as.numeric(RetornoMercado$RacMercado)
```

```
Conclusao

}

################################################################################
################

write.csv2(Conclusao[,c(1:3)],"C:\\Users\\aluno\\Desktop\\Kernels\\Gaussiano.csv")

################################################################################
##############

{

##        acrecentar metricas.

#          Graficos

library(ggplot2)

##  ------------------------------  Grefico retorno acumulado----------------------------

DATA<-(as.character(Conclusao[,1]))

DATA<-Conclusao[,1]


ggplot(Conclusao,aes(x=DATA))+

  ggtitle("Cumulative Return")+

  labs(x="Time",y=" Return % ")+

  geom_line(aes(y =Conclusao$Retorno_AC_Portifolio*100, colour ="SVM" )) +

  geom_line(aes(y =Conclusao$Retorno_AC_Mercado*100, colour = "Market "))+

  labs(colour = "")+ scale_x_date(breaks = "8 month", minor_breaks = "month")


ggplot(Conclusao,aes(x=DATA))+

  ggtitle("Cumulative Return")+

  labs(x="Time",y=" Return % ")+

  geom_line(aes(y =Conclusao$Retorno_Portifolio*100, colour ="SVM" )) +

  geom_line(aes(y =Conclusao$Retorno_Mercado*100, colour = "Market "))+
```

```
  labs(colour = "")+ scale_x_date(breaks = "8 month", minor_breaks = "month")

###-----------------------------------------------------------------------------

###-----------------------------------------------------------------------------

##      analise de risco

# acrecentar mais metricas


library(PerformanceAnalytics)

library(robustbase)

R<-(as.ts(Conclusao$Retorno_Portifolio))

VaRPort<-VaR(R=R,p=0.95,method=c("historical"),clean = c("geltner"))


R<-as.ts((Conclusao$Retorno_Mercado))

VaRMerc<-VaR(R=R,p=0.95,method=c("historical"),clean = c("geltner"))


VaRPort

VaRMerc

}
```

REFERÊNCIAS

Abu-Mostafa, Y. S., & Atiya, A. F. (1996). Introduction to financial forecasting. *Applied Intelligence, 6*(3), 205-213.

Albuquerque, P. H. M. (2014). *Previsão de Séries Temporais Financeiras or meio de Máquinas de Suporte Vetorial e Ondaletas.* (Pós-Doutorado), Universidade de São Paulo.

Ali, S., & Smith, K. A. (2003). *Automatic parameter selection for polynomial kernel.* Paper presented at the Information Reuse and Integration, 2003. IRI 2003. IEEE International Conference on.

Beltrami, M., Loch, G. V., & Silva, A. (2011). Comparação Das Técnicas De Support Vector Regression E Redes Neurais Na Precificação De Opções. *XIlii Sbpo*, 572-583.

Bengio, Y., & LeCun, Y. (2007). Scaling learning algorithms towards AI. *Large-scale kernel machines, 34*(5).

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). *A training algorithm for optimal margin classifiers.* Paper presented at the Proceedings of the fifth annual workshop on Computational learning theory.

Boughorbel, S., Tarel, J.-P., & Boujemaa, N. (2005). *Conditionally positive definite kernels for svm based image recognition.* Paper presented at the Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on.

Cai, X., Hu, S., & Lin, X. (2012). *Feature extraction using restricted Boltzmann machine for stock price prediction.* Paper presented at the Computer Science and Automation Engineering (CSAE), 2012 IEEE International Conference on.

Cho, Y., & Saul, L. K. (2009). *Kernel methods for deep learning.* Paper presented at the Advances in neural information processing systems.

Cho, Y., & Saul, L. K. (2010). Large-margin classification in infinite neural networks. *Neural computation, 22*(10), 2678-2697.

Cho, Y., & Saul, L. K. (2011). Analysis and extension of arc-cosine kernels for large margin classification. *arXiv preprint arXiv:1112.3712*.

Ciresan, D. C., Meier, U., Masci, J., Maria Gambardella, L., & Schmidhuber, J. (2011). *Flexible, high performance convolutional neural networks for image classification.* Paper presented at the IJCAI Proceedings-International Joint Conference on Artificial Intelligence.

Dahl, G., Mohamed, A.-r., & Hinton, G. E. (2010). *Phone recognition with the mean-covariance restricted Boltzmann machine.* Paper presented at the Advances in neural information processing systems.

Drucker, H., Burges, C., Kaufman, L., Smola, A., & Vapnik, V. (1997). Support vector regression machines, advances in neural information processing systems 9: MIT Press, Cambridge.

Emir, S., Dinçer, H., & Timor, M. (2012). A Stock Selection Model Based on Fundamental and Technical Analysis Variables by Using Artificial Neural Networks and Support Vector Machines. *Review of Economics & Finance*, 106-122.

Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P., & Bengio, S. (2010). Why does unsupervised pre-training help deep learning? *The Journal of Machine Learning Research, 11*, 625-660.

Fan, A., & Palaniswami, M. (2001). *Stock selection using support vector machines.* Paper presented at the Neural Networks, 2001. Proceedings. IJCNN'01. International Joint Conference on.
Ferreira, T. A. (2011). *Previsão da Volatilidade de Séries Temporais Financeiras via Máquina de Suporte Vetorial.*, Universidade de São Paulo.
Genton, M. G. (2002). Classes of kernels for machine learning: a statistics perspective. *The Journal of Machine Learning Research, 2*, 299-312.
Gerlein, E. A., McGinnity, M., Belatreche, A., & Coleman, S. (2016). Evaluating machine learning classification for financial trading: An empirical approach. *Expert Systems with Applications, 54*, 193-207. doi:http://dx.doi.org/10.1016/j.eswa.2016.01.018
Gupta, P., Mehlawat, M. K., & Mittal, G. (2012). Asset portfolio optimization using support vector machines and real-coded genetic algorithm. *Journal of Global Optimization, 53*(2), 297-315.
Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation, 18*(7), 1527-1554.
Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735-1780.
Hofmann, T., Schölkopf, B., & Smola, A. J. (2008). Kernel methods in machine learning. *The annals of statistics*, 1171-1220.
Hsu, C.-W., Chang, C.-C., & Lin, C.-J. (2003). A practical guide to support vector classification.
Huang, C.-F. (2012). A hybrid stock selection model using genetic algorithms and support vector regression. *Applied Soft Computing, 12*(2), 807-818.
Huang, S.-C., Chuang, P.-J., Wu, C.-F., & Lai, H.-J. (2010). Chaos-based support vector regressions for exchange rate forecasting. *Expert Systems with Applications, 37*(12), 8590-8598.
Huerta, R., Corbacho, F., & Elkan, C. (2013). Nonlinear support vector machines can systematically identify stocks with high and low future returns. *Algorithmic Finance, 2*(1), 45-58.
Jarrett, K., Kavukcuoglu, K., Ranzato, M. A., & LeCun, Y. (2009). *What is the best multi-stage architecture for object recognition?* Paper presented at the Computer Vision, 2009 IEEE 12th International Conference on.
Kamath, U., Shehu, A., & De Jong, K. A. (2010). Feature and kernel evolution for recognition of hypersensitive sites in DNA sequences *Bio-Inspired Models of Network, Information, and Computing Systems* (pp. 213-228): Springer.
Kim, K.-j. (2003). Financial time series forecasting using support vector machines. *Neurocomputing, 55*(1), 307-319.
Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). *Imagenet classification with deep convolutional neural networks.* Paper presented at the Advances in neural information processing systems.
Lai, K. K., Yu, L., Wang, S., & Zhou, C. (2006). *A double-stage genetic optimization algorithm for portfolio selection.* Paper presented at the Neural Information Processing.
LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436-444.
Marcelino, S., Henrique, P. A., & Albuquerque, P. H. M. (2015). PORTFOLIO SELECTION WITH SUPPORT VECTOR MACHINES IN LOW ECONOMIC PERSPECTIVES IN EMERGING MARKETS. *Economic Computation & Economic Cybernetics Studies & Research, 49*(4).

Marcelino, S. S. d. F. (2015). Formação de portfólio por meio de Máquinas De Suporte Vetorial.
Markowitz, H. (1952). Portfolio selection. *The journal of finance, 7*(1), 77-91.
Mercer, J. (1909). Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical transactions of the royal society of London. Series A, containing papers of a mathematical or physical character, 209*, 415-446.
Micchelli, C. A. (1986). Interpolation of scattered data: distance matrices and conditionally positive definite functions. *Constructive approximation, 2*(1), 11-22.
Pires, M. M., & Marwala, T. (2005). *American option pricing using Bayesian multi-layer perceptrons and Bayesian support vector machines.* Paper presented at the Computational Cybernetics, 2005. ICCC 2005. IEEE 3rd International Conference on.
Platt, J. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers, 10*(3), 61-74.
Rahimi, A., & Recht, B. (2007). *Random features for large-scale kernel machines.* Paper presented at the Advances in neural information processing systems.
Rüping, S. (2001). *SVM kernels for time series analysis*. Retrieved from
Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks, 61*, 85-117.
Soman, K., Loganathan, R., & Ajay, V. (2009). *Machine learning with SVM and other kernel methods*: PHI Learning Pvt. Ltd.
Strobl, E. V., & Visweswaran, S. (2013). *Deep multiple kernel learning.* Paper presented at the Machine Learning and Applications (ICMLA), 2013 12th International Conference on.
Tang, Y. (2013). Deep learning using support vector machines. *CoRR, abs/1306.0239*.
Tay, F. E., & Cao, L. (2001). Application of support vector machines in financial time series forecasting. *Omega, 29*(4), 309-317.
Thomason, M. (1999). The practitioner methods and tool. *Journal of Computational Intelligence in Finance, 7*(3), 36-45.
Trafalis, T. B., & Ince, H. (2000). *Support vector machine for regression and applications to financial forecasting.* Paper presented at the ijcnn.
Vapnik, V. N., & Vapnik, V. (1998). *Statistical learning theory* (Vol. 1): Wiley New York.
Vedaldi, A., & Zisserman, A. (2012). Efficient additive kernels via explicit feature maps. *Pattern Analysis and Machine Intelligence, IEEE Transactions on, 34*(3), 480-492.
Veropoulos, K., Campbell, C., & Cristianini, N. (1999). *Controlling the sensitivity of support vector machines.* Paper presented at the Proceedings of the international joint conference on AI.
Wei, C.-C. (2012). Wavelet kernel support vector machines forecasting techniques: Case study on water-level predictions during typhoons. *Expert Systems with Applications, 39*(5), 5189-5199.
White, H. (2000). A reality check for data snooping. *Econometrica, 68*(5), 1097-1126.
Wolfe, P. (1961). A duality theorem for nonlinear programming. *Quarterly of applied mathematics, 19*(3), 239-244.

Yu, S.-W., Lu, R.-S., & Chang, C.-H. (2008). *A study on application of smooth support vector classification to stock selection in taiwan's stock market.* Paper presented at the The International Conference on Computational Intelligence in Economics and Finance, Taiwan.

Yu, S.-W. L., R.-S.; Chang, C.-H.A. (2008). A study on application of smooth support vector classification to stock selection in taiwan's stock market.

Zhang, Z., & Zhao, Q. (2010). The application of SVMs method on exchange rates fluctuation. *Discrete Dynamics in Nature and Society, 2009*.