POTENCIAL DO USO DE TÉCNICAS DE MACHINE LEARNING PARA O ENRIQUECIMENTO SEMÂNTICO DE MODELOS BIM PARA A CLASSIFICAÇÃO DE ESPAÇOS RESIDENCIAIS

RAÍ LUZ BARBOSA

DISSERTAÇÃO DE MESTRADO

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA CIVIL E AMBIENTAL
PROGRAMA DE PÓS-GRADUAÇÃO EM ESTRUTURAS E CONSTRUÇÃO CIVIL

POTENCIAL DO USO DE TÉCNICAS DE MACHINE LEARNING PARA O ENRIQUECIMENTO SEMÂNTICO DE MODELOS BIM PARA A CLASSIFICAÇÃO DE ESPAÇOS RESIDENCIAIS

RAÍ LUZ BARBOSA

DISSERTAÇÃO DE MESTRADO APRESENTADO AO PROGRAMA DE PÓS-GRADUAÇÃO EM ESTRUTURAS E CONSTRUÇÃO CIVIL COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM ESTRUTURAS E CONSTRUÇÃO CIVIL.

_____

Raí Luz Barbosa
(Mestrando)

_____

Prof. Francisco Evangelista Junior, Ph.D. (Universidade de Brasília)
(Orientador)

_____

Prof. Antônio Carlos de Oliveira Miranda, Dr. (Universidade de Brasília)
(Examinador interno)

_____

Prof. Kléos Magalhães Lenz César Júnior, Ph.D. (Universidade Federal de Viçosa)
(Examinador externo)

BRASÍLIA – DF
SETEMBRO DE 2022

"O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001".

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

SUMÁRIO

RESUMO

O processo BIM surgiu na indústria da AEC com a proposta de desenvolver um modelo digital de uma edificação para que este pudesse auxiliar nas etapas da concepção do projeto, construção, gerenciamento e manutenção. Estes modelos possuem uma grande quantidade de informações que, constantemente, precisam ser trocadas entre as diversas disciplinas envolvidas. Como essas disciplinas utilizam ferramentas diferentes para a leitura das informações do modelo, é importante garantir a troca de informações de forma completa, para se evitar o trabalho de correção e/ou acréscimo de dados. O padrão de troca de informações aberto e não proprietário IFC foi criado para que houvesse interoperabilidade entre os diferentes softwares que se baseiam no processo BIM. Entretanto, devido à complexidade dos modelos digitais e à grande variedade de softwares que possam a vir se envolver no projeto, ainda é comum que haja perda de informações. Surgiu-se, então, a necessidade de utilização de metodologias que auxiliassem no processo de interoperabilidade e promoção do enriquecimento semântico de arquivos IFC. Entre essas metodologias, o *machine learning* (ML) vem ganhando espaço, com suas diversas técnicas que utilizam o processo de aprendizagem por treinamento para a predição de novas informações. Este trabalho propõe e valida modelos de ML e estratégias de análise de dados como potencial para o enriquecimento semântico de modelos BIM para melhorar a interoperabilidade entre projetos de arquitetura e engenharia. Diversas técnicas ML (*k-Nearest Neighbor*, *Bagged Tree*, SVM-Gaussiana, SVM-Quadrática e SVM-Cúbica) foram implementadas na classificação de espaços em modelos residenciais. Além dos tradicionais classificadores binários, foram propostas duas técnicas por votação, multiclasse-binário e *ensemble*, para melhorar a acurácia e diminuir o *overfitting* durante o processo de classificação. Os resultados demonstram que as técnicas por votação podem ser usadas com sucesso como estratégias de enriquecimento semântico para classificar, de forma precisa, classes de espaços residenciais a partir de variáveis simples e fáceis de se obter de arquivos IFC. As estratégias e técnicas propostas podem melhorar a interoperabilidade de modelos BIM, classificando de forma eficiente espaços residenciais sem a necessidade direta de intervenção humana.

Palavras-chave: *BIM, enriquecimento semântico, machine learning, classificação multiclasse, ensemble.*

ABSTRACT

The BIM process emerged in the AEC industry with the proposal to develop a digital model of a building to assist the project conception, the building process and the management and maintenance of the building. These models have a large amount of information that constantly needs to be exchanged between the various disciplines involved in the project. As these disciplines use different tools for reading the model's information, it is important to guarantee a complete exchange of information to avoid correction and/or addition of data. The non-proprietary IFC file was created so that there would be interoperability between the different software that are based on the BIM process. However, due to the complexity of the digital models and the wide variety of software that may be involved in the project, it is still common for information to be lost. Because of that, there was a need to use methodologies that would help in the interoperability process and promote the semantic enrichment of IFC files. Among these methodologies, machine learning (ML) has been gaining ground, with its various techniques that use the learning from experience process to predict new information. This paper proposes and validates ML models and strategies of data analysis to promote the semantic enrichment of Building Information Modeling (BIM) to improve interoperability among architectural and engineering applications. Various ML techniques (k-Nearest Neighbor, Bagged Tree, SVM-Gaussian, SVM-Quadratic e SVM-Cubic) were implemented to classify room types of residential building models. In addition to traditional binary classifiers, it was proposed two voting techniques, namely multiclass-binary and ensemble, to improve the accuracy of the classifications and decrease overfitting during the training process. Real architectural design data of residential buildings with 8 different classes of rooms were used to train the classifiers in scenarios with different feature variables selection. The results demonstrate that the voting techniques can be successfully used as semantic enrichment strategies to accurately predict the residential room classes using few and easy-to-obtain features from IFC files. The proposed strategies and techniques improved the interoperability of BIM models in efficiently classifying residential rooms without the need for human intervention.

Keywords: *BIM, semantic enrichment, machine-learning, multiclass classification, ensemble.*

1. INTRODUÇÃO

A Modelagem de Informação da Construção, do inglês *Building Information Modeling* (BIM), vai além do simples conceito de representação gráfica de um empreendimento na construção civil. Com a tecnologia BIM, o modelo de uma edificação é construído digitalmente de forma mais precisa, contendo informações das diversas fases do empreendimento, permitindo um melhor gerenciamento quando comparado com os processos baseados em desenhos bidimensionais. Este modelo virtual, além da representação geométrica da edificação, contém informações importantes para o desenvolvimento do empreendimento, tais como quantitativos de materiais, custos de materiais e serviços, e cronogramas. (SACKS et al., 2018)

A tecnologia BIM veio para substituir os padrões mais usuais na indústria da construção, os modelos CAD – *Computer-Aided Design* – que contém prioritariamente informações geométricas, seja em 2 ou 3 dimensões. As demais informações são obtidas a partir destes modelos CAD. Assim, softwares que utilizam o conceito BIM integram o trabalho que usualmente era realizado por diversas ferramentas.

Além do seu uso para desenvolvimento de novos projetos, o BIM também tem se desenvolvido em atividades relacionadas aos patrimônios, tais como investigação, conservação, documentação, reconstrução virtual e gerenciamento. Como exemplo, cita-se o levantamento feito para a criação de um modelo BIM da Basílica di Collemaggio para a avaliação do comportamento estrutural, econômica e restauração final após esta ser atingida por um terremoto. (ORENI et al., 2014)

São diversos os softwares que se baseiam no conceito BIM para a elaboração de um projeto na construção civil. Diferentes disciplinas, sejam elas Arquitetura, Estrutura, Instalações, utilizam ferramentas que melhor atendem suas necessidades. O desenvolvimento do projeto deve ser feito de forma integrada por essas diferentes disciplinas, sendo necessária a constante troca de informações entre elas. Sendo assim, a *Building Smart*, uma organização internacional sem fins lucrativos, criou um padrão que auxilia na troca de dados entre os diferentes softwares, o ISO 16739-1:2018 *Industry Foundation Classes* (IFC). O IFC é um padrão não proprietário e aberto de dados que visa padronizar as informações em sistemas orientados a objetos, possibilitando o compartilhamento de dados entre os diversos aplicativos (NASCIMENTO, 2004). A partir deste padrão, criou-se o formato de arquivo IFC, que permite a interoperabilidade entre as

diferentes ferramentas utilizadas na indústria de arquitetura, engenharia e construção (AEC) que trabalham com o conceito BIM.

De acordo com Jacoski (2003), para que as informações entre os softwares sejam interoperáveis, algumas características devem ser observadas. Uma determinada ferramenta deve possuir uma estrutura capaz de operar formatos de arquivos de outras ferramentas, permitindo assim a troca de dados. Além disso, é preciso que haja uniformidade para a interação com o usuário, garantindo uma padronização na forma em que os modelos são construídos. Devido a integração de trabalho entre diferentes usuários, as ferramentas devem produzir informações de forma transparente e similar, fazendo com que qualquer um possa trabalhar no modelo utilizando as mesmas convenções. A inobservância dessas características pode prejudicar a troca de informações. (JACOSKI, 2003)

A tecnologia BIM se baseia em uma modelagem parametrizada de objetos em que cada componente contém informações sobre geometria, especificações de material, tipologia, forma de agregação. Ao se exportar um modelo de um determinado software para outro através do IFC, muitas dessas informações podem ser perdidas ou repassadas de forma incorreta. Isso ocorre porque o modelo IFC pode não ter suporte para determinado tipo de informação (JEONG et al., 2009). Assim, é necessária a verificação dos dados importados para que se possa fazer as correções necessárias, possibilitando a obtenção de um modelo com informações pertinentes ao processo de trabalho.

Essa verificação, que pode ser feita manualmente, gera um trabalho adicional, fazendo com que o projetista perca tempo em algo que poderia ser evitado caso os dados fossem transferidos de forma eficiente. Com o intuito de aprimorar a checagem dos dados, muitas pesquisas têm sido feitas com o objetivo de acrescentar as informações não contidas nos arquivos IFC, chamado de enriquecimento semântico, de forma automatizada. De uma forma mais geral, o enriquecimento semântico engloba, segundo Bloch (2018), a identificação, classificação, reconstrução e adição de informações dos objetos presentes no modelo exportado. Este processo permite o melhor manuseio do modelo na ferramenta para a qual este está sendo exportado.

Uma metodologia que tem ganhado espaço na automatização de análise de dados é o *Machine Learning* (ML). Ele se baseia em um processo de aprendizado por experiência em que, através

do treinamento de um algoritmo, é possível fazer predições ou classificações a partir de um banco de dados com soluções definidas para o problema em análise. Após esse treinamento, o algoritmo é utilizado, então, para encontrar respostas para problemas que ainda não possuem soluções definidas. São diversas as técnicas de ML existentes, e cada uma terá um comportamento diferente perante os diversos tipos de análises. Sendo assim, para cada tipo de problema, há uma técnica que produzirá melhores resultados.

Sendo assim, o ML se demonstra uma ferramenta promissora para o enriquecimento semântico de modelos BIM, uma vez que será possível identificar informações pertinentes dos diversos elementos do modelo. O fato destes elementos serem caracterizados por diversos atributos e, também, possuírem relações com outros elementos do modelo, faz com que estes dados sejam bastante viáveis para o estudo com ML. Uma vez tratados e corretamente manuseados, as novas informações podem ser, então, inseridas novamente nos modelos.

1.1. OBJETIVO GERAL

Desenvolvimento de modelos de *Machine Learning* para a classificação de ambientes residenciais, a partir de dados de projetos arquitetônicos, como proposta de técnica para o potencial enriquecimento semântico de arquivos IFC. Além disso, busca-se estratégias de análise de dados que auxiliem na melhora do desempenho destes modelos.

1.2. OBJETIVOS ESPECÍFICOS

- Aplicação de técnicas de ML, sendo elas *Support Vector Machine* (SVM), k-Nearest Neighbor (k-NN) e Bagged Tree (BT), para o treinamento de algoritmos de classificação. A partir da técnica SVM, foram analisados algoritmos com diferentes funções de covariância e, então, foram propostas 3 novas técnicas: SVM-Gaussiana, SVM-Quadrática e SVM-Cúbica;
- Elaboração de um algoritmo de treinamento, na linguagem Matlab, para a classificação de ambientes residenciais, deixando essa informação disponível para o enriquecimento semântico de arquivos IFC;
- Definir as variáveis de análise, sejam individuais ou de correlação, mais adequadas para se obter o melhor desempenho na classificação;

- Analisar quais fatores gerais influenciam na performance do algoritmo, construindo assim um modelo que consiga fornecer os melhores resultados possíveis para a classificação;
- Analisar, dentre as técnicas de ML propostas, qual apresenta melhor performance para o problema proposto.

1.3. IMPACTOS E SUGESTÕES PARA TRABALHOS FUTUROS

Com o desenvolvimento deste trabalho espera-se:

- Contribuir para o processo de enriquecimento semântico de arquivos IFC, uma vez que as técnicas analisadas neste estudo irão proporcionar alternativas eficientes para o processo de automatização na manipulação de dados;
- Melhorar o fluxo de informações entre os diferentes softwares (interoperabilidade) que utilizam o processo BIM.

Como proposta de continuidade deste trabalho, pode-se citar:

- Desenvolvimento de um algoritmo que extraia os dados analisados neste trabalho diretamente dos arquivos IFC;
- Desenvolvimento de um algoritmo que insira dentro dos arquivos IFC as informações obtidas dos modelos de classificação.

2. REVISÃO BIBLIOGRÁFICA

2.1. ESTADO DA ARTE

Alguns estudiosos fizeram testes que demonstravam os diversos problemas que ocorriam ao se transferir informações entre os diferentes softwares fundamentados no processo BIM. Andrade e Ruschel (2009) estudaram a interoperabilidade de aplicativos BIM por meio do formato IFC no âmbito de projetos arquitetônicos. Nesse estudo, foi feita a modelagem de um edifício de dois pavimentos nos softwares ArchiCAD e Revit. Os modelos foram, então, exportados para outro *software*, através de arquivos IFC, em que foram feitas análises das informações transferidas. Observou-se que ocorreu perdas qualitativas dos modelos devido às limitações das informações dos aplicativos e à falta de padronização na definição das propriedades dos objetos. As perdas mais significativas foram observadas em atributos não geométricos, tais como código de identificação, tipo de material, disposição dos elementos e custos de material, evidenciando a falta de robustez dos arquivos na transferência de dados. (ANDRADE; RUSCHEL, 2009)

Jeong et al. (2009) realizou testes com o objetivo de analisar a troca de informações entre ferramentas BIM de modelagem arquitetônica e de fabricação de pré-moldados. Foram utilizados como modelos, painéis de fachada em concreto pré-moldado, contendo diferentes objetos com geometrias complexas e uma variedade de materiais. Estes foram exportados por meio de arquivos de formato neutro, como o IFC. O conteúdo dos dados foi classificado em dois tipos: dados geométricos e propriedades do objeto. Assim, foi analisada a capacidade de edição de arquivos exportados por ferramentas utilizadas em modelagem arquitetônica por ferramentas utilizadas no processo de fabricação de pré-moldados. Observou-se falhas tanto na exportação quanto na importação dos dados devido, em sua maior parte, à não uniformidade da maneira como os objetos e as propriedades são organizados nos arquivos IFC.

Müller (2011) fez uma série de exportações e importações de dados de modelos estruturais em concreto armado entre os softwares Revit, TQS e um visualizador de arquivos IFC, para a análise das inconformidades que ocorrem nas trocas de dados. Nos casos em que havia cargas aplicadas à estrutura, esta informação foi perdida no processo de troca e, em estruturas com armaduras, estas foram transferias apenas como uma informação de volume de aço, porém sem especificações pertinentes, como bitola, tipo de aço. Notou-se também que, vigas com furos,

curvadas ou contínuas com apoios intermediários, foram divididas em partes ao serem exportadas de uma determinada ferramenta para outra. Sabendo da importância do comportamento monolítico da estrutura numa análise estrutural, este erro de exportação deve ser ajustado para a correta análise do modelo. (MULLER, 2011)

No campo de estudos de construções patrimoniais, um problema recorrente da aplicação do BIM é a falta de informações não geométricas que são de grande importância para quem trabalha nessa área. Com isso, o enriquecimento semântico vem sendo utilizado também nessa área. Apesar do arquivo IFC não permitir a inserção de algumas informações desse tipo, é possível fazer com que algumas informações sejam inseridas de forma forçada neste arquivo. Simeone et al. (2019) analisou a possibilidade de inserção de dados diversos em modelos BIM a partir de ontologias. Assim, é possível criar um modelo construtivo de uma determinada construção patrimonial contendo informações que se concentram em um único documento. (SIMEONE; CURSI; ACIERNO, 2019)

Diversos são os estudos com o objetivo de melhorar a interoperabilidade entre ferramentas BIM, possibilitando o enriquecimento semântico de arquivos no formato IFC. Entretanto, quando se fala em aplicações de técnicas de ML para tal ou estudos que buscam a identificação de ambientes residenciais em modelos BIM, as limitações são maiores.

Bloch e Sacks (2018) fizeram um estudo comparativo entre as técnicas ML e *rule-based inferencing* para a classificação de ambientes em apartamentos residenciais. Na abordagem utilizando *rule-based inferencing*, foram montadas matrizes contendo informações sobre a geometria e função dos espaços, bem como a forma de relacionamento entre eles. De um total de 15 tipos de ambientes, obteve-se a correta classificação de cinco destes. Ao se utilizar a metodologia do ML, diversos modelos com ambientes já identificados são tomados como exemplo para treinamento do algoritmo para posterior classificação de modelos desejados. Nesse estudo, foi utilizado um total de 150 ambientes, dos quais 70% foram utilizados para treinamento e 30% para validação, em que se atingiu 82% de acurácia. Em modelos de apartamentos testados para a classificação após o treinamento, obteve-se acurácia de até 100%.

Noi e Kappas (2018) compararam a performance de três técnicas de ML, sendo elas *support vector machine* (SVM), *k-nearest neighbor* (k-NN) e *random forest* (RF), aplicados à classificação de coberturas e uso de terras através de imagens. As análises mostraram que, para

um banco de dados de treinamento grande o suficiente, SVM, k-NN e RF obtiveram acurácia semelhante e acima de 93%, independentemente se os dados eram balanceados ou não. Notou-se também que SVM obteve melhor performance, com menos sensibilidade ao tamanho do banco de dados de treinamento. De uma forma geral, quanto maior o banco de dados de treinamento, melhor a performance das análises. (NOI; KAPPAS, 2018)

Koo et al. (2019) também aplicaram uma técnica do ML, o SVM, para o estudo da integridade semântica de arquivos IFC. Foram utilizados seis modelos arquitetônicos BIM para o treinamento do SVM, contendo mais de 4.000 elementos. O treinamento se deu em dois estágios: no primeiro foi feita a classificação geral dos objetos com base em 8 classes distintas do IFC, enquanto no segundo, classificou-se subtipos de elementos de 3 classes escolhidas. Os resultados obtidos demonstraram alta taxa de performance da classificação, mesmo com algumas limitações observadas, tal como desproporcionalidade na quantidade de diferentes classes nos modelos utilizados para treinamento. (KOO et al., 2019)

Zeng et al. (2019) realizaram um estudo de classificação de espaços residenciais, porém utilizando redes neurais multitarefa em uma base de dados composta de imagens de plantas baixas arquitetônicas. Inicialmente eles trabalharam em delimitar os diferentes espaços para, posteriormente, nomeá-los. Os resultados deste estudo foram promissores, obtendo uma acurácia de aproximadamente 90%, o que possibilitou uma performance melhor que outros métodos comparados no estudo. (ZENG et al., 2019)

Outros estudiosos, tais como Gimenez et al. (2016), Turner e Zakhor (2014) e Ahmed et al. (2012), também investigaram formas de identificar, de forma automática, os diferentes espaços de plantas baixas reconstruídas através de imagens. Foram diversos os intuitos dessa identificação, seja pela melhor representação do modelo ou pelo uso desse tipo de informação em análises pertinentes da construção civil, como estudo de conforto ambiental. Assim, a assimilação dessa informação em modelos arquitetônicos é uma importante contribuição no enriquecimento semântico. (GIMENEZ et al., 2016)(TURNER; ZAKHOR, 2014)(AHMED et al., 2012)

Algumas outras técnicas também tiveram seu destaque na promoção do enriquecimento semântico de modelos BIM. Belsky at al. (2016) desenvolveu uma ferramenta, SeeBIM, que promove o enriquecimento semântico em modelos BIM aplicando a inferência baseada em

regras (rule-based inferencing). A partir de uma série de regras, denotadas como *IF-THEN rules*, foram feitos testes para a classificação de juntas e conexões de elementos em concreto pré-moldado e para a agregação de lajes pré-moldadas. A partir de uma série de operadores, são checadas a constituição, a funcionalidade, a geometria e a orientação espacial de determinado objeto, além da verificação de como dois objetos se relacionam (se um objeto faz parte de outro). Implementado em linguagem C#, o SeeBIM adiciona novas informações ao modelo extraídas de sua base de dados interna. Essa base de dados é formada por regras definidas pelo usuário e armazenadas em um arquivo de texto. Um modelo estrutural de um estacionamento foi utilizado para a verificação da ferramenta e todas as juntas, conexões e formas de agregação foram identificadas corretamente. (BELSKY; SACKS; BRILAKIS, 2016)

A ferramenta SeeBIM também foi utilizada em um estudo desenvolvido por Sacks at al. (2017) com o intuito de melhorar sua capacidade devido a algumas limitações observadas. Foram desenvolvidos novos procedimentos para a geração de regras, além da melhoria dos operadores para a identificação de geometrias complexas. Para a validação, utilizou-se modelos de pontes gerados através de dados obtidos por scanners a laser (SACKS et al., 2017). Outro estudo com enriquecimento semântico em modelos 3D de dados obtidos por scanners à laser foi realizado por Xiong at al. (2013). Nesse artigo, foi proposto uma metodologia para a automatização da conversão dos dados obtidos pelo scanner para um modelo BIM enriquecido. O algoritmo implementado foi capaz de reconhecer os principais componentes visíveis de um ambiente interno, tais como paredes, teto, piso, portas e janelas, apesar do alto nível de oclusão e falta de alguns dados. (XIONG et al., 2013)

Ma (2018) sugere alguns passos para a classificação de objetos parametrizados em modelos BIM. Primeiro, com base no conhecimento de um experiente da área, deve-se obter uma estrutura de dados contendo informações sobre a localização espacial dos objetos e seus principais atributos. Estas informações devem estar estruturadas de forma que sejam facilmente interpretadas por um computador. Em seguida, com o auxílio de algoritmos, as informações devem ser guardadas. Por fim, a classificação se dá de forma automatizada através da equiparação dos modelos com a base de dados. Com essa metodologia, foi possível realizar a correta classificação de todos os objetos de duas pontes em concreto. (MA et al., 2018)

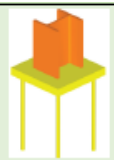
Nota-se que tanto o *machine learning* quanto o rule-based inferencing vem sendo usados no processo de enriquecimento semântico do IFC. A abordagem do ML aplica o raciocínio indutivo, no qual as conclusões são feitas por um processo de observação. Por outro lado, a abordagem do rule-based inferencing se baseia no raciocínio dedutivo, onde as conclusões são baseadas em uma série de regras pré-determinadas. A técnica mais adequada fica, então, dependente do tipo de problema a ser estudado, uma vez que, apesar das falsas conclusões que se pode obter com o raciocínio indutivo, alguns problemas relacionados à AEC são de difícil tradução em formas de regras. (BLOCH; SACKS, 2018)

2.2. PROCESSO BIM

BIM, acrônimo da expressão em inglês *Building Information Modeling*, traduzido como Modelagem da Informação da Construção, é definido por Sakcs et al. (2018) como uma tecnologia de modelagem e conjunto de processos associados para produzir e analisar modelos de edificações, bem como trocar informações entre os participantes deste processo. A ISO 19650 (2018, apud UK BIM ALLIANCE, 2019) diz que o BIM busca benefícios através de uma melhor especificação e entrega da quantidade necessária de informações a respeito do projeto, construção e gerenciamento de empreendimentos utilizando ferramentas adequadas. Os modelos BIM são caracterizados por representações digitais dos componentes de uma edificação, definido como objetos, em que apresentam informações gráficas, de dados e regras paramétricas. Com isso, é possível construir elementos que carregam as informações necessárias ao processo construtivo e pertinentes às diversas disciplinas envolvidas, passíveis de serem manipulados de forma inteligente e consistente, garantindo sua atualização de forma prática em todo o modelo.

A quantidade e qualidade das informações existentes em elementos de um modelo BIM vai depender das etapas e fases em que um empreendimento se encontra. Quanto mais avançado este se encontrar, maior será o detalhamento de seus elementos, em que estes apresentarão informações pertinentes à etapa em que se encontra. O Governo do Estado de Santa Catarina divulgou um caderno com especificações que deverão ser adotados pelos prestadores de serviços do estado, onde este faz uma correlação entre a representação dos elementos e a etapa de desenvolvimento do empreendimento, como mostrado na Figura 2.1.

Figura 2.1 - Correlação da representação de um elemento em modelo BIM com as etapas de um empreendimento (GOVERNO DO ESTADO DE SANTA CATARINA, 2018).

REPRESENTAÇÃO									– Execução da obra – *"As Built"* – Realidade – Como executado		
DESCRIÇÃO	– Levantamento de informações (urbanísticas, ambientais, fundiárias e econômicas); – Identificação das necessidades; – Esboço; e – Estudo de Massa.			– Desenhos esquemáticos; – Volumetria geral edifício; – Análise do prédio inteiro (volume, orientação, custos de metragem quadrada); – Predefinição dos componentes e elementos/objetos dos ambientes;	– Desenvolvimento do desenho e do modelo; – Sistemas/conjuntos genéricos (quantidades aproximadas, tamanho, forma, localização, orientação); – Análise de desempenho do sistema selecionado.	– Desenvolvimento da modelagem da construção; – Criação da documentação pela geração de desenhos tradicionais; – Análise dos elementos/sistemas; – Inclusão de atributos e parâmetros definidos.	– Finalização da modelagem da construção; – Construção da documentação; – Modelos finais sem as informações e detalhes de montagens, suas especificações com os correspondentes desenhos; – Análise detalhada de elementos/sistemas; – Inclusão de atributos e parâmetros definidos.	– Planejamento e administração da construção; – Modelos finais com as informações, detalhes de montagens e suas especificações com os correspondentes desenhos; – Tabelas de quantitativos precisas, que incluem tamanhos, formas, localização e orientação dos elementos e objetos do projeto; – Representações virtuais dos elementos propostos, adequados para construção, fabricação e montagem.	– Conclusão da execução da obra do Projeto; – Registro nos projetos e documentação de como foi construído e suas condições (*As-built*); – O modelo deve estar reajustado e configurado para ser usado como base de dados central para a integração nos sistemas de manutenção e operações do empreendimento; – As entidades devem conter os parâmetros e atributos, conforme especificado pela CONTRATANTE, ao tempo da execução, instalação ou montagem.		
ETAPAS	Levantamento de Dados (LV)	Programa de Necessidades (PN)	Estudo de Viabilidade (EV)	Estudo Preliminar (EP)	Anteprojeto (AP)	Projeto Legal (PL)	Projeto Básico (PB)	Projeto Executivo (PE)	Licitação da Obra	Contratação da Obra	Obra Concluída
FASES	Concepção do Produto			Definição do Produto	Identificação e Solução de Interfaces			Projeto de Detalhamento de Especialidades	Pós–Entrega do Projeto		

Devidos aos diversos *softwares* disponíveis quem utilizam o conceito BIM, onde muitos são desenvolvidos com o foco em determinadas disciplinas, sejam arquitetura, estrutura, instalações, foi desenvolvido o formato IFC para que fosse possível a troca de informações entre eles. Este formato possui uma estrutura específica, muitas vezes denominado de esquema, que procura garantir a transferência de todas as informações necessárias. Há um constante desenvolvimento desse formato de arquivo para que a interoperabilidade seja garantida entre as mais diversas disciplinas e os mais variados programas.

2.2.1. Protocolo IFC

Baseado na ISO-10.303 (STEP), que trata sobre a linguagem de definição de modelos de produto, o IFC, do inglês *Industry Foundation Classes*, é um protocolo desenvolvido para trocas de informações entre ferramentas pertinentes ao ramo da arquitetura, engenharia e construção civil. Ele engloba diversas informações, seja de objetos específicos ou de dados em geral, que podem ser utilizados durante todo o ciclo de vida da edificação (SACKS et al., 2018). Estas informações são armazenadas, especificamente, nas diversas entidades definidas pelo protocolo IFC.

Sendo assim, para acessar informações específicas de determinados elementos, tais como paredes, vigas, portas, espaços, é preciso acessar as entidades definidas para estes objetos, por exemplo *IfcWall, IfcBeam, IfcDoor* e *IfcSpace* respectivamente. Assim, é possível adquirir informações geométricas, como área, perímetro, volume, entre outros. O protocolo IFC permite também saber como que os diferentes elementos se correlacionam, através da entidade *IfcRelationship*.

A entidade *IfcSpace*, que representa o elemento espaço, estudado para o embasamento das análises deste trabalho, possui diversos atributos que carregam informações de identificação, de quantidades e de posicionamento (BSI STANDARDS PUBLICATION, 2014). Outros atributos podem, também, ser associados ao elemento espaço, tais como acabamentos de piso, teto e parede, dentre outros. A Tabela 2.1 abaixo mostra alguns dos principais atributos associado a este elemento e que contribuiu para a escolha das variáveis utilizadas neste trabalho.

Tabela 2.1 - Tabela com alguns dos principais atributos da entidade *IfcSpace* (Adaptado de BSI STANDARDS PUBLICATION, 2014).

Nome	Tipo	Exemplo	Descrição
GrossPerimeter	Numérico	11.546,8	Perímetro bruto, incluindo partes do perímetro criado por limites virtuais e aberturas.
InteriorOrExteriorSpace	Categórico	Interno	Espaço Interior ou Exterior
NetPerimeter	Numérico	11.546,8	Perímetro líquido, que exclui o perímetro crido por limites virtuais e aberturas.
CeilingCovering	Categórico	Nenhum	Material de acabamento do teto.
FloorCovering	Categórico	Carpete	Material de acabamento do piso.
FinishCeilingHeight	Numérico	2.500,0	Altura do topo do piso à face inferior do acabamento do teto.
FinishFloorHeight	Numérico	0	Altura da laje até o acabamento do piso.
GrossCeilingArea	Numérico	12,50	Soma de toda a área de teto do espaço, incluindo a área coberta por elementos internos, como pilares. Esta é a área real (e não a projetada).
GrossVolume	Numérico	97,25	Volume bruto do espaço, incluindo o volume de elementos internos.
GrossWallArea	Numérico	56,75	Soma da área de todas as paredes do espaço, incluindo a área dos elementos contidos nas paredes, como portas e janelas.
NetCeilingArea	Numérico	11,75	Soma de toda a área de teto do espaço, excluindo a área coberta por elementos internos, como pilares. Esta é a área real (e não a projetada).
NetVolume	Numérico	95,50	Volume bruto do espaço, excluindo o volume de elementos internos.
NetWallArea	Numérico	113,95	Soma da área de todas as paredes do espaço, excluindo a área dos elementos contidos nas paredes, como portas e janelas.
WallCovering	Categórico	Nenhum	Material de acabamento da parede.

Nota-se, na Figura 2.2, que o elemento espaço possui relação com diversos outros elementos, possibilitando identificar as paredes limites do espaço, bem como portas e janelas que pertencem a este. Uma vez que é possível identificar essas paredes e que para determinada parede é possível identificar as portas e janelas pertencentes a esta, é possível obter, de forma indireta, as portas e janelas que pertencem a determinados ambientes.

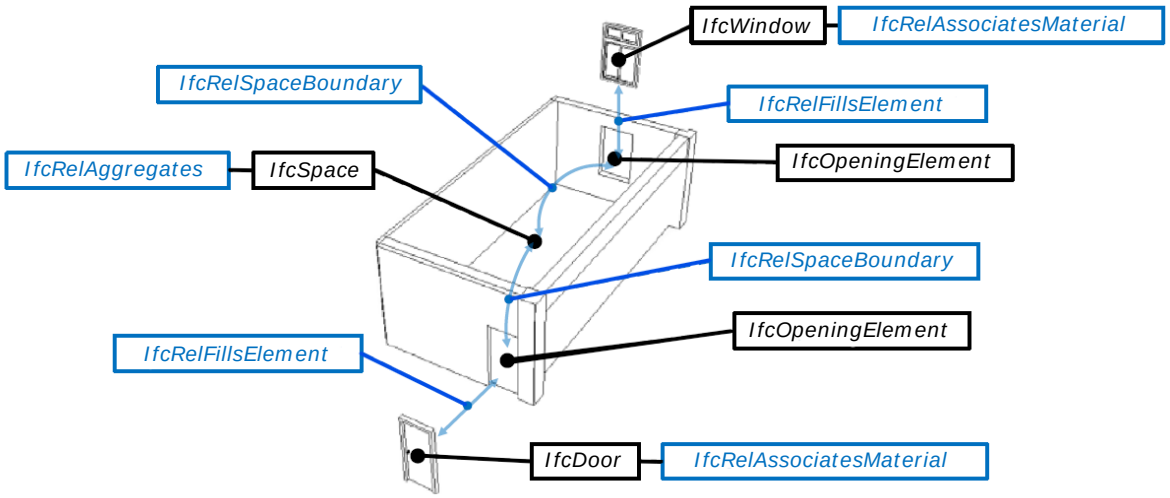


Figura 2.2 – Estrutura da entidade IfcSpace. (Adaptado de TEO; CHO, 2016)

O IFC, por ser um arquivo não proprietário, cujo principal objetivo é a troca de informações entre qualquer software, é passível a diversos erros na tentativa de promover a interoperabilidade (VENUGOPAL et al., 2012);(VENUGOPAL et al., 2010). Estes erros ocorrem pelo fato de o arquivo IFC não conseguir abranger, de forma precisa, todos os elementos definidos nos diversos softwares aplicados ao BIM. Percebe-se, então, a importância do enriquecimento semântico para o seu adequado uso. De forma geral, existem três tipos de metodologia de enriquecimento semântico (XUE; WU; LU, 2021):

a) Raciocínio semântico: baseado em regras pré-determinadas ou ontologias;
b) Registro semântico: baseado em uma série de procuras por tentativa e erro;
c) Segmentação semântica: baseado em um banco de dados de treinamento.

2.3. *MACHINE LEARNING* - ML

Machine Learning é um método que procura respostas por um processo de aprendizado através de um banco de dados, como descrito na eq. (1), com soluções já definidas.

$$T = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N} \tag{1}$$

em que $\boldsymbol{x} \in R^m$ é o vetor de treinamento no espaço de atributos m-dimensional, $\boldsymbol{y}_i$ é a correspondente classe da i-ésima amostra de treinamento $\boldsymbol{x}_i$, e N é o total de amostras utilizadas para treinamentos.

Este aprendizado se dá por meio do treinamento, de um algoritmo, realizado com os dados de entrada, sendo possível, após o treinamento, encontrar soluções para novos dados (BISHOP, 2006). As técnicas de ML são adequadas para problemas de resolução muito complexa por abordagens tradicionais ou que não tenham algoritmos conhecidos para resolvê-los. São diversas as técnicas de ML utilizadas no treinamento de algoritmos, sendo algumas delas o *Support Vector Machine*, o k-Nearest Neighbor e o Bagged Tree. (GÉRON, 2019)

2.3.1. *Support Vector Machine* - SVM

SVM foi estudado pioneiramente por Vapnik e, desde então, vem sendo aplicado em diversos estudo de classificação e regressão. De forma geral, SVM alcança resultados competitivos quando aplicados em um conjunto de dados linearmente separáveis. Entretanto, para dados que não se enquadram nesse perfil é possível utilizar funções de covariância, que levarão os dados não separáveis para um espaço de alta dimensão, tornando-os linearmente separáveis. Um desafio para o SVM é a seleção das funções de covariância e a determinação de seus parâmetros. (THARWAT, 2019)

Lin e Wang (2002) apresentam uma simples definição para o funcionamento do algoritmo que utiliza a técnica de SVM, para dados não linearmente separáveis, em que este mapeia os dados de entrada para um espaço de características de alta dimensão e procura por um hiperplano de separação que maximize a margem entre as classes nesse espaço, como mostra a Figura 2.3. Nela, é possível notar, em (a), que os dados, representados pelos pontos vermelhos e azuis, não podem ser separados por uma reta no plano em que elas estão representadas. Neste caso, estes dados são classificados como não linearmente separáveis. Entretanto, ao aplicar uma transformação nestes dados, através das funções de covariância, estes podem ser levados para um outro espaço dimensional, como mostrado em (b). Neste novo espaço, já é possível traçar um plano que separe estes dados. Maximizar a margem entre as classes é procurar por um hiperplano que separe as amostras das diferentes classes e esteja o mais distante possível dos dados. (LIN; WANG, 2002)

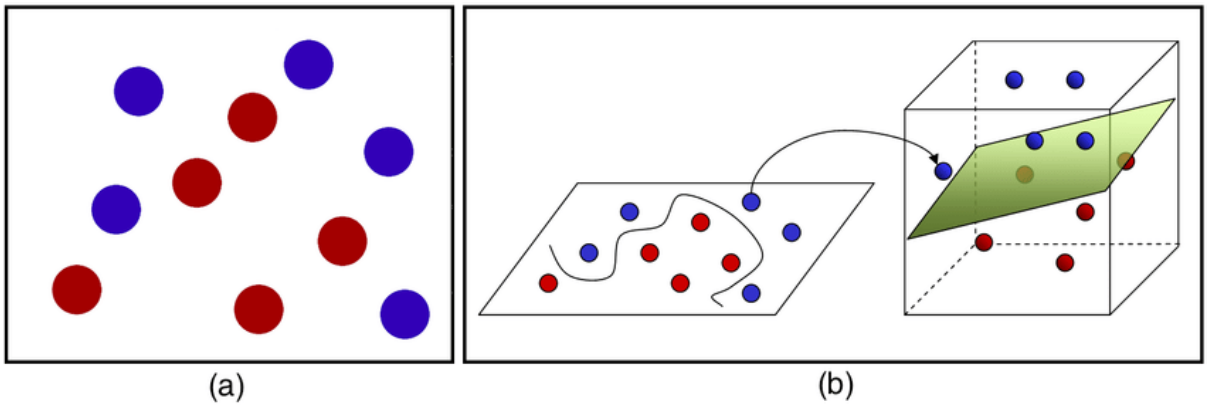


Figura 2.3 - Conjunto de dados não linearmente separáveis (a) levados para um espaço de alta dimensão, tornando-os linearmente separáveis por um hiperplano (b).

Apenas alguns pontos de treinamento são levados em consideração na determinação do limite de decisão, definido pelo hiperplano de separação. Estes pontos são chamados de vetores de suporte, do inglês *support vectors*. Segundo Muller e Guido (2017), para definir a classe à qual um novo ponto pertence, é levado em consideração a distância desse ponto em relação aos vetores de suporte, bem como a importância destes vetores, que foram aprendidas durante o treinamento. (MULLER; GUIDO, 2017)

De acordo com Tharwat (2019), para dados linearmente separáveis, tem-se a seguinte equação que descreve a linha de separação entre classes:

$$\boldsymbol{w}^T \boldsymbol{x} + b = 0 \tag{2}$$

em que $\boldsymbol{w}$ representa o vetor peso, b determina o deslocamento do hiperplano em relação à origem.

Um modelo SVM procura por valores de $\boldsymbol{w}$ e b que oriente um hiperplano o mais longe possível das amostras mais próximas. Além disso, o SVM busca, ainda, construir dois planos H_1 e H_2, como mostra as equações (3) e (4):

$$H_1 \rightarrow \boldsymbol{w}^T \boldsymbol{x}_i + b = +1 \text{ para } y_i = +1 \tag{3}$$

$$H_2 \rightarrow \boldsymbol{w}^T \boldsymbol{x}_i + b = -1 \text{ para } y_i = -1 \tag{4}$$

em que $\boldsymbol{w}^T\boldsymbol{x}_i+b \geq +1$ representa o plano para a classe positiva e $\boldsymbol{w}^T\boldsymbol{x}_i+b \leq -1$ representa o plano para a classe negativa.

A Figura 2.4 mostra um conjunto de dados linearmente separáveis no espaço de características com a identificação do hiperplano ótimo de separação entre duas classes em um modelo SVM. Nesta figura, também é possível identificar os planos H_1 e H_2, que distam $d_1 = 1/\|\boldsymbol{w}\|$ e $d_2 = 1/\|\boldsymbol{w}\|$, respectivamente, do hiperplano ótimo, sendo $\|\boldsymbol{w}\| = (\boldsymbol{w}^T\boldsymbol{w})^{1/2}$. Estes planos definem os limites das classes no espaço de características e foram determinados com bases nos vetores de suportes, nos quais recaem sobre estes planos. A distância entre os planos H_1 e H_2 é conhecido como margem e seu valor é dado pela soma das distâncias d_1 e d_2, que será igual a $2/\|\boldsymbol{w}\|$.

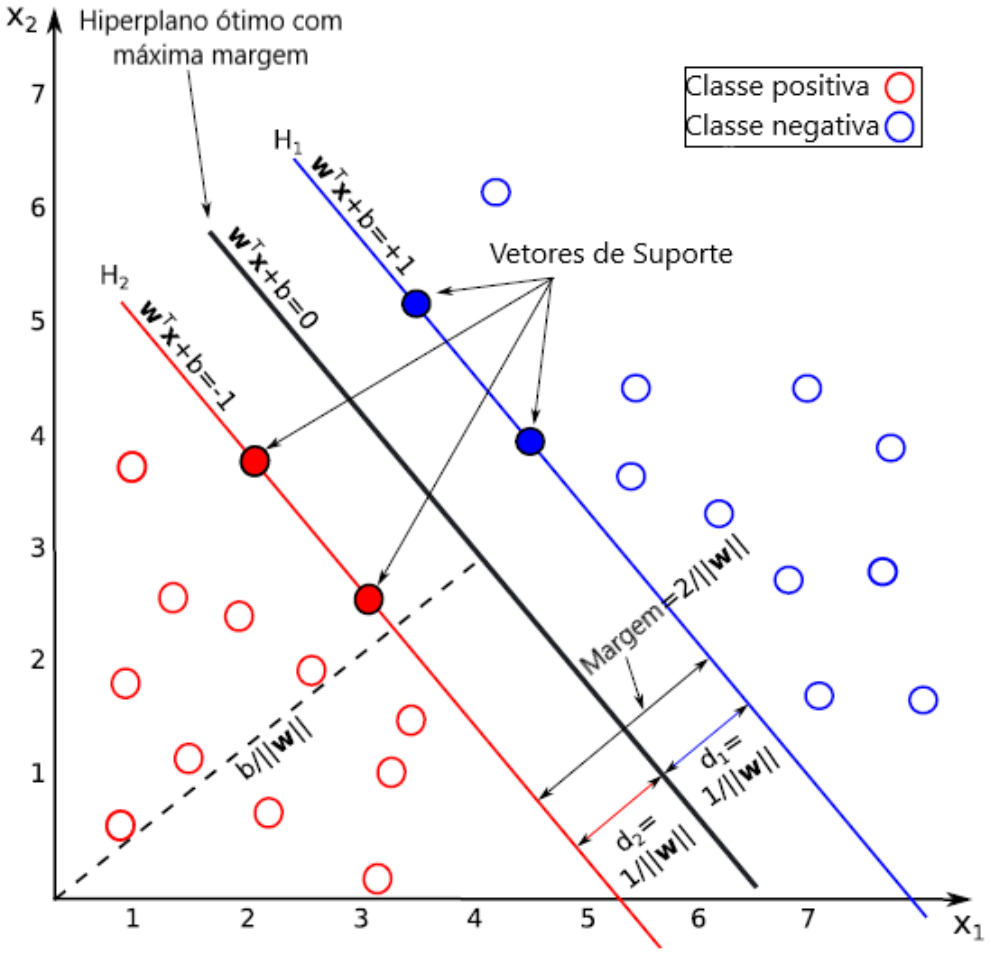


Figura 2.4 - Hiperplano de separação (Adaptado de Tharwat (2019)).

Em um modelo SVM, a largura da margem deve ser maximizada segundo a função objetivo mostrada na equação (5), ficando restringida pela equação (6):

$$\min \frac{1}{2}\|\boldsymbol{w}\|^2 \tag{5}$$

$$y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) - 1 \geq 0 \ \forall_i = 1,\ 2, ..., N \tag{6}$$

Se o conjunto de dados são não separáveis, uma variável de folga, $\varepsilon_i \geq 0$, é introduzida para diminuir as restrições de um modelo SVM linear, como mostram as equações (7) e (8) abaixo:

$$\boldsymbol{w}^T \cdot \boldsymbol{x}_i + b \geq +1 - \varepsilon_i \text{ para } y_i = +1 \tag{7}$$

$$\boldsymbol{w}^T \cdot \boldsymbol{x}_i + b \leq -1 + \varepsilon_i \text{ para } y_i = -1 \tag{8}$$

Após a adição da variável de folga, a função objetivo pode ser descrita pela equação (9) e será restringida pela equação (10).

$$\min \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N} \varepsilon_i \tag{9}$$

$$y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) - 1 + \varepsilon_i \geq 0 \ \forall_i = 1,\ 2, ..., N \tag{10}$$

em que C é o parâmetro de penalidade no qual controla a troca entre o tamanho da margem e a penalidade da variável de folga, ou seja, é responsável por quão tolerante a erros de classificação será o modelo treinado. Um alto valor de C pode ocasionar *overfitting*, resultando numa separação mais completa entre os dados. Em contrapartida, um baixo valor de C pode resultar em *underfitting*, possibilitando maiores erros na fronteira de decisão.

Se o conjunto de dados são não linearmente separáveis, funções de covariância são utilizadas para a transformação dos dados, em que estes são levados para um espaço de alta dimensão através de uma função não linear, ϕ. Neste novo espaço, os dados se tornam linearmente separáveis. Assim, a função objetivo, eq. (9), fica submetida a uma nova restrição, dada pela eq. (11) abaixo:

$$y_i(\boldsymbol{w}\phi(\boldsymbol{x}_i) + b) - 1 + \varepsilon_i \geq 0 \ \forall_i = 1,\ 2, ..., N \tag{11}$$

A função de covariância é definida pelo produto escalar de funções não lineares $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$. As funções de covariância mais utilizadas em modelos SVM são (SCHOLKOPF; SMOLA, 2003):

a) Função Linear, definida por (12):

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle \tag{12}$$

b) Função de Base Radial, do inglês *Radial Basis Function* (RDB), também conhecida como Função Gaussiana, definida por (13):

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(\frac{-\| \boldsymbol{x}_i - \boldsymbol{x}_j \|^2}{2\sigma^2}\right) \tag{13}$$

com $\sigma \in R^+_0$ uma largura de banda escolhida apropriadamente.

c) Função de covariância Polinomial de ordem *d*, definida por:

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle)^d \tag{14}$$

2.3.2. k-Nearest Neighbor - k-NN

Esta técnica leva em consideração a quantidade de pontos vizinhos, armazenados no banco de dados usado para treinamento, próximos ao ponto que se queira determinar a classe. O caso mais simples para esse algoritmo é quando se leva em conta apenas um ponto vizinho para a classificação, ou seja, *k* igual a 1. Neste caso, a classe de um novo ponto que se queira determinar será igual a classe do ponto mais próximo deste.

Quando o valor de *k* é configurado para valores acima de 1, a determinação da classe é realizada através de uma votação. Sendo assim, a classe de um novo ponto será determinada como sendo a classe mais comum entre os *k* vizinhos mais próximos deste. Apesar do k-NN ser uma técnica bastante simples, ela consegue apresentar uma acurácia elevada em diversos problemas, podendo ser aplicada tanto para problemas de classificação binária como para problemas de múltiplas classes.

De forma matemática, seja o conjunto de *k* dados mais próximos de um elemento *x'* de classe *y'* desconhecida dados por (GOU et al., 2012):

$$T = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{k} \tag{15}$$

a distância Euclidiana entre *x'* e x_i é dada pela eq. (16):

$$d(x', x_i) = \sqrt{(x' - x_i)^T (x' - x_i)}. \tag{16}$$

Após o cálculo das distâncias, a determinação da classe é feita com o voto majoritário expresso pela eq. (17):

$$y' = \arg\max_{y} \sum_{(x_i, y_i) \in T'} \delta(y = y_i). \tag{17}$$

onde *y* é uma determinada classe, y_i é a classe do *i-ésimo* vizinho mais próximo e $\delta(y = y_i)$ é a função de Dirac, que assume o valor de 1 se $y = y_i$ e 0 caso contrário.

2.3.3. Bagged Tree - BT

A técnica Bagged Tree (BT) se baseia em outra técnica do ML, que é a Árvore da Decisão, da expressão em inglês *Decision Tree* (DT). A base do processo de classificação da DT se resume a uma série de "perguntas" feitas ao banco de dados, com o intuito de dividir este em diferentes *subsets* até que se chegue a uma decisão final sobre a classe pertencente de cada exemplar. A Figura 2.5 mostra um fluxograma esquemático de uma DT, que é comumente comparado a uma árvore. Ao treinar um algoritmo, este procura por um conjunto de perguntas que produz a melhor performance do processo de classificação.

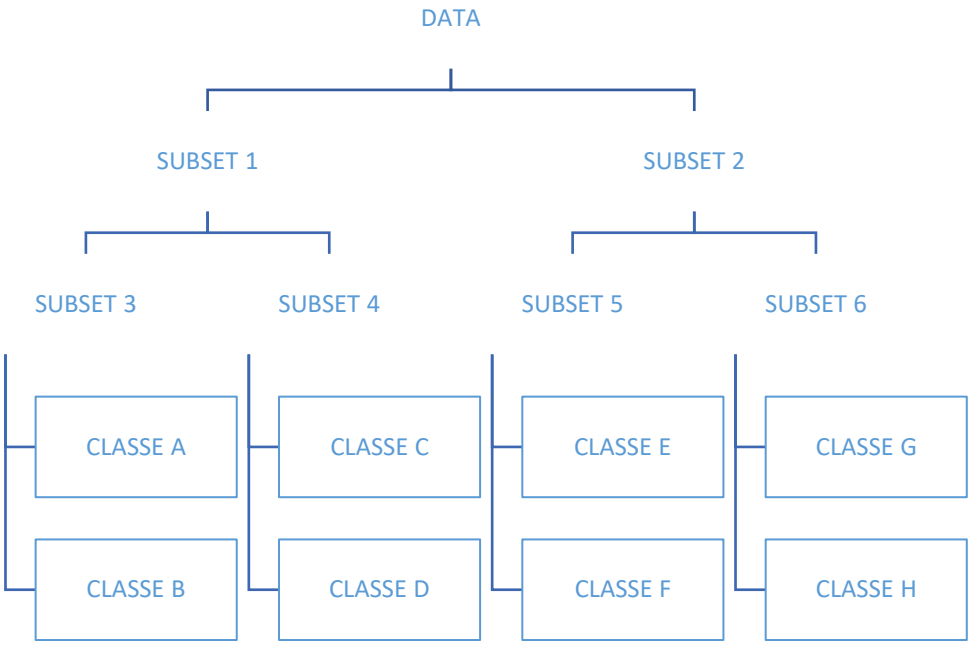


Figura 2.5 - Fluxograma da Decision Tree.

A BT é construída, então, como uma combinação de diversos bancos de dados $p \in R^{+}$, ou "árvores", formados, segundo Breiman (1996), através de um processo chamado *bagging*, um acrônimo da expressão em inglês *bootstrap aggregating*, aplicado ao banco geral de dados. Cada banco de dados *p* contém a mesma quantidade *N* de exemplares que o banco de dados original. Entretanto, para diferenciar um banco de dados do outro, estes podem conter alguns exemplares repetidos e outros faltando. A determinação da classe de cada exemplar se dá por um processo de voto majoritário, onde a decisão da maioria das *p* árvores individuais é levada em consideração. (BREIMAN, 1996)

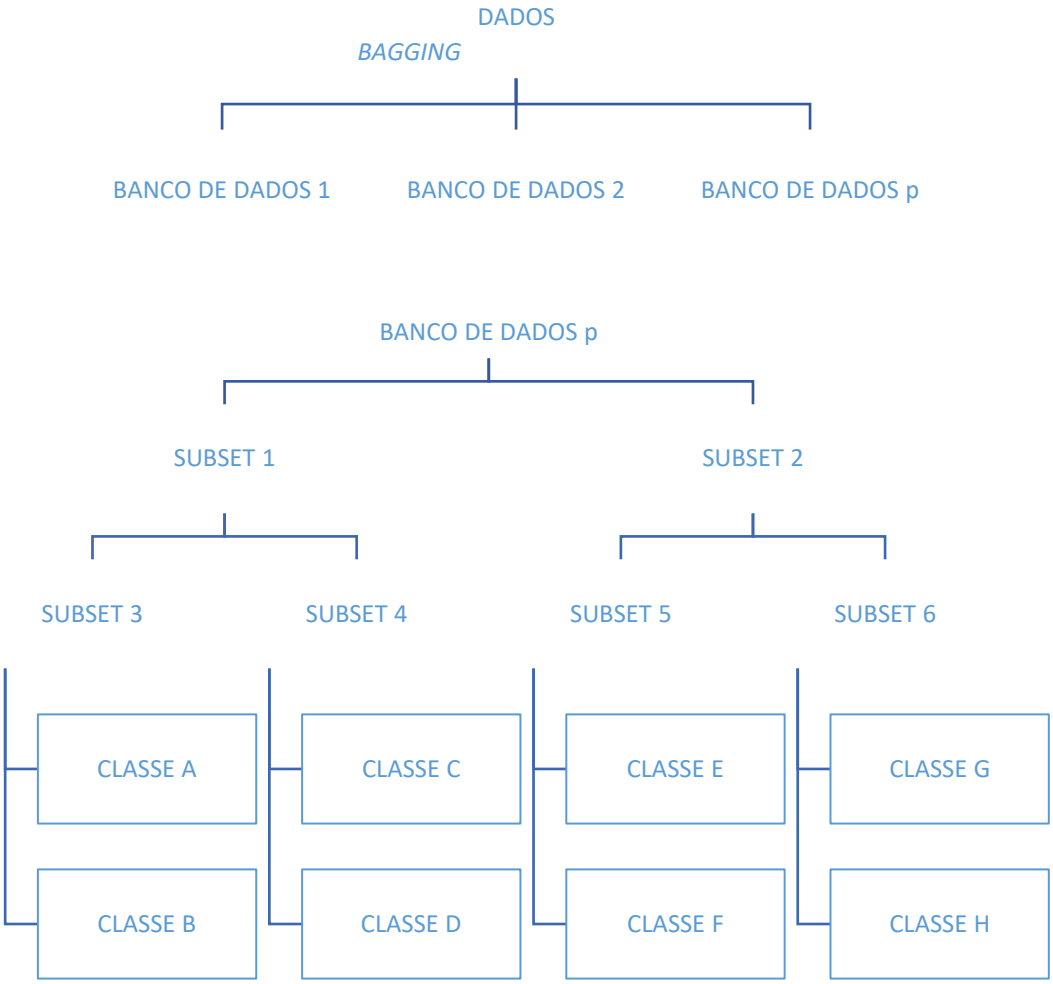


Figura 2.6 - Fluxograma da técnica BT, com a aplicação do *bagging* aos dados de treinamento.

Seja um banco de dados de treinamento baseado no banco de dados *T* definido como:

$$\boldsymbol{X} = (x_1, x_2, x_3, ..., x_N) \tag{18}$$

pode-se montar *B subsets*, aleatórios e com reposição a partir de $\boldsymbol{X}$, definidos como:

$$\boldsymbol{X}^b = (x_1^b, x_2^b, ..., x_N^b) \tag{19}$$

com $b = 1, 2, ..., B$.

Para cada *subset*, monta-se um classificador $C^b(\boldsymbol{x})$, para que se possa obter a classe final pela regra de decisão mostrada na eq. (20) (SKURICHINA; DUIN, 2002).

$$\beta(x) = \arg\max_{y \in \{-1,1\}} \sum_{b=1}^{B} \delta_{\text{sgn}(C^b(\boldsymbol{x})),y} \tag{20}$$

onde y ∈ {-1,1} é uma decisão do classificador e

$$\delta_{i,j} = \begin{cases} 1, i = j \\ 0, i \neq j \end{cases} \tag{21}$$

onde $i = \text{sgn}(C^b(\boldsymbol{x}))$ e $j = y$.

3. METODOLOGIA

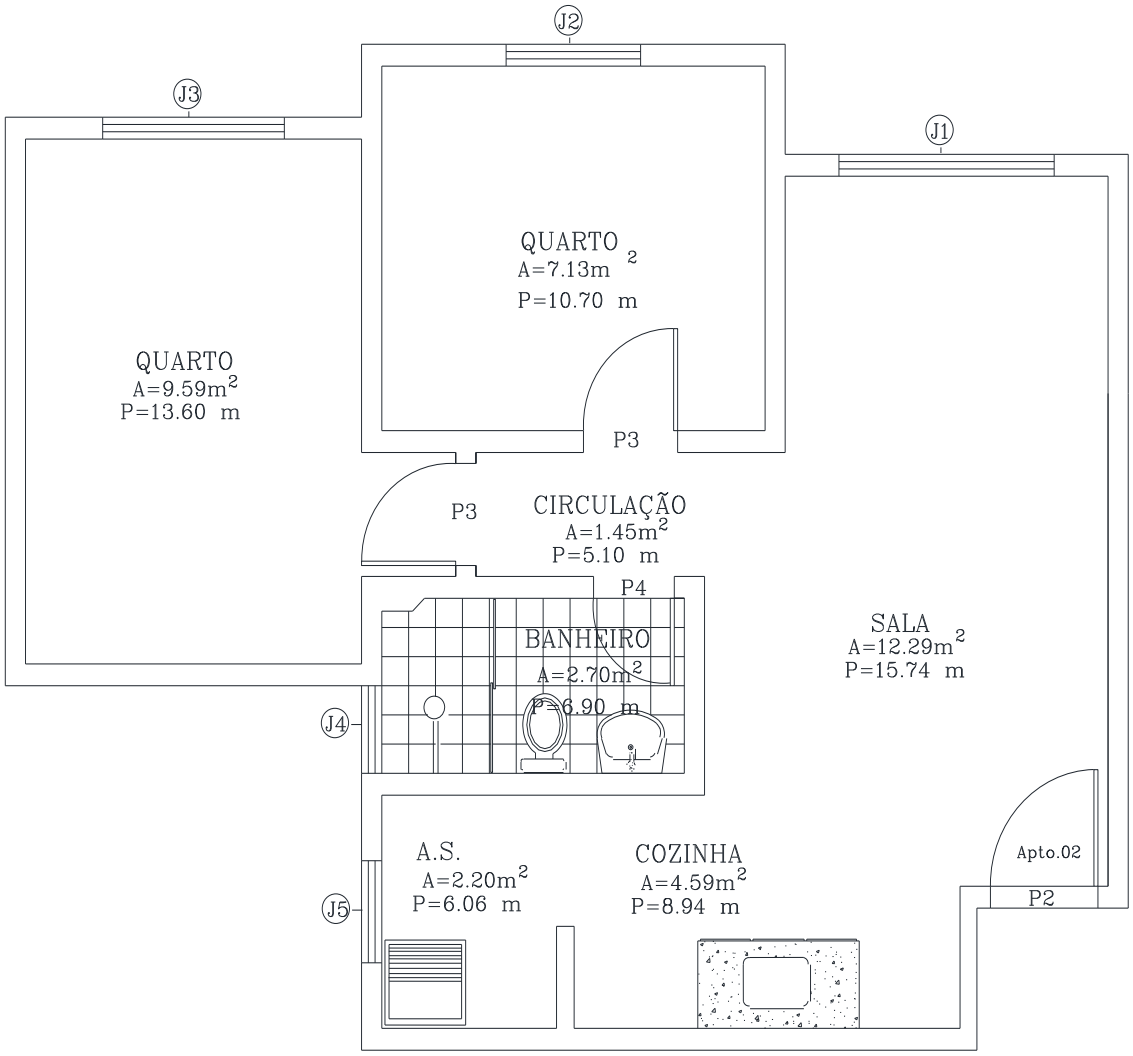
3.1. BANCO DE DADOS

O banco de dados deste trabalho foi montado a partir de um conjunto de projetos arquitetônicos elaborados em CAD, obtidos de diversas fontes da internet e verificados pelo autor. Os modelos utilizados neste estudo são residências unifamiliares, possuindo de um a três quartos, casas térreas, na sua grande maioria, estando presente alguns apartamentos ou casas de dois andares. As áreas totais variam de, aproximadamente, 46 a 138 metros quadrados. Todas as informações extraídas para este estudo são informações que também podem ser acessadas a partir de arquivos IFC.

A partir de uma ferramenta CAD, foram obtidas então informações dos diversos ambientes, tais como área e perímetro. Por inspeção, foram identificadas outras informações como quantidade de portas, janelas bem como a existência, ou não, de torneira e vaso sanitário. Estas informações, que passarão a ser tratadas como variáveis do nosso estudo, foram inseridas em uma planilha de dados, a partir da qual foi possível obter mais variáveis, tais como a relação entre a área e o perímetro de cada ambiente, e a fração da área de cada ambiente em relação a área total de cada unidade residencial. Esta última variável foi inserida de modo a reduzir possíveis erros de classificação para residências de tamanhos diferentes, uma vez que residências maiores tendem a ter ambientes maiores.

A Figura 3.1 mostra um dos modelos utilizados para este estudo, em que é possível identificar os ambientes com suas respectivas áreas e perímetros.

Figura 3.1 - Exemplo de um dos modelos utilizados no estudo.



A estruturação da planilha foi feita da seguinte forma: os diferentes ambientes considerados no estudo, tais como sala, banheiro, quarto, cozinha, dentre outros, foram dispostos em linhas; os atributos, ou variáveis, de cada espaço, tais como informações geométricas, foram organizados em colunas. Tomando o modelo da Figura 3.1 como exemplo, os dados de entrada para o treinamento do algoritmo são mostrados na Tabela 3.1 abaixo:

Tabela 3.1 - Dados de entrada do algoritmo para o modelo da Figura 3.1.

Espaço	A (m^2)	P (m)	Fração Área	Relação A/P	Nº Janelas	Nº Portas	Torneira	Vaso Sanitário
Quarto 1	9,59	13,60	0,7051	0,2401	1	1	0	0
Quarto 2	7,13	10,70	0,6663	0,1785	1	1	0	0
Circulação	1,45	5,10	0,2843	0,0363	0	3	0	0
Banheiro	2,70	6,90	0,3913	0,0676	1	1	1	1
Sala	12,29	15,74	0,7808	0,3076	1	1	0	0
Cozinha	4,59	8,94	0,5134	0,1149	0	0	1	0
A.S.	2,20	6,06	0,3630	0,0551	1	0	1	0

Os dados de área e perímetro são dados em m² e m, respectivamente. A fração da área do ambiente em relação ao total do modelo (Fração Área) e a relação entre a área e o perímetro de cada ambiente (Relação A/P) são adimensionais. As quantidades das janelas e portas são dadas em unidades, e para torneira e vaso sanitário tem-se que 0 indica a ausência e 1 a presença deste item no ambiente.

Após a coleta das informações de todos os modelos, o banco de dados foi dividido em dois grupos. O primeiro grupo, que representa 80% de todos os ambientes analisados, é o grupo de treinamento, no qual será utilizado para o treinamento do algoritmo. Para garantir a representatividade do grupo de treinamento, este foi montado de tal forma que contivesse os valores máximos e mínimos das variáveis "área", "perímetro", "relação A/P" e "fração da área". O segundo grupo, constituído pelos 20% restantes dos dados, é o grupo de teste. Este grupo será utilizado para testagem das predições a partir do algoritmo treinado.

3.2. ALGORITMO

De posse do banco de dados, deu-se início à implementação do algoritmo de treinamento com a utilização do *software* Matlab. Este *software* possui um aplicativo específico com diferentes técnicas de classificação para o treinamento de algoritmos embasados no *machine learning*. Dentre essas técnicas, três foram escolhidas para o desenvolvimento deste estudo: o *Support Vector Machine* (SVM), o *k-Nearest Neighbor* (k-NN) e o *Bagged Tree* (BT). Estas técnicas foram escolhidas dentre as demais por suas relevâncias nas literaturas e por apresentarem, numa rápida avaliação entre as técnicas disponíveis no aplicativo do Matlab, as melhores performances para a classificação de ambientes.

3.2.1. Configurações iniciais

Para cada técnica, é possível que sejam ajustados diversos parâmetros para a realização do treinamento. Estes parâmetros podem ser ajustados de forma manual, um por um, ou através de um processo de otimização, onde este ajuste é feito forma automatizada para que se obtenha os melhores resultados para classificação. Para o SVM, um destes parâmetros ajustáveis é a função de covariância, que pode assumir quatro diferentes formas: linear, gaussiana, quadrática e cúbica. Para este trabalho, apenas três destas quatro funções, foram consideradas, sendo elas a gaussiana, a quadrática e a cúbica. Novamente, a escolha foi feita com base numa rápida avaliação de classificação com o banco de dados.

Considerando, então, para uma melhor análise dos resultados, a subdivisão da técnica SVM em três técnicas, este estudo apresentará uma análise comparativa de cinco diferentes técnicas:

a) *k-Nearest Neighbor* (k-NN);
b) *Bagged Tree* (BT);
c) *Support Vector Machine* com função de covariância gaussiana (SVM-Gaussiana);
d) *Support Vector Machine* com função de covariância quadrática (SVM-Quadrática) e
e) *Support Vector Machine* com função de covariância cúbica (SVM-Cúbica).

Devido à grande quantidade de parâmetros que podem ser ajustados na implementação do código, uma otimização completa poderia resultar em um grande tempo computacional para o treinamento. Assim, apenas os parâmetros indicados na tabela abaixo foram otimizados.

Tabela 3.2 - Parâmetros otimizados do código.

Técnica	Parâmetro	Configuração	Variável no código
Bagged Tree	Número de ciclos de aprendizagem	Otimizado	NumLearningCyles
	Número máximo de ramificações	Otimizado	MaxNumSplits
	Número mínimo de nó folhas	Otimizado	MinLeafSize
	Número de variáveis por amostra	Otimizado	NumVariblesToSample
k-NN	Número de vizinhos	Otimizado	NumNeighbors
	Distância	Otimizado	Distance
	Função peso da distância	Otimizado	DistanceWeight
SVM-Gausiana	Restrição de caixa	Otimizado	BoxConstraint
	Fator de escala da função de covariança	Auto	KernelScale
SVM-Quadrática	Restrição de caixa	Otimizado	BoxConstraint
	Fator de escala da função de covariança	Auto	KernelScale
SVM-Cúbica	Restrição de caixa	Otimizado	BoxConstraint
	Fator de escala da função de covariança	Auto	KernelScale

O algoritmo também foi configurado de forma a normalizar todos os dados de entrada para o treinamento com base nas suas médias e desvios-padrões. Os demais parâmetros não mencionados foram mantidos a configuração *default* do Matlab.

Para o processo de validação do algoritmo realizada ainda na etapa de treinamento, foi utilizado o método de validação cruzada *k-fold*, em que o banco de dados é dividido em *k* partes iguais. Numa primeira fase de treinamento e validação, (k – 1) partes serão usadas para treinamento, enquanto uma parte será usada para a validação. Em seguida, realiza-se mais fases de treinamentos e validações, onde uma nova parte do banco de dados será utilizada para validação e as demais para treinamento. Isso se repete até que todas as partes sejam utilizadas para os testes de validação.

Neste estudo, adotou-se um valor de *k* igual a 5. Na Figura 3.2 tem-se um esquema da divisão do banco de dados para validação *k-fold*, destacando-se todas as fases e partes do conjunto de dados utilizados como validação e treinamento.

	Parte 1	*Parte 2*	*Parte 3*	*Parte 4*	*Parte 5*
Fase 1	Validação	Treinamento	Treinamento	Treinamento	Treinamento
Fase 2	Treinamento	Validação	Treinamento	Treinamento	Treinamento
Fase 3	Treinamento	Treinamento	Validação	Treinamento	Treinamento
Fase 4	Treinamento	Treinamento	Treinamento	Validação	Treinamento
Fase 5	Treinamento	Treinamento	Treinamento	Treinamento	Validação

Figura 3.2 – Estrutura de divisão de um banco de dados para validação *k-fold*.

3.2.2. Repetições do processo de treinamento

Em alguns testes do algoritmo, enquanto este era implementado, notou-se que a acurácia possuía uma certa variação a cada novo processo de treinamento. Para se ter uma acurácia mais representativa deste trabalho, o código foi implementado de forma a realizar várias repetições do processo de treinamento/validação. O número de repetições foi definido como 100. Procurou-se um número de repetições grande o suficiente para se ter uma acurácia mais representativa possível, porém não tão grande para não se ter um tempo de treinamento muito oneroso. A cada treinamento, a acurácia foi calculada e, no fim, é calculada a acurácia médias de todas as repetições. A Figura 3.3 apresenta o esqueleto principal do algoritmo.

Figura 3.3 - Estrutura principal do algoritmo.

É importante ressaltar que, apesar de haver 100 repetições do processo de treinamento, a realização de novas predições será feita com base em um único processo de treinamento. Nota-se, também, que o processo de otimização é realizado apenas uma vez, mantendo, assim, os mesmos parâmetros para todas as 100 repetições. Com o algoritmo treinado, é possível então, prever a classificação de um determinado espaço com base nas variáveis de entrada.

3.2.3. Classificador multiclasse

O algoritmo para a realização do treinamento, como descrito em 3.2.1 e 3.2.2, foi utilizado, inicialmente, para a montagem de um modelo classificador que aqui será chamado de classificador multiclasse. Neste classificador, todas as classes de ambientes são levadas em consideração no processo de treinamento. O algoritmo recebe como entrada os dados com todas as classes em análise e define os parâmetros que melhor classifique todas estas classes simultaneamente. Na Figura 3.4 tem-se o fluxograma deste classificador, em que a partir do banco de dados, o classificador determina as classes de todos os elementos.

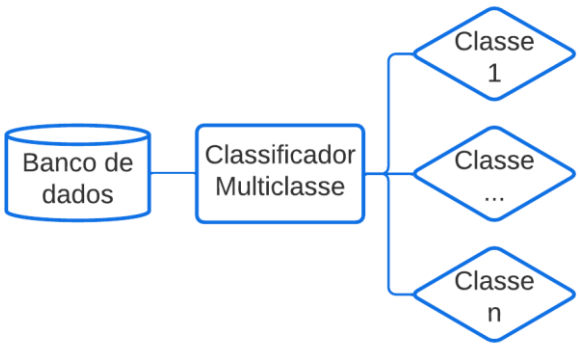


Figura 3.4 - Fluxograma do classificador multiclasse.

3.2.4. Classificador binário e classificador multiclasse-binário

Foi criado, também, um algoritmo para a realização de classificação binária de ambientes, ou seja, o banco de dados deve apresentar apenas duas classes de ambientes. Dessa forma, o classificador binário irá determinar os parâmetros de classificação que melhor distinga uma determinada classe das demais. Na Figura 3.5 tem-se o fluxograma dos classificadores binários. Assim, haverá um classificador para cada classe.

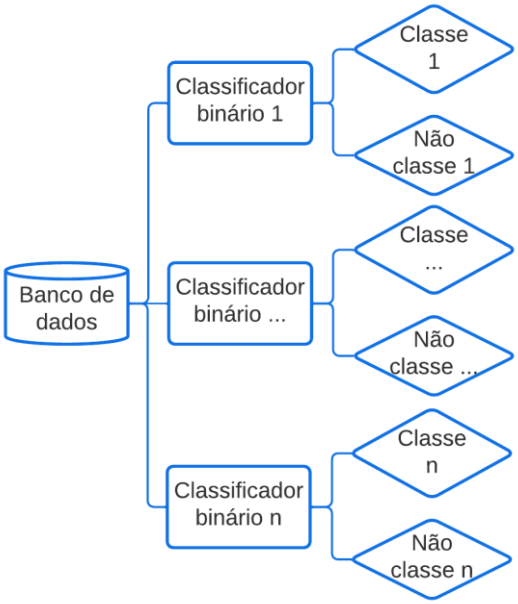


Figura 3.5 - Fluxograma do classificador binário.

Uma vez que, no estudo aqui proposto, há diversas classes de ambientes a serem determinados, foi implementado um algoritmo complementar para que se pudesse proceder com a classificação do todas as classes de espaços residenciais. Este algoritmo complementar se baseia em uma metodologia de classificação denominada *One versus All* (OvA), em que uma classe é comparada com todas as demais classes, e será tratado como algoritmo de classificação multiclasse-binária, uma vez que fará a classificação de todas as classes baseado nos resultados do classificador binário.

A classificação multiclasse-binária, neste estudo, é um processo complementar à classificação binária que utilizará o os resultados deste classificador para a determinação de todas as classes presentes no banco de dados. Assim, o procedimento para classificação funciona da seguinte forma. Primeiramente, utiliza-se o classificador binário em que o banco de dados utilizado para treinamento é replicado *n* vezes, em que *n* representa o total de classes presente nele. Em cada réplica tem-se apenas duas classes definidas, uma classe representa um ambiente em análise e a outra classe representa os demais ambientes. Haverá, então, *n* treinamentos binários. Em seguida, será realizado predições com todos os algoritmos binários treinados para a obtenção dos *scores* de classificação. Esses *scores* medem, direta ou indiretamente, a probabilidade de determinado ambiente pertencer a determinada classe e variam de 0 a 1. Por fim, com o

classificador multiclasse-binário, é feita a comparação entre os *scores* das predições de todos os classificadores binários para se determinar a classe na qual o ambiente pertence.

Por exemplo, para um banco de dados com três classes, A, ..., N, será realizado *n* treinamentos binários, cada um para a diferenciação entre os ambientes “N” e os demais. Com esses classificadores treinados, será utilizado a função de predição para a obtenção do score de cada classe. Essa função, ao passar pelo primeiro algoritmo (“A” x “demais”), determinará a probabilidade de o ambiente pertencer à classe A. Ao passar pelos demais algoritmos, retornará a probabilidade de o ambiente pertencer à cada uma das demais classes. Estes *scores* serão então comparados e o ambiente será classificado como da classe que retornou o maior valor. Na Figura 3.6 tem-se o fluxograma do classificador multiclasse-binário.

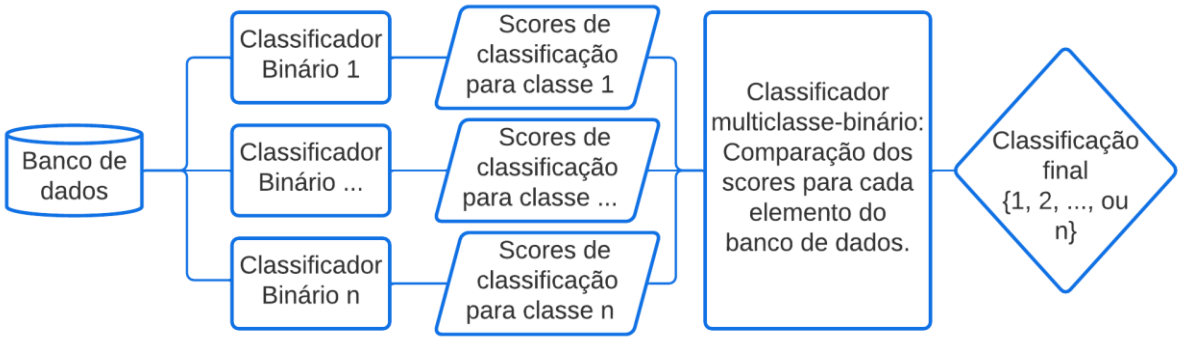


Figura 3.6 - Fluxograma do classificador multiclasse-binário.

3.2.5. Classificador *Ensemble*

A fim de se obter um classificador mais eficiente e mais consistente, é comum, no campo de estudos de *machine learning*, a utilização de métodos *ensemble*, na qual são combinados diversos algoritmos em um único modelo preditivo para realizar determinada classificação. Neste trabalho, foi implementado um modelo de classificador *ensemble* que combina a predição das cinco técnicas aqui analisadas a partir dos resultados do classificador multiclasse-binário.

Para cada uma das cinco técnicas, foram realizados os processos de classificação binária e classificação multiclasse-binária mencionados em 3.2.4. Para a determinação da classe de um determinado ambiente, o classificador *ensemble* utilizará um processo de votação simples. Ou seja, a classe predita pela maioria dos cinco classificadores multiclasse-binária para cada

amostra será a classe definida pelo classificador *ensemble*. Na Figura 3.7 tem-se o fluxograma deste classificador.

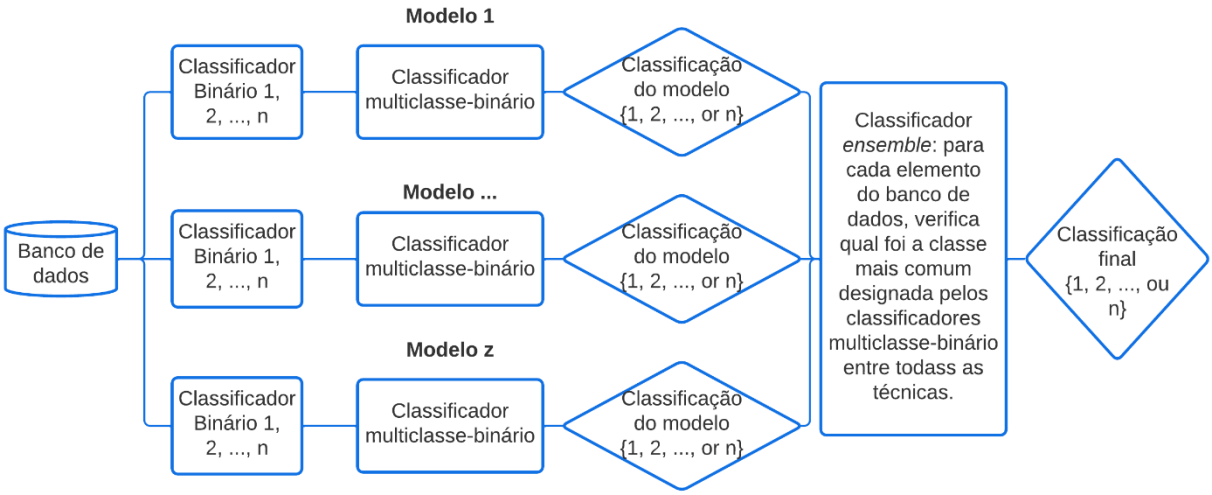


Figura 3.7 - Fluxograma do classificador *ensemble*.

3.3. MEDIDORES DE PERFORMANCE

As performances dos algoritmos foram medidas e analisadas a partir da matriz de confusão, da curva ROC e dos seguintes medidores, acurácia (AC) e *F1-score*. A matriz de confusão fornece uma análise detalhada, onde é possível visualizar como cada um dos dados foram classificados, sendo possível obter a quantidade de verdadeiros positivos (VP), verdadeiros negativos (VN), falsos positivos (FP) e falsos negativos (FN). Segundo Chicco e Jurman (2020), os VP's são os dados que pertencem à classe positiva e foram classificados como positivos, enquanto os VN's são os dados que pertencem à classe negativa e foram classificados como negativos. Os FP's são os dados que pertencem à classe negativa e foram classificados como positivos e os FN's são os dados que pertencem à classe positiva e foram classificados como negativos. São denominados dados positivos aqueles que pertencem à classe em análise e dados negativos os demais dados. A matriz de confusão também pode ser visualizada como mostra a Figura 3.8.

Figura 3.8 - Esquema de uma matriz de confusão.

		Categoria Predita		
		A	**B**	**C**
Categoria verdadeira	**A**	*Ambientes que pertencem à classe A e foram classificados como classe A*	*Ambientes que pertencem à classe A e foram classificados como classe B*	*Ambientes que pertencem à classe A e foram classificados como classe C*
	B	*Ambientes que pertencem à classe B e foram classificados como classe A*	*Ambientes que pertencem à classe B e foram classificados como classe B*	*Ambientes que pertencem à classe B e foram classificados como classe C*
	C	*Ambientes que pertencem à classe C e foram classificados como classe A*	*Ambientes que pertencem à classe C e foram classificados como classe B*	*Ambientes que pertencem à classe C e foram classificados como classe C*

A curva ROC, do inglês *Receiver Operating Characteristics*, é uma curva bidimensional que mostra a taxa de verdadeiros positivos, representada no eixo *y*, e a taxa de falsos positivos, representada no eixo *x*. A curva é construída em diversos passos, onde cada um representa um ponto da curva. Em cada passo, um valor limite de score é determinado e este valor varia do maior ao menor score. As amostras são então classificadas como positivas ou negativas com base nestes valores limites e as taxas de falso positivo e verdadeiro positivos são plotadas. Ao fim das análises, tem se a curva, de forma semelhante à mostrada na Figura 3.9. Quanto maior a área abaixo da curva, ou seja, mais distante da linha tracejada, melhor é a classificação. O classificador ideal seria dado pela curva que passa pelo ponto (0,1). (THARWAT, 2018)

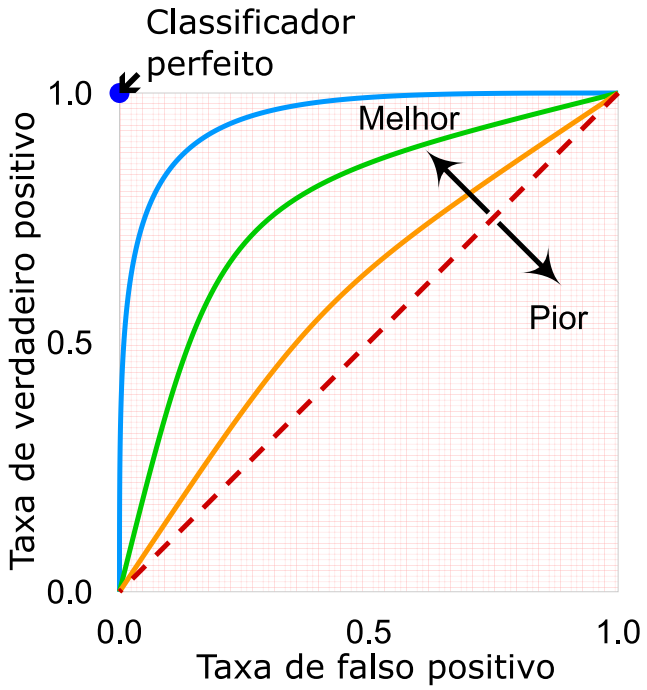


Figura 3.9 - Curva ROC (Adaptado de cmglee, MartinThoma).

A acurácia mede a porcentagem de dados corretamente classificados em relação aos dados analisados, podendo variar de 0% a 100%, e é dado pela equação abaixo. Este medidor é importante para banco de dados homogêneos e, também, quando os dados classificados como VP possuem importância significativa nas análises.

$$AC = \frac{VP + VN}{VP + VN + FP + FN} \tag{22}$$

O *F1-score*, definido como a média harmônica de outros dois dados estatísticos, precisão e revocação, varia de 0 a 1 e, quanto mais próximo de 1, melhor o classificador. Este medidor é importante para banco de dados não homogêneo e, também, quando os dados classificados como FP e FN são relevantes para as análises. Ele pode ser calculado da seguinte forma (CHICCO; JURMAN, 2020):

$$F_1 = \frac{2VP}{2VP + FP + FN} \tag{23}$$

3.4. SAÍDA DE DADOS

A Figura 3.10 apresenta o fluxograma geral do processo de classificação. A partir dos dados de entrada, obtidos de modelos arquitetônicos, será possível, com o modelo ML treinado e validado, determinar a classe à qual um espaço residencial pertence. Essa informação ficará, então, disponível para que possa ser inserida novamente no modelo de origem, possibilitando o enriquecimento semântico.

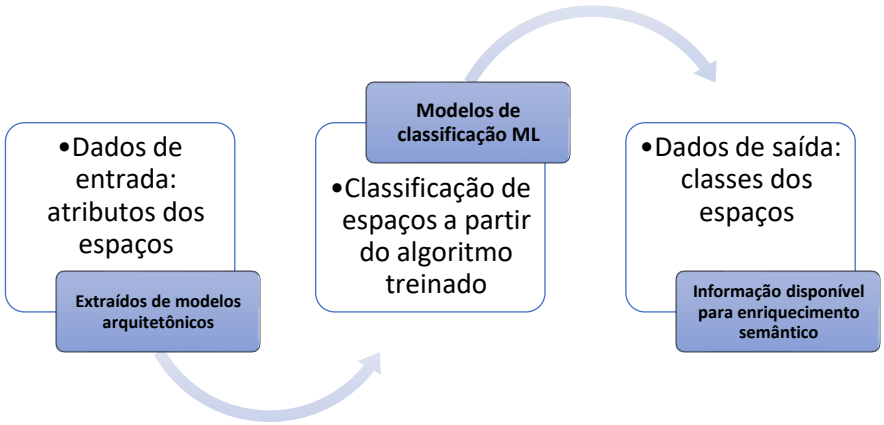


Figura 3.10 - Processo geral de classificação.

4. RESULTADOS

Os resultados são apresentados em 3 cenários:

a) Cenário I: treinamento realizado com um classificador multiclasse e com o banco de dados sem nenhuma modificação;
b) Cenário II: treinamento realizado com classificadores binário, multiclasse-binário e *ensemble* após a homogeneização do banco de dados e inserção de novos atributos em relação ao cenário I.
c) Cenário III: treinamento realizado com classificadores binário, multiclasse-binário e *ensemble* após a homogeneização do banco de dados e inserção de novos atributos em relação ao cenário II.

A Tabela 4.1 mostra os atributos que foram analisados em cada um dos cenários, bem como o tipo de atributo, classificado como numérico ou categórico.

Atributos dos ambientes	Tipo	Cenário
Área	Numérico	I, II, III
Perímetro	Numérico	I, II, III
Fração da área	Numérico	I, II, III
Relação A/P	Numérico	II, III
Número de janelas	Numérico	I, II, III
Número de portas	Numérico	I, II, III
Presença de torneira	Categórico	III
Presença de vaso sanitário	Categórico	III

Tabela 4.1 - Atributos dos ambientes.

4.1. CENÁRIO I - TREINAMENTO COM O CLASSIFICADOR MULTICLASSE

4.1.1. Dados de entrada

Originalmente, o banco de dados possuía 435 exemplares obtidos de 47 diferentes modelos de edificações residenciais com 12 classes distintas de espaços, distribuídos como mostrado na Tabela 4.2, em que A.S representa a área de serviço.

	Classes	Exemplares
1	Quarto	111
2	Circulação	52
3	Banheiro	70
4	Sala	59
5	Cozinha	47
6	A.S.	30
7	Garagem	17
8	Varanda	28
9	Escada	6
10	Closet	8
11	Depósito	5
12	Lavabo	2
	Total	**435**

Tabela 4.2 - Classes e quantidade de exemplares do banco de dados original.

Entretanto, devido à pequena quantidade de exemplares dos espaços “Escada”, “Closet”, “Depósito” e “Lavabo”, estes foram excluídos das análises, o que resultou em um banco de dados para treinamento de 414 exemplares e 8 classes distintas de espaços, como apresentados na Tabela 4.3:

	Classes	Exemplares
1	Quarto	111
2	Circulação	52
3	Banheiro	70
4	Sala	59
5	Cozinha	47
6	A.S.	30
7	Garagem	17
8	Varanda	28
	Total	**414**

Tabela 4.3 - Classes e quantidade de exemplares do banco de dados inicial – cenário I.

Numa primeira análise, foram utilizados como dados de entrada para o treinamento as seguintes variáveis: área, perímetro, fração da área do espaço em relação à área total do modelo, número de portas e número de janelas. A Tabela 4.4 a seguir apresenta um resumo dos dados de entrada com alguns dados estatísticos:

	Mínimo	Máximo	Média	Desvio Padrão	1° Quartil	2° Quartil	3° Quartil
Área (m^2)	1,23	49,83	10,71	7,74	4,94	9,10	14,00
Perímetro (m)	4,00	34,07	13,36	5,12	9,41	12,60	15,72
Fração da área	0,0051	0,5096	0,1121	0,0843	0,0515	0,0875	0,1476
Número de janelas	0	6	-	-	-	-	-
Número de portas	0	5	-	-	-	-	-

Tabela 4.4 - Resumo dos dados de entrada do banco de dados do cenário I.

Para o treinamento deste classificador, foi utilizado todo o banco de dados mostrado na Tabela 4.3, sem dividi-lo em um grupo para treinamento e outro para teste. Nota-se também, que as variáveis correspondentes da relação entre área e perímetro, presença de torneira e vaso sanitário ainda não foram levadas em consideração.

4.1.2. Performance do treinamento

Foi feito, então, o treinamento do algoritmo de classificação multiclasse, paras as cinco técnicas analisadas neste trabalho, utilizando-se a validação cruzada k-fold, com k igual a 5. A Tabela 4.5 mostra alguns resultados obtidos deste treinamento, em que são mostrados a AC média geral de todas as 100 repetições para cada técnica, bem como o desvio padrão das acurácias, o F1-score e o tempo total de treinamento para cada uma das técnicas.

Técnica	AC média %	*F1*	Desvio padrão	Tempo (s)
BT	61,96	0,5256	0,0146	250,80
k-NN	63,21	0,5782	0,0108	70,40
SVM-Gaussiana	63,45	0,5666	0,0144	176,40
SVM-Quadrática	63,72	0,5446	0,0112	944,40
SVM-Cúbica	59,99	0,5129	0,0176	5.186,00

Tabela 4.5 – Performance do classificador multiclasse – cenário I.

Analisando a performance do classificador multiclasse no cenário I, na Tabela 4.5, nota-se que as acurácias para as cinco técnicas não foram satisfatórias. Essa baixa performance pode estar ligada à heterogeneidade do banco de dados, uma vez que o banco de dados possui 111 exemplares de quarto e 17 de garagem, e à baixa quantidade de variáveis consideradas em comparação com a quantidade de classes de ambientes. Apesar disso, para este classificador, é possível notar que as técnicas k-NN, SVM-Gaussiana e SVM-Quadrática apresentaram os

melhores desempenho, sendo que a técnica SVM-Quadrática obteve uma acurácia um pouco maior.

Com relação ao tempo de treinamento, nota-se que a técnica k-NN levou menos tempo no processo de otimização e treinamento que as demais técnicas, ao passo que a técnica SVM-Cúbica apresentou um tempo bem maior que os demais. Comparando o tempo de treinamento e as acurácias, nota-se que, apesar de simples, a técnica k-NN se mostrou bastante eficaz. Já a técnica SVM-Cúbica, se mostrou mais complexa e com baixa eficácia.

Na busca de resultados melhores que os apresentados pelo classificador multiclasse, foi feito o treinamento do banco de dados com os classificadores binário, multiclasse-binário e *ensemble*, que serão mostrados a seguir.

4.2. CENÁRIO II – CLASSIFICADORES BINÁRIO, MULTICLASSE-BINÁRIO E *ENSEMBLE*

Para este treinamento, as seguintes mudanças foram adotadas em relação ao classificador multiclasse, com o intuito de se obter uma melhor performance:

a) Foi feita uma homogeneização do banco de dados, uma vez que algumas classes apresentam menores quantidades de exemplares em relação à outras, tais como área de serviço, garagem e varanda. Essa desproporcionalidade no quantitativo de exemplares para as diferentes classes pode gerar um classificador tendencioso. Assim, uma das formas de contornar este problema é aumentando o quantitativo das classes minoritárias replicando sua amostra, de forma aleatória (KAUR; PANNU; MALHI, 2019)(BLAGUS; LUSA, 2010)(JAPKOWICZ; STEPHEN, 2002). Assim, a quantidade de exemplares da área de serviço e varanda foram duplicados, enquanto a quantidade de exemplares da garagem foi triplicada;
b) Foi incluída a variável "Relação A/P" nas análises, como indicado na Tabela 4.1;
c) O banco de dados foi dividido em grupo de treinamento e grupo de teste, representando 80% e 20% do bando de dados geral, respectivamente. Nessa divisão, foi mantido os valores máximos e mínimos dos dados no grupo de treinamento.

Assim, o banco de dados passa a ter as seguintes quantidades de exemplares:

	Classes	**Exemplares**		
		Total	**Treinamento (80%)**	**Teste (20%)**
1	A.S.	60	88	23
2	Banheiro	70	42	10
3	Circulação	52	56	14
4	Cozinha	47	41	10
5	Garagem	51	47	12
6	Quarto	111	48	12
7	Sala	59	38	9
8	Varanda	56	45	11
	Total	**506**	**405**	**101**

Tabela 4.6 - Classes e quantidade de exemplares do banco de dados total, de treinamento e de teste – cenário II.

4.2.1. Dados de entrada

Com as modificações feitas, o banco de dados utilizado para treinamento passa a ter as seguintes características:

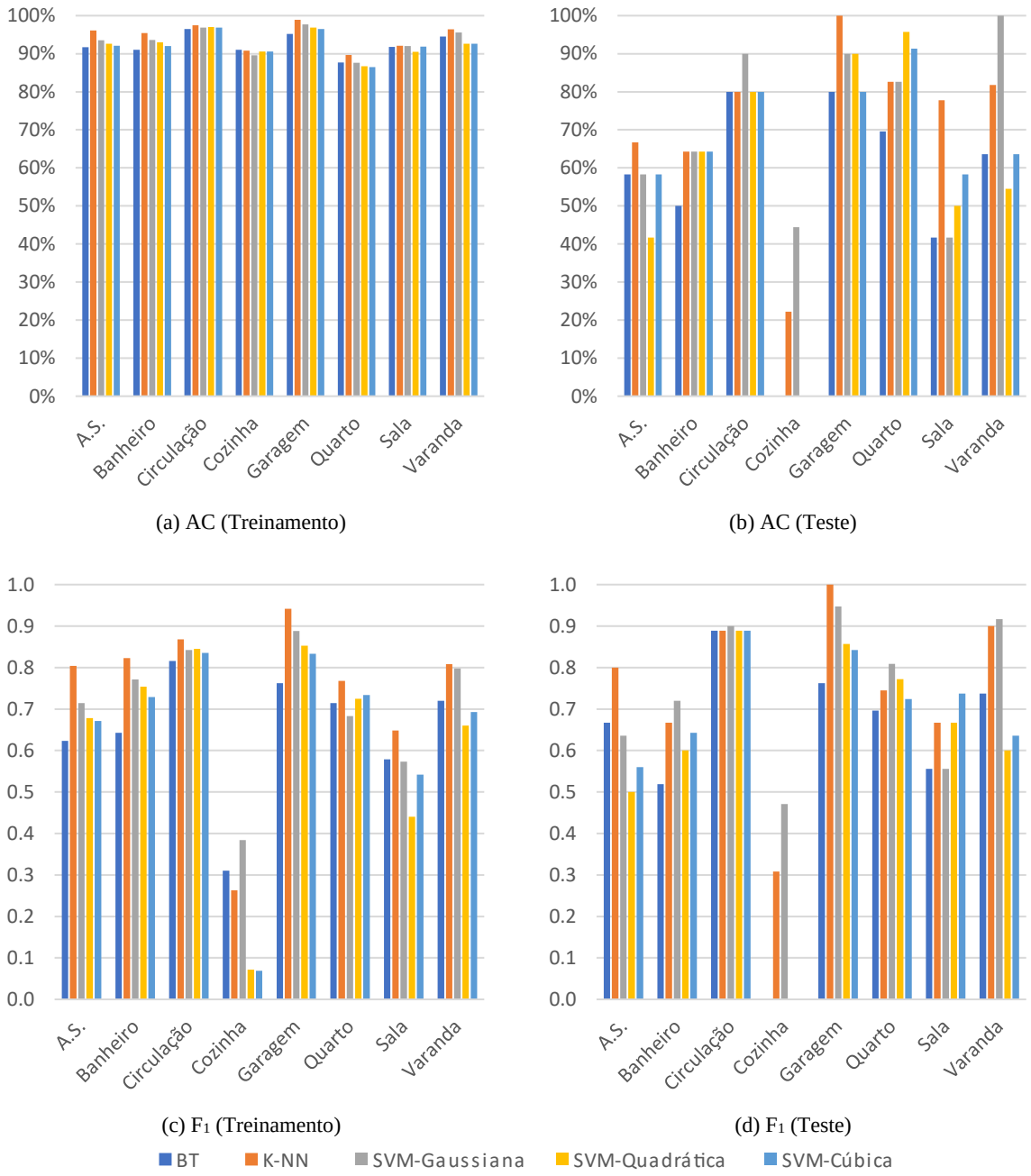
	Mínimo	Máximo	Média	Desvio Padrão	1° Quartil	2° Quartil	3° Quartil
Área (m^2)	1,23	49,83	11,30	8,81	4,95	9,04	15,02
Perímetro (m)	4,00	34,07	13,73	5,62	9,40	12,61	16,20
Fração da área	0,0051	0,5096	0,1115	0,0844	0,0514	0,0841	0,1477
Relação A/P	0,1246	1,8284	0,7313	0,2856	0,5000	0,6937	0,9099
Número de janelas	0	6	-	-	-	-	-
Número de portas	0	5	-	-	-	-	-

Tabela 4.7 - Resumo dos dados de entrada do banco de treinamento – cenário II.

A partir destes dados, foi feito, então, o treinamento dos algoritmos com os classificadores binário, multiclasse-binário e ensemble.

4.2.2. Performance do treinamento com o classificador binário

Por se tratar de um classificador binário, foram realizados 8 treinamentos, um para cada classe de ambiente. Em cada treinamento, foi seguido as etapas do algoritmo, como mostrado na Figura 3.3. Sendo assim, para cada técnica analisada, cada classe apresentará medidas de AC e *F1*, que medem a performance do classificador para distinguir se o ambiente pertence à classe em análise (classe positiva) ou à outra classe (classe negativa). A Figura 4.1 apresenta a média da AC e do *F1* dos treinamentos realizados para todas as 8 classes de ambientes analisadas dos grupos de teste e treinamento. A Tabela A.1, no apêndice, apresenta os resultados, em forma tabular, para todas as técnicas do grupo de treinamento e do grupo de teste, este último com valores entre parêntese.



(a) AC (Treinamento)

(b) AC (Teste)

(c) F_1 (Treinamento)

(d) F_1 (Teste)

Figura 4.1 - Performance do classificador binário para os grupos de treinamento e teste – Cenário II.

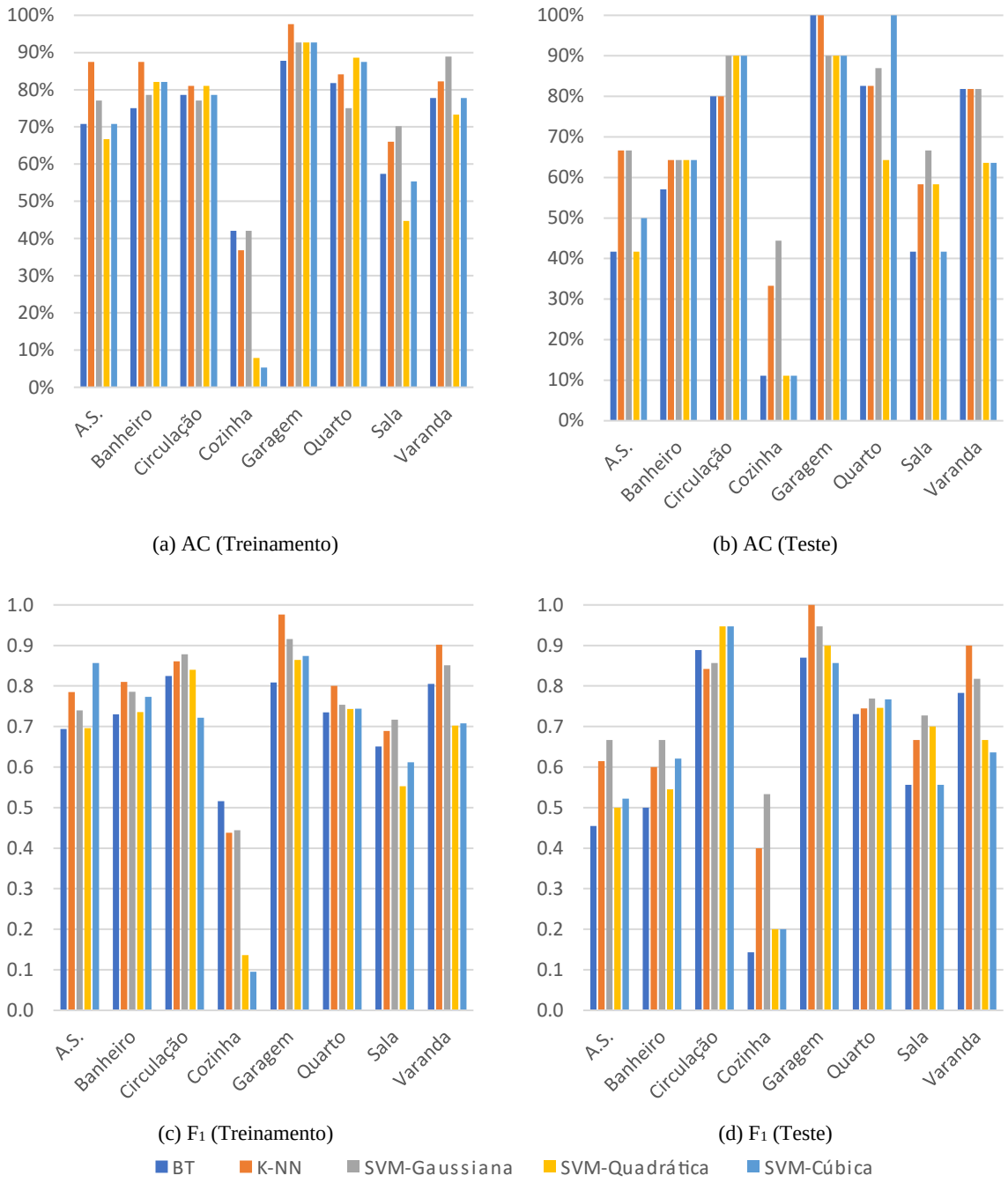
Nota-se que as AC alcançaram valores por volta de 90%, apesar de que para algumas técnicas no grupo de testes, esta ficou por volta de 50%. Em uma classificação binária, AC de 50% significa uma classificação aleatória, na qual não seria adequada para resolver este tipo de problema. Ainda, pela Figura 4.1, a técnica k-NN obteve a melhor performance para a grande maioria das classes analisadas. Nota-se, também, que algumas AC e *F1* foram iguais a 0 para o grupo de testes, e que o desempenho para este grupo foi menor em relação ao grupo de treinamento, o que pode indicar que a quantidade de dados analisados não foi grande o

suficiente para que o algoritmo pudesse realizar o treinamento adequado. Outra explicação para este resultado é o que se chama de *overfitting*, que é quando um modelo consegue descrever bem o que acontece com os dados de treinamento, mas quando é utilizado para descrever novos dados, apresenta uma diminuição da performance. O ambiente "Quarto" foi o que apresentou menores AC, enquanto o ambiente "Cozinha" apresentou menores *F1*, o que significa que as variáveis analisadas não foram suficientes para a distinção destas classes.

4.2.3. Performance do classificador multiclasse-binário

Como dito anteriormente, o classificador multiclasse-binário compara os *scores* de cada classificador binário para a determinação da classe de cada exemplar. Essa comparação foi feita tanto com os dados utilizados no treinamento quanto os dados utilizados no teste. Os *scores* dos dados de treinamento são gerados durante o próprio treinamento do algoritmo, enquanto, para a obtenção dos *scores* dos dados de teste, utilizou-se os classificadores binários já treinados para a realização da predição.

A Figura 4.2 apresenta os valores de AC e *F1*, tanto para o grupo de treinamento quanto para o grupo de teste. O classificador k-NN ainda apresentou a melhor performance dentre os métodos de classificação para o grupo de treinamento, mas para o grupo de teste, o classificador SVM-Gaussiana já apresentou o melhor desempenho. Assim como no classificador binário, a "Cozinha" e é a classe que apresentou o pior desempenho para o classificador multiclasse-binário, enquanto a "Garagem" apresentou o melhor desempenho. Ainda, a diferença de valores entre os grupos de treinamento e teste foram menores para o classificador multiclasse-binário do que para o classificador binário, indicando que a estratégia adotada diminuiu o problema de *overfitting*. A Tabela A.2, no apêndice, resume a performance de todas as técnicas analisadas, na forma tabular, para cada classe.



(a) AC (Treinamento)

(b) AC (Teste)

(c) F_1 (Treinamento)

(d) F_1 (Teste)

Figura 4.2 - Performance do classificador multiclasse-binário para os grupos de treinamento e teste – Cenário II.

Na Tabela 4.8, tem-se a performance geral deste classificador, tanto para os dados do grupo de treinamento quanto para os dados do grupo de teste, entre parênteses. Para o classificador multiclasse-binário do cenário II, apesar das acurácias serem menores que as acurácias do classificador binário, percebe-se que houve uma melhora no desempenho, se comparado com a performance do classificador multiclasse. Essa diferença de desempenho entre os classificadores binários e multiclasse-binário já é de se esperar, uma vez que, no classificador binário, apenas duas classes são levadas em consideração em cada análise, enquanto no

classificador multiclasse-binário, todas as 8 classes são levadas em consideração. Além disso, nota-se que a técnica k-NN apresentou o melhor desempenho, seguido pela técnica SMV-Gaussiana.

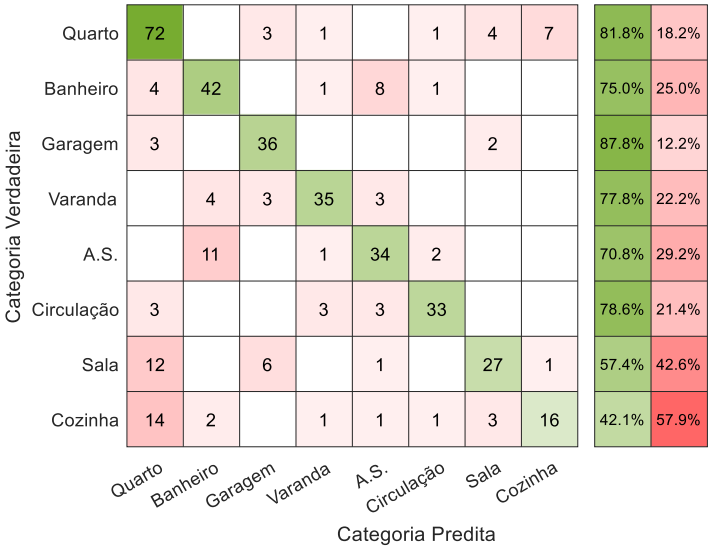
BT	k-NN	SVM-Gaussiana	SVM-Quadrática	SVM-Cúbica
72.8% (64.5%)	79.3% (72.3%)	76.5% (75.3%)	70.4% (68.3%)	71.9% (68.3%)
0.721 (0.616)	0.783 (0.721)	0.761 (0.748)	0.659 (0.651)	0.673 (0.638)

Tabela 4.8 - Performance geral do classificador multiclasse-binário – cenário II.

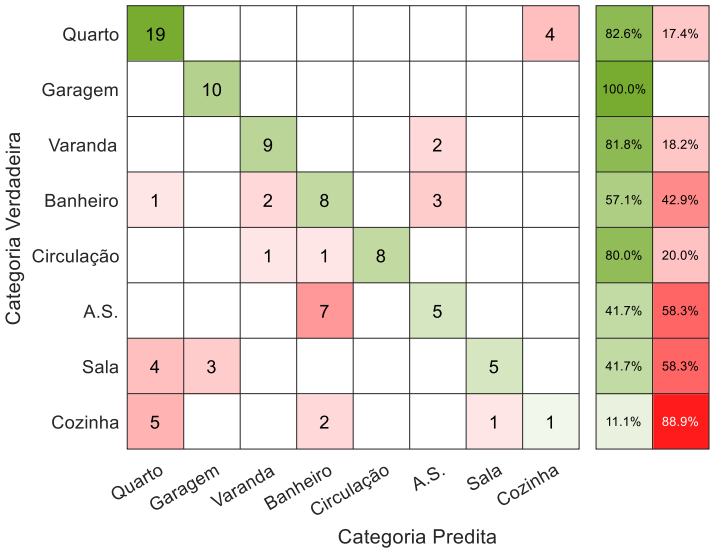
A partir da determinação das classes, tem-se as matrizes de confusão, apresentadas a seguir, para todas as técnicas. As colunas representam as classes de ambientes preditas pelo classificador e as linhas representam as classes verdadeiras. Por exemplo, observando a interseção da primeira linha com a primeira coluna da Figura 4.3, sabe-se que 72 “Quartos” foram preditos corretamente como “Quartos”. A interseção da primeira linha com a última coluna mostra que sete “Quartos” foram preditos como “Cozinha”.

Junto com cada matriz de confusão, é possível observar, nas duas últimas colunas, as porcentagens de acerto e erro, respectivamente, de cada classe, para o classificador multiclasse-binário. A coluna em verde representa a porcentagem de acertos (acurácia) enquanto a coluna em vermelho representa a porcentagem de erros. Assim, observando a primeira linha da Figura 4.3, de todos os “Quartos” presente no banco de dados de treinamento, 81,8% foram classificados corretamente, ao passo que, pela última linha, nota-se que a “Cozinha” teve uma porcentagem de 42,1% de exemplares classificados corretamente.

Figura 4.3 - Matriz de confusão do classificador multiclasse-binário da técnica BT para o grupo de (a) treinamento e (b) teste – cenário II.

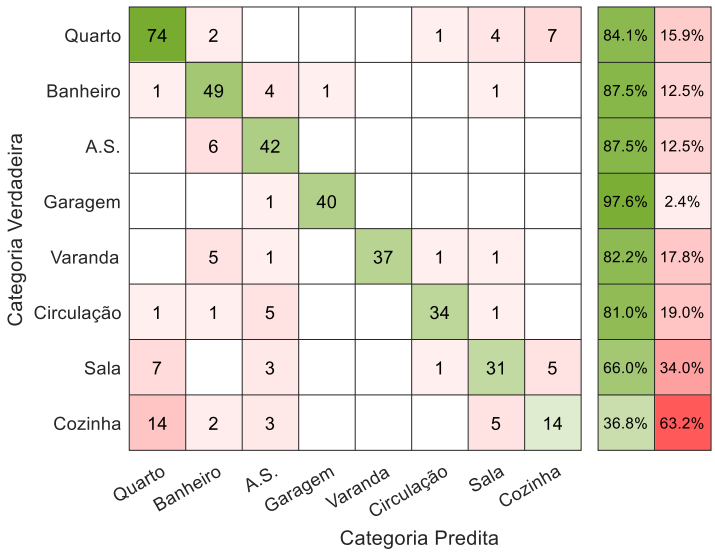


(a)

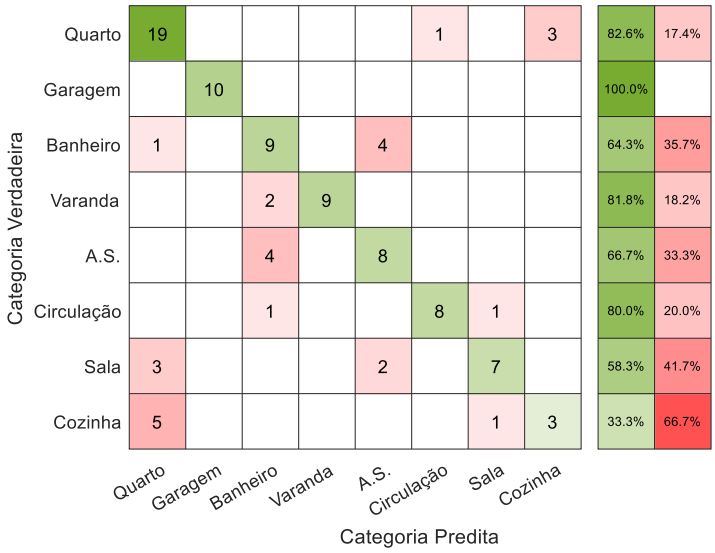


(b)

Figura 4.4 - Matriz de confusão do classificador multiclasse-binário da técnica k-NN para o grupo de (a) treinamento e (b) teste – cenário II.



(a)



(b)

Figura 4.5 - Matriz de confusão do classificador multiclasse-binário da técnica SVM-Gaussiana para o grupo de (a) treinamento e (b) teste – cenário II.

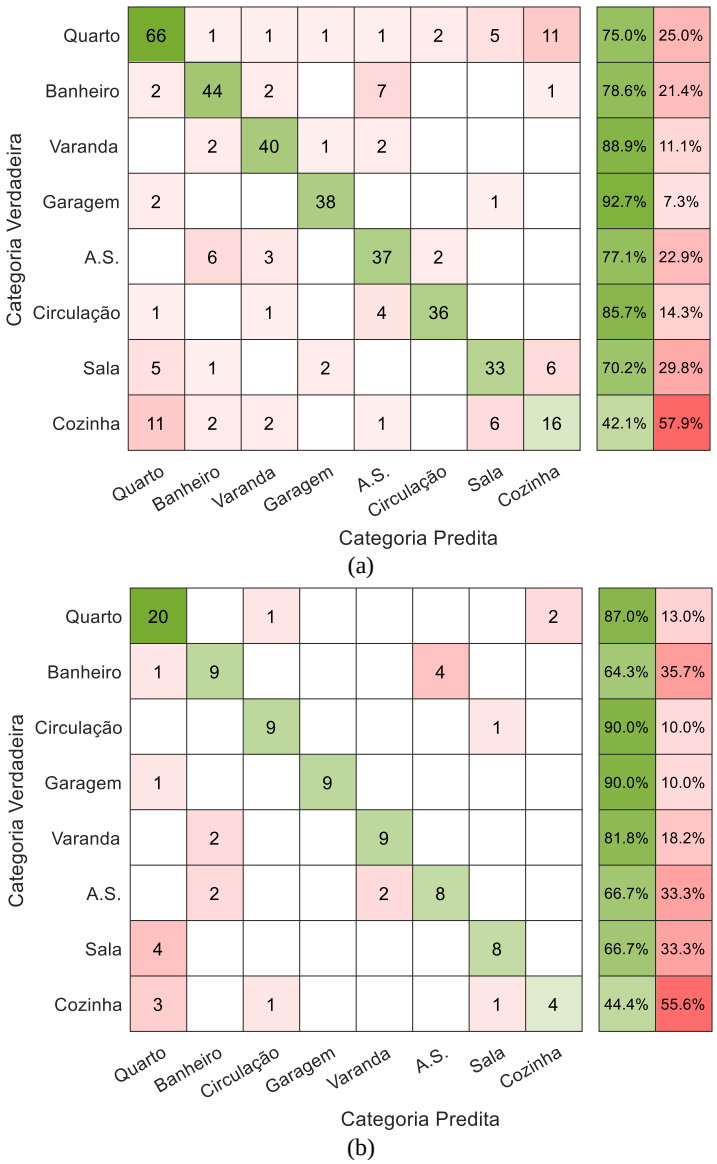
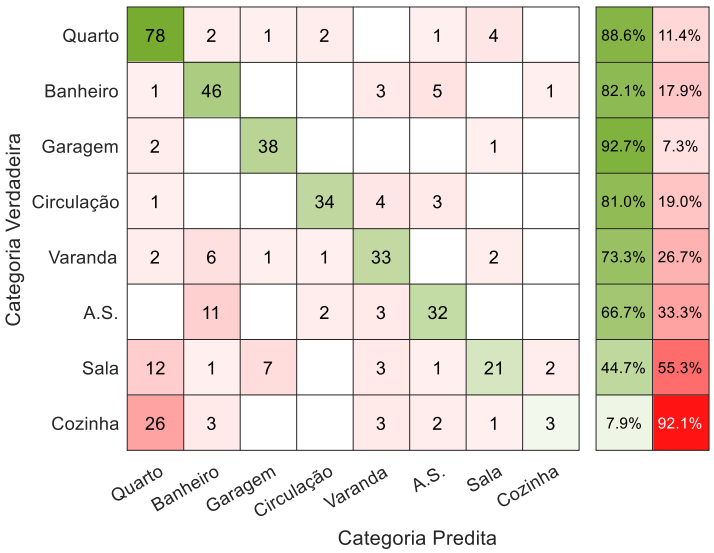
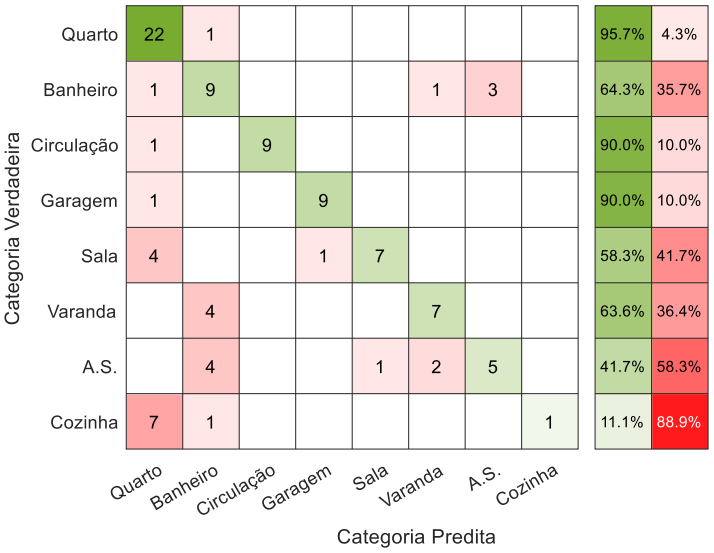


Figura 4.6 - Matriz de confusão do classificador multiclasse-binário da técnica SVM-Quadrática para o grupo de (a) treinamento e (b) teste – cenário II.

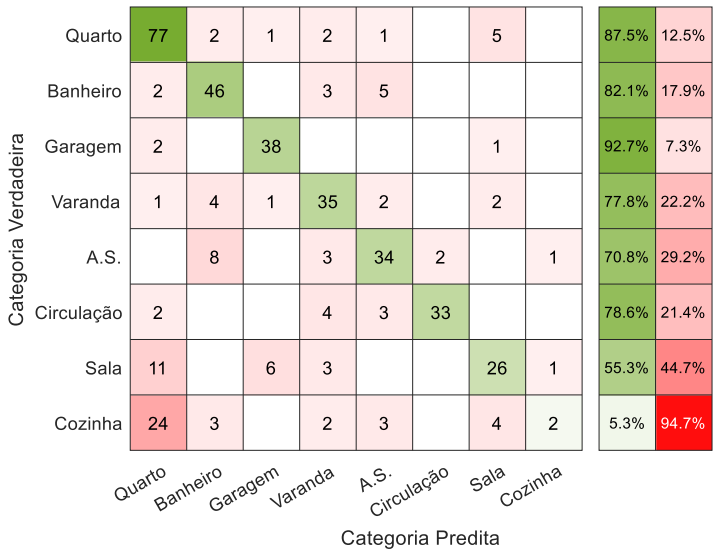


(a)

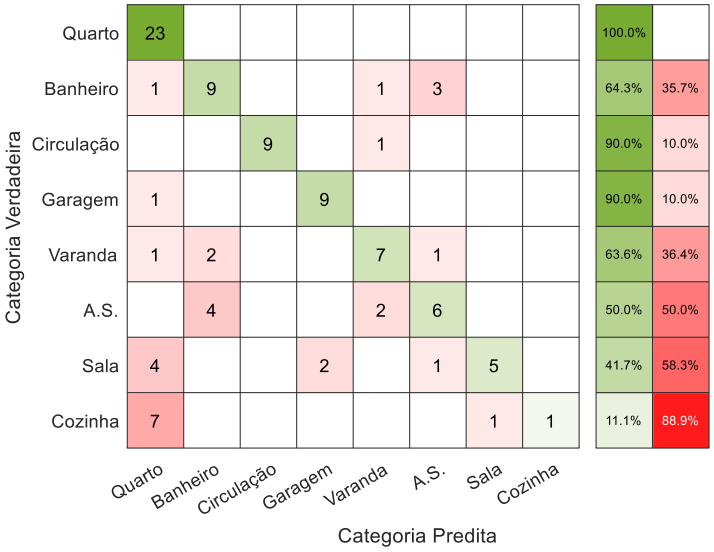


(b)

Figura 4.7 - Matriz de confusão do classificador multiclasse-binário da técnica SVM-Cúbica para o grupo de (a) treinamento e (b) teste – cenário II.



(a)



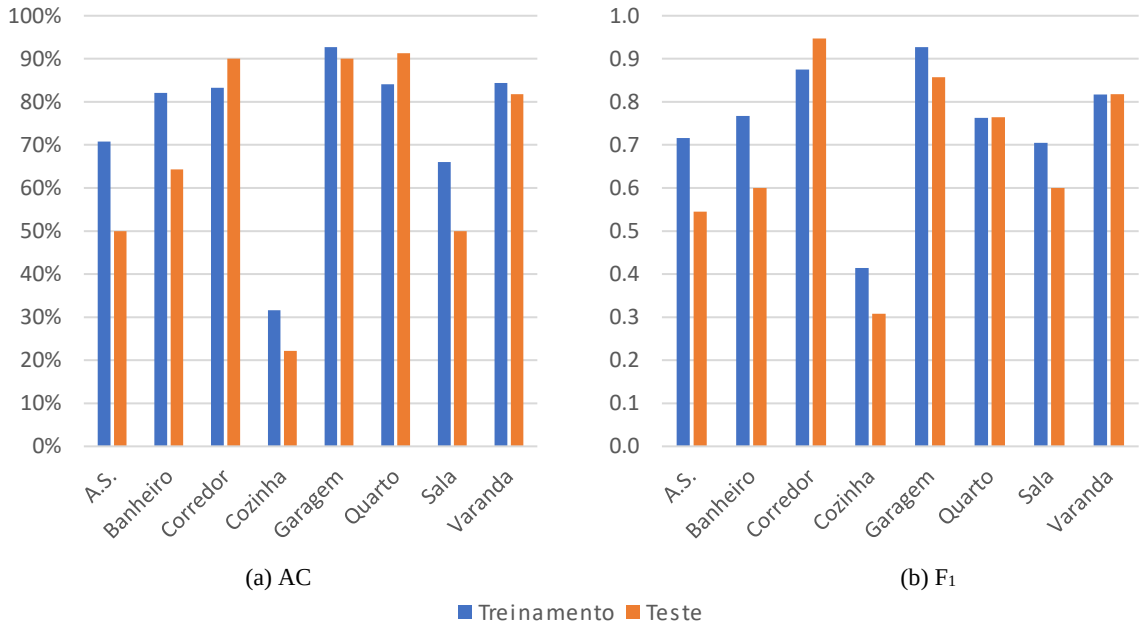
(b)

Fazendo uma análise das matrizes de confusão (Figura 4.3 à Figura 4.9), nota-se que o ambiente “Cozinha” apresentou uma porcentagem de acertos bem abaixo dos demais ambientes, para todas as técnicas, e que, para algumas técnicas, os ambientes “Sala” e “Área de Serviço” não obtiveram uma margem de acertos tão boas quanto as demais classes. Percebe-se, então, que as variáveis utilizadas nesse treinamento não descrevem adequadamente estes ambientes, para algumas técnicas.

Em apêndice, encontram-se as curvas ROC para cada ambiente de cada técnica analisada deste classificador.

4.2.4. Performance do classificador *Ensemble*

Após a classificação de todos os ambientes a partir dos resultados do treinamento de cada técnica, é possível determinar as classes de cada exemplar por votação simples, que é a proposta do classificador *ensemble*. Novamente, esse método de classificação foi aplicado tanto para os dados do grupo de treinamento quanto para o grupo de teste. Com isso, obtém-se as AC e *F1* de cada classe, mostradas na Figura 4.8 para os grupos de treinamento e teste. Observa-se que as AC apresentaram valores acima de 50% para a maioria das classes, entretanto, a classe "Cozinha" ainda apresentou um baixo desempenho.



(a) AC (b) F_1

Figura 4.8 – Performance do classificador *ensamble* para os grupos de treinamento e teste – Cenário II.

Na Tabela 4.9 é apresentado o desempenho geral do grupo de treinamento para o classificador *ensemble*, em que os valores são maiores que 70%, sendo menores que as AC para as técnicas k-NN e SVM-Gaussiana do classificador multiclasse-binário (Tabela 4.8). Entre parênteses é apresentado o desempenho para o grupo de teste.

AC	***F1***
76.1% (70.3%)	0.748 (0.680)

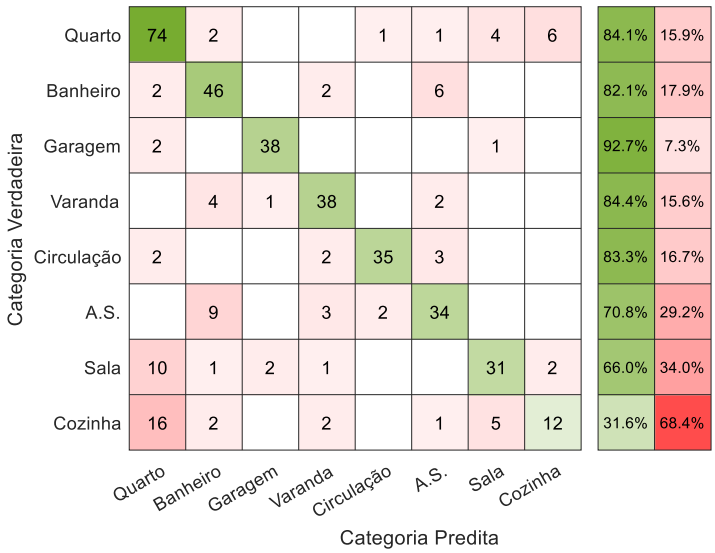
Tabela 4.9 - Performance geral do classificador *ensemble* – cenário II.

Fazendo uma média das acurácias das cinco técnicas do classificador multiclasse-binário no cenário II, Tabela 4.8, temos uma acurácia média de 74,17% para o grupo de treinamento e

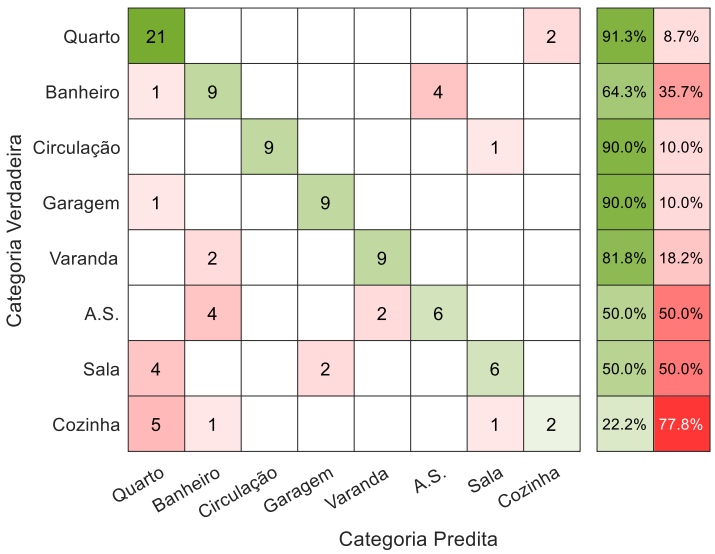
69,71% para o grupo de teste. Comparando essa média com a acurácia do classificador *ensemble*, Tabela 4.9, onde temos acurácias de 76,05% e 70,30% paras os grupos de treinamento e teste, respectivamente, nota-se uma pequena melhora na performance deste classificador, sendo essa melhora mais significante para o grupo de treinamento.

A Figura 4.9 apresenta as matrizes de confusão para o grupo de treinamento (a) e teste (b). Nelas, é possível notar que a classe "Cozinha" é comumente confundida com a classe "Quarto". Assim, para que haja uma melhor classificação desta classe, é necessário que seja incluído um atributo que possa diferenciar estes dois ambientes.

Figura 4.9 - Matriz de confusão do classificador e*nsemble* para o grupo de (a) treinamento e (b) teste – cenário II.



(a)



(b)

4.3. CENÁRIO III – CLASSIFICADORES BINÁRIO, MULTICLASSE-BINÁRIO E *ENSEMBLE*

Procurando melhorar a performance do algoritmo, duas novas variáveis foram inseridas nas análises, como indicado na Tabela 4.1:

a) Variável que indica a presença de torneira;
b) Variável que indica a presença de vaso sanitário.

Essas variáveis foram escolhidas uma vez que são itens que devem ser representados nos modelos arquitetônicos de edificações residenciais, além de ser um diferencial para melhorar a classificação da cozinha, ambiente este que apresentou baixa performance. As demais considerações foram mantidas semelhantes às análises realizadas no cenário II. A Tabela 4.10 resume os dados para este novo cenário.

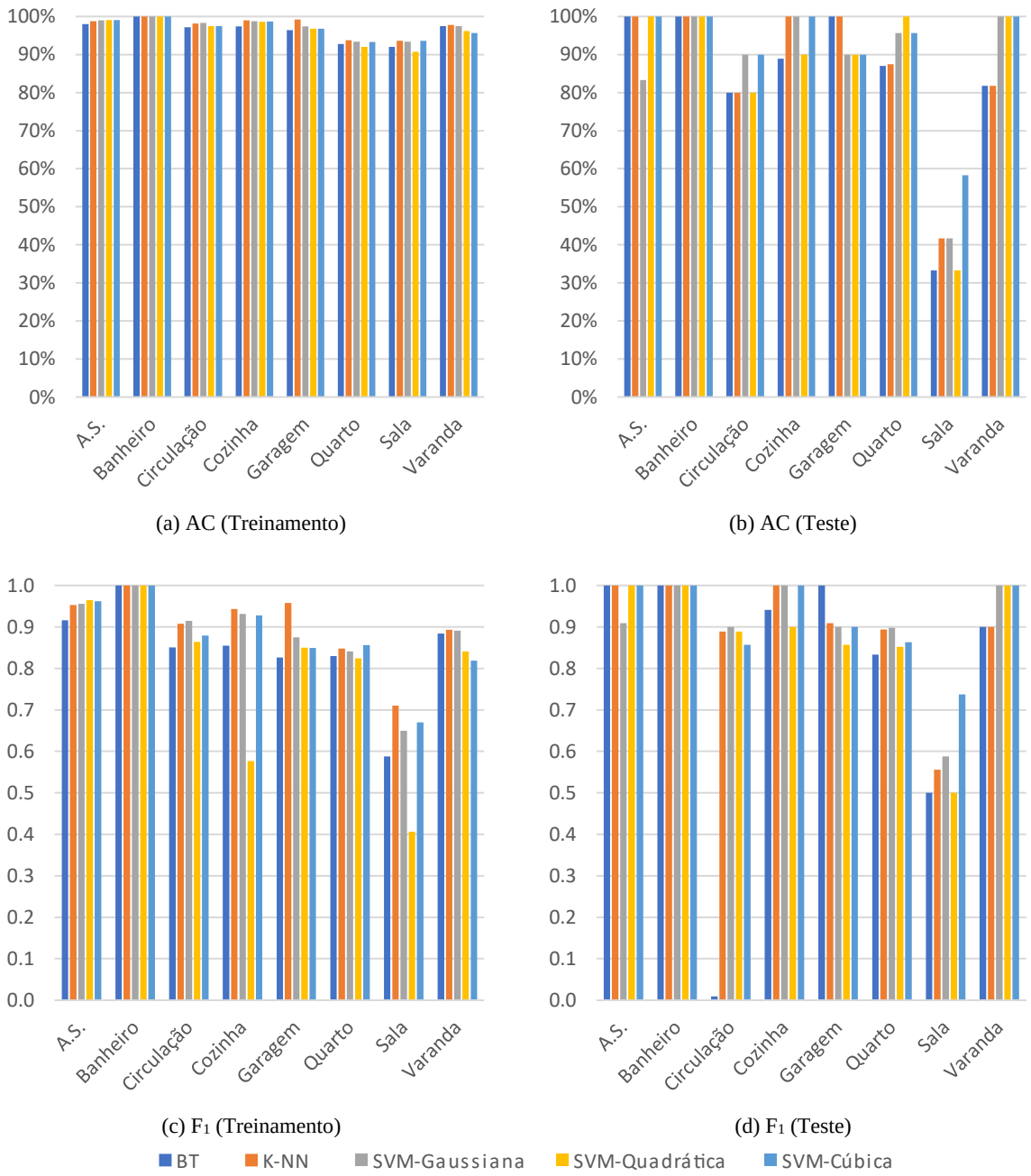
	Mínimo	**Máximo**	**Média**	**Desvio Padrão**	**1° Quartil**	**2° Quartil**	**3° Quartil**
Área (m^2)	1,23	49,83	11,30	8,81	4,95	9,04	15,02
Perímetro (m)	4,00	34,07	13,73	5,62	9,40	12,61	16,20
Fração da área	0,0051	0,5096	0,1115	0,0844	0,0514	0,0841	0,1477
Relação A/P	0,1246	1,8284	0,7313	0,2856	0,5000	0,6937	0,9099
Número de janelas	0	6	-	-	-	-	-
Número de portas	0	5	-	-	-	-	-
Torneira	Não	Sim	-	-	-	-	-
Vaso Sanitário	Não	Sim	-	-	-	-	-

Tabela 4.10 - Resumo dos dados de entrada do banco de treinamento – cenário III.

4.3.1. Performance do treinamento com o classificador binário

A Figura 4.10 apresenta o desempenho do classificador binário, para este cenário, de todas as classes e técnicas analisadas. Por meio desta, tem-se que os classificadores binários apresentaram melhoras significativas nas acurácias em relação ao Cenário II, com os ambientes "Quarto" e "Sala" apresentando os menores desempenhos e a técnica k-NN ainda apresentando os melhores resultados. Para o grupo de treinamento, todas as classes apresentaram AC acima de 90%, enquanto para o grupo de teste, algumas classes ainda apresentaram baixo desempenho, como é o caso da classe "Sala". A classe "Corredor" também apresentou baixo desempenho com relação ao *F1* para a técnica BT. Ambos os desempenhos podem ser

explicados pelo *overfitting* e pelo reduzido número de amostras utilizadas no grupo de teste. A Tabela A.4, no apêndice, mostra as AC deste classificador, para as cinco técnicas analisadas, em forma tabular, considerando a inclusão das novas variáveis.



(a) AC (Treinamento)

(b) AC (Teste)

(c) F_1 (Treinamento)

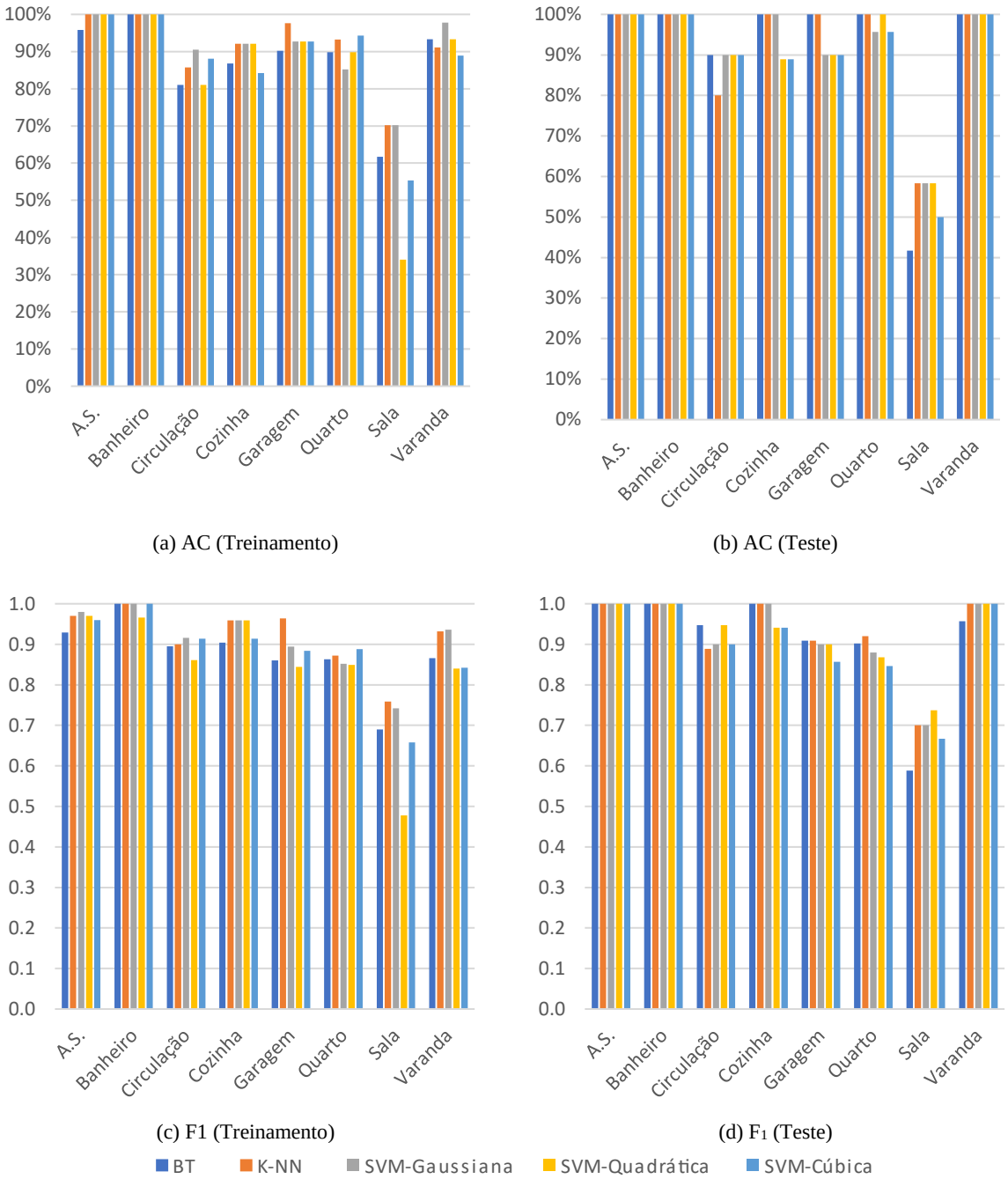
(d) F_1 (Teste)

Figura 4.10 - Performance do classificador binário para os grupos de treinamento e teste – Cenário III.

4.3.2. Performance do classificador multiclasse-binário

A Figura 4.11 apresenta a performance do classificador multiclasse-binário para cada classe analisada. Pode-se observar que houve uma melhora significativa nos resultados, em quem a

AC para algumas classes alcançou 100% e, para a maioria das demais, alcançaram valores acima de 80%. Os valores de *F1* também alcançaram valores satisfatórios, próximos a 1. Entretanto, é possível notar que a classe “Sala” ainda apresentou baixa performance.



(a) AC (Treinamento)

(b) AC (Teste)

(c) F1 (Treinamento)

(d) F_1 (Teste)

Figura 4.11 - Performance do classificador multiclasse-binário para os grupos de treinamento e teste – Cenário III.

A Tabela 4.11 apresenta a performance geral do classificador multiclasse-binário para cada técnica analisada. Nota-se que os valores de AC atingiram cerca de 90% e a técnica k-NN apresentou o melhor desempenho, enquanto o SVM-Quadrático apresentou o pior. Os

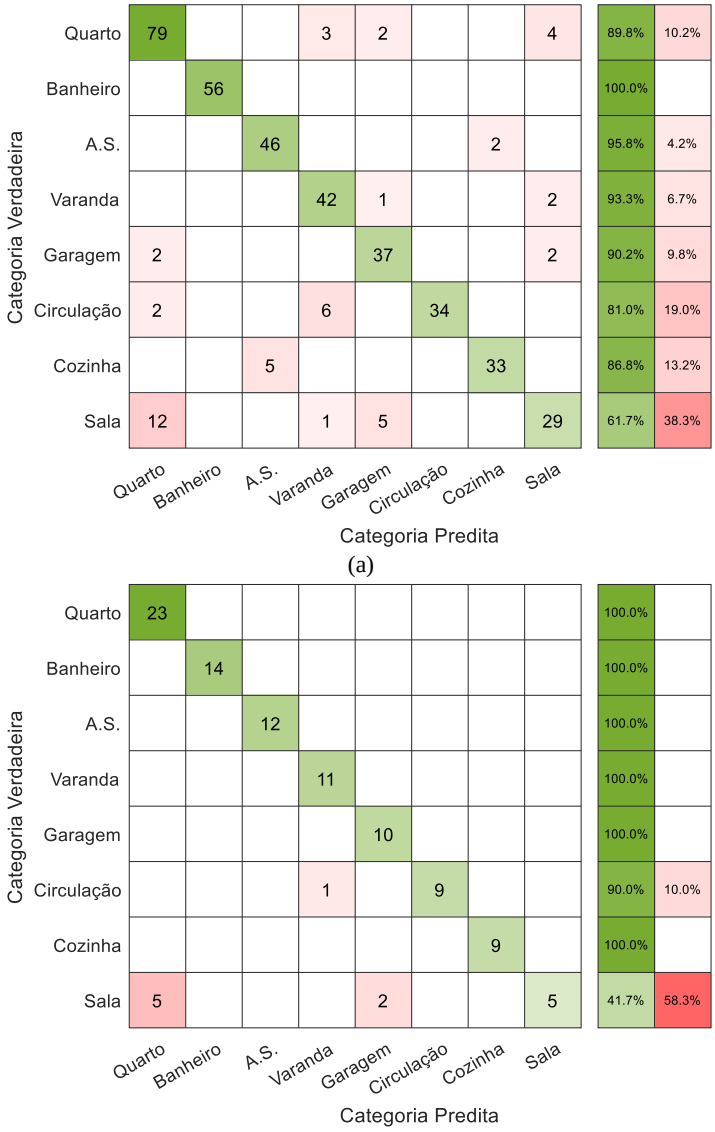
resultados apresentados nesta tabela foram melhores que os resultados do cenário II (Tabela 4.8), onde as AC ficaram abaixo de 80% e *F1* abaixo de 0,750, para a maioria das técnicas.

BT	k-NN	SVM-Gaussiana	SVM-Quadrática	SVM-Cúbica
87.9% (92.11%)	91.6% (93.1%)	90.6% (92.1%)	85.9% (92.1%)	88.9% (90.1%)
0.876 (0.913)	0.919 (0.927)	0.910 (0.923)	0.846 (0.924)	0.883 (0.901)

Tabela 4.11 – Performance geral do classificador multiclasse-binário – cenário III.

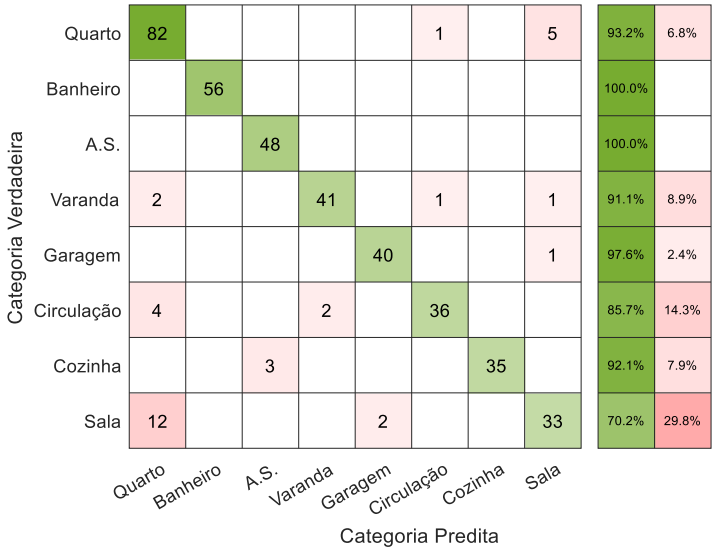
As figuras a seguir apresentam as matrizes de confusão das cinco técnicas, tanto para o grupo de treinamento quanto para o grupo de teste.

Figura 4.12 - Matriz de confusão do classificador multiclasse-binário da técnica BT para o grupo de (a) treinamento e (b) teste – cenário III.

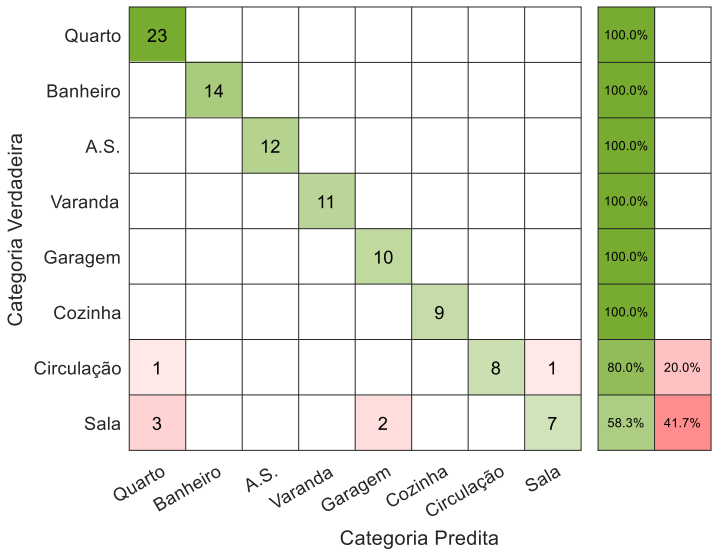


(b)

Figura 4.13 - Matriz de confusão do classificador multiclasse-binário da técnica k-NN para o grupo de (a) treinamento e (b) teste – cenário III.

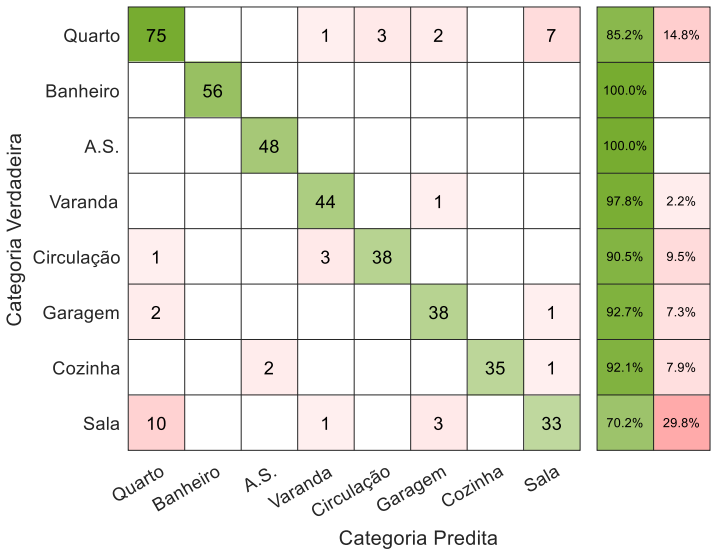


(a)

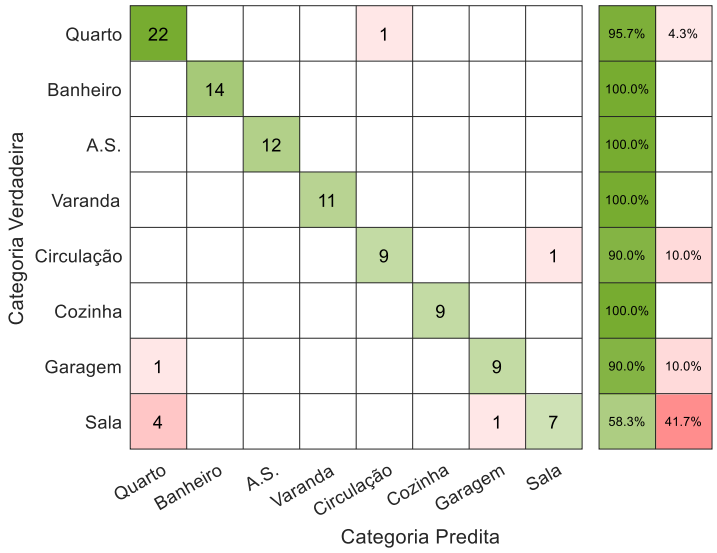


(b)

Figura 4.14 - Matriz de confusão do classificador multiclasse-binário da técnica SVM-Gaussiana para o grupo de (a) treinamento e (b) teste – cenário III.



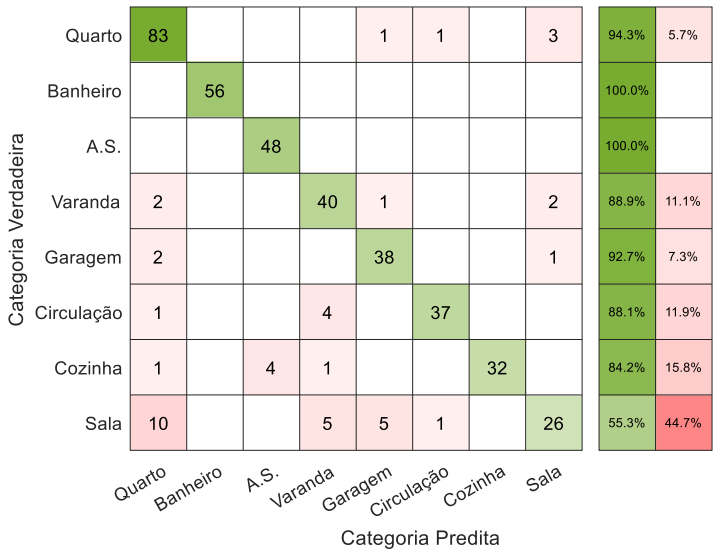
(a)



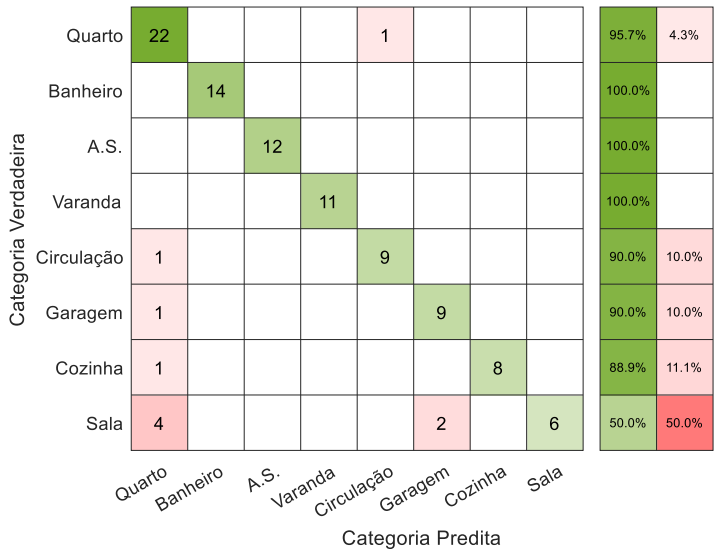
(b)

Figura 4.15 - Matriz de confusão do classificador multiclasse-binário da técnica SVM-Quadrática para o grupo de (a) treinamento e (b) teste – cenário III.

Figura 4.16 - Matriz de confusão do classificador multiclasse-binário da técnica SVM-Cúbica para o grupo de (a) treinamento e (b) teste – cenário III.



(a)



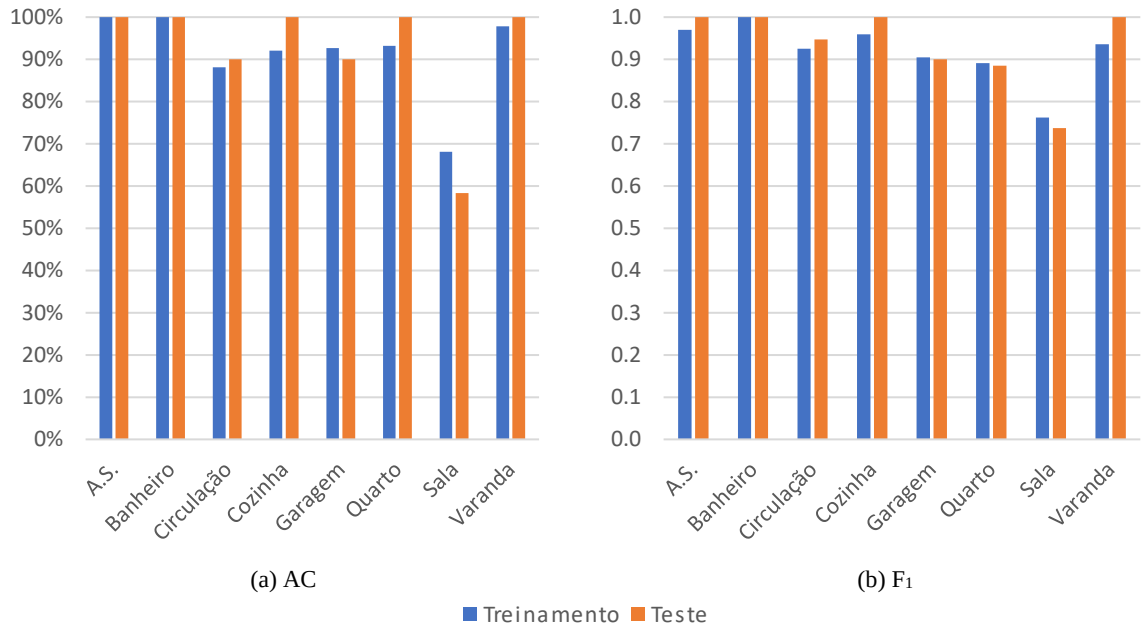
(b)

Em apêndice, são mostradas o desempenho de todas as técnicas analisadas, na Tabela A.5, bem como as curvas ROC.

4.3.3. Performance do classificador *Ensemble*

A Figura 4.17 mostra as medidas de desempenho do classificador *ensemble*, em que, novamente, o desempenho apresentou melhora em relação ao cenário I e os desempenhos dos grupos de treinamento e teste foram semelhantes, indicando que não ocorreu *overfitting* no modelo. Como se pode ver, a classe “Sala” ainda apresentou o menor desempenho entre todas

as classes, com AC em torno de 60% e *F1* em torno de 0,700. As demais classes atingiram AC por volta de 90% ou maior e *F1* em torno de 0,900 ou maior.



(a) AC (b) F_1

Figura 4.17 - Performance do classificador *ensamble* para os grupos de treinamento e teste – Cenário III.

Na Tabela 4.12 é mostrada a performance geral, em que os desempenhos, tanto para os grupos de treinamento quanto para os de teste, ficaram acima de 90% para AC e 0,900 para F1, indicando uma boa performance para o modelo. Além disso, os desempenhos de treinamento e teste foram semelhantes, o que indica que não ocorreu *overfitting* no modelo.

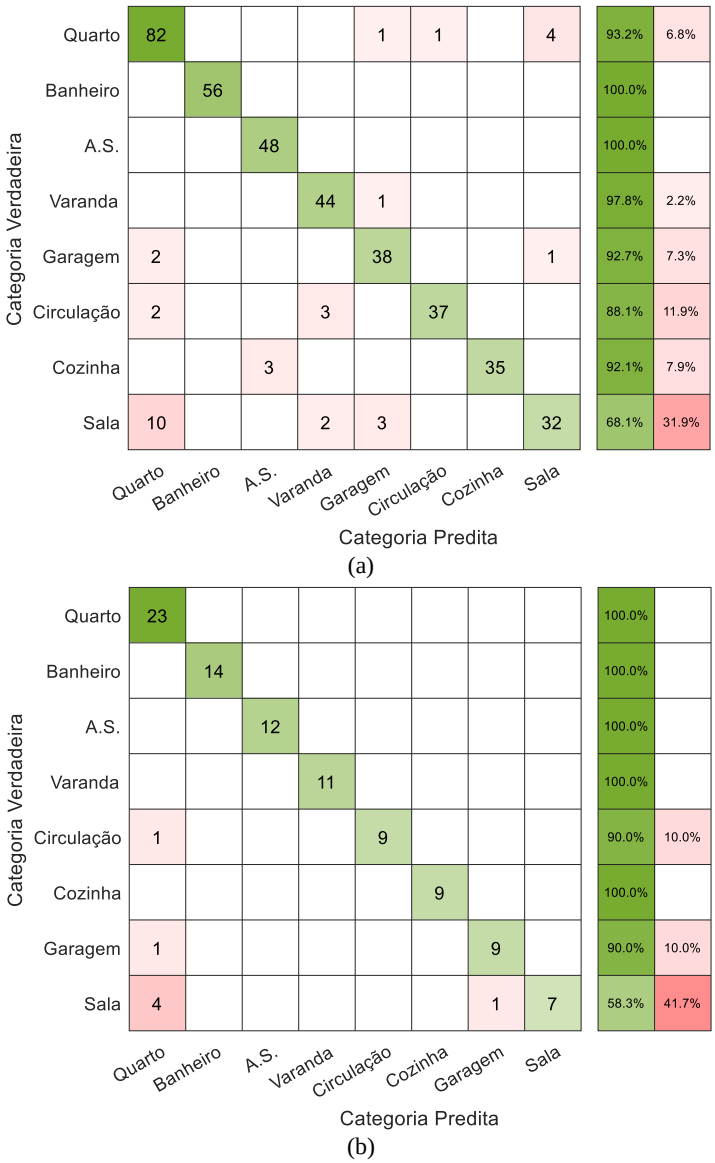
AC	***F1***
91.9% (93.0%)	0.919 (0.934)

Tabela 4.12 - Performance geral do classificador e*nsemble* – cenário III.

Comparando as AC do classificador *ensemble*, apresentados acima na Tabela 4.12, com a média das AC de todas as técnicas do classificador multiclasse-binário, apresentados anteriormente na Tabela 4.11, que foram de 89,0% e 91,9% para os grupos de treinamento e teste, respectivamente, observou-se que houve uma melhora na classificação dos ambientes.

A Figura 4.18 mostra as matrizes de confusão do classificador *ensemble* para este cenário III. Percebe-se que, com relação ao erro de classificação, é comum o ambiente sala ser classificado como quarto. Sendo assim, o acréscimo de uma nova variável que pudesse ajudar na distinção destes dois ambientes traria uma melhora significativa na performance dos algoritmos.

Figura 4.18- Matriz de confusão do classificador ensemble para o grupo de (a) treinamento e (b) teste – cenário III.

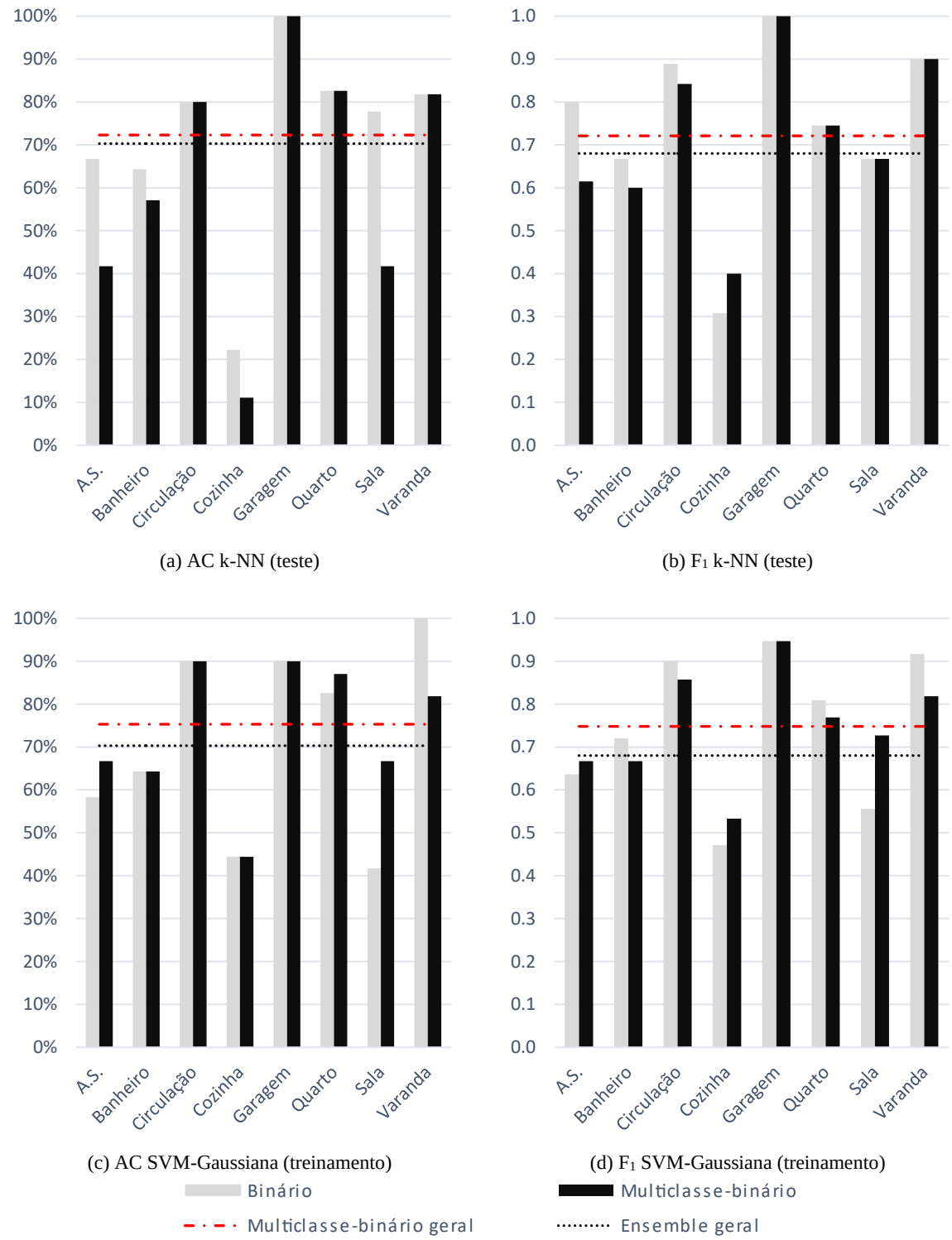


4.4. COMPARAÇÃO ENTRE CENÁRIOS II E III

A Figura 4.19 e a Figura 4.20 resumem os resultados apresentados anteriormente para as técnicas k-NN e SVM-Gaussiana, as duas com melhor desempenho. Nelas são apresentados os desempenhos para os cenários II e III dos classificadores binário e multiclasse-binário, para todas as classes, além do desempenho geral do classificador multiclasse-binário e *ensemble*.

Comparando os desempenhos, nota-se, claramente, a melhora que ocorreu do cenário II para o cenário III, especialmente para a classe “Cozinha”. Para o cenário II, a classe “Garagem” apresentou os melhores resultados, enquanto a classe “Cozinha”, o pior. Para o cenário III,

algumas classes alcançaram o desempenho ideal, tais como “Banheiro” e “Cozinha”, enquanto a classe “Sala” apresentou o pior resultado. Analisando a performance geral para o classificador multiclasse-binário e *ensemble*, eles apresentaram resultados bastante semelhantes no cenário III, enquanto no cenário II, o classificador multiclasse-binário se sobressaiu.



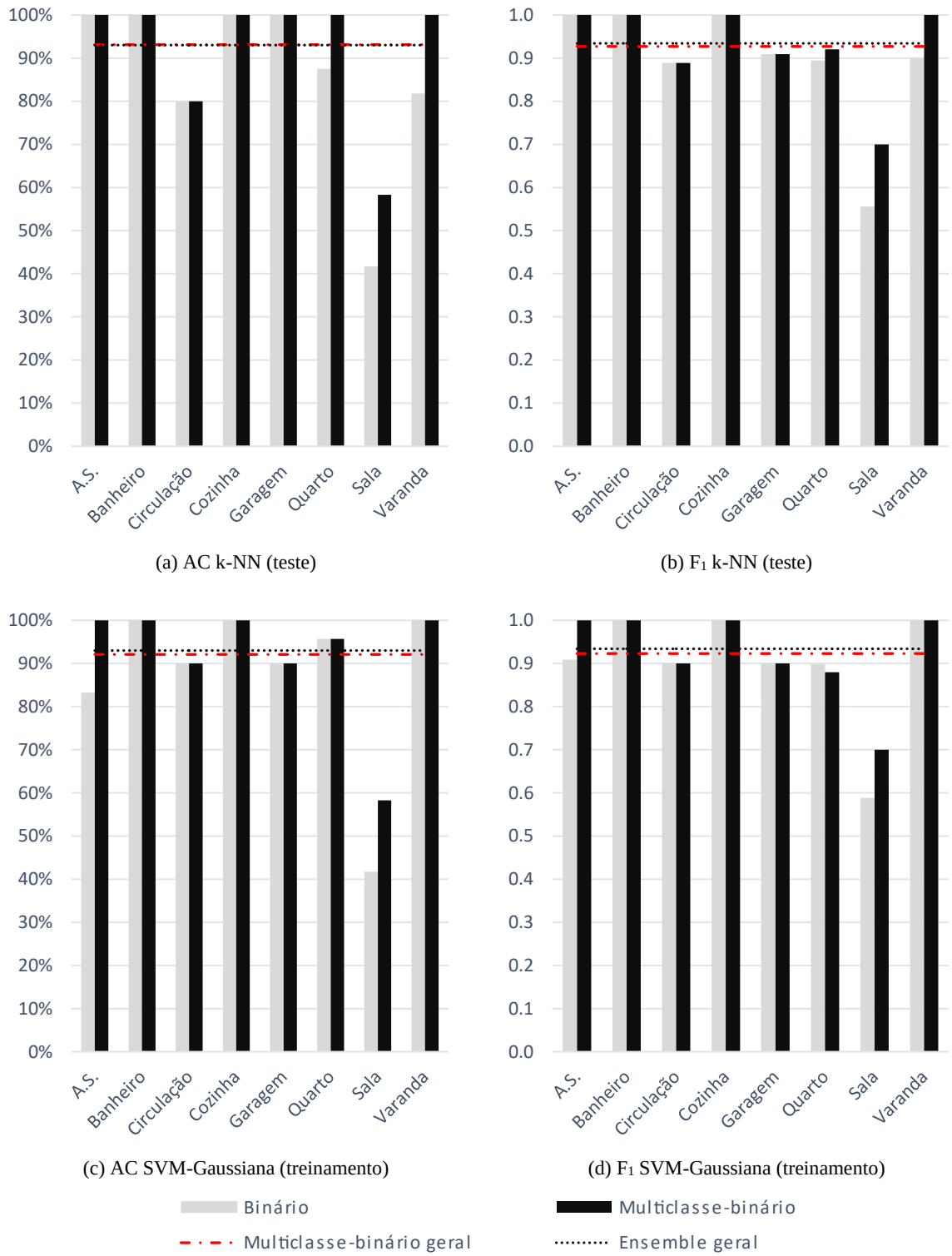
(a) AC k-NN (teste)

(b) F_1 k-NN (teste)

(c) AC SVM-Gaussiana (treinamento)

(d) F_1 SVM-Gaussiana (treinamento)

Figura 4.19 - Performance para os classificadores binário, multiclasse-binário e *ensemble* – Cenário II.



(a) AC k-NN (teste)

(b) F_1 k-NN (teste)

(c) AC SVM-Gaussiana (treinamento)

(d) F_1 SVM-Gaussiana (treinamento)

Figura 4.20 - Performance para os classificadores binário, multiclasse-binário e *ensemble* – Cenário III.

A Tabela 4.13 compara o percentual das acurácias, tanto do grupo de treinamento quanto do grupo de teste, dos cenários II e III. Esse comparativo é feito tanto para as técnicas analisadas no classificador multiclasse-binário quanto para o classificador *ensemble*. É possível notar o aumento percentual das acurácias em todas as situações, sendo que no grupo de testes, o aumento foi mais expressivo.

	Multiclasse-binário					***Ensemble***
	BT	**k-NN**	**SVM-Gaussiana**	**SVM-Quadrática**	**SVM-Cúbica**	
Cenário II - Treinamento	72,8%	79,3%	76,5%	70,4%	71,9%	76,1%
Cenário III - Treinamento	87,9%	91,6%	90,6%	85,9%	88,9%	91,9%
Aumento	*20,7%*	*15,6%*	*18,4%*	*22,1%*	*23,7%*	*20,8%*
Cenário II - Teste	64,4%	72,3%	75,3%	68,3%	68,3%	70,3%
Cenário III - Teste	92,1%	93,1%	92,1%	92,1%	90,1%	93,1%
Aumento	*43,1%*	*28,8%*	*22,4%*	*34,8%*	*31,9%*	*32,4%*

Tabela 4.13 - Comparação das acurácias entre os cenários II e III.

Na Tabela 4.14 tem-se o comparativo dos *F1*, de forma semelhante ao comparativo das acurácias. Novamente, houve uma melhora dos resultados do cenário II para o cenário III, sendo essa melhora mais expressiva para o grupo de teste. Como já discutido, isso indica que, no cenário III, o problema de *overfitting* foi diminuído.

	Multiclasse-binário					***Ensemble***
	BT	**k-NN**	**SVM-Gaussiana**	**SVM-Quadrática**	**SVM-Cúbica**	
Cenário II - Treinamento	0,721	0,783	0,761	0,659	0,673	0,748
Cenário III - Treinamento	0,876	0,919	0,910	0,846	0,883	0,919
Aumento	*21,6%*	*17,5%*	*19,6%*	*28,4%*	*31,1%*	*22,8%*
Cenário II - Teste	0,616	0,721	0,748	0,651	0,638	0,680
Cenário III - Teste	0,913	0,927	0,923	0,924	0,901	0,934
Aumento	*48,3%*	*28,6%*	*23,3%*	*42,0%*	*41,2%*	*37,3%*

Tabela 4.14 - Comparação dos valores de *F1* entre os cenários II e III.

Fazendo uma comparação das matrizes de confusão do classificador *ensemble*, apresentadas na Figura 4.9 e na Figura 4.18 para os cenários II e III, respectivamente, nota-se que o número de amostras classificadas incorretamente diminuiu do cenário II para o cenário III, exceto para as classes “Garagem” e “Corredor”. Isso significa que as variáveis introduzidas nas análises para o cenário III melhoram a performance do classificador, de forma geral. No cenário II, a maioria dos exemplares classificadas incorretamente foram confundidos com as classes “Quarto” e “Banheiro”, enquanto no cenário III, estes exemplares foram confundidos com as classes “Quarto” e “Varanda”. As classes “Sala” e “Cozinha” foram as que obtiveram os maiores quantitativos de exemplares classificados incorretamente no cenário II, enquanto no cenário III, apesar da melhora geral da performance, a classe “Sala” continuou como sendo a classe com maior quantitativo de exemplares classificados incorretamente.

Uma vez que as novas variáveis introduzidas nas análises do cenário III não estão presentes no ambiente sala, a performance desta classe não apresentou uma melhora significativa para o classificador *ensemble*. Esta estabilidade do desempenho é reforçada pelo fato de que esta classe é comumente confundia com a classe “Quarto”, quem também não apresenta estas variáveis.

5. CONCLUSÃO

Este trabalho se propôs a coletar e preparar um banco de dados de projetos arquitetônicos que pudessem ser utilizadas na classificação de espaços residenciais em modelos BIM sem a necessidade de um operador humano. Esta classificação se daria por meio de variáveis estrategicamente selecionadas do banco de dados e de fácil obtenção de modelos BIM e IFC, tais como áreas e perímetros, número de portas e janelas, e presença de torneiras e/ou vasos sanitários.

Baseado nesses dados de entrada, vários modelos de ML foram utilizados no enriquecimento semântico de modelos BIM por meio de quatro estratégias de classificação: multiclasse, binária, multiclasse-binária e *ensemble*. Estas estratégias foram usadas para a predição de 8 classes de espaços como quarto, corredor, banheiro, cozinha, sala, varanda, área de serviço e garagem. Como esperado, as variáveis de entrada tiveram um bom impacto na performance de todos os modelos e técnicas, como pode ser notado nas diferenças de resultados entre os diferentes cenários analisados. O cenário que considerou a simples adição de duas novas variáveis (presença de torneira e vaso sanitário) melhorou, significativamente, a performance de todos os modelos. Isto é uma informação valiosa para o enriquecimento semântico de modelos BIM, uma vez que estas variáveis são fáceis de se obter de arquivos IFC.

De uma forma geral, pode-se notar a influência que o tratamento dos dados de entrada tem sobre a performance de cada técnica analisada, uma vez que houve uma melhora na performance das análises após os dados passarem por modificações. Por exemplo, na análise de classificação multiclasse, a técnica SVM-Quadrática apresentou a melhor performance, enquanto nas demais análises seu desempenho já não foi um dos melhores. Sendo assim, não se pode afirmar de antemão qual técnica é melhor para uma análise de classificação dos ambientes, uma vez que há a influência das variáveis levadas em consideração nos estudos, bem como do tratamento dos dados.

Analisando o cenário II, dada a metodologia utilizada, percebe-se a necessidade de incluir variáveis que possam melhor descrever os ambientes de forma a melhorar o desempenho dos classificadores aqui analisados. Caso houvesse a necessidade de se classificar um ou outro ambiente, o classificador binário já seria uma boa solução. Sua utilização seria viável, por exemplo, para a determinação de algum ambiente específico, como por exemplo o ambiente

quarto, que apresentou bons resultados de uma forma geral, e, a partir desse ambiente já classificado, incluir novas variáveis, como posição relativa dos demais ambientes em relação a este ambiente.

Para o cenário III, nota-se uma melhor performance do classificador k-NN em comparação com os demais, tanto para o modelo de classificação binária quanto para o modelo de classificação multiclasse-binário, neste último para ambos os grupos, de treinamento e teste. Além disso, percebe-se que as acurácias do grupo de teste foram maiores e próximas das acurácias do grupo de treinamento, indicando que já não houve problemas de *overfitting* nessa análise.

Sobre os modelos *machine learning*, quando eles são utilizados como classificadores binários, a técnica k-NN obteve, no geral, resultados melhores que as técnicas BT e SVM, apesar de ser um dos algoritmos mais simples de ser implementado. Por outro lado, as técnicas BT e SVM-Quadrática apresentaram os piores resultados no geral. Alguns classificadores binários apresentaram *overfitting* nos dados de treinamento.

Os resultados mostraram que é difícil concluir sobre o melhor ou pior modelo *machine learning* quando se usa o classificador binário, uma vez que sua performance varia entre os diferentes modelos e classes de espaço. Entretanto, o uso destas técnicas de *machine learning* para os classificadores multiclasse-binário (classificado de acordo com o *score* obtido do classificador binário de uma mesma técnica) e *ensemble* (classificado por votação dentre as diferentes técnicas do modelo multiclasse-binário), melhoraram a performance das medidas de acurácia e F1-score para as classificações. Ambas as técnicas diminuíram o *overfitting* que ocorreu no classificador binário, permitindo desempenhos similares para os grupos de treinamento e teste. Além disso, o desempenho geral para o classificador *ensemble* superou o desempenho do classificador multiclasse-binário quando considerado a média do desempenho para as oito classes analisadas neste estudo.

Pode-se concluir, também, que o modelo em que se utilizou os classificadores binário e multiclasse-binário para a classificação de classes múltiplas (cenários II e III) apresentaram melhores resultados se comparado com o classificador multiclasse (cenário I). Além disso, o classificador *ensemble* pode ser de grande valia, uma vez que este tende a minimizar anormalidades em resultados de uma técnica específica.

Referências Bibliográficas

AHMED, S. et al. **Automatic room detection and room labeling from architectural floor plansProceedings - 10th IAPR International Workshop on Document Analysis Systems, DAS 2012**, 2012.

ANDRADE, M. L. V. X. DE; RUSCHEL, R. C. Interoperabilidade de aplicativos BIM usados em arquitetura por meio do formato IFC. **Gestão & Tecnologia de Projetos**, v. 4, n. 2, p. 76–111, 2009.

BELSKY, M.; SACKS, R.; BRILAKIS, I. Semantic Enrichment for Building Information Modeling. **Computer-Aided Civil and Infrastructure Engineering**, v. 31, n. 4, p. 261–274, 2016.

BISHOP, C. M. **Pattern Recognition and Machine Learning**. [s.l: s.n.].

BLAGUS, R.; LUSA, L. Class prediction for high-dimensional class-imbalanced data. **BMC Bioinformatics**, v. 11, n. 523, p. 17, 2010.

BLOCH, T.; SACKS, R. Comparing machine learning and rule-based inferencing for semantic enrichment of BIM models. **Automation in Construction**, v. 91, n. July 2017, p. 256–272, 2018.

BREIMAN, L. Bagging Predictors. **Machine Learning**, v. 24, p. 123–140, 1996.

BSI STANDARDS PUBLICATION. **BS 1192-4:2014 - Collaborative production of Information - Part 4: Fulfilling employer's information exchange requirements using COBie - Code of Practice.**, 2014.

CHICCO, D.; JURMAN, G. **The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluationBMC Genomics**, 2020.

GÉRON, A. **Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems**. Second ed. [s.l.] O'Reilly,

2019.

GIMENEZ, L. et al. Automatic reconstruction of 3D building models from scanned 2D floor plans. **Automation in Construction**, v. 63, p. 48–56, 2016.

GOU, J. et al. A New Distance-weighted k -nearest Neighbor Classifier. **Journal of Information and Computational Science**, p. 1429–1436, 2012.

GOVERNO DO ESTADO DE SANTA CATARINA. **Caderno de especificações de projetos em BIM**. [s.l: s.n.].

JACOSKI, C. A. **Integração e Interoperabilidade em Projetos de Edificações - Uma Implemnetação com IFC/XML**. [s.l.] Universidade Federal de Santa Catarina, 2003.

JAPKOWICZ, N.; STEPHEN, S. The class imbalance problem: A systematic study. **Intelligent Data Analysis**, v. 6, n. 5, p. 429–449, 2002.

JEONG, Y. S. et al. Benchmark tests for BIM data exchanges of precast concrete. **Automation in Construction**, v. 18, n. 4, p. 469–484, 2009.

KAUR, H.; PANNU, H. S.; MALHI, A. K. A systematic review on imbalanced data challenges in machine learning: Applications and solutions. **ACM Computing Surveys**, v. 52, n. 4, p. 36, 2019.

KOO, B. et al. Using support vector machines to classify building elements for checking the semantic integrity of building information models. **Automation in Construction**, v. 98, n. October 2018, p. 183–194, 2019.

LIN, C.-F.; WANG, S.-D. Fuzzy Support Vector Machines. **IEEE Transactions on Neural Networks**, v. 13, n. 2, p. 464–471, 2002.

MA, L. et al. 3D Object Classification Using Geometric Features and Pairwise Relationships. **Computer-Aided Civil and Infrastructure Engineering**, v. 33, n. 2, p. 152–164, 2018.

MULLER, A. C.; GUIDO, S. **Introduction to Machine Learning with Python: A Guide for Data Scientists**. [s.l.] O'Reilly, 2017.

MULLER, M. F. **A Interoperabilidade entre Sistemas CAD de Projeto de Estruturas de Concreto Armado Baseada em Arquivos IFC**Universidade Federal do Paraná, , 2011.

NASCIMENTO, L. A. DO. **Proposta de um sistema de recuperação de informação para extranet de projeto**Universidade de São Paulo, , 2004.

NOI, P. T.; KAPPAS, M. Comparison of Random Forest, k-Nearest Neighbor, and Support Vector Machine Classifiers for Land Cover Classification Using Sentinel-2 Imagery. **Sensors**, v. 18, n. 18, p. 1–20, 2018.

ORENI, D. et al. Survey turned into HBIM: The restoration and the work involved concerning the Basilica di Collemaggio after the earthquake (L'Aquila). **ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences**, v. 2, n. 5, p. 267–273, 2014.

SACKS, R. et al. Semantic Enrichment for Building Information Modeling: Procedure for Compiling Inference Rules and Operators for Complex Geometry. **Journal of Computing in Civil Engineering**, v. 31, n. 6, 2017.

SACKS, R. et al. **BIM Handbook: A Guide to Building Information Modeling for Owners, Designers, Engineers, Constractors and Facility Managers**. Terceira ed. [s.l.] John Wiley & Sons, 2018.

SCHOLKOPF, B.; SMOLA, A. J. **Learning With Kernels: Support Vector Machines, Regularization, Optimization, and Beyond**. [s.l: s.n.]. v. 98

SIMEONE, D.; CURSI, S.; ACIERNO, M. BIM semantic-enrichment for built heritage representation. **Automation in Construction**, v. 97, n. August 2018, p. 122–137, 2019.

SKURICHINA, M.; DUIN, R. P. W. Bagging , Boosting and the Random Subspace Method for Linear Classifiers. **Pattern Analysis and Applications**, v. 5, p. 121–135, 2002.

TEO, T.-A.; CHO, K.-H. BIM-oriented indoor network model for indoor and outdoor combined route planning. **Advanced Engineering Informatics**, v. 30, n. 3, p. 268–282, 2016.

THARWAT, A. Classification assessment methods. **Applied Computing and Informatics**, v.

17, n. 1, p. 168–192, 2018.

THARWAT, A. Parameter investigation of support vector machine classifier with kernel functions. **Knowledge and Information Systems**, v. 61, n. 3, p. 1269–1302, 2019.

TURNER, E.; ZAKHOR, A. **Floor plan generation and room labeling of indoor environments from Laser range data**. GRAPP 2014 - Proceedings of the 9th International Conference on Computer Graphics Theory and Applications. **Anais**...SCITEPRESS, 2014

UK BIM ALLIANCE. **Information Management according to BS EN ISO 19650 - Guidance Part 1: ConceptsUK BIM Alliance**, 2019.

VENUGOPAL, M. et al. Engineering Semantics of Model Views for Building. **Proceedings of the CIB W78 2010: 27th International Conference –Cairo**, 2010.

VENUGOPAL, M. et al. Semantics of model views for information exchanges using the industry foundation classes schema. **Advanced Engineering Informatics**, v. 26, n. 2, p. 411–428, 2012.

XIONG, X. et al. Automatic creation of semantically rich 3D building models from laser scanner data. **Automation in Construction**, v. 31, p. 325–337, 2013.

XUE, F.; WU, L.; LU, W. Semantic enrichment of building and city information models : A ten-year review. **Advanced Engineering Informatics**, v. 47, n. December 2020, p. 101245, 2021.

ZENG, Z. et al. **Deep Floor Plan Recognition Using a Multi-Task Network with Room-Boundary-Guided Attention**. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). **Anais**...IEEE, 2019

A. APÊNDICE

a) TABELAS DE ACURÁCIAS E F1 SCORES

	BT	K-NN	SVM-Gaussiana	SVM-Quadrática	SVM-Cúbica
A.S.	91.7% (58.3%)	96.1% (66.7%)	93.5% (58.3%)	92.6% (41.7%)	92.1% (58.3%)
	0.623 (0.667)	0.804 (0.800)	0.714 (0.636)	0.678 (0.500)	0.671 (0.560)
Banheiro	91.0% (50.0%)	95.4% (64.3%)	93.6% (64.3%)	93.0% (64.3%)	92.0% (64.3%)
	0.643 (0.519)	0.823 (0.667)	0.771 (0.720)	0.754 (0.600)	0.729 (0.643)
Circulação	96.5% (80.0%)	97.5% (80.0%)	96.9% (90.0%)	97.0% (80.0%)	96.9% (80.0%)
	0.816 (0.889)	0.868 (0.889)	0.842 (0.900)	0.845 (0.889)	0.835 (0.889)
Cozinha	91.0% (0.0%)	90.8% (22.2%)	89.6% (44.4%)	90.6% (0.0%)	90.6% (0.0%)
	0.310 (0.000)	0.263 (0.308)	0.384 (0.471)	0.072 (0.000)	0.069 (0.000)
Garagem	95.2% (80.0%)	98.9% (100.0%)	97.7% (90.0%)	96.9% (90.0%)	96.5% (80.0%)
	0.762 (0.762)	0.942 (1.000)	0.888 (0.947)	0.853 (0.857)	0.833 (0.842)
Quarto	87.7% (69.6%)	89.7% (82.6%)	87.6% (82.6%)	86.7% (95.7%)	86.5% (91.3%)
	0.714 (0.696)	0.768 (0.745)	0.683 (0.809)	0.725 (0.772)	0.734 (0.724)
Sala	91.8% (41.7%)	92.1% (77.8%)	92.0% (41.7%)	90.5% (50.0%)	91.9% (58.3%)
	0.579 (0.556)	0.648 (0.667)	0.573 (0.556)	0.440 (0.667)	0.542 (0.737)
Varanda	94.5% (63.6%)	96.4% (81.8%)	95.6% (100.0%	92.6% (54.5%)	92.6% (63.6%)
	0.720 (0.737)	0.808 (0.900)	0.798 (0.917)	0.660 (0.600)	0.693 (0.636)

Tabela A.1 - Performance do classificador binário para cada classe, em que os números entre parênteses apresentam os valores para o grupo de teste – cenário II.

	BT	K-NN	SVM-Gaussiana	SVM-Quadrática	SVM-Cúbica
A.S.	70.8% (41.7%)	87.5% (66.7%)	77.1% (66.7%)	66.7% (41.7%)	70.8% (50.0%)
	0.694 (0.455)	0.785 (0.615)	0.740 (0.667)	0.696 (0.500)	0.857 (0.522)
Banheiro	75.0% (57.1%)	87.5% (64.3%)	78.6% (64.3%)	82.1% (64.3%)	82.1% (64.3%)
	0.730 (0.500)	0.810 (0.600)	0.786 (0.667)	0.736 (0.545)	0.773 (0.621)
Circulação	78.6% (80.0%)	81.0% (80.0%)	77.1% (90.0%)	81.0% (90.0%)	78.6% (90.0%)
	0.825 (0.889)	0.861 (0.842)	0.878 (0.857)	0.840 (0.947)	0.722 (0.947)
Cozinha	42.1% (11.1%)	36.8% (33.3%)	42.1% (44.4%)	7.9% (11.1%)	5.3% (11.1%)
	0.516 (0.143)	0.438 (0.400)	0.444 (0.533)	0.136 (0.200)	0.095 (0.200)
Garagem	87.8% (100.0%)	97.6% (100.0%)	92.7% (90.0%)	92.7% (90.0%)	92.7% (90.0%)
	0.809 (0.870)	0.976 (1.000)	0.916 (0.947)	0.864 (0.900)	0.874 (0.857)
Quarto	81.8% (82.6%)	84.1% (82.6%)	75.0% (87.0%)	88.6% (64.3%)	87.5% (100.0%)
	0.735 (0.731)	0.800 (0.745)	0.754 (0.769)	0.743 (0.746)	0.744 (0.767)
Sala	57.4% (41.7%)	66.0% (58.3%)	70.2% (66.7%)	44.7% (58.3%)	55.3% (41.7%)
	0.651 (0.556)	0.689 (0.667)	0.717 (0.727)	0.553 (0.700)	0.612 (0.556)
Varanda	77.8% (81.8%)	82.2% (81.8%)	88.9% (81.8%)	73.3% (63.6%)	77.8% (63.6%)
	0.805 (0.783)	0.902 (0.900)	0.851 (0.818)	0.702 (0.667)	0.708 (0.636)

Tabela A.2 - Performance do classificador multiclasse-binário para cada classe, em que os números entre parênteses apresentam os valores para o grupo de teste – cenário II.

	Ensemble
A.S.	70.8% (50.0%)
	0.716 (0.545)
Banheiro	82.1% (64.3%)
	0.767 (0.600)
Circulação	83.3% (90.0%)
	0.875 (0.947)
Cozinha	31.6% (22.2%)
	0.414 (0.308)
Garagem	92.7% (90.0%)
	0.927 (0.857)
Quarto	84.1% (91.3%)
	0.763 (0.764)
Sala	66.0% (50.0%)
	0.705 (0.600)
Varanda	84.4% (81.8%)
	0.817 (0.818)

Tabela A.3 - Performance do classificador *ensemble* para cada classe, em que os números entre parênteses apresentam os valores para o grupo de teste – cenário II.

	BT	K-NN	SVM-Gaussiana	SVM-Quadrática	SVM-Cúbica
A.S.	98.0% (100.0%)	98.8% (100.0%)	99.0% (83.3%)	99.1% (100.0%)	99.1% (100.0%)
	0.916 (1.000)	0.953 (1.000)	0.956 (0.909)	0.965 (1.000)	0.962 (1.000)
Banheiro	100.0% (100.0%)	100.0% (100.0%)	100.0% (100.0%)	100.0% (100.0%)	100.0% (100.0%)
	1.000 (1.000)	1.000 (1.000)	1.000 (1.000)	1.000 (1.000)	1.000 (1.000)
Circulação	97.2% (80.0%)	98.2% (80.0%)	98.3% (90.0%)	97.5% (80.0%)	97.5% (90.0%)
	0.851 (0.889%)	0.908 (0.889)	0.915 (0.900)	0.864 (0.889)	0.879 (0.857)
Cozinha	97.4% (88.9%)	99.0% (100.0%)	98.8% (100.0%)	98.6% (90.0%)	98.7% (100.0%)
	0.855 (0.941)	0.943 (1.000)	0.931 (1.000)	0.577 (0.900)	0.928 (1.000)
Garagem	96.4% (100.0%)	99.2% (100.0%)	97.4% (90.0%)	96.8% (90.0%)	96.8% (90.0%)
	0.826 (1.000)	0.958 (0.909)	0.875 (0.900)	0.850 (0.857)	0.849 (0.900)
Quarto	92.8% (87.0%)	93.8% (87.5%)	93.4% (95.7%)	92.0% (100.0%)	93.3% (95.7%)
	0.830 (0.833)	0.848 (0.894)	0.841 (0.898)	0.824 (0.852)	0.856 (0.863)
Sala	92.0% (33.3%)	93.6% (41.7%)	93.4% (41.7%)	90.7% (33.3%)	93.6% (58.3%)
	0.588 (0.500)	0.710 (0.556)	0.650 (0.588)	0.406 (0.500)	0.670 (0.737)
Varanda	97.5% (81.8%)	97.8% (81.8%)	97.5% (100.0%)	96.2% (100.0%)	95.7% (100.0%)
	0.884 (0.900)	0.893 (0.900)	0.891 (1.000)	0.841 (1.000)	0.819 (1.000)

Tabela A.4 – Performance do classificador binário para cada classe, em que os números entre parênteses apresentam os valores para o grupo de teste – cenário III.

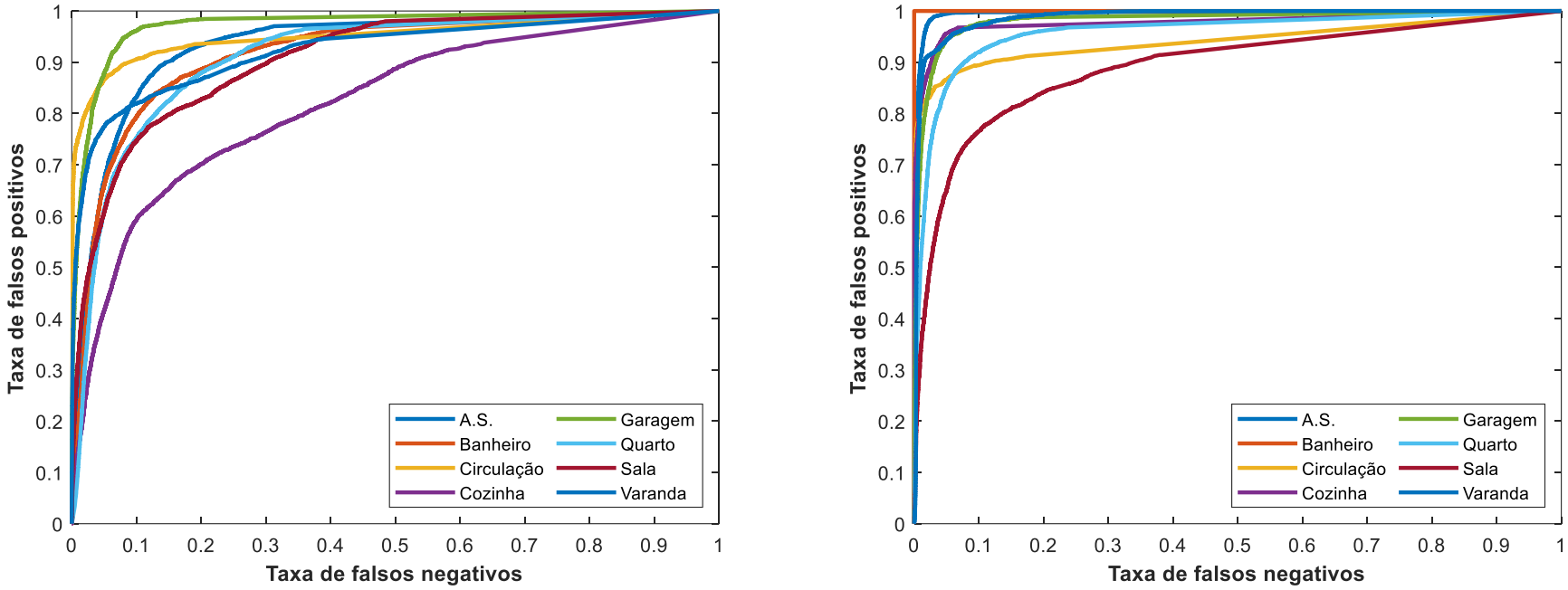
	BT	K-NN	SVM-Gaussiana	SVM-Quadrática	SVM-Cúbica
A.S.	95.8% (100.0%)	100.0% (100.0%)	100.0% (100.0%)	100.0% (100.0%)	100.0% (100.0%)
	0.929 (1.000)	0.970 (1.000)	0.980 (1.000)	0.970 (1.000)	0.960 (1.000)
Banheiro	100.0% (100.0%)	100.0% (100.0%)	100.0% (100.0%)	100.0% (100.0%)	100.0% (100.0%)
	1.000 (1.000)	1.000 (1.000)	1.000 (1.000)	0.966 (1.000)	1.000 (1.000)
Circulação	81.0% (90.0%)	85.7% (80.0%)	90.5% (90.0%)	81.0% (90.0%)	88.1% (90.0%)
	0.895 (0.947)	0.900 (0.889)	0.916 (0.900)	0.861 (0.947)	0.914 (0.900)
Cozinha	86.8% (100.0%)	92.1% (100.0%)	92.1% (100.0%)	92.1% (88.9%)	84.2% (88.9%)
	0.904 (1.000)	0.959 (1.000)	0.959 (1.000)	0.959 (0.941)	0.914 (0.941)
Garagem	90.2% (100.0%)	97.6% (100.0%)	92.7% (90.0%)	92.7% (90.0%)	92.7% (90.0%)
	0.860 (0.909)	0.964 (0.909)	0.894 (0.900)	0.844 (0.900)	0.884 (0.857)
Quarto	89.8% (100.0%)	93.2% (100.0%)	85.2% (95.7%)	89.8% (100.0%)	94.3% (95.7%)
	0.863 (0.902)	0.872 (0.920)	0.852 (0.880)	0.849 (0.868)	0.888 (0.846)
Sala	61.7% (41.7%)	70.2% (58.3%)	70.2% (58.3%)	34.0% (58.3%)	55.3% (50.0%)
	0.690 (0.588)	0.759 (0.700)	0.742 (0.700)	0.478 (0.737)	0.658 (0.667)
Varanda	93.3% (100.0%)	91.1% (100.0%)	97.8% (100.0%)	93.3% (100.0%)	88.9% (100.0%)
	0.866 (0.957)	0.932 (1.000)	0.936 (1.000)	0.840 (1.000)	0.842 (1.000)

Tabela A.5 - Performance do classificador multiclasse-binário para cada classe, em que os números entre parênteses apresentam os valores para o grupo de teste – cenário III.

	Ensemble
A.S.	100.0% (100.0%)
	0.970 (1.000)
Banheiro	100.0% (100.0%)
	1.000 (1.000)
Circulação	88.1% (90.0%)
	0.925 (0.947)
Cozinha	92.1% (100.0%)
	0.959 (1.000)
Garagem	92.7% (90.0%)
	0.905 (0.900)
Quarto	93.2% (100.0%)
	0.891 (0.885)
Sala	68.1% (58.3%)
	0.762 (0.737)
Varanda	97.8% (100.0%)
	0.936 (1.000)

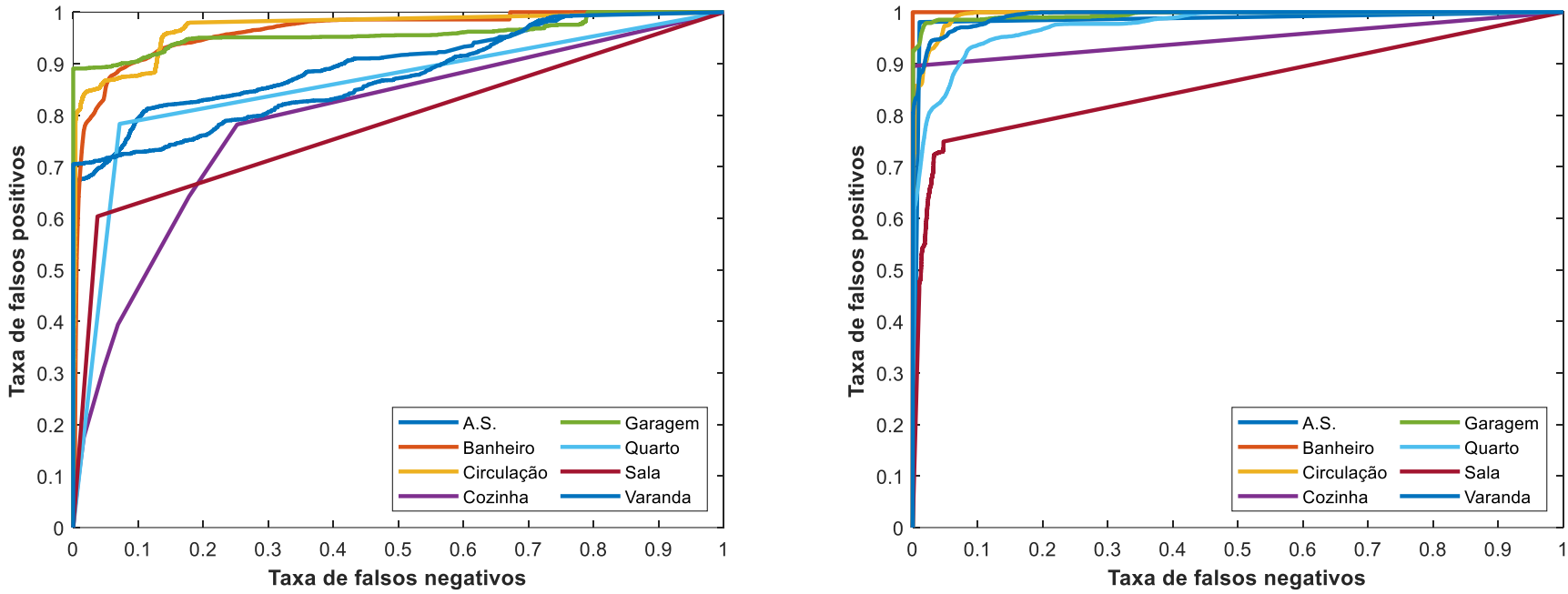
Tabela A.6 - Performance do classificador *ensemble* para cada classe, em que os números entre parênteses apresentam os valores para o grupo de teste – cenário III.

b) CURVAS ROC



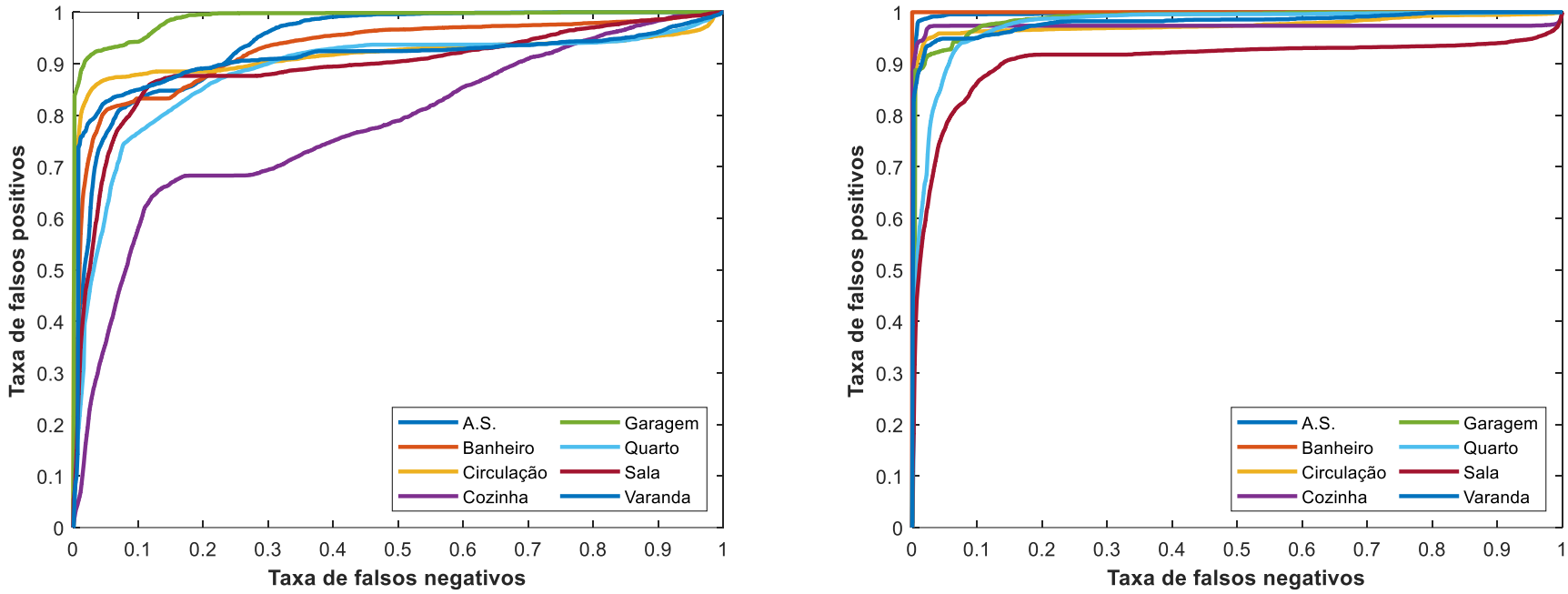
(a) (b)

Figura A.1 – Curvas ROC para o classificador multiclasse-binário, técnica BT, cenário II (a) e cenário III (b).



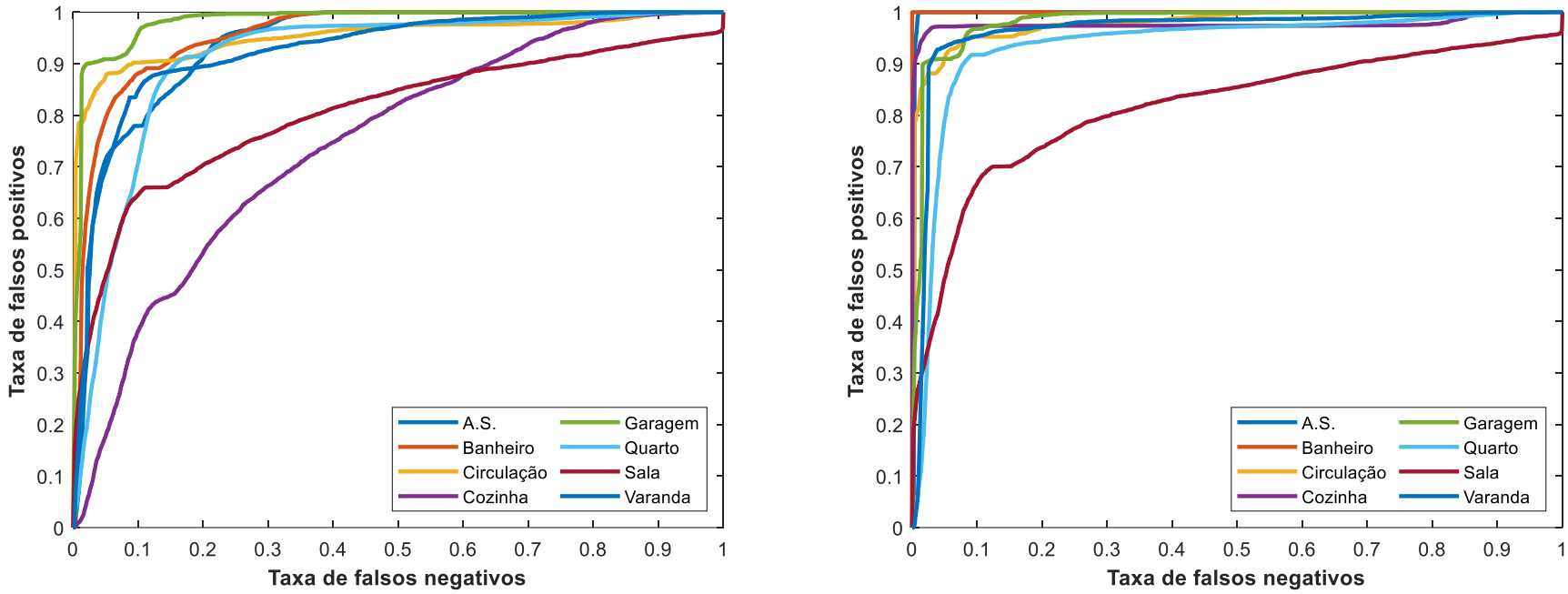
(a) (b)

Figura A.2 – Curvas ROC para o classificador multiclasse-binário, técnica KNN, cenário II (a) e cenário III (b).



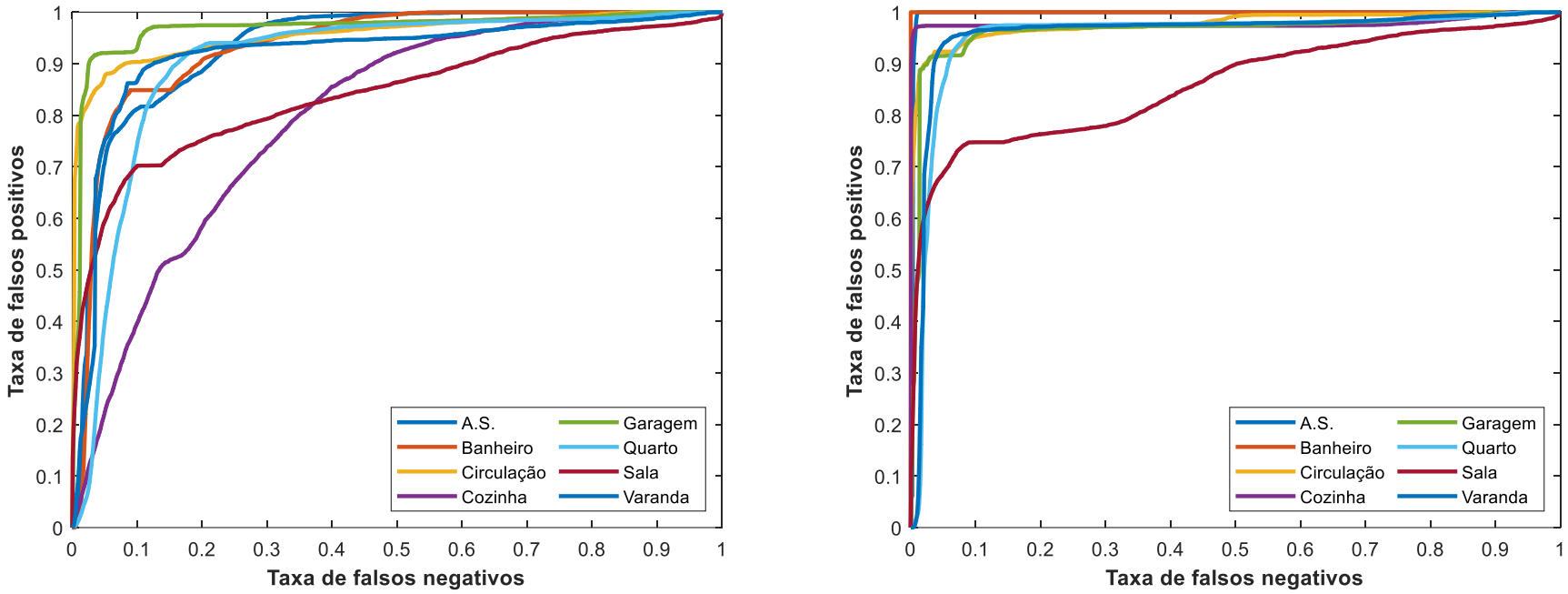
(a) (b)

Figura A.3 – Curvas ROC para o classificador multiclasse-binário, técnica SVM-Gaussiana, cenário II (a) e cenário III (b).



(a) (b)

Figura A.4 – Curvas ROC para o classificador multiclasse-binário, técnica SVM-Quadrática, cenário II (a) e cenário III (b).



(a) (b)

Figura A.5– Curvas ROC para o classificador multiclasse-binário, técnica SVM-Cúbica, cenário II (a) e cenário III (b).

c) PARÂMETROS DOS MODELOS ANALISADOS

	Configuração	Variável no código	Valor do parâmetro
	Bagged Tree		
Número de ciclos de aprendizagem	Otimizado	NumLearningCyles	10
Número máximo de ramificações	Otimizado	MaxNumSplits	355
Número mínimo de nó folhas	Otimizado	MinLeafSize	3
Número de variáveis por amostra	Otimizado	NumVariblesToSample	3
	k-NN		
Número de vizinhos	Otimizado	NumNeighbors	3
Distância	Otimizado	Distance	mahalanobis
Função peso da distância	Otimizado	DistanceWeight	inverse
	SVM-Gausiana		
Restrição de caixa	Otimizado	BoxConstraint	14,097
Fator de escala da função de covariança	Auto	KernelScale	-
	SVM-Quadrática		
Restrição de caixa	Otimizado	BoxConstraint	0,109
Fator de escala da função de covariança	Auto	KernelScale	-
	SVM-Cúbica		
Restrição de caixa	Otimizado	BoxConstraint	0,001
Fator de escala da função de covariança	Auto	KernelScale	-

Tabela A.7 - Parâmetros para o modelo de classificação multiclasse – cenário I.

	Configuração	Variável no código	Valor do parâmetro			
Bagged Tree			**A.S.**	**Banheiro**	**Circulação**	**Cozinha**
Número de ciclos de aprendizagem	Otimizado	NumLearningCyles	12	10	13	201
Número máximo de ramificações	Otimizado	MaxNumSplits	87	156	32	149
Número mínimo de nó folhas	Otimizado	MinLeafSize	1	12	1	4
Número de variáveis por amostra	Otimizado	NumVariblesToSample	5	5	3	6
k-NN						
Número de vizinhos	Otimizado	NumNeighbors	203	112	17	5
Distância	Otimizado	Distance	cityblock	mahalanobis	minkowski	cityblock
Função peso da distância	Otimizado	DistanceWeight	inverse	squaredinverse	inverse	equal
SVM-Gausiana						
Restrição de caixa	Otimizado	BoxConstraint	9,297	73,007	47,734	36,615
Fator de escala da função de covariança	Auto	KernelScale	-			
SVM-Quadrática						
Restrição de caixa	Otimizado	BoxConstraint	278,790	14,535	11,160	0,134
Fator de escala da função de covariança	Auto	KernelScale	-			
SVM-Cúbica						
Restrição de caixa	Otimizado	BoxConstraint	0,327	329,110	0,002	0,002
Fator de escala da função de covariança	Auto	KernelScale	-			

Tabela A.8 - Parâmetros para o modelo de classificação binária do cenário II.

	Configuração	Variável no código	Valor do parâmetro			
Bagged Tree			**Garagem**	**Quarto**	**Sala**	**Varanda**
Número de ciclos de aprendizagem	Otimizado	NumLearningCyles	11	10	101	11
Número máximo de ramificações	Otimizado	MaxNumSplits	21	133	15	115
Número mínimo de nó folhas	Otimizado	MinLeafSize	4	4	1	1
Número de variáveis por amostra	Otimizado	NumVariblesToSample	3	5	5	6
k-NN						
Número de vizinhos	Otimizado	NumNeighbors	154	1	7	203
Distância	Otimizado	Distance	hamming	cityblock	minkowski	jaccard
Função peso da distância	Otimizado	DistanceWeight	squaredinverse	squaredinverse	equal	squaredinverse
SVM-Gausiana						
Restrição de caixa	Otimizado	BoxConstraint	17,616	27,190	2,660	998,970
Fator de escala da função de covariança	Auto	KernelScale	-	-	-	-
SVM-Quadrática						
Restrição de caixa	Otimizado	BoxConstraint	123,080	0,129	0,293	0,403
Fator de escala da função de covariança	Auto	KernelScale	-	-	-	-
SVM-Cúbica						
Restrição de caixa	Otimizado	BoxConstraint	2,503	0,007	0,115	0,862
Fator de escala da função de covariança	Auto	KernelScale	-	-	-	-

Tabela A.9 - Parâmetros para o modelo de classificação binária do cenário II (continuação).

	Configuração	Variável no código	Valor do parâmetro			
Bagged Tree			**A.S.**	**Banheiro**	**Circulação**	**Cozinha**
Número de ciclos de aprendizagem	Otimizado	NumLearningCyles	10	150	15	10
Número máximo de ramificações	Otimizado	MaxNumSplits	364	100	12	7
Número mínimo de nó folhas	Otimizado	MinLeafSize	2	11	1	1
Número de variáveis por amostra	Otimizado	NumVariblesToSample	8	3	6	7
k-NN						
Número de vizinhos	Otimizado	NumNeighbors	1	148	203	1
Distância	Otimizado	Distance	correlation	correlation	correlation	cosine
Função peso da distância	Otimizado	DistanceWeight	squaredinverse	squaredinverse	inverse	inverse
SVM-Gausiana						
Restrição de caixa	Otimizado	BoxConstraint	437,860	222,990	999,370	6,391
Fator de escala da função de covariança	Auto	KernelScale	-			
SVM-Quadrática						
Restrição de caixa	Otimizado	BoxConstraint	1,479	475,750	0,025	4,126
Fator de escala da função de covariança	Auto	KernelScale	-			
SVM-Cúbica						
Restrição de caixa	Otimizado	BoxConstraint	0,747	475,750	71,984	0,106
Fator de escala da função de covariança	Auto	KernelScale	-			

Tabela A.10 - Parâmetros para o modelo de classificação binária do cenário III.

	Configuração	Variável no código	Valor do parâmetro			
Bagged Tree			**Garagem**	**Quarto**	**Sala**	**Varanda**
Número de ciclos de aprendizagem	Otimizado	NumLearningCyles	12	10	12	45
Número máximo de ramificações	Otimizado	MaxNumSplits	23	354	206	26
Número mínimo de nó folhas	Otimizado	MinLeafSize	1	6	1	1
Número de variáveis por amostra	Otimizado	NumVariblesToSample	8	7	4	5
k-NN						
Número de vizinhos	Otimizado	NumNeighbors	202	203	2	198
Distância	Otimizado	Distance	correlation	chebychev	seuclidean	euclidean
Função peso da distância	Otimizado	DistanceWeight	inverse	squaredinverse	inverse	squaredinverse
SVM-Gausiana						
Restrição de caixa	Otimizado	BoxConstraint	24,515	0,873	4,291	13,882
Fator de escala da função de covariança	Auto	KernelScale	-	-	-	-
SVM-Quadrática						
Restrição de caixa	Otimizado	BoxConstraint	995,080	0,009	0,037	5,883
Fator de escala da função de covariança	Auto	KernelScale	-	-	-	-
SVM-Cúbica						
Restrição de caixa	Otimizado	BoxConstraint	202,130	0,714	0,209	0,189
Fator de escala da função de covariança	Auto	KernelScale	-	-	-	-

Tabela A.11 - Parâmetros para o modelo de classificação binária do cenário III (continuação).