Faculdade de Ciência e Tecnologia de Montes Claros

**Felipe Israel Corrêa**

**SISTEMA AGREGADOR PARA ANÁLISE DE CARACTERÍSTICAS DE IMÓVEIS COM RASTREAMENTO WEB E APRENDIZADO DE MÁQUINA**

Montes Claros- MG

2018

**Felipe Israel Corrêa**

**SISTEMA AGREGADOR PARA ANÁLISE DE CARACTERÍSTICAS DE IMÓVEIS COM RASTREAMENTO WEB E APRENDIZADO DE MÁQUINA**

Projeto de graduação apresentado ao Curso de Engenharia da Computação, da Faculdade de Ciência e Tecnologia de Montes Claros, como parte dos requisitos para obtenção do título do Engenheiro da Computação.

Orientador: **PROF. DR. RENATO DOURADO MAIA.**

Montes Claros - MG

2018

FUNDAÇÃO EDUCACIONAL MONTES CLAROS

**Faculdade de Ciência e Tecnologia de Montes Claros**

Felipe Israel Corrêa

**SISTEMA AGREGADOR PARA ANÁLISE DE CARACTERÍSTICAS DE IMÓVEIS COM RASTREAMENTO WEB E APRENDIZADO DE MÁQUINA**

Este projeto de graduação foi julgado adequado como parte dos requisitos para a obtenção do diploma de Engenheiro da Computação aprovada pela banca examinadora da Faculdade de Ciência e Tecnologia de Montes Claros.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Prof. Dr. Maurílio José Inácio**

Coord. do Curso de Engenharia da Computação

**Banca Examinadora**

Prof. Dr. Renato Dourado Maia, FACIT / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Orientador)

Prof. Examinador 1 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Examinador 2 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Montes Claros, xx de Junho de 2018.

*Para Lavínia, amor maior.*

AGRADECIMENTOS

“Não se espante com a altura do voo. Quanto mais alto, mais longe do perigo. Quanto mais você se eleva, mais tempo há de reconhecer uma pane. É quando se está próximo do solo que se deve desconfiar.”

Santos Dumont

**RESUMO**

A democratização do acesso a internet tem proporcionado uma geração de dados, sobretudo digitais, nunca antes vista na história da humanidade. Nunca se produziu tanta informação em tão pouco tempo. A partir desta perspectiva este estudo tem por objetivo coletar os dados sobre imóveis residenciais, da cidade de Montes Claros, adquiridos em sites de empresas imobiliárias e desenvolver um sistema que os agregue e gere observações relevantes sobre as características destes empreendimentos. A aquisição das informações será feita por meio de um Rastreador Web e a análise dos dados ocorrerá através da aplicação de Aprendizagem de Máquina no que tange a utilização de um algoritmo regressor e um de recomendação e busca. Para alcance do objetivo foram estudados os conceitos que englobam os processos de rastreamento e aprendizagem de máquina, verificadas as condições para desenvolvimento do trabalho e descritos os algoritmos de codificação. Como proposto, o sistema obteve êxito em predizer valores de um imóvel com base em suas características, tais como bairro, número de quartos, banheiros, vagas de estacionamento e tamanho da área, além de efetuar a recomendação e busca de imóveis. Todo ele foi confeccionado através da linguagem de programação *Python* no ambiente de desenvolvimento *Visual Studio Code*, em conjunto com as bibliotecas *Requests* e *BeautifulSoup* para o rastreador e as bibliotecas científicas e gráficas *Scikit-learn*, Pandas, Numpy, Matplotlib e *Bokeh* para aprendizagem de máquina.

**Palavras-Chave:** *Dados, Imóveis, Rastreador Web, Aprendizagem de Máquina, Python.*

**ABSTRACT**

The popularization of Internet access has provided a generation of data, especially digital data, never before seen in the history of mankind. Never has so much information been produced in such a short time. From this perspective, this study aims to collect data on real estate acquired on websites of real estate companies and develop a system that aggregates and generates relevant observations about the characteristics of these developments. The information acquisition will be done through a Web Tracker and data analysis will take place through the Machine Learning application. In order to reach the objective, the concepts that comprise the processes of machine tracking and learning have been studied, the conditions for the development of the work were verified and the coding algorithms were described. As proposed the system is able to predict values of a property based on its characteristics, such as neighborhood, number of rooms, bathrooms, parking spaces and size of the area, in addition to recommending and searching for real estate. The system was made using the Python programming language in the Visual Studio Code development environment, together with the Requests and BeautifulSoup libraries for the tracker, and the Scikit-learn, Pandas, Numpy, Matplotlib and Bokeh scientific and graphic libraries for learning machine. At the end, this work was successful in developing the system.

**Keywords:** *Data, Real Estate, Web Tracker, Machine Learning, Python.*

**LISTA DE FIGURAS**

FIGURA 1 - *Document Object Module*………………………………… 15

FIGURA 2 - Código fonte e *tags* HTML……………………………………….. 16

FIGURA 3 - Diagrama de fluxo de um rastreador………………………… 16

FIGURA 4 - Fluxo de operações de um sistema de AM………………. 21

FIGURA 5 - Representação intercepto e coeficiente de inclinação………….. 22

FIGURA 6 - Representação do erro………………………………………… 23

FIGURA 7 - Linha de regressão da relação entre variáveis A e B………… 24

FIGURA 8 - Árvore de decisão……………………………………………. 24

FIGURA 9 - Árvore de decisão e divisões no espaço………………….. 25

FIGURA 10 - Distância calculada entre vetores a e b……………………. 29

FIGURA 11 - Similaridade do cosseno entre os ponto a e b………………… 29

FIGURA 12 - Vetores a e b distantes 90º……………………………………… 30

FIGURA 13 - Vetores a e b com ângulo igual a 0º………………….... 30

FIGURA 14 - K-vizinhos mais próximos em classificação…………………... 31

FIGURA 15 - Diagrama do compilador Python……………………………….. 33

FIGURA 16 - Exemplos de gráficos produzidos com biblioteca Matplotlib……... 37

FIGURA 17 - Exemplos de gráficos criados com a biblioteca Bokeh………… 39

FIGURA 18 - Fluxograma das aplicações…………………………………... 40

FIGURA 19 - Script rastreador web……………………………………….. 41

FIGURA 20 - Exemplo de busca de tags para rastreador web, 42

FIGURA 21 - Exemplo de definição das tags para busca dos dados, 42

FIGURA 22 - Código para armazenamento dos dados, 43

FIGURA 23 - Estrutura de criação da tabela no servidor MySQL, 43

FIGURA 24 - Códigos para ajuste dos nomes dos bairros, 45

FIGURA 25 - Porcentagem de valores nulos por coluna, 46

FIGURA 26 - Código para imputação de valores da média nos elementos nulos, por coluna, 46

FIGURA 27 - Variáveis independentes no hiperplano, 48

FIGURA 28 - Matriz de correlação entre as variáveis, 48

FIGURA 29 - Processo de validação cruzada, 50

FIGURA 30 - Separação do conjunto de dados entre treino e teste, 50

FIGURA 31 - Método de avaliação do modelo linear, 51

FIGURA 32 - Construção do modelo, 52

FIGURA 33 - Método de avaliação e definição da profundidade da árvore de decisão, 52

FIGURA 34 - Teste do modelo com 100 árvores, 53

FIGURA 35 - Método de cálculo para floresta aleatória, 54

FIGURA 36 - Erro global entre valores reais e preditos, 56

FIGURA 37 - Método para a média das características dos imóveis, 57

FIGURA 38 - Método para predição de preço do imóvel, 58

FIGURA 39 - Estimativas de preços para apartamentos de 1 quarto, 58

FIGURA 40 - Estimativas de preços para apartamentos de 2 quartos, 59

FIGURA 41 - Estimativas de preços para apartamentos de 3 quartos, 59

FIGURA 42 - Estimativas de preços para apartamentos de 4 quartos, 60

FIGURA 43 - Estimativas de preços para apartamentos de 5 quartos, 60

FIGURA 44 - Código para busca e armazenamento das coordenadas geográficas, 62

FIGURA 45 - Leitura e atribuição às variáveis correspondentes dos conjuntos de dados, 63

FIGURA 46 - Método de inserção das coordenadas geográficas, 63

FIGURA 47 - Método calculador de distância, 64

FIGURA 48 - Método para selecionar imóveis de acordo com o bairro, 64

FIGURA 49 - Métodos de busca e recomendação de apartamentos, 65

FIGURA 50 - Chamada dos métodos de busca e recomendação, 66

FIGURA 51 - Resultado dos testes de busca e recomendação para métrica euclidiana, 66

FIGURA 52 - Resultado dos testes de busca e recomendação para métrica do cosseno, 67

FIGURA 53 - Itens recomendados e valor da dissimilaridade, 68

FIGURA 54 - Implementação para cálculo da precisão e *recall*, 69

FIGURA 55 - Implementação fórmula F1-*score*, 70

FIGURA 56 - Imóveis apresentados ao usuário calculados pela distância euclidiana, 70

FIGURA 57 - Opções de imóveis apresentados ao usuário calculados pela similaridade do cosseno, 71

FIGURA 58 - Quantidade de imóveis por bairro, 72

FIGURA 59 - Distribuição de imóveis pela cidade, 73

FIGURA 60 - Quantidade de imóveis, por bairro, com base na quantidade de quartos, 74

FIGURA 61 - Porcentagem de banheiros e vagas de garagem, 74

FIGURA 62 - Estimativa para o menor apartamento, 75

FIGURA 63 - Estimativa para o maior apartamento, 75

FIGURA 64 - Estimativa com área aumentada em 10m², 76

FIGURA 65 - Diferença no valor do preço do imóvel a cada aumento, 76

FIGURA 66 - Diferença de preços baseada no aumento do número de quartos, 77

FIGURA 67 - Diferença de preços baseada no aumento do número de banheiros, 78

FIGURA 68 - Diferença de preços baseada no aumento do número de vagas, 79

**LISTA DE TABELAS**

TABELA 1 - Característica dos imóveis…………………………………………………... 01

TABELA 2 - Conjunto de dados após pré-processamento…………………………….. 02

TABELA 3 - Resultado do cálculo das métricas do modelo linear…………………….. 03

TABELA 4 - Resultado das métricas da árvore de decisão, de acordo com sua profundidade…………………………………………………………………………………. 04

TABELA 5 - Resultado das métricas da floresta aleatória, de acordo com quantidade de árvores e profundidade ………………………………………………………………… 05

TABELA 6 - Média das características dos apartamentos com base na quantidade de quartos………………………………………………………………………………………… 06

TABELA 7 - Resultado cálculos de eficiência das recomendações ………………….. 07

**SUMÁRIO**

**INTRODUÇÃO, 13**

**CAPÍTULO 1 SISTEMA AGREGADOR PARA ANÁLISE DE CARACTERÍSTICAS**

**DE IMÓVEIS COM RASTREAMENTO WEB E APRENDIZADO DE MÁQUINA, 16**

1.1 Rastreador Web, 16

1.2 Fluxo de Busca, 17

1.3 Aprendizado de Máquina, 19

1.3.1 Aplicações, 20

1.4 Tarefas de Aprendizado, 20

1.5 Tipos de Aprendizado, 21

1.5.1 Aprendizagem Supervisionada, 21

1.5.2 Aprendizagem Não Supervisionada, 22

1.6 Fluxo de Funcionamento de um Sistema AM, 22

1.7 Sistema de Regressão, 23

1.7.1 Regressão Linear Simples e Múltipla, 24

1.7.2 Árvore de Decisão e Floresta Aleatória, 26

1.8 Sistema de Recomendação, 28

1.9 Tipos de Recomendação, 29

1.9.1 Filtragem Colaborativa, 29

1.9.2 Recomendação Baseada em Conteúdo, 29

1.10 Técnicas de Recomendação, 30

1.10.1 Recomendação Baseada em Vizinhança, 30

1.11 Distância Euclidiana, 30

1.12 Similaridade do Cosseno, 31

1.13 K-Vizinhos Mais Próximos, 32

**CAPÌTULO 2 MATERIAIS E MÉTODOS, 34**

2.1 Ambiente de Desenvolvimento, 34

2.2 Linguagem Python, 35

2.3 Bibliotecas, 36

2.3.1 Requests, 36

2.3.2 BeautifulSoup, 37

2.3.3 Googlemaps Geocoding, 37

2.3.4 Mysql.connector, 37

2.3.5 Scikit-learn, 37

2.3.6 Numpy, 38

2.3.7 Pandas, 38

2.3.8 Matplotlib, 39

2.3.9 Bokeh, 40

2.4 Questionário, 40

**CAPÍTULO 3 RESULTADOS: APRESENTAÇÃO, ANÁLISE E DISCUSSÃO, 41**

3.1 Rastreador Web, 42

3.2 Regressor, 46

3.3 Sistema de Recomendação e Busca, 62

3.4 Análise Exploratória dos Dados, 73

**CAPÍTULO 4 APLICAÇÃO, 80**

**CONSIDERAÇÕES FINAIS, 81**

**REFERÊNCIAS, 83**

**INTRODUÇÃO**

O constante crescimento dos centros urbanos tem ocasionado o aumento de demanda por uma moradia proporcionalmente, tornando este fato extremamente importante para empresas do ramo imobiliário. Esse mercado sofreu uma desaceleração causada pela recessão sofrida pelo país a partir de 2012, mas para 2018, tais empresas prevêem um retorno de crescimento, consequência da baixa da taxa SELIC e as novas regras definidas pelo Governo Federal em relação ao programa Minha Casa Minha Vida (EXAME, 2017).

A oferta de imóveis muitas vezes é feita através de ferramentas tecnológicas, principalmente por meio dos sites mantidos pelas empresas deste segmento. Gera-se então um imenso volume de dados digitais que podem chegar a milhares de gigabytes de informação, contendo as características de cada imóvel e a transação comercial disponibilizada, seja de compra, venda, aluguel, dentre outras. E esse aumento na produção de dados digitais é uma das características que podemos imputar a toda humanidade atual, pois em um estudo realizado pela empresa IDC, especialista em análise de dados, para a empresa EMC, apontou que o volume de dados digitais tem dobrado a cada dois anos, chegando a 4,4 trilhões de gigabytes de informação no ano de 2013, e que em 2020 os dados criados e copiados chegarão à marca de 44 trilhões de gigabytes (EMC, 2014).

Por meio desta perspectiva o uso da Aprendizagem de Máquina, técnica de computação baseada em Inteligência Artificial, desenvolvida em meados da década de 70 Faceli et. al (2011), passou a ser usada para tratativa de conjuntos de dados que são impossíveis de serem analisados mecânica e mentalmente por um humano, com o objetivo de aprendizado e predição de padrões.

O problema surge quando ao buscar uma informação referente a um imóvel, o usuário se depara com inúmeras opções de sites e empresas, consequentemente, uma imensa quantidade de aspectos, em grande parte repetidos, e que não conseguem expor de forma clara as vantagens e desvantagens da aquisição do imóvel, principalmente se o valor fornecido pela imobiliária é condizente com a região na qual o imóvel se encontra.

Como então obter informações adicionais, por meio de todos esses dados e que ainda façam sentido para o usuário e possa auxiliá-lo na decisão de escolher o melhor imóvel, ou seja, o que mais atende a seus objetivos a curto e longo prazo?

Este estudo recolheu os dados referentes aos imóveis e aplicou técnicas de Aprendizado de Máquina, tais como Regressão Linear, Árvores de Decisão e Sistemas de Recomendação, retirando aspectos que adicionam valor às informações que são disponibilizadas atualmente pelas empresas do setor imobiliário.

Através de um conjunto de dados de imóveis residenciais obtidos automaticamente em sites imobiliários da cidade de Montes Claros, este estudo construiu um sistema que pôde fornecer aos usuários não somente informações sobre os imóveis, mas também sobre o bairro e cidade. Para tal foram necessários o atendimento dos seguintes objetivos específicos:

1. criação de um rastreador web, para busca dos dados;
2. armazenamento correto dos dados obtidos;
3. análise e manipulação correta do conjunto de dados;
4. criação de um sistema de regressão e recomendação consistentes;
5. análise exploratória satisfatória dos dados;

Este estudo possibilitou retirar, dos dados disponibilizados, ideias que complementem o conhecimento prévio dos usuários, abrangendo não somente o imóvel, mas também o bairro e a cidade, de forma que auxiliem este a tomar a melhor decisão, ou a decisão que achar pertinente para o momento. Outro benefício é que o modelo foi criado através de ferramentas de análise e programação gratuitas.

No total o trabalho foi composto por quatro capítulos, sendo o capítulo um o detalhamento dos conceitos teóricos referentes às técnicas de rastreamento web e aprendizagem de máquina. O capítulo dois detalhada a metodologia e os materiais empregados para elaboração do sistema. O capítulo três apresenta os resultados obtidos pelas ferramentas desenvolvidas juntamente com a discussão destes. E no capítulo quatro é apontada uma indicação de qual área e tipos de usuário este trabalho é destinado.

**CAPÍTULO 1 SISTEMA AGREGADOR PARA ANÁLISE DE CARACTERÍSTICAS DE IMÓVEIS COM RASTREAMENTO WEB E APRENDIZADO DE MÁQUINA**

Neste capítulo serão discutidos os conceitos e teorias dos componentes que, juntos, foram utilizados para a criação do Sistema Agregador.

Farão parte os seguintes temas:

1. Rastreador Web.
2. Aprendizagem de Máquina.
3. Sistema de Regressão.
4. Sistema de Recomendação.
5. Algoritmo K-vizinhos mais próximos.

**1.1. Rastreador Web**

A democratização do acesso à internet e as comodidades trazidas por esta plataforma ao mundo moderno, provocou o ingresso de vários setores aos serviços *online*. Atualmente são mais de 4 bilhões de usuários conectados, o que corresponde a 53% da população mundial, segundo dados do site *Internet World Stats* (2017), navegando em mais de 966 milhões de páginas (SMITH, 2017).

Páginas de compra, venda, entretenimento são, hoje, indispensáveis para o público em geral. Frente a este crescimento e de ser humanamente impossível consultar todos os sites manualmente, mesmo aqueles que sejam específicos para um usuário ou grupo de usuários, foi desenvolvida a técnica conhecida como rastreamento web, que segundo Bernard (2017), consiste em coletar automaticamente dados de páginas web, extrair informações específicas e armazená-los para uso posterior. O site *Data for thoughts* (2014), classifica esta prática como um subgrupo de uma técnica conhecida como *Data Scraping* (Raspagem de Dados), em que se procura extrair informações legíveis, de um software para outro.

Esta técnica possui alguns outros nomes, alguns já em desuso, que são *screen scraping*, *web harvesting* e similares (MITCHELL, 2015).

Normalmente estes dados são apresentados de forma não estruturada, o que impede de serem utilizados em bases que possam ser interpretadas por algoritmos, portanto se faz necessário transformá-los em dados estruturados e que sejam semanticamente compreensíveis e interpretáveis pela máquina (BARREIRA, 2014).

**1.2. Fluxo de Busca**

O método mais utilizado para busca das informações, é feito através das *tags* disponíveis nos códigos fonte das páginas HTML ou XML (HEMENWAY e CALISHAIN, 2003). Este método tira proveito da forma semiestruturada das páginas HTML, por serem parecidas como uma árvore (BARREIRA, 2014).

Ainda segundo Barreira (2014), a estrutura das *tags* de uma página HTML, representam diferentes níveis, formando uma estrutura hierárquica, com isto utiliza-se o *Document Object Module* (DOM). O DOM define a estrutura lógica de um documento e a forma como ele deve ser acessado e manipulado W3C (2004), conforme mostrada na figura 1 abaixo:

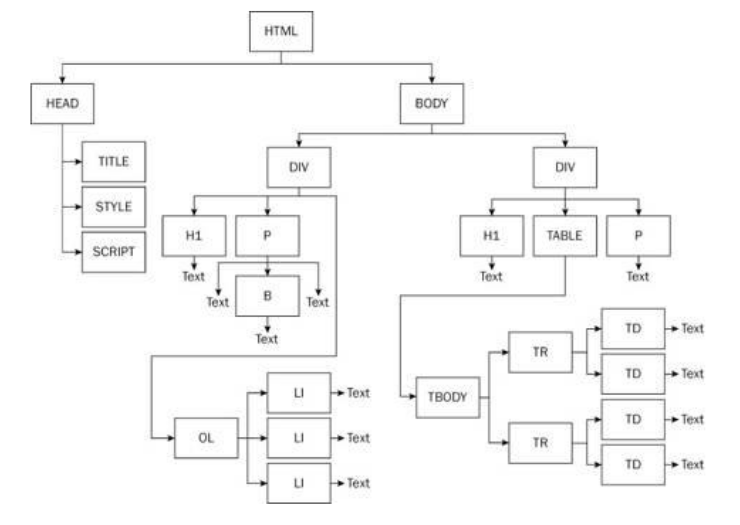


FIGURA 1: *Document Object Module*.

Fonte: BARREIRA (2004, p. 8)

A figura 2, detalha o código fonte de uma página e as *tags* referentes a cada conteúdo:

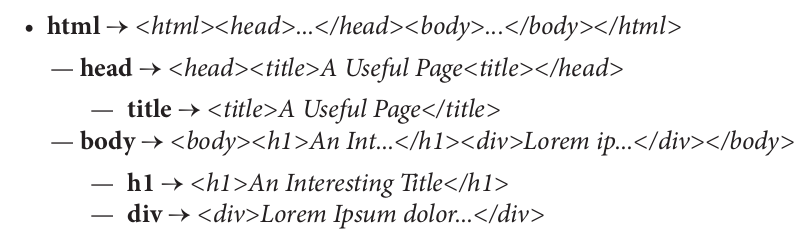


FIGURA 2: Código fonte e *tags* HTML.

Fonte: MITCHELL (2015, p. 8)

Para se buscar um determinado texto, a figura 3 apresenta o fluxo do utilizado pelo rastreador:

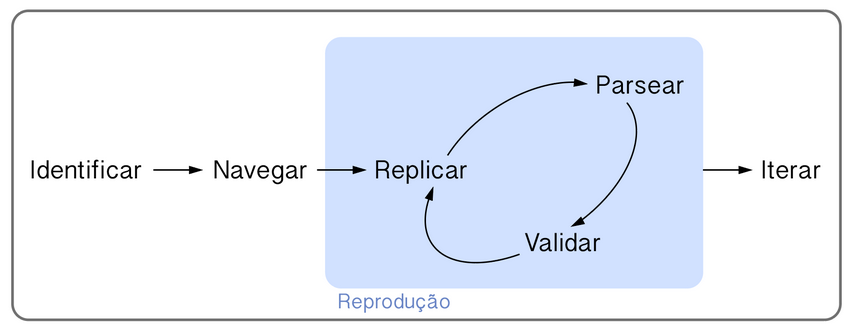


FIGURA 3: Diagrama de fluxo de um rastreador.

Fonte: LENTE (http://curso-r.com/blog/2018/02/18/2018-02-18-fluxo-scraping/)

Cada um destes verbos indica uma das fases do processo e a caixa azulada indica uma iteração que será feita até que a coleta funcione (LENTE, 2018).

Abaixo são detalhados, segundo Lente (2018), os processos executados por cada verbo:

1. Identificar:Processo em que se identifica a informação a se coletar, bem como entender a estrutura da página web para traçar a forma como o dado será extraído.
2. Navegar:Neste processo, utiliza-se a ferramenta de desenvolvedor do próprio navegador, a fim de encontrar as fontes dos dados que desejamos coletar, tais como uma *tag*, uma função JavaScript, ou outro tipo de marcador.
3. Replicar:Processo que serve para executar várias solicitações, para obtenção da informação desejada, ou seja, as solicitações são replicadas. Em suma é a solicitação de acesso à página.
4. Parsear:Palavra originada do verbo, em inglês, *to parse*, pode significar estudar ou analisar e, no contexto de um rastreador, significa extrair os dados de um arquivo HTML ou XML.
5. Validar:A validação é feita reproduzindo os procedimentos acima para outras páginas com o objetivo de verificar se a informação está sendo extraída de forma correta. Caso sejam encontrados erros é necessário voltar ao passo de replicação, rever o comportamento da página e parsear os dados corretamente.
6. Iterar:Consiste em, finalizados os procedimentos acima com sucesso, colocar o rastreador para busca contínua dos dados desejados. Na maioria das vezes cria-se uma função que recebe uma série de *links* e aplica o mesmo procedimento em cada um deles de forma automática.

Todo este fluxo, em geral, é feito através de uma biblioteca ou uma *Application Programming Interface* (API), traduzida como Interface de Programação de Aplicações, dentro de uma determinada linguagem de programação (MITCHELL, 2015).

**1.3. Aprendizado de Máquina**

Em todo o histórico da evolução das espécies, uma das características essenciais a qualquer grupo é o aprendizado. Através deste processo as espécies foram capazes de se adaptar, criar laços e se estabelecer em um meio. Aprender, segundo o dicionário Ferreira (2005), significa tornar-se capaz de algo graças a estudo, observação e experiência.

Tendo em vista esses pontos, Fernandes (2003), estabelece que um sistema aprende, caso este seja hábil em identificar a informação por ele já reconhecida e processá-la logo se torne disponível.

Assim Faceli et. al (2011), constatam que uma máquina resolve problemas por meio de um algoritmo que define os passos de sua resolução, portanto o Aprendizado de Máquina (AM) tenta criar algoritmos, baseados no processo de aprendizagem à partir de uma experiência passada ou reconhecimento de padrões, que sejam capazes de organizar um novo conhecimento, atingindo o que é chamado de comportamento inteligente.

Várias são as definições de AM encontradas na literatura, em uma delas Mueller e Massaron (2016), o consideram um subárea da Inteligência Artificial tendo como base teorias de matemática e estatística.

Para outros como Raschka e Mirjalili (2017), o AM oferece uma alternativa de melhorar a predição de modelos e tomar decisões baseadas nos dados, por meio do conhecimento obtido através destes.

**1.3.1. Aplicações**

Alpaydin (2010), descreve alguns exemplos de aplicações de aprendizado de máquina mais utilizadas:

1. Aprendizado por associações: consiste em aprender a condição de um elemento a outro. Exemplo: um consumidor ao comprar um produto X, tem mais propensão a comprar também o produto Y e não o produto Z. Tem como fórmula matemática de probabilidade condicional P(X|Y), em que Y é o produto que deseja-se condicionar a X. E X o conjunto de produtos que o cliente já comprou.
2. Classificação: como o nome sugere, consiste em classificar um dado novo em uma determinada classe de acordo com a classificação dada para dados analisados anteriormente.
3. Regressão: consiste em estimar um valor de saída para associado ao dado, baseado em suas características inseridas.

**1.4. Tarefas de Aprendizado**

O paradigma de aprendizado é um dos critérios utilizados para definir o tipo de tarefa desenvolvido pelo AM, sendo, de acordo com este critério, divididas em tarefas preditivas ou descritivas (FACELI et. al 2011).

Alpaydin (2010), define uma tarefa preditiva quando esta é capaz de estimar dados futuros, com base nos dados passados e do tipo descritiva quando da possibilidade de aprender e/ou reconhecer padrões. Existe ainda a possibilidade de mesclar estes dois tipos de abordagem.

**1.5. Tipos de Aprendizagem**

Para Russell e Norvig (2013), um sistema estará aprendendo se for capaz de aprimorar suas estimativas futuras após realizar observações sobre o mundo.   
 Eles elencam três principais técnicas de aprendizagem, que são:

1. Aprendizagem supervisionada.
2. Aprendizagem não-supervisionada.
3. Aprendizagem por reforço.

Neste estudo discutiremos somente as técnicas não-supervisionada e supervisionada, com ênfase maior nesta última por se tratar da técnica utilizada para criação do Sistema Agregador.

**1.5.1. Aprendizagem Supervisionada**

Este tipo de aprendizagem é baseada em um conjunto de dados, cujas saídas ou algum outro tipo de representação do comportamento que o sistema deve apresentar já são previamente conhecidos (CASTRO e FERRARI, 2016). Em linhas gerais, o sistema observa amostra de dados em pares de entrada e saída e aprende uma função que faz a rota entre eles.

A técnica supervisionada é descrita por Russell e Norvig (2013), da seguinte maneira:

1. É fornecido previamente um conjunto de treinamento com N pares de entrada e saída (), (), (), ..., (), onde a geração de cada y é fornecida por uma função não conhecida .
2. Com essa relação observada é objetivo do sistema gerar uma outra função que se assemelhe à função verdadeira .
3. Sendo a função uma hipótese, a aprendizagem se torna uma busca por hipóteses possíveis que terá uma boa eficiência mesmo caso um novo dado seja inserido.
4. Para que a eficiência da hipótese seja observada, deve-se fornecer conjuntos de testes e conjuntos de treinamentos, sendo estes diferentes.
5. A hipótese será eficiente se prevê de forma correta o valor de para novos dados.
6. Em geral, sendo uma saída de valores finitos, como ensolarado, nublado ou chuvoso, a aprendizagem recebe o nome de classificação, caso seja um número, como o valor da temperatura em determinada hora do dia, a aprendizagem é chamada de regressão.

**1.5.2. Aprendizagem Não Supervisionada**

Neste tipo de aprendizagem a saída dos dados é desconhecida, portanto o objetivo é identificar padrões Alpaydin (2010), ou seja, o algoritmo deve categorizar ou rotular os dados.

O sistema é capaz de aprender padrões com dados de entrada, mesmo não sendo fornecido um valor correspondente de saída. Tem como tarefa mais recorrente o agrupamento, ou seja, reconhecer grupos com os dados de entrada potencialmente úteis (RUSSELL e NORVIG, 2013).

**1.6. Fluxo de Funcionamento de um Sistema de AM**

Em geral, todos os sistemas baseados em AM obedecem a um fluxo de operações, divididos em 4 partes que são:

1. Pré-processamento dos dados: os dados, muitas vezes, podem conter valores incorretos, inconsistentes, duplicados, apresentar diferentes características, formatos e dimensões, o que atrapalha a interpretação destes pela máquina. O pré-processamento, visa melhorar qualidade destes dados minimizando ou eliminando os problemas citados (FACELI et. al, 2011). Nesta etapa também, o conjunto de dados é dividido, aleatoriamente, entre elementos de treino e teste, objetivando evitar que a máquina não tenha um bom desempenho somente com o conjunto de treino (RASCHKA e MIRJALILI, 2017).
2. Definição do modelo e aprendizagem: nesta etapa se define o tipo de modelo, ou seja, o algoritmo de AM, com base na tarefa a ser resolvida, e se efetua a aprendizagem da máquina, com o conjunto de dados de treino (RASCHKA e MIRJALILI, 2017).
3. Avaliação: após o treinamento, para verificar a performance do treino, o modelo recebe os dados de teste. Caso esta performance seja satisfatória, o modelo está apto para receber dados novos (RASCHKA e MIRJALILI, 2017).
4. Estimação: nesta última etapa o modelo consiste em obter os resultados de valores novos estimados pelo modelo (RASCHKA e MIRJALILI, 2017).

A figura 4, apresenta o fluxo descrito acima:

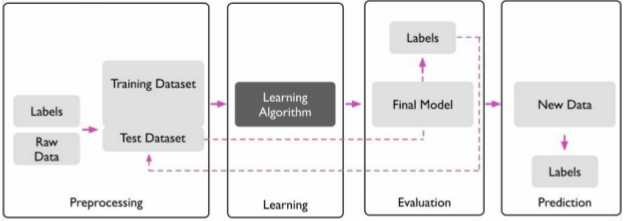


FIGURA 4: Fluxo de operações de um sistema de AM.

Fonte: RASCHKA e MIRJALILI (2017, p. 53 - adaptada)

**1.7. Sistema de Regressão**

Bishop (2006), afirma que modelos de regressão têm por objetivo estimar o valor de uma ou mais variáveis alvo contínuas, através de um vetor de entrada -dimensional.

Existem inúmeros algoritmos de AM, alicerçados em conceitos matemáticos, usados para o propósito de regressão, tais como Regressão Linear, Árvores de Decisão, Regressão Polinomial e Máquina de Vetores de Suporte (SVR em inglês), mas neste estudo serão analisados somente os algoritmos de Regressão Linear Simples e Múltipla, além de Árvore de Decisão e Floresta Aleatória, pois estes modelos foram utilizados para criar o Sistema de Regressão, do Sistema Agregador.

**1.7.1. Regressão Linear Simples e Múltipla**

Segundo Castro e Ferrari (2016), modelos de regressão linear simples são capazes de estabelecer a relação entre uma variável dependente e uma independente, por meio da determinação de uma equação de linha reta que seja capaz de representar esta relação.

A esta linha dá-se o nome de linha de regressão e a equação é denominada equação de regressão.

A equação de regressão é escrita por Lattin et. al (2011) como:

em que é chamado de intercepto e coeficiente de inclinação, parâmetros estes a serem estimados.

O coeficiente de inclinação representa a variação média de , para o aumento de uma unidade da variável e o intercepto o ponto do eixo das ordenadas cortado pela reta (BUSSAB e MORETTIN, 2010). A figura 5, apresenta estes parâmetros:

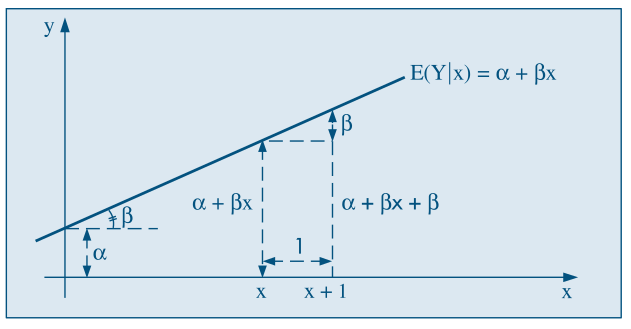


FIGURA 5: Representação intercepto e coeficiente de inclinação.

Fonte: BUSSAB e MORETTIN (2010, p. 450)

Pelo fato desta relação não ser perfeitamente exata, à fórmula adiciona-se um erro, que reflete um ruído, ou seja, a diferença entre o valor real da variável dependente e o valor estimado (LATTIN et. al 2011). Por fim a equação final é dada por:

O erro também é conhecido como offset e é representado por , onde corresponde ao valor real e o valor estimado (RASCHKA e MIRJALILI, 2016) . Conforme a figura 6, eles são representados pelas linhas verticais.

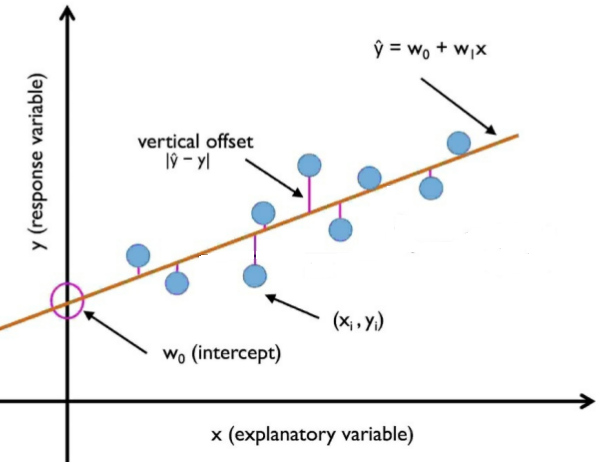


FIGURA 6: Representação do erro.

Fonte: RASCHKA e MIRJALILI (2017, p. 450 - adaptada)

Segundo Faceli et. al (2011), este erro, também conhecido como erro da hipótese ƒ, é medido, geralmente, através de duas métricas conhecidas como erro quadrático médio (MSE - *mean squared error*) e distância absoluta média (MAD - *mean absolute distance*), dadas pelas fórmulas:

Ambas as medidas são valores não negativos e quanto menores seus resultados, melhor o modelo de regressão (FACELI et. al, 2011).

A figura 7, apresenta a linha de regressão, dada pela relação entre a variável independente A e a dependente B:

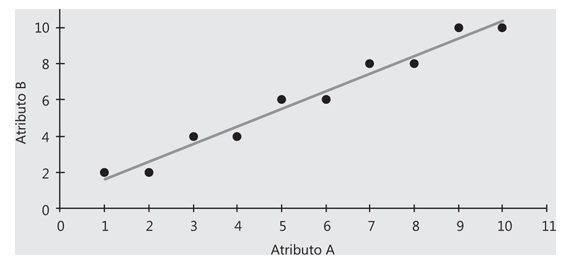


FIGURA 7: Linha de regressão da relação entre variáveis A e B.

Fonte: CASTRO e FERRARI (2016, p. 160)

A regressão linear múltipla se baseia nos mesmos conceitos, métricas e técnicas de avaliação da regressão linear simples. Ela é a generalização do modelo linear para mais de uma variável independente e pode ser definida pela fórmula (RASCHKA e MIRJALILI, 2017):

**1.7.2. Árvore de Decisão e Floresta Aleatória**

Alpaydin (2010), define uma árvore de decisão como uma estrutura hierárquica, composta por nós de decisão e folhas, conforme figura 8, baseada na estratégia de conquistar e dividir. Isto significa que um problema complexo é dividido em problemas mais simples e recursivamente se aplica a mesma estratégia, começando da raiz até ao ponto em que a folha é escolhida, ou seja, na folha a saída é definida por um valor.

Tem-se na parte mais elevada da árvore o nó chamado raiz e os caminhos deste nó raiz até a um nó folha corresponde a uma regra de classificação ou regressão (CASTRO e FERRARI, 2016).

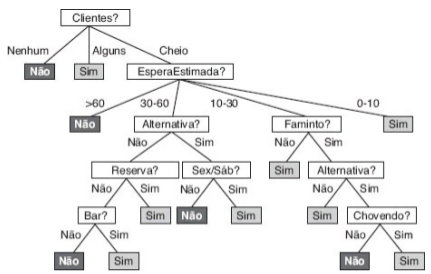


FIGURA 8: Árvore de decisão.

Fonte: RUSSELL e NORVIG (2013, p. 812)

Na figura 8, Russell e Norvig (2013) apresentam uma árvore que decide a espera ou não de uma mesa em um determinado restaurante e conta com os seguintes atributos:

1. Alternativa: Se existe um restaurante alternativo por perto.
2. Bar: Se existe área de bar confortável no restaurante para espera.
3. Sex/Sáb: Se funciona às sextas e sábados.
4. Faminto: Se cliente com fome.
5. Clientes: Quantidade de pessoas no restaurante.
6. Chovendo: Se chove na área externa do restaurante.
7. Reserva: Se cliente fez reserva.
8. EsperaEstimada: Tempo de espera estimada pelo gerente.

Segundo Faceli et. al, (2011), em cada nó de decisão existe um teste condicional referente aos valores de atributo e de acordo com o resultado um dos ramos é escolhido. Na maioria das vezes o teste efetua a comparação do valor de atributo a uma constante e após o resultado encaminha o objeto para um próximo nó de decisão ou um nó folha (CASTRO e FERRARI, 2016).

Ainda segundo Castro e Ferrari (2016), os nós folhas fornecem uma classificação ou valor, nos casos de regressão, a todo objeto que atinge a folha.

Em problemas de regressão a média dos valores contidos na folha é utilizada para determinar o valor do objeto e nos casos de classificação se utiliza a moda das classes contidas também na folha (FACELI et. al, 2011).

A figura 9, apresenta uma árvore de decisão e a divisão no espaço referente aos atributos e e cada nó corresponde a uma região neste espaço.

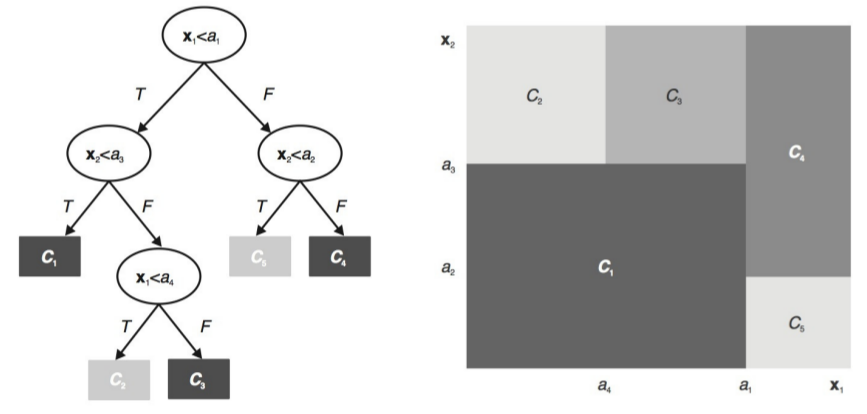


FIGURA 9: Árvore de decisão e divisões no espaço.

Fonte: FACELI et al. (2011, p. 84)

Ainda segundo Faceli et. al, (2011), pela árvore abranger todo espaço em instâncias ela se torna capaz de efetuar predições para qualquer exemplo de entrada e cada uma das divisões é estipulada, em geral, através da métrica SDR (*Standard Deviation Reduction*), através dos seguintes processos efetuados em um conjunto com amostras:

1. Calcula-se a variância da variável y, dependente, pela expressão:
2. Divide-se o conjunto em dois subconjuntos e com cardinalidades e .
3. Como a variância de será sempre igual ou menor à sua variância antes da divisão, estima-se a redução desta pela fórmula:
4. É calculado o valor da redução da variância para cada um dos atributos e em cada teste possível no valor do atributo.
5. O teste que possuir a maior redução na variância é escolhido como nó de decisão.

Já o cálculo do valor de predição é dado pela média dos elementos pertencentes ao espaço do qual a folha pertence, aplicando a fórmula (RASCHKA e MIRJALILI, 2016):

onde é a quantidade de elementos do conjunto e os valores contidos no conjunto.

Uma floresta aleatória é o conjunto de árvores de decisão, sendo definido pelo usuário, tendo o valor de predição calculado à partir da média de todos os valores preditos por cada árvore (RASCHKA e MIRJALILI, 2017).

**1.8. Sistema de Recomendação**

Ricci et. al (2015), definem um sistema de recomendação como um conjunto de ferramentas e técnicas capaz de fornecer sugestões de itens, os quais sejam o mais parecido possível ao interesse de um usuário em particular, seja qual música ouvir, qual produto comprar, dentre outros. Item nesse contexto é um termo genérico que se refere ao produto, serviço, ou outros, fornecido pelo sistema.

Seu princípio básico consiste em observar as dependências significativas entre o item e o usuário, ou seja, se um usuário se interessa por um documentário, muito provavelmente este se interessaria mais por outro documentário ou um programa educacional do que por um filme de ação (AGGARWAL, 2016).

**1.9. Tipos de Sistema de Recomendação**

Um sistema de recomendação possui vários tipos de abordagem, e neste estudo serão detalhados os funcionamentos dos tipos filtragem colaborativa e baseada em conteúdo.

**1.9.1. Filtragem Colaborativa**

Composta por duas técnicas, que são a baseada em usuário e em item, a filtragem colaborativa visa efetuar as recomendações de itens para determinados usuários fundamentadas no histórico de usuários que compartilhem gostos similares (GORAKALA , 2016).

Na filtragem colaborativa baseada em usuário, após a avaliação de determinados itens por um usuário A, a ideia básica é localizar as avaliações semelhantes aos mesmos itens avaliados por A, por um grupo de usuários B e recomendar para A itens desconhecidos que B avaliou (AGGARWAL, 2016).

Segundo Segaran (2007), na filtragem colaborativa baseada em item, primeiro, são observadas as similaridades entre todos os itens. A recomendação ocorre com os itens que, após consideradas as avaliações de um determinado usuário, apresentam maior similaridade entre si.

**1.9.2. Recomendação Baseada em Conteúdo**

Gorakala (2016), estabelece que nesta abordagem o conteúdo dos itens é utilizado para efetuar a recomendação. A similaridade entre eles é calculada através da característica associada a cada item por meio de comparação entre todos os itens.

Em linhas gerais, neste tipo de recomendação os itens tem suas características comparadas e se recomenda ao usuário os que forem mais similares ao item que este já avaliou.

Neste estudo esta foi a abordagem utilizada para criação do Sistema de Recomendação e Busca.

**1.10. Técnicas de Recomendação**

Existem inúmeras técnicas utilizadas para criação de um sistema de recomendação, tais como a técnica baseada em modelo, baseada em restrições e baseada em vizinhança. Esta última foi a técnica escolhida para criação do sistema de recomendação e busca, portanto será abordada e suas características detalhadas.

**1.10.1. Recomendação Baseada em Vizinhança**

Esta técnica é uma das mais antigas utilizadas e se apoia no princípio da similaridade entre pares de elementos, sejam eles, preferências e avaliações de usuários, itens, dentre outros (RICCI et al., 2015). Os pares de itens ou avaliações, preferências, ou qualquer outra fonte de dados, conforme Gorakala (2016), são considerados vetores e um cálculo de dimissilaridade é aplicado a cada um deles para determinar o quão próximos estão.

Para o cálculo da similaridade é comum o uso das distâncias euclidiana, Minkowski, similaridade do cosseno, de Jaccard, coeficiente de correlação Pearson, e outras métricas. Sendo que neste estudo será explorado os aspectos das distâncias euclidiana e similaridade do cosseno, por terem sido estas utilizadas.

**1.11. Distância Euclidiana**

A distância euclidiana calcula a distância entre dois pontos, no espaço denominado, espaço de entrada (FACELI et. al, 2011). É definida pela seguinte equação:

É uma das medidas mais populares para cálculo de distância e conforme a figura 10, pode-se ver a distância calculada entre dois pontos a e b:

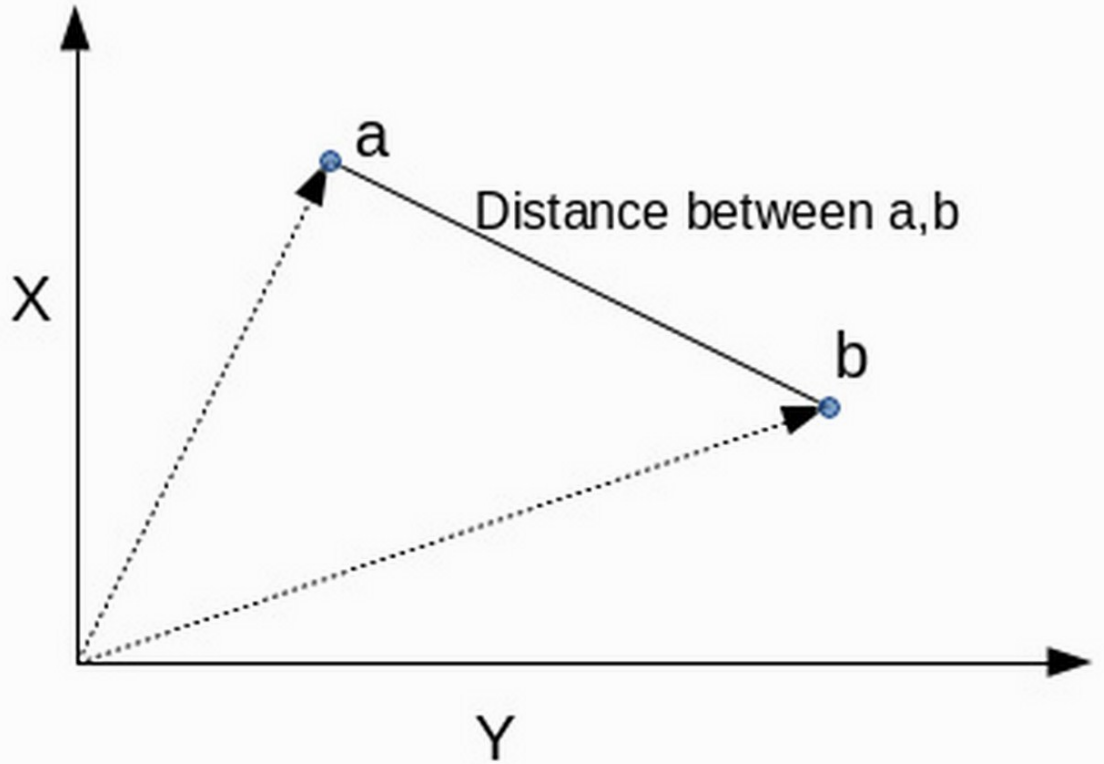


FIGURA 10: Distância calculada entre vetores a e b.

Fonte: GORAKALA (2016, p. 70)

**1.12. Similaridade do Cosseno**

Segundo Gorakala (2016), a similaridade do cosseno, conhecida também como medida de separação angular, efetua o cálculo da similaridade entre os pontos a e b através do ângulo existente entre eles. É dada pela fórmula:

A figura 11, apresenta o cálculo da distância entre os dois vetores a e b:

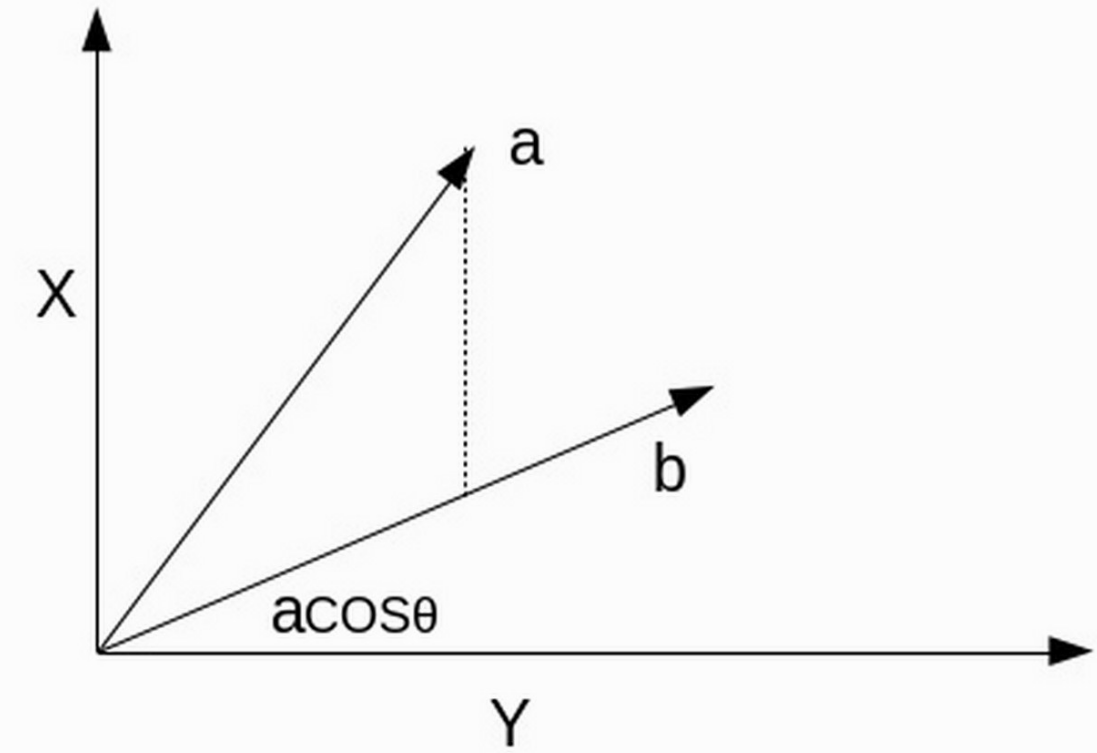


FIGURA 11: Similaridade do cosseno

entre os pontos a e b.

Fonte: GORAKALA (2016, p. 71)

Caso a distância entre os dois vetores seja 90º, o resultado de todos os pontos entre eles será 0, pois , significando que os elementos estão muito distantes entre si. O contrário ocorre quando o ângulo entre os elementos é reduzido, ou seja, com a redução deste ângulo eles se tornam mais similares (GORAKALA, 2016). As figuras 12 e 13, a seguir, exemplificam as situações descritas:

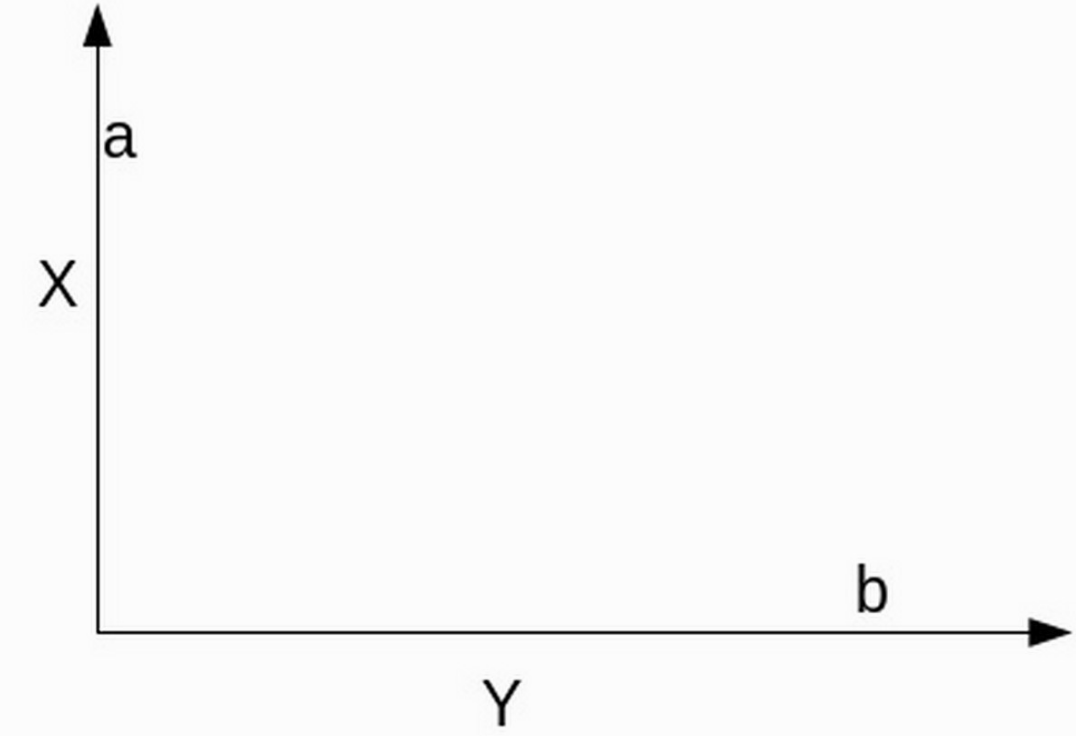


FIGURA 12: Vetores a e b distantes 90º.

Fonte: GORAKALA (2016, p. 71)

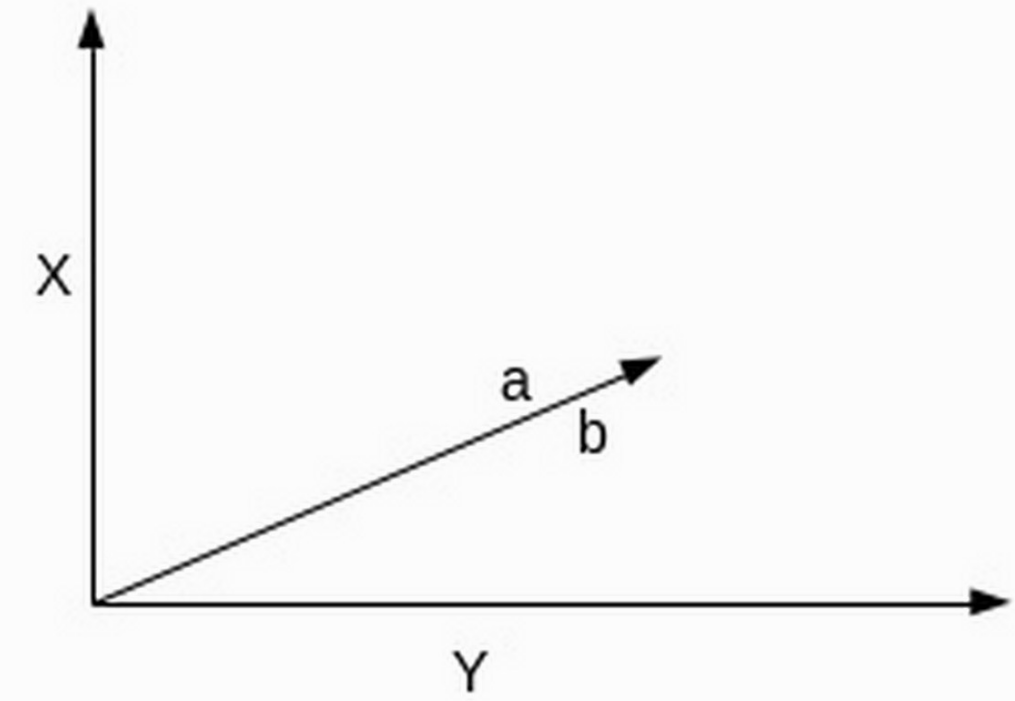


FIGURA 13: Vetores a e b com ângulo a 0º.

Fonte: GORAKALA (2016, p. 72)

A aplicação de algoritmos de AM para sistemas de recomendação, são em sua maioria baseados em conceitos de agrupamento e este estudo fez uso do algoritmo denominado k-vizinhos mais próximos (*K-Nearest Neighbors*), o qual será detalhado.

**1.13. K-Vizinhos Mais Próximos**

Este algoritmo consiste em observar k pontos vizinhos mais próximos em relação a um ponto x. Portanto para implementação é importante a medida de distância entre pontos pares de observação (SMOLA e VISHWANATHAN, 2010). É também conhecido como algoritmo de aprendizagem preguiçosa (*lazy learning algorithm*), pois conforme Raschka e Mirjalili (2016), ele memoriza os dados do conjunto de treinamento ao invés de aprender uma função discriminante que os defina. Por esse motivo é categorizado como modelo não paramétrico, com abordagem baseada em exemplos ou memória (RUSSELL e NORVIG, 2013).

O funcionamento deste algoritmo pode ser definido pelas seguintes etapas:

1. Define-se a quantidade de k vizinhos, sendo que, Faceli et al. (2011) estipulam um número pequeno e ímpar para evitar empates.
2. Para cada ponto do conjunto de dados, calcula-se a dissimilaridade ao k-ésimo vizinho mais próximo (LATTIN et al, 2011).
3. Identifica-se os k vizinhos mais próximos, através do resultado da dissimilaridade, ou seja, quanto menor a dissimilaridade mais próximos os elementos estão (GORAKALA, 2016).

A seguir, a figura 14 apresenta o uso do algoritmo em problemas de classificação, onde a classe escolhida é a que possui mais representantes:

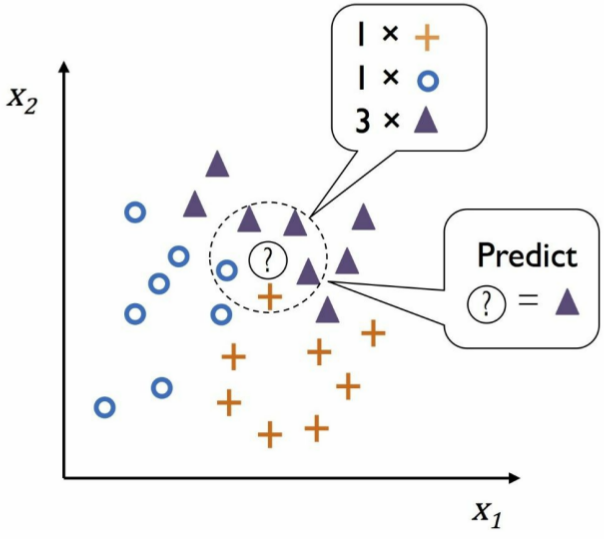


FIGURA 14: K-vizinhos mais próximos em

classificação.

Fonte: RASCHKA e MIRJALILI (2017, p. 174)

Segundo Russell e Norvig (2013), para problemas de regressão pode se tirar a média ou mediana dos k-vizinhos.

**CAPÍTULO 2 MATERIAIS E MÉTODOS**

Neste capítulo serão discutidos e detalhados as tecnologias e métodos para criação do Sistema Agregador.

A primeira parte deste estudo consistiu em analisar e escolher as ferramentas mais apropriadas para elaboração do sistema. Após feita a análise, definiu-se pela codificação em linguagem de programação Python, banco de dados Mysql e plataforma de desenvolvimento VSCode.

Este estudo possuiu aspectos quantitativo e qualitativo, pois fará uso tanto de dados referentes às características dos apartamentos, que foram adquiridos de forma automática, quanto de dados adquiridos por meio de pesquisa.

Foi uma pesquisa aplicada, pois segundo Gerhardt e Silveira (2009), este tipo de pesquisa gera conhecimentos para aplicação prática, o que foi justamente a proposta deste estudo, aplicar as técnicas estudadas para análise de dados.

Além de aplicada é uma pesquisa explicativa, pois tem-se o interesse de explicar os resultados através dos procedimentos efetuados.

Metodologicamente o estudo obedeceu a seguinte sequência:

1. Escolha da linguagem de programação e bibliotecas.
2. Criação do rastreador web e obtenção dos dados.
3. Armazenamento dos dados obtidos em banco de dados.
4. Criação dos modelos de aprendizado de máquina, tanto para regressão, como para busca e recomendação.
5. Análise exploratória dos dados.

A seguir são detalhados todos os materiais utilizados.

**2.1. Ambiente de Desenvolvimento**

O ambiente de desenvolvimento escolhido para este estudo foi o Visual Studio Code*,* mais conhecido como VSCode da empresa Microsoft, pois possui uma grande comunidade de usuários na internet, além de várias ferramentas que aumentam a produtividade do programador, como atalhos para identação, comentários, execução do código e versionamento.

**2.2. Linguagem Python**

A linguagem *Python* foi escolhida, pela sua versatilidade e grande quantidade de bibliotecas para uso científico, principalmente para manipulação de dados e aplicações de aprendizado de máquina.

É através desta linguagem que os dados adquiridos serão manipulados, analisados e ao final, apresentados os resultados.

A linguagem *Python* foi desenvolvida, a partir da linguagem ABC, em 1990 por Guido Van Rossum no Instituto Nacional de Pesquisa para Matemática e Ciência da Computação da Holanda. Foi criada como o objetivo inicial de suprir as necessidades de usuários como físicos e engenheiros (BORGES, 2010).  
 Segundo Cruz (2015), *Python* é uma linguagem de alto nível interpretada. Possui tipagem forte e dinâmica, gerenciamento de memória automático e escopo léxico. Ela também é capaz de suportar inúmeros paradigmas de programação, tais como imperativo, orientado a objetos e funcional.  
 É uma linguagem de código aberto, (licença *General Public License* – GLP), e por ter menos restrições, pode ser usada em produtos proprietários. Sua especificação e controle de versões é mantida pela *Python Software Foundation* – PSF (BORGES, 2010). O processo de interpretação é exibido na figura 15:

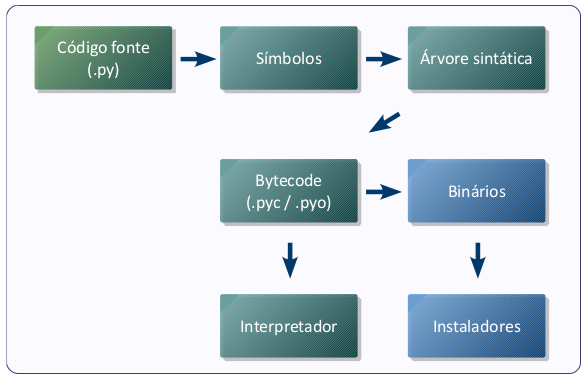


FIGURA 15: Diagrama do interpretador *Python*.

Fonte: BORGES (2010, p. 16)

Borges (2010), descreve que ao ser interpretado pelo *Python*, o código fonte é traduzido para *bytecode*, que é um formato binário com as instruções para o interpretador. Para que não necessite traduzir o código novamente, o interpretador armazena o *bytecode* em disco, fazendo com que o tempo de carga na execução seja menor.

Este *bytecode* é armazenado com as extensões .pyc (normal) ou .pyo (otimizado).

**2.3.** **Bibliotecas**

Ao longo de todo o projeto foram utilizadas bibliotecas, pertencentes à linguagem de programação escolhida, e que auxiliaram no desenvolvimento de tarefas específicas.

Ao todo foram utilizadas nove bibliotecas, que são:

1. Requests: responsável por acessar e resgatar o conteúdo HTML das páginas.
2. BeautifulSoup: buscar os dados desejados, no conteúdo HTML das páginas.
3. Googlemaps Geocoding: retornar os dados sobre as coordenadas geográficas, latitude e longitude, com base no nome dos bairros.
4. Mysql.connector: responsável por criar uma conexão com o banco de dados.
5. Scikit-learn: criar os modelos regressores e métricas de eficiência.
6. Numpy: responsável pelas funções matemáticas e manipulação de matrizes.
7. Pandas: manipular os dados do conjunto de dados.
8. Matplotlib e Bokeh: responsáveis pela geração de gráficos.

Conforme descrito na introdução do capítulo, as bibliotecas Requests, BeautifulSoup e GoogleMaps foram utilizadas para elaboração do rastreador web e obtenção dos dados.

**2.3.1 Requests**

A Requests é uma biblioteca que está sob a licença Apache2, portanto é de código aberto e pode ser utilizada em códigos proprietários.

Ela é capaz de realizar requisições em sites http com ou sem requisição, além de ser disponibilizada como biblioteca *built-in*, ou seja, ao instalar a linguagem Python ela já faz parte do sistema, sendo desnecessária a instalação de qualquer outra ferramenta.

**2.3.2. BeautifulSoup**

Esta biblioteca é capaz de retirar, de arquivos HTML e XML, dados de acordo com o desejo do usuário.

Possui uma gama de métodos, todos voltados para este objetivo, além de converter os dados automaticamente para o padrão UTF-8.

Atua sob licença MIT e é de código aberto.

**2.3.3. Googlemaps Geocoding**

Biblioteca sob licença Apache2, que possibilita resgatar informações de geocodificação, processo capaz de converter endereços em coordenadas geográficas e geocodificação reversa, neste caso quando fornecidos as coordenadas geográficas retorna o endereço.

**2.3.4. Mysql.connector**

Biblioteca capaz de criar um *driver* de comunicação entre o algoritmo em linguagem Python e um servidor MySQL.

Com esta ferramenta foi possível armazenar os dados obtidos em um servidor de banco de dados, concluindo-se assim a terceira parte do projeto.

**2.3.5. Scikit-learn**

Scikit-learn é uma biblioteca para Python, que integra uma grande variedade de algoritmos para aprendizagem de máquina supervisionados ou não supervisionados (PEDREGOSA, et al., 2011).

Esta biblioteca começou a ser desenvolvida em 2007 por David Cournapeau como um projeto da Google Summer of Code e possui vários algoritmos de classificação, regressão, agrupamento (*clustering*), *gradient boosting* e outras técnicas. Atua sobre a licença *Berkeley Software Distribution* – BSD, portanto é de código aberto e permite que seja usada comercialmente.

**2.3.6.** **Numpy**

A entidade Numpy.org (2018), define esta biblioteca como sendo matemática possibilitando a linguagem Python a trabalhar com arranjos, vetores e matrizes de N dimensões. Possui vários recursos, incluindo:

1. Implementação de arranjos multidimensionais.
2. Ferramentas para cálculos de matrizes.
3. Ferramentas para álgebra linear.
4. Transformadas básicas de Fourier.
5. Ferramentas de geração de números aleatórios sofisticadas.

Além da aplicação científica, pode ser usada para abrigar dados genéricos multidimensionais, permitindo a integração com inúmeros tipos de conjuntos de dados (NUMPY.ORG, 2018).

Atua sob a licença *Berkeley Software Distribution* – BSD.

Com o uso das bibliotecas Scikit-learn e Numpy, concluiu-se a etapa quatro do projeto, conforme descrito na introdução.

As próximas bibliotecas listadas foram utilizadas para elaboração das etapas três e quatro do trabalho, no que diz respeito a manipulação do conjunto de dados e criação de gráficos.

**2.3.7. Pandas**

Pandas é uma biblioteca, *open-source* licença BSD, que permite à Python uma alta performance e facilidades na operação e manipulação de dados estruturados. Segundo Pandas (2018), entre seus recursos, podem-se destacar:

1. eficiente objeto *DataFrame* para manipulação de dados.
2. ferramentas para leitura e gravação de dados para diferentes formatos.
3. remodelagem e rotacionamento de conjunto de dados.
4. alta performance em mesclar e unir conjuntos de dados.
5. colunas podem ser inseridas e deletadas das estruturas de dados.
6. manuseio de dados faltantes e alinhamento de dados.
7. manipulação de dados confusos de forma ordenada.

**2.3.8. Matplotlib**

A Matplotlib foi desenvolvida por Jhon Hunter à partir de 2003, com atuação sob a licença BSD. Alguns exemplos de gráficos podem ser vistos na figura 16.

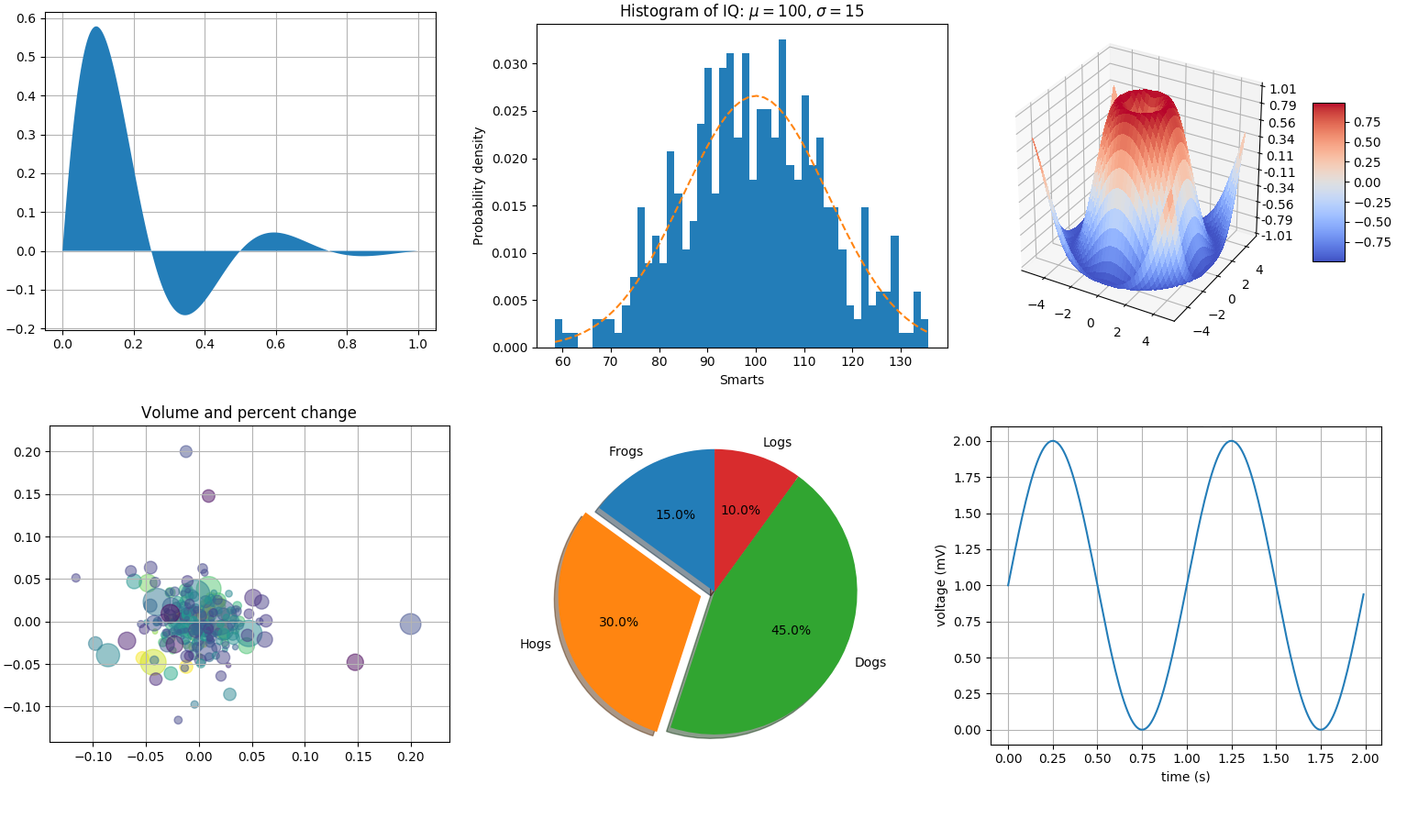


FIGURA 16: Exemplos de gráficos produzidos com a biblioteca Matplotlib.

Fonte: Matplotlib (https://matplotlib.org/users/screenshots.html -adaptada)

É uma biblioteca de produção de gráficos em 2D para Python, com capacidade de gerar, além de gráficos de linhas, histogramas, espectros de potência, gráficos de barra, erros e dispersão (MATPLOTLIB.ORG, 2018).

Para trabalhos em 3D existe a possibilidade de instalação da kit de ferramenta mplot3d.

**2.3.9. Bokeh**

Assim como a Matplotlib, a biblioteca Bokeh, possui a finalidade de criação de gráficos. A diferença é sua capacidade em gerar gráficos interativos sendo criados como objetos HTML, ou seja, são criados com a possibilidade de serem usados em navegadores web (BOKEH, 2018).

A figura 17, apresenta alguns exemplos:

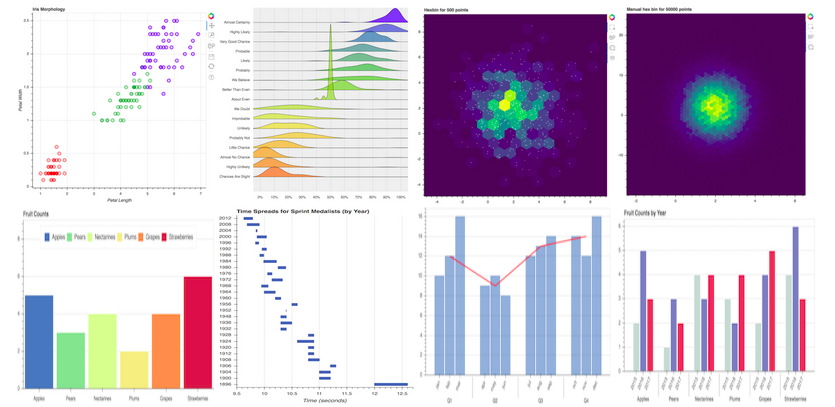


FIGURA 17: Exemplos de gráficos criados com a biblioteca Bokeh.

Fonte: Bokeh (https://bokeh.pydata.org/en/latest/docs/gallery.html - adaptada)

É também um projeto *open-source* e atua sob licença BSD.

**2.4. Questionário**

O questionário aplicado neste estudo, foi desenvolvido através da plataforma da empresa Google, chamada Google Forms, que foi escolhida pela facilidade de criação, envio e resgate das respostas. Foi disponibilizado durante os dias 20/04/2018 a 23/04/2018, obtendo 111 respostas.

As respostas foram coletadas e armazenadas em uma tabela através da própria plataforma e os usuários entrevistados foram escolhidos aleatoriamente mas todos com idade acima de dezoito anos.

Este questionário foi utilizado na quarta etapa deste trabalho, no que diz respeito a elaboração de um sistema de recomendação e busca.

**CAPÍTULO 3 RESULTADOS: APRESENTAÇÃO, ANÁLISE E DISCUSSÃO**

Neste capítulo serão apresentadas as etapas de construção dos algoritmos, que juntos constituem o sistema agregador. Este sistema agregador objetiva, através dos dados de imóveis coletados em cinco sites de imobiliárias da cidade de Montes Claros, fornecer estimativas de preços, gráficos de análises, recomendação e busca, com base na característica de cada imóvel.

É composto por quatro algoritmos principais, sendo eles:

1. Rastreador web.
2. Sistema de regressão.
3. Sistema de recomendação e busca.
4. Análise exploratória do conjunto de dados.

Através da figura 18, é apresentada o fluxograma do funcionamento de cada algoritmo:

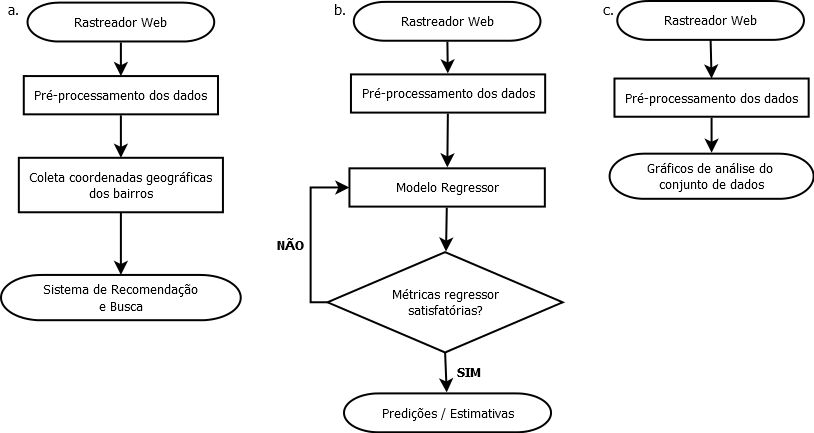


FIGURA 18: Fluxogramas das aplicações. a) Fluxograma do sistema de recomendação e busca. b) Fluxograma sistema de regressão. c) Fluxograma de análise exploratória do conjunto de dados.

Fonte: O Autor

**3.1. Rastreador Web**

O rastreador web é a parte do sistema responsável por obter e salvar dados disponibilizados pelos sites das empresas imobiliárias.

Foram escolhidos cinco sites, os quais possuíam a maior quantidade de imóveis cadastrados, para atuação do rastreador, o que gerou cinco *scripts* em linguagem Python, utilizando as bibliotecas BeautifulSoup e Requests, para a aplicação.

A lógica para o *script* de elaboração do sistema rastreador é apresentada na figura 19:

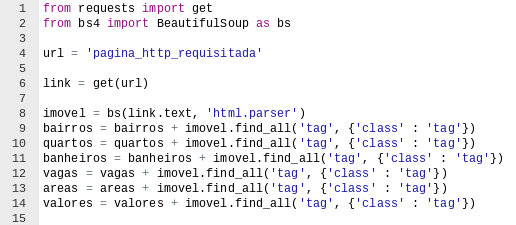


FIGURA 19: Script rastreador web.

Fonte: O Autor

Primeiro são adicionadas as bibliotecas, sendo que a requests, através do método *get*, é responsável por acessar a página web, onde seu conteúdo é armazenado na variável intitulada ‘link’.

À variável ‘imovel’ é atribuída um objeto da classe BeautifulSoup, renomeada como bs para melhor legibilidade do código.

Para criação do objeto são passados como parâmetros o conteúdo textual da variável ‘link’ e o tipo de parser, ou seja, o tipo de estrutura de arquivo que o parser deve atuar, neste caso um arquivo do tipo HTML.

De acordo com o dicionário Collins (2018), um parser é um programa capaz de interpretar os dados de entrada, reconhecendo palavras chave e analisando sua estrutura.

Da classe denominada BeautifulSoup é utilizado também o método ‘find\_all’, responsável por buscar as informações das *tags* definidas pelo usuário de acordo com a forma com que são disponibilizadas no *site*.

Neste estudo, as *tags* escolhidas para rastreamento foram as que apresentavam as características dos apartamentos, como bairro, quantidade de quartos, banheiros e vagas de garagem, além da área e o valor de venda.

A figura 20, abaixo, exemplifica como os marcadores são disponibilizados e encontrados quando efetua-se uma exploração do código HTML da página:



FIGURA 20: Exemplo de busca de *tags* para rastreador web

Fonte: O Autor

Com estes dados em mãos, a busca dos dados se dará de acordo com a figura 21:

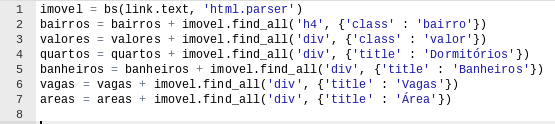


FIGURA 21: Exemplo de definição das tags para busca dos dados.

Fonte: O Autor

Ao final todos os dados obtidos foram armazenados em um banco de dados no software MySQL, através da biblioteca mysql.connector, que como o nome mostra, criar uma conexão com o servidor de banco de dados.

O código para armazenamento é exemplificado na figura 22:

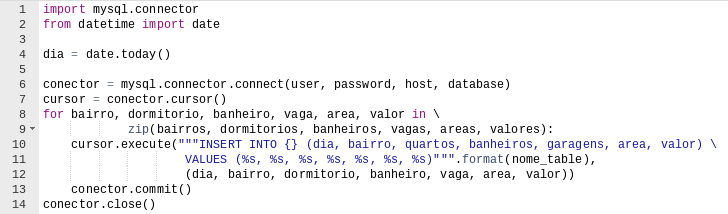


FIGURA 22: Código para armazenamento dos dados.

Fonte: O Autor

Importam-se as bibliotecas para conexão no servidor MySQL e para obtenção da data corrente, através da biblioteca datetime.

À variável ‘conector’ atribui-se um objeto mysql.connector, que recebe como parâmetros, o nome do usuário, senha, endereço do servidor e nome do banco de dados.

Após criada a conexão, a variável ‘cursor’ recebe um objeto do tipo conector.cursor, responsável por apontar para uma determinada linha na tabela SQL e executar uma ação.

Com todos os elementos obtidos na página é feita uma iteração, conforme linhas 8 e 9, executando uma solicitação de inserção de dados, conforme linhas 10 e 11.

A cada inserção é efetuado um comando ‘commit’, linha 13, para conclusão deste processo na tabela.

Após todos os elementos serem inseridos na tabela, a conexão com o banco de dados é fechada conforme comando definido na linha 14. As tabelas criadas no banco de dados possuem a estrutura apresentada na figura 23:

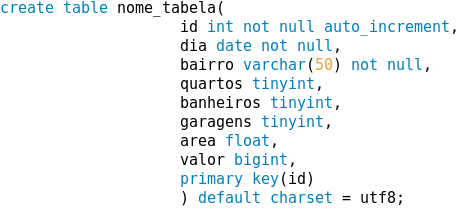


FIGURA 23: Estrutura de criação da tabela no servidor MySQL.

Fonte: O Autor

Finalizadas as inserções de dados, gerou-se cinco tabelas com o seguinte padrão:

TABELA 1

Características dos imóveis

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | DIA | BAIRRO | QUARTOS | BANHEIROS | GARAGENS | AREA | VALOR |
| 1 | 2018-04-02 | Augusta Mota | 3 | 1 | 2 | 90 | 320000 |
| 2 | 2018-04-02 | Augusta Mota | 2 | 1 | 1 | 85 | 300000 |
| 3 | 2018-04-02 | Ibituruna | 3 | 2 | 2 | 0 | 287500 |
| 4 | 2018-04-02 | Canelas | 2 | 1 | 1 | 62.5 | 198000 |
| 5 | 2018-04-02 | São José | 4 | 3 | 1 | 0 | 285000 |

Fonte: O Autor

Ao todo neste estudo foram obtidos dados relativos a 670 imóveis.

Estas tabelas foram utilizadas para o sistema regressor, análise exploratória e sistemas de recomendação e busca.

A formatação dos dados conforme mostrado na tabela 1, foi imprescindível para a criação e funcionamento do sistema agregador, pois através dela e da forma como os dados foram dispostos, a máquina foi capaz de interpretar e processar as informações.

**3.2. Regressor**

O regressor foi concebido objetivando a predição dos preços dos imóveis com base em suas características. Através dele será possível ao usuário ter uma ideia do valor de um determinado imóvel em um bairro específico, sem a necessidade de consultar várias imobiliárias.

Para que funcionasse como esperado foi necessário, primeiramente, efetuar o pré-processamento dos dados que serão utilizados.

Este pré-processamento foi dividido em 3 partes:

1. Ajuste dos nomes dos bairros.
2. Verificação da quantidade de valores nulos por coluna e exclusão das colunas que possuíam mais de 80% de valores nulos.
3. Imputação de valores aos valores nulos, utilizando o critério de média.

Os nomes dos bairros cadastrados nos sites das imobiliárias, em sua grande maioria, possuem diferenças, seja de acentuação, seja no uso de letras maiúsculas. Portanto na primeira parte do pré-processamento, eliminou-se a acentuação e os nomes dos bairros passou a possuir letra maiúscula somente na inicial de cada nome inclusive dos elementos conectores, tais como ‘de’, ‘o’, ‘os’.

Abaixo a figura 24 apresenta o código desenvolvido para tal procedimento:

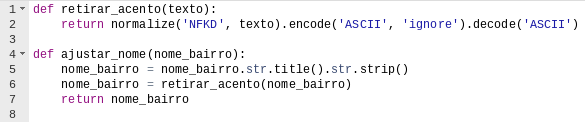


FIGURA 24: Códigos para ajuste dos nomes dos bairros.

Fonte: O Autor

O próximo passo foi verificar a quantidade de valores nulos por coluna e optou-se por não eliminar colunas com menos de 80% de valores nulos. A figura 25 apresenta a porcentagem encontrada em cada uma delas:

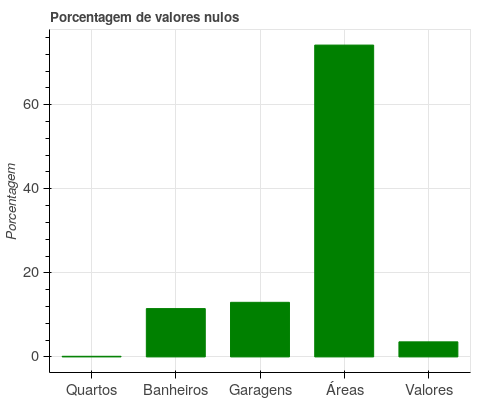


FIGURA 25: Porcentagem de valores nulos por coluna.

Fonte: O Autor

Neste caso nenhuma das colunas atingiu 80% de valores nulos, portanto não ocorreram exclusões.

O último passo do pré-processamento, consistiu em imputar valores, aos que se encontravam nulos no conjunto de dados. Este procedimento foi feito atribuindo a média de um mesmo conjunto dos valores ausentes, com exceção da coluna ‘garagens’, pois é possível que um apartamento não possua vaga.

A figura abaixo apresenta o código para tal procedimento:

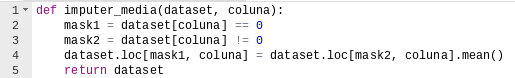


FIGURA 26: Código de imputação de valores da média.

Fonte: O Autor

Ao final, os dados ficaram disponíveis para manipulação conforme a tabela seguinte demonstra:

TABELA 2

Conjunto de dados após pré-processamento

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | DIA | BAIRRO | QUARTOS | BANHEIROS | GARAGENS | AREA | VALOR |
| 1 | 2018-02-04 | Planalto | 2 | 1 | 1 | 48 | 333075.79 |
| 2 | 2018-02-04 | Todos Os Santos | 2 | 2 | 0 | 51.215 | 333075.79 |
| 3 | 2018-02-04 | Cidade Santa Maria | 3 | 2 | 2 | 120 | 480000 |
| 4 | 2018-02-04 | Sao Jose | 4 | 3 | 1 | 90.752 | 285000 |
| 5 | 2018-02-04 | Centro | 4 | 5 | 2 | 149.838 | 750000 |
| 6 | 2018-02-04 | Morada Do Sol | 3 | 1 | 2 | 155.432 | 370000 |

Fonte: O Autor

Finalizado o pré-processamento a próxima etapa foi a escolha do modelo de regressor.

Para definição deste modelo foi necessário verificar, por meio de um diagrama de dispersão, a distribuição dos valores das variáveis no hiperplano, além do índice de correlação entre as variáveis independentes e a dependente, ou seja, a relação entre as variáveis ‘quartos’, ‘banheiros’, ‘garagens’ e ‘áreas’ com a variável ‘valor’.

Na figura 27 são apresentados os diagramas dos valores das variáveis independentes com relação a variável dependente:

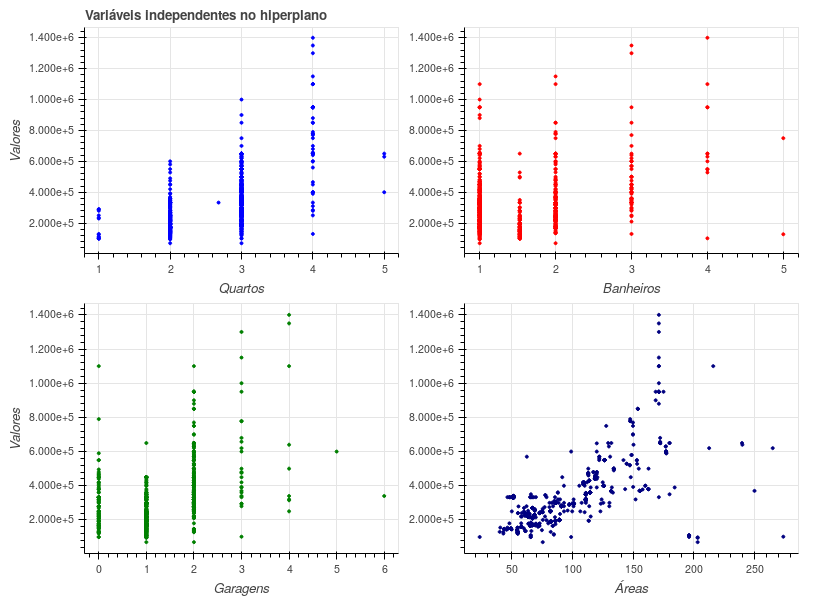


FIGURA 27: Variáveis independentes no hiperplano.

Fonte: O Autor

Através da figura 27, observa-se que este conjunto de dados possui uma baixa correlação linear com a variável dependente ‘valor’, pois conforme Morettin e Bussab (2010) a correlação linear é mais forte quando os dados das variáveis envolvidas apresentam-se próximos a uma reta. A matriz de correlação, conforme figura 28 evidencia o baixo índice de correlação linear:

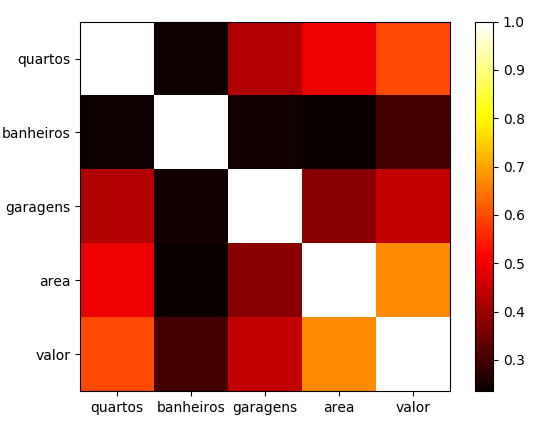


FIGURA 28: Matriz de correlação entre as variáveis.

Fonte: O Autor

Conforme observado, a variável independente ‘area’ é a que possui o maior índice de correlação, algo em torno de 0,7, indicando que esta abordagem não seria a mais ideal. O coeficiente de correlação, segundo Bussab e Morettin, 2010, é uma medida de relação linear entre duas variáveis e varia entre e .

Seja a correlação entre e dada por . Ela é dita perfeita se , isto ocorre pois , portanto se e se (BUSSAB e MORETTIN, 2010).

Sendo assim, os valores de correlação das variáveis se mostraram baixos para se estabelecer uma associação linear consistente.

Outras métricas para avaliar o modelo de regressão linear, foram as listadas abaixo:

1. Média do resultado da validação cruzada.
2. Desvio-padrão da validação cruzada.
3. Média do erro absoluto.
4. Valor de coeficiente de determinação (R²).

A validação cruzada divide, aleatoriamente, os dados utilizados para treino, em partes iguais, definidas pelo usuário. Com isto a cada iteração, uma destas partes é utilizada para teste enquanto as outras para treino. Este processo ocorre até que todas as partes sejam treinadas e testadas. Ao final se obtém a média geral de acurácia de todos os treinos e testes além do desvio-padrão médio, ou seja, a média total de quanto cada resultado de teste e treino está distante da média geral (MUELLER e MASSARON, 2016). Idealmente o valor de média da acurácia deve ser próximo de 1 e o desvio-padrão o menor possível. A figura seguinte, demonstra o funcionamento deste processo:

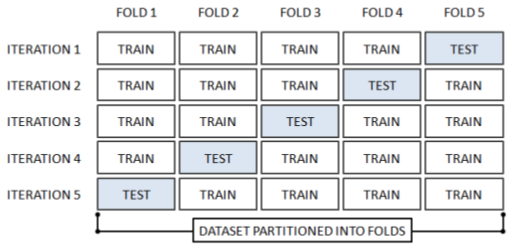


FIGURA 29: Processo de validação cruzada.

Fonte: Mueller e Massaron (2016, p.192)

Conforme descrito, o valor da média de acurácia da validação cruzada, foi utilizado para verificar a efetividade do modelo.

O erro absoluto é o valor real de subtraído pelo valor estimado de , ou seja, (REAMAT, 2018). Portanto o ideal é que este valor seja o menor possível.

O valor de R² fornece uma medida absoluta de ajuste do modelo aos dados fornecidos, ou seja, o quanto que uma variável dependente pode ser explicada pela variável independente e o ideal é que este valor seja próximo de 1 (JAMES et.al, 2017).

Para cálculo das métricas o conjunto de dados necessitou ser dividido entre treinamento e teste de acordo com a figura a seguir:

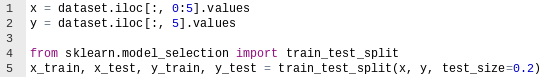


FIGURA 30: Separação do conjunto de dados entre treino e teste.

Fonte: O Autor

De acordo com o apresentado na figura, as linhas 1 e 2 são responsáveis por atribuir as variáveis ‘x’ e ‘y’ os valores contidos no *dataset*, em forma de um *array*.

Importa-se da biblioteca o método responsável, linha 4. Por fim atribui-se às variáveis ‘x\_train’, ‘x\_test’, ‘y\_train’, ‘y\_test’ o resultado da separação, que neste

caso foi configurado, pelo parâmetro test\_size, que o conjunto de teste teria 20% do tamanho total deste conjunto.

A figura seguinte apresenta o método para cálculo e exibição das métricas:

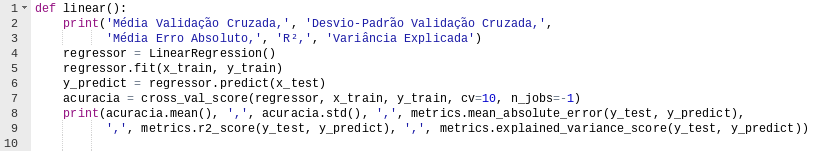


FIGURA 31: Método de avaliação do modelo linear.

Fonte: O Autor

Nas linhas 4 e 5 cria-se um objeto regressor linear e efetua-se o ajuste do conjunto de dados para treinamento. A linha 6 executa o método de predição com os conjunto de teste e na linha 7 a variável ‘acuracia’, recebe os resultados executados pela função de validação cruzada com os conjunto de treinamento, sendo que são efetuados 10 testes, setados pelo parâmetro ‘cv’.

Nas linhas 8 e 9 solicita-se a impressão do resultado dos testes.

Na tabela seguinte, são apresentados os resultados dos cálculos das métricas do modelo linear.

TABELA 3

Resultado do cálculo das métricas do modelo linear

|  |  |  |  |
| --- | --- | --- | --- |
| Média Validação Cruzada | Desvio-Padrão da Validação Cruzada | Média do Erro Absoluto (R$) | R² |
| 0,6389 | 0,0916 | 67.268,09 | 0,4527 |

Fonte: O Autor

Conforme visto na tabela 3, o valores das métricas da validação cruzada e do R² se mantiveram baixas, fazendo com que outra técnica para o sistema regressor fosse escolhida.

Após análises de alguns modelos escolheu-se a técnica de árvore de decisão, pois uma de suas vantagens é poder ser utilizada em sistemas não lineares (RASCHKA e MIRJALILI, 2017).

A construção da árvore de decisão foi efetuada em quatro etapas:

1. Pré-processamento dos dados.
2. Separação do conjunto de dados entre treinamento e teste.
3. Construção do modelo.
4. Definição da profundidade, através dos resultados apresentados após cada treinamento dos dados de treinamento e predição com o conjunto de teste.

O pré-processamento dos dados e divisão do conjunto de dados entre treinamento e teste, foi feita da mesma forma que para o regressor linear.

A construção do modelo, etapa três, se deu conforme apresentado na figura 32:



FIGURA 32: Construção do modelo.

Fonte: O Autor

A última etapa consistiu em definir a profundidade do nível da árvore de decisão e para isto foi gerado um método que alterava o valor de profundidade, treinava o modelo, com o conjunto de treinamento, e efetuava a predição com os dados de teste.

Após cada treinamento e teste o resultado dos indicadores da efetividade do modelo eram impressos. Os indicadores foram os mesmos utilizados para o modelo linear.

A figura 33 apresenta o método:

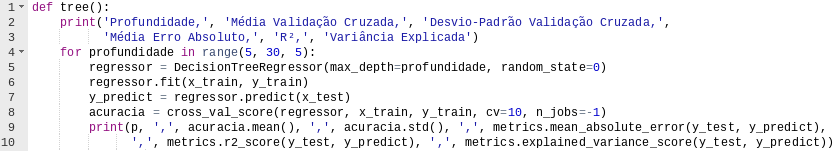


FIGURA 33: Método de avaliação e definição da profundidade da árvore de decisão.

Fonte: O Autor

Os níveis de profundidade da árvore foram testados, conforme linha 4, dos valores de 5 até 25.

O valor máximo de 25 foi escolhido, pois conforme a figura 34, em um teste com 100 árvores, as maiores variações de métrica ocorrem até, aproximadamente, este valor.

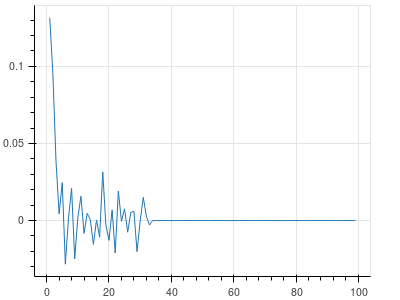


FIGURA 34: Teste do modelo com 100 árvores.

Fonte: O Autor

Para setar o nível de profundidade o parâmetro ‘*max\_depth*’, recebeu o valor corrente da variável ‘p’ e para garantir que o conjunto de dados fosse treinado sempre na mesma ordem o parâmetro ‘*random\_state*’, recebeu o valor 0.

Na linha 7 a variável ‘y\_predict’, recebe os resultados, referentes a predição do conjunto de teste.

As linhas 9 e 10 são responsáveis por receber e imprimir o resultado das métricas da eficiência de cada nível de profundidade da árvore.

Abaixo, a tabela 4, exibe os resultados da métricas de cada nível:

TABELA 4

Resultado das métricas da árvore de decisão, de acordo com sua profundidade

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Profundidade | Média Validação Cruzada | Desvio-Padrão da Validação Cruzada | Média do Erro Absoluto (R$) | R² |
| 5 | 0,7683 | 0,1230 | 51.808,86 | 0,5145 |
| 10 | 0,7623 | 0,1556 | 37.086,16 | 0,5738 |
| 15 | 0,7769 | 0,1398 | 35.937,07 | 0,5805 |
| 20 | 0,7797 | 0,1308 | 32.791,86 | 0,6265 |
| 25 | 0,7714 | 0,1506 | 36.885,14 | 0,5408 |

Fonte: O Autor

Com base nos valores apresentados, na tabela 4, o nível de profundidade com maior eficiência é o 20, pois apresentou o maior valor de média da acurácia da validação cruzada e R² e os menores valores de desvio-padrão e média do erro absoluto. Mas apesar da melhora em relação ao modelo linear, os valores de média de validação cruzada e R² se mantiveram bem abaixo do 1 ideal, o que indicou que somente uma árvore não era suficiente para garantir um regressor consistente.

Por causa deste motivo o mesmo método de cálculo de métricas foi aplicado em uma floresta aleatória, em que a quantidade de árvores e profundidade eram alterados automaticamente e o resultado exibido ao final.

A seguir na figura 35, tem-se o método criado para teste de uma floresta aleatória:

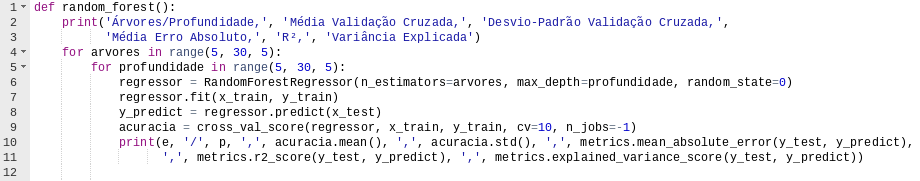


FIGURA 35: Método de cálculo para floresta aleatória.

Fonte: O Autor

Em relação ao método anterior as únicas diferenças se encontram das linhas 4 a 6, pois nas linhas 4 e 5 são feitas duas iterações que alteram o número de árvores e a profundidade. Na linha 6 cria-se um regressor de Floresta Aleatória.

Os resultados das métricas são apresentados na tabela seguinte.

TABELA 5

Resultado das métricas da floresta aleatória, de acordo com quantidade de árvores e profundidade

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Árvores / Profundidade | Média Validação Cruzada | Desvio-Padrão da Validação Cruzada | Média do Erro Absoluto (R$) | R² |
| 5 / 5 | 0,8048 | 0,1071 | 51.561,28 | 0,6585 |
| 5 / 10 | 0,8358 | 0,8300 | 34.714,23 | 0,7047 |
| 5 / 15 | 0,8491 | 0,0747 | 36.565,53 | 0,6918 |
| 5 / 20 | 0,8405 | 0,0813 | 34.840,55 | 0,6929 |
| 5 / 25 | 0,8442 | 0,0797 | 35.548,32 | 0,7051 |
| 10 / 5 | 0,8085 | 0,1094 | 47.564,53 | 0,7132 |
| 10 / 10 | 0,8393 | 0,0885 | 33.223,20 | 0,7607 |
| 10 / 15 | 0,8467 | 0,0872 | 33.581,44 | 0,7517 |
| 10 / 20 | 0,8381 | 0,0952 | 32.772,93 | 0,7498 |
| 10 / 25 | 0,8405 | 0,0932 | 33.817,69 | 0,7524 |
| 15 / 5 | 0,8174 | 0,0947 | 46.055,14 | 0,7065 |
| 15 / 10 | 0,8500 | 0,0768 | 32.568,22 | 0,7523 |
| 15 / 15 | 0,8575 | 0,0739 | 32.370,09 | 0,7647 |
| 15 / 20 | 0,8496 | 0,0811 | 31.824,14 | 0,7507 |
| 15 / 25 | 0,8523 | 0,0781 | 32.647,13 | 0,7541 |
| 20 / 5 | 0,8240 | 0,0867 | 45.210,21 | 0,7284 |
| 20 / 10 | 0,8555 | 0,0694 | 32.516,06 | 0,7598 |
| 20 / 15 | 0,8606 | 0,0706 | 31.518,09 | 0,7833 |
| 20 / 20 | 0,8545 | 0,0766 | 30.885,83 | 0,7714 |
| 20 / 25 | 0,8560 | 0,0734 | 31.539,07 | 0,7711 |
| 25 / 5 | 0,8253 | 0,0859 | 43.852,90 | 0,7406 |
| 25 / 10 | 0,8583 | 0,0673 | 31.711,47 | 0,7657 |
| 25 / 15 | 0,8610 | 0,0681 | 30.933,56 | 0,7945 |
| 25 / 20 | 0,8577 | 0,0749 | 30.708,28 | 0,7711 |
| 25 / 25 | 0,8568 | 0,0737 | 30.828,96 | 0,7860 |

Fonte: O Autor

Pode-se verificar pelas métricas que o melhor resultado é o que contém 25 árvores com profundidade de 15 folhas, pois conforme visto na tabela 5 este modelo figura entre os menores valores de média de erro absoluto e desvio-padrão, além de possuir os maiores valores de média da acurácia da validação cruzada e R².

Para verificar a efetividade deste regressor, mais um teste foi feito, que pode ser observado no histograma apresentado na figura 36. Este teste consistiu em avaliar a diferença entre os valores reais do conjunto de teste e os valores estimados através deste mesmo conjunto.

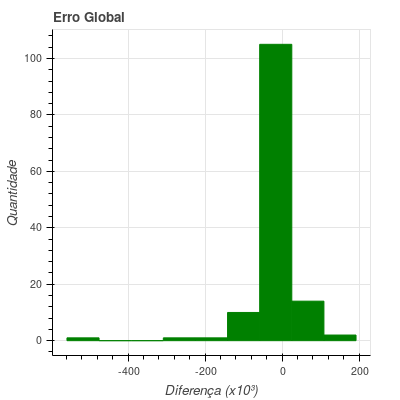


FIGURA 36: Erro global entre valores reais e preditos.

Fonte: O Autor

Nota-se que a maioria dos valores de diferença se compreendem próximos a zero, o que indicava uma ótima performance do regressor.

Um último teste foi efetuado com o objetivo de verificar o comportamento do regressor com relação ao bairro onde o imóvel se encontra e para tal estimou-se valores de preços de imóveis com características semelhantes, alterando-se somente o bairro.

A definição dos aspectos dos imóveis para este teste foi feita por meio da média das características de cada um deles encontradas no conjunto de dados, ou seja, para o imóvel de dois quartos verificou-se, antes de estimar, a média da quantidade de banheiros, garagens e o tamanho da área que um imóvel deste possui.

O método para resgate destas informações se encontra na figura 37.

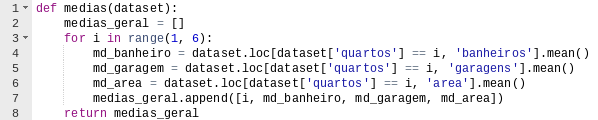


FIGURA 37: Método para a média das características dos imóveis.

Fonte: O Autor

Neste método acontece uma iteração de 1 a 5 em que os valores das médias são armazenados na variável ‘medias\_geral’, conforme as linhas 3 a 7.

Na tabela que segue são apresentados o resultado das médias dos imóveis.

TABELA 6

Média das características dos apartamentos com base na quantidade de quartos

|  |  |  |  |
| --- | --- | --- | --- |
| Quartos | Média Banheiros | Média Garagens | Média Área (m²) |
| 1 | 1,28 | 0,53 | 70,43 |
| 2 | 1,39 | 1,02 | 73,40 |
| 3 | 1,53 | 1,54 | 104,85 |
| 4 | 2,20 | 2,34 | 150,68 |
| 5 | 2,66 | 2 | 128,37 |

Fonte: O Autor

Conforme apresentado pela tabela 6 o tamanho da área de apartamentos de cinco quartos é menor que o de apartamentos de quatro quartos, isto ocorre devido à baixa disponibilidade de imóveis deste tipo à venda, conforme observado na seção 3.4 deste capítulo. Esta situação é parecida com os imóveis de um quarto em relação aos apartamentos de dois quartos, que possuem áreas de tamanho parecidas, devido a baixa disponibilidade de imóveis de um quarto.

Baseado nos valores de média apresentados na tabela 6 e para um resultado mais preciso, os valores de quantidades de banheiros e garagens foram arredondados para um número inteiro.

Na figura 38 é apresentado o método criado para predição do valor de imóvel:



FIGURA 38: Método para predição de preço do imóvel.

Fonte: O Autor

Das figuras 39 a 43, são exibidos os resultados de um mesmo tipo de apartamento em todos os bairros disponíveis no conjunto de dados.

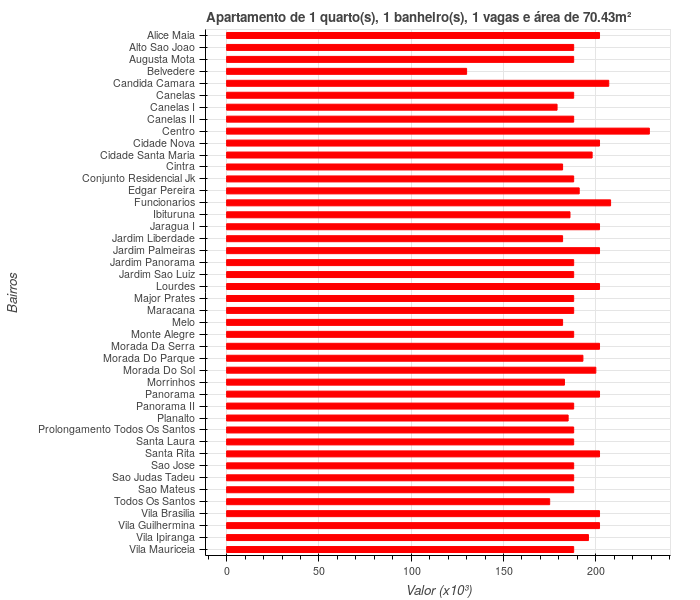


FIGURA 39: Estimativas de preços para apartamentos de 1 quarto.

Fonte: O Autor

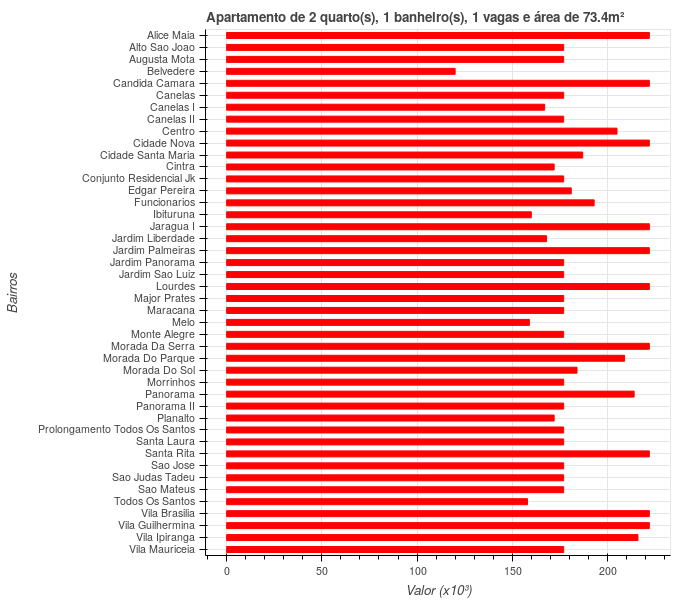


FIGURA 40: Estimativas de preços para apartamentos de 2 quartos.

Fonte: O Autor

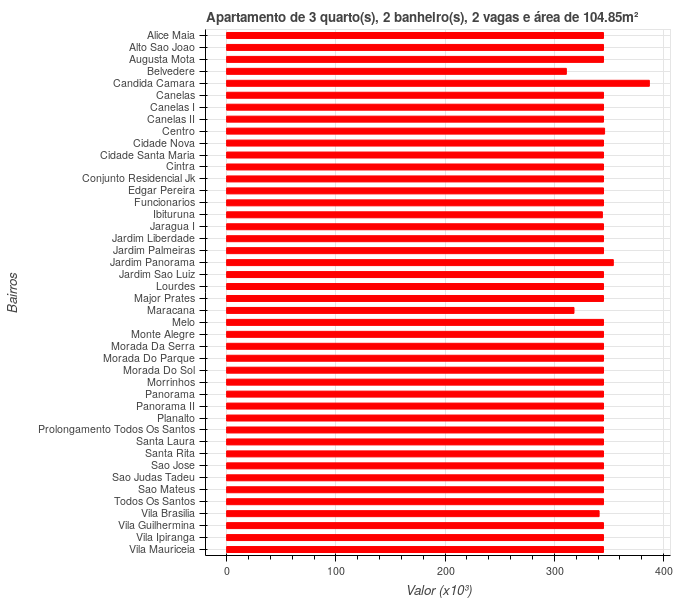


FIGURA 41: Estimativas de preços para apartamentos de 3 quartos.

Fonte: O Autor

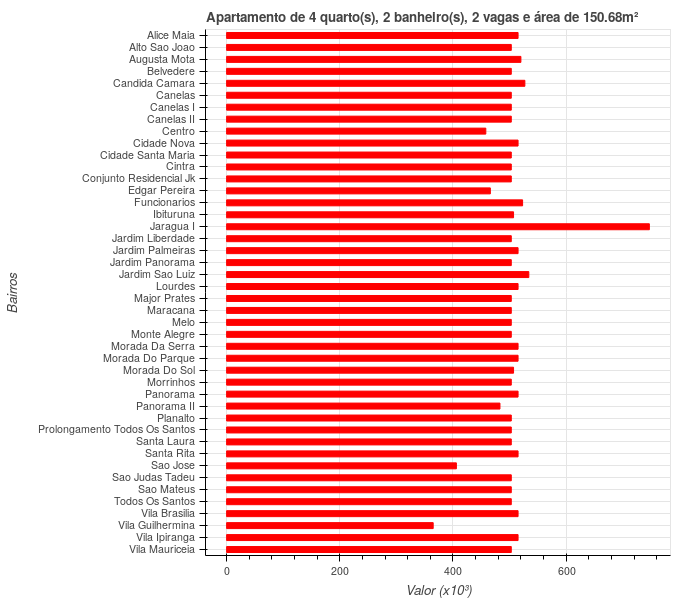


FIGURA 42: Estimativas de preços para apartamentos de 4 quartos.

Fonte: O Autor

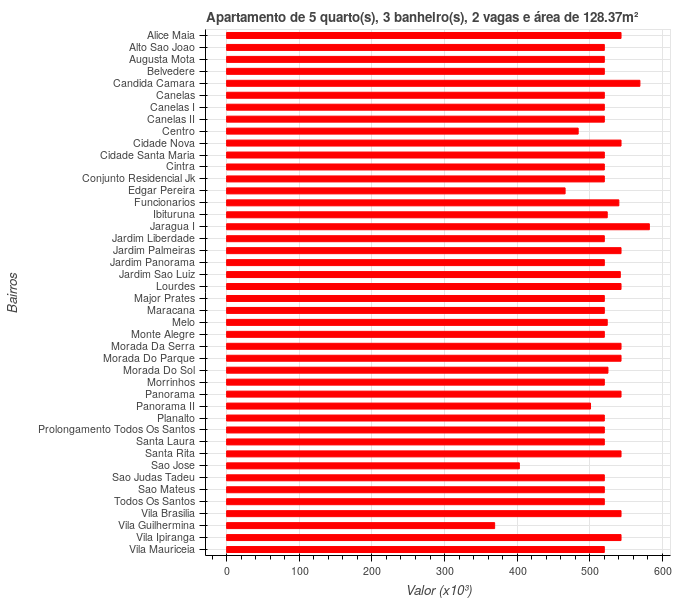


FIGURA 43: Estimativas de preços para apartamentos de 5 quartos.

Fonte: O Autor

Como é possível verificar nos testes, os apartamentos, apesar de possuírem características semelhantes, recebem preços diferentes de acordo com o bairro. Este comportamento denota a aprendizagem adquirida pelo regressor, pois conforme observa-se, os valores estimados de um apartamento em um bairro mantiveram-se, em geral, estáveis, ou seja, os apartamentos com preços mais altos permaneceram figurando entre os mais caros em todas as amostras e o mesmo ocorre para apartamentos mais baratos.

Comprova-se assim o funcionamento esperado do modelo regressor.

**3.3. Sistema de Recomendação e Busca**

O sistema de recomendação e busca foi criado com o propósito de apresentar ao usuário opções de imóveis que sejam mais similares no que tange suas características, seja número de quartos, vagas de garagem, o bairro e até mesmo o valor.

No processo de busca são levados em consideração somente os imóveis pertencentes ao mesmo bairro, já o processo de recomendação consulta bairros diferentes.

Para efetuar estes procedimentos, além das características dos imóveis, tais como quartos, vagas e valor, foram acrescidas as coordenadas geográficas, latitude e longitude, dos bairros. Este acréscimo tem como objetivo possibilitar a identificação de bairros que sejam menos distantes, ou seja, tenham coordenadas geográficas mais similares.

Para obtenção das coordenadas foi utilizada uma Interface de Programação de Aplicações, sigla API em inglês, disponibilizada pela empresa Google em seu repositório de aplicativos.

O código desenvolvido é exibido na figura a seguir:

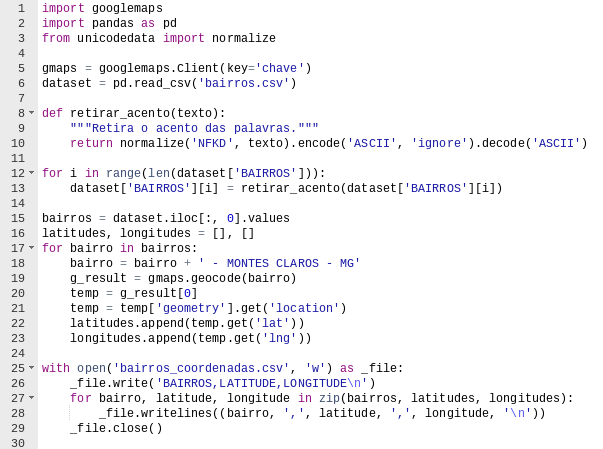


FIGURA 44: Código para busca e armazenamento das

coordenadas geográficas dos bairros.

Fonte: O Autor

As bibliotecas importadas possuem as seguintes funções:

1. Linha 1: API do Google, responsável pela busca das coordenadas geográficas.
2. Linha 2: Biblioteca responsável por manipulação de conjuntos de dados.
3. Linha 3: Responsável pela manipulação de textos.

Na linha 5 é feita a autenticação na API e na 6 a variável ‘dataset’, recebe o conjunto de dados, com o nome de todos os bairros.

Por divergências encontradas nos nomes dos bairros, principalmente pela acentuação, optou-se por eliminar quaisquer tipos de acentos. Este procedimento tem por objetivo eliminar qualquer obstáculo ao efetuar a busca das coordenadas pela API, procedimento executado das linhas 8 a 13.

A partir da linha 15 até a 22, é efetuado o procedimento de busca das coordenadas, através de uma iteração do conjunto de nomes atribuída a variável ‘bairros’.

Da linha 25 até 29 é gerado um arquivo chamado ‘bairros\_coordenadas.csv’, que recebe o nome dos bairros, bem como as coordenadas.

Finalizados tais procedimentos, o arquivo gerado está pronto para ser utilizado no processo de recomendação e busca.

O código final deste sistema é apresentado nas figuras seguinte:

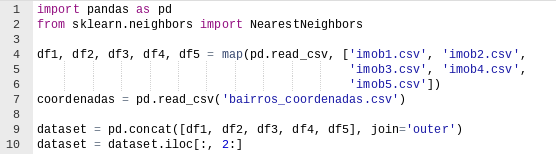


FIGURA 45: Leitura e atribuição às variáveis correspondentes dos

conjuntos de dados

Fonte: O Autor

Pode-se ver na linha 2 a importação da biblioteca sklearn.neighbors que contém a classe ‘*NearestNeighbors’*. Esta classe é responsável por criar o objeto que efetua o cálculo dos imóveis mais similares.

Na figura 45, pode-se ver que foram atribuídas às variáveis de ‘df1’ a ‘df5’ os conjuntos de dados dos imóveis e na linha 7 os dados das coordenadas.

Na linha 9 é feita uma concatenação dos conjuntos de dados e na 10 foram retiradas as colunas de ‘id’ e ‘dia’ da variável ‘dataset’, pois estas não são necessárias.

O método, mostrado na figura 46, é responsável por adicionar as informações de coordenadas geográficas no conjunto de dados.

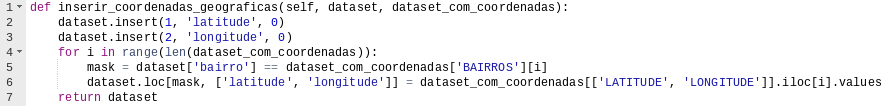


FIGURA 46: Método de inserção das coordenadas geográficas.

Fonte: O Autor

Já com as coordenadas geográficas inseridas no conjunto de dados, o próximo passo foi criar um método para efetuar o cálculo de similaridade entre os imóveis.

A figura 47 seguinte, exibe o método:



FIGURA 47: Método calculador de distância.

Fonte: O Autor

Conforme observado na figura 47, na linha 2 a variável ‘neigh’ recebe um objeto do tipo ‘*NearestNeighbors’*, que efetua o cálculo de distância entre todos os elementos do conjunto de dados. Este objeto recebe a quantidade de elementos que se deseja agrupar e a métrica, definida pelo parâmetro ‘*metric*’, desejada.

Na linha 3 são passadas as características dos imóveis e retorna-se a posição dos elementos que são mais similares.

O próximo método, exibido na figura 48, foi criado para selecionar os imóveis que devem ser utilizados para busca e recomendação, ou seja, se tratando da busca devem ser consultados somente apartamentos no mesmo bairro, caso seja recomendação somente apartamentos de bairros diferentes.

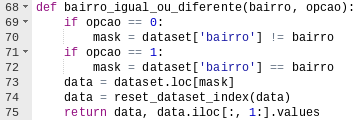


FIGURA 48: Método para selecionar imóveis de

acordo com o bairro.

Fonte: O Autor

Na figura 49, são mostrados os métodos que efetuam a busca e a recomendação dos imóveis e ao final imprime-se o resultado.

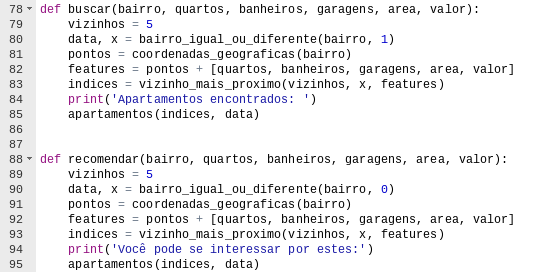


FIGURA 49: Métodos de busca e recomendação de

apartamentos.

Fonte: O Autor

Após finalizadas as criações dos métodos, foram feitos alguns testes com bairros diferentes, escolhidos aleatoriamente, mas que possuíam características iguais.

Estes primeiros testes objetivavam verificar o funcionamento dos métodos e portanto o parâmetro ‘metric’, exibido na figura 47, foi setado para dois tipos de distâncias, a euclidiana (*‘euclidean’*) e a cosseno (*‘coseno’*), distâncias estas suportadas pela classe ‘*NeireastNeighbors’*.

A figura a seguir, apresenta os imóveis escolhidos para busca.

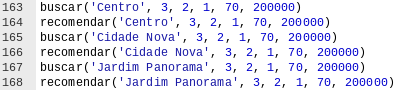


FIGURA 50: Chamada dos métodos de busca e

recomendação.

Fonte: O Autor

Na próxima figura, são exibidos os resultados utilizando a métrica euclidiana:

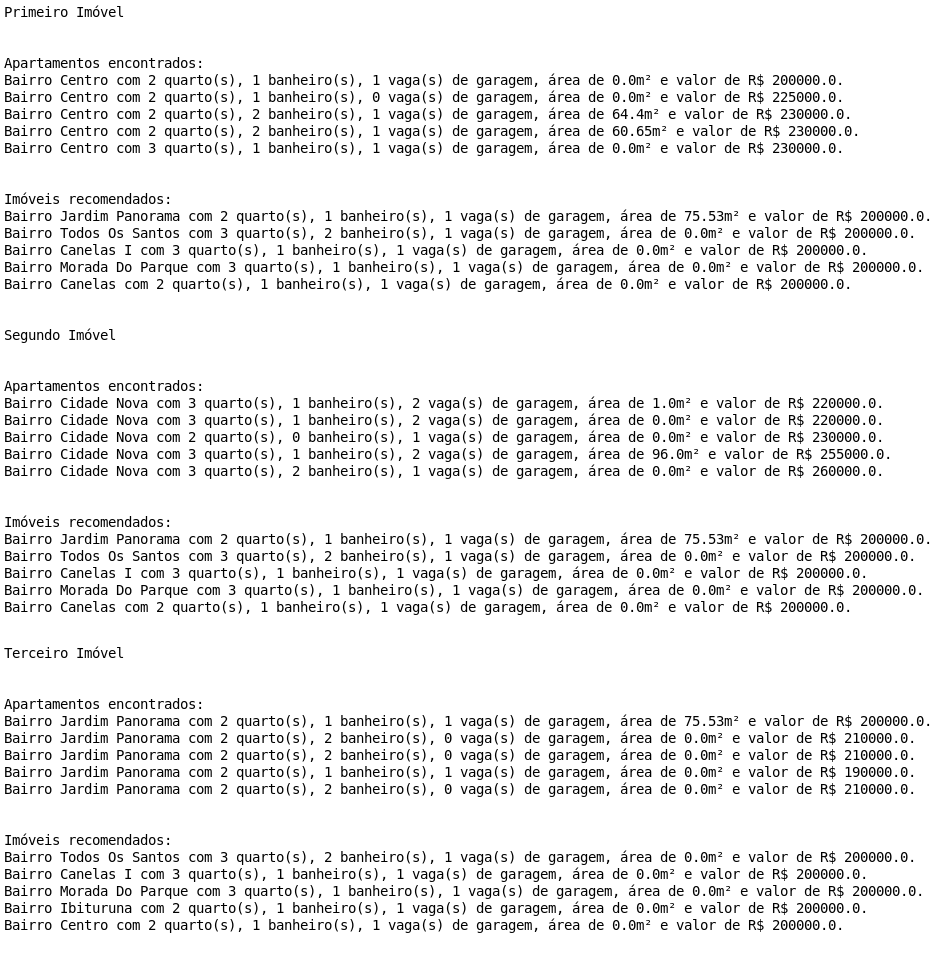


FIGURA 51: Resultado dos testes de busca e recomendação

para métrica euclidiana.

Fonte: O Autor

E na figura seguinte são exibidos os resultados para busca e recomendação utilizando métrica de distância do cosseno.



FIGURA 52: Resultado dos testes de busca e recomendação para métrica

do cosseno.

Fonte: O Autor

A quantidade de cinco imóveis para busca e recomendação foi definida, pois conforme a figura 53, a dissimilaridade entre o imóvel buscado e o imóvel apresentado aumenta a cada unidade recomendada, causando a recomendação de imóveis com aspectos muito diferentes do desejado.

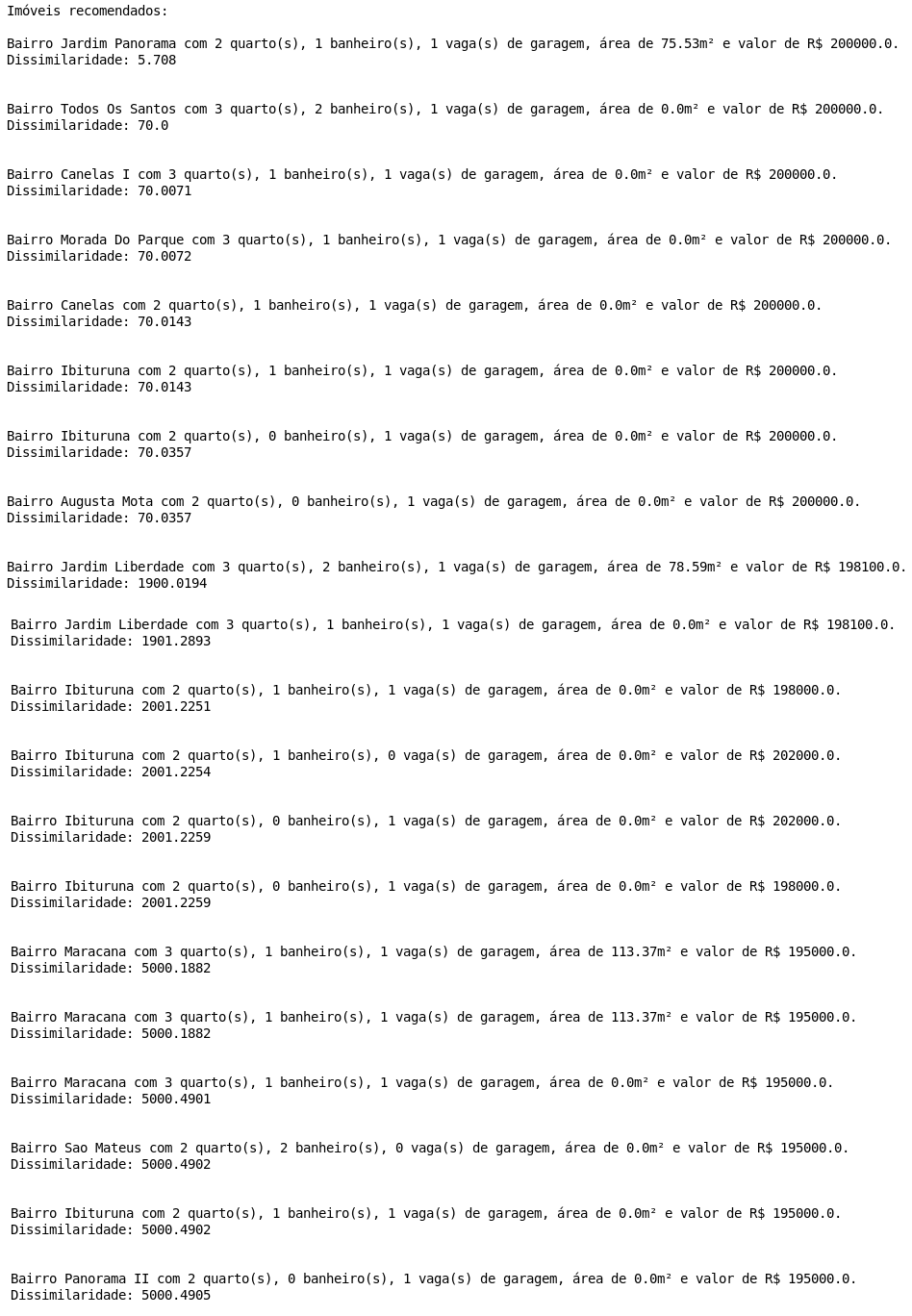


FIGURA 53: Itens recomendados e valor da dissimilaridade.

Fonte: O Autor

Através destes resultados fica comprovado o funcionamento dos sistema de recomendação e busca.

A última parte de implementação do sistema de recomendação, consistiu em avaliar a eficiência das recomendações através de uma métrica.

A métrica escolhida para este estudo foi a de precisão-*recall*, pois segundo Aggarwal (2016), este tipo de métrica é capaz de avaliar e comparar a eficiência entre tipos de medidas de distância.

A precisão representa a porcentagem de itens recomendados que realmente é relevante e o *recall* a porcentagem de todos os itens relevantes que foi recomendada (AGGARWAL, 2016).

Os cálculos de precisão e *recall* são efetuados da seguinte maneira:

A figura a seguir apresenta a implementação dos cálculos em código.

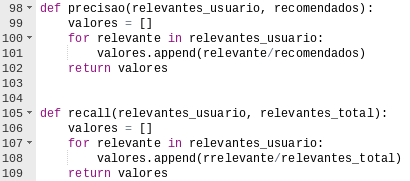


FIGURA 54: Implementação para cálculo da precisão e recall.

Fonte: O Autor

Ainda segundo Aggarwal (2016), existe a métrica F1-*score*, que combina os valores de precisão e *recall* em uma média harmônica.

O valor de F1-*score* é obtido através da seguinte fórmula:

A implementação desta fórmula é apresentada na figura seguinte:

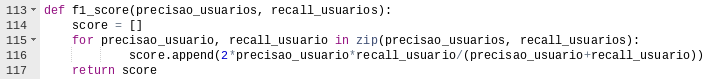


FIGURA 55: Implementação fórmula F1-score.

Fonte: O Autor

A avaliação da eficiência entre as métricas de distância euclidiana e similaridade do cosseno se deu através de um formulário de pesquisa.

Neste formulário o usuário era apresentado a um texto sobre a compra de um imóvel com as características de ser localizado no bairro Centro, possuir três quartos, dois banheiros, uma vaga de garagem e um valor de R$ 230000,00.

Neste mesmo texto o usuário era informado que não existia no bairro Centro, nenhum imóvel à venda, mesmo com características diferentes e era apresentado a ele dez opções de imóveis semelhantes, sendo cinco calculados através da distância euclidiana e cinco através da similaridade do cosseno.

Solicitava-se então ao usuário escolher as opções recomendadas as quais ele achava mais relevante.

As figuras 56 e 57, exibem as opções apresentadas para o usuário.

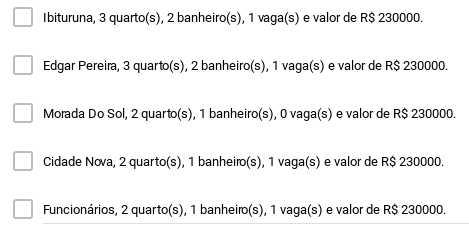


FIGURA 56: Imóveis apresentados ao usuário calculados pela distância

euclidiana.

Fonte: O Autor

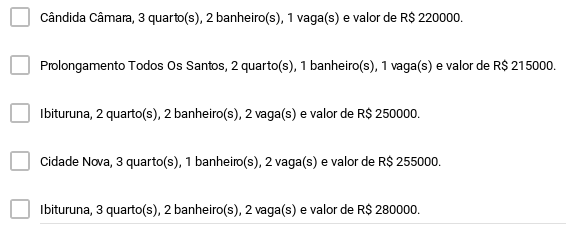


FIGURA 57: Opções de imóveis apresentados ao usuário calculados pela similaridade

do cosseno.

Fonte: O Autor

O formulário ficou disponível durante os dias 20/04/2018 a 23/04/2018 e no total foram obtidas 111 respostas, as quais foram utilizadas para efetuar os cálculos precisão, *recall* e F1-*score*.

O valor da média de F1-*score* foi utilizada para definição da métrica de distância mais eficiente.

A tabela 7 a seguir, apresenta os resultados:

TABELA 7

Resultado cálculos de eficiência das recomendações

|  |  |  |  |
| --- | --- | --- | --- |
|  | Média F1-score | Média Precisão | Média Recall |
| Distância Euclidiana | 0,2322 | 0,2090 | 0,2613 |
| Distância do Cosseno | 0,1959 | 0,1566 | 0,2613 |

Fonte: O Autor

Conforme mostrado na tabela 7 a média de F1-*score* da distância euclidiana foi maior e portanto esta medida foi definida, com base nos itens mais relevantes escolhidos por usuários, como padrão no sistema de recomendação e busca.

**3.4. Análise Exploratória dos Dados**

A análise exploratória dos dados possuiu o objetivo de apresentar algumas observações a respeito do conjunto de dados.

Nesta primeira etapa da análise exploratória o pré-processamento foi feito somente para regularizar o nome dos bairros, os demais elementos do conjunto de dados permaneceram iguais.

No estudo, este conjunto, obtido após atuação do rastreador web e armazenamento em banco de dados, possui 670 amostras de apartamentos, e na figura 58, são apresentadas as quantidades de imóveis por bairro.

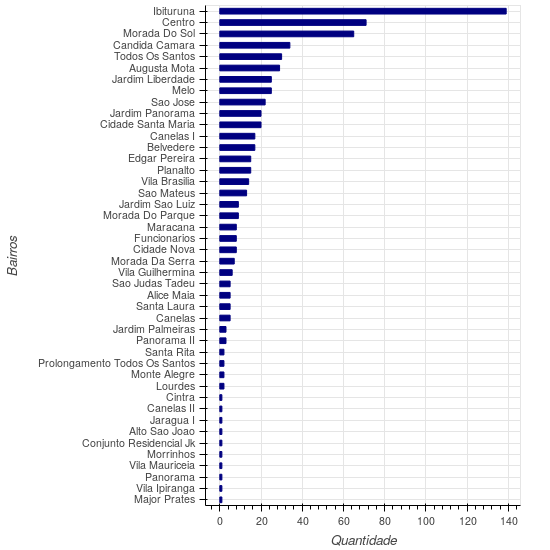


FIGURA 58: Quantidade de imóveis por bairro.

Fonte: O Autor

A figura 59 destaca a distribuição destes imóveis pela cidade, considerados os seguintes aspectos:

1. O raio dos círculos é proporcional a quantidade de apartamentos.
2. Nos círculos vermelhos a quantidade mínima de apartamentos é 30.
3. Círculos azuis possuem entre 20 e 29 apartamentos.
4. Círculos verdes possuem entre 10 e 19 apartamentos.
5. Círculos pretos possuem de 1 a 9 apartamentos.

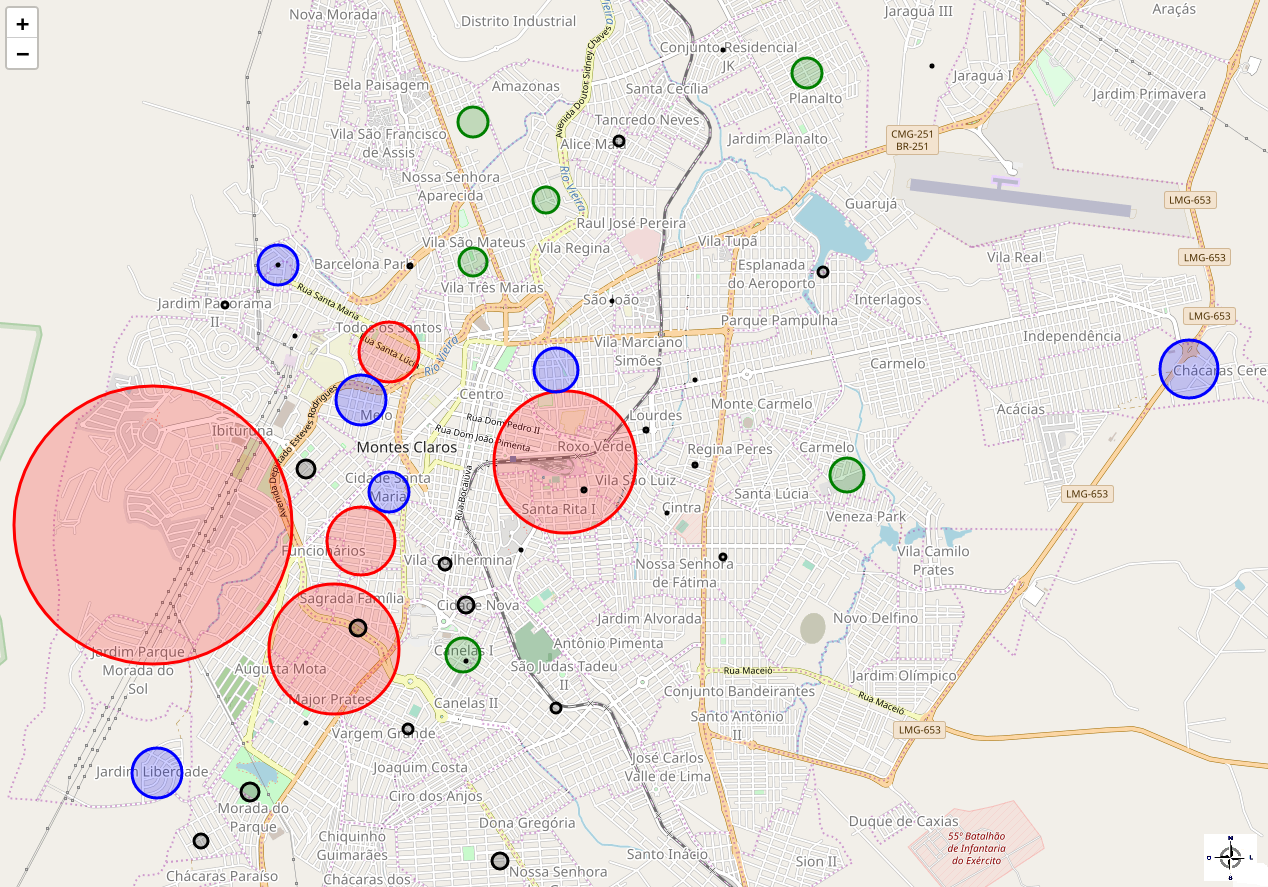


FIGURA 59: Distribuição de imóveis pela cidade.

Fonte: O Autor

Nota-se através das figura 59 uma hegemonia dos imóveis compreendida na região oeste da cidade de Montes Claros.

A figura seguinte, apresenta, com base na quantidade de quartos, o número de imóveis em cada bairro:

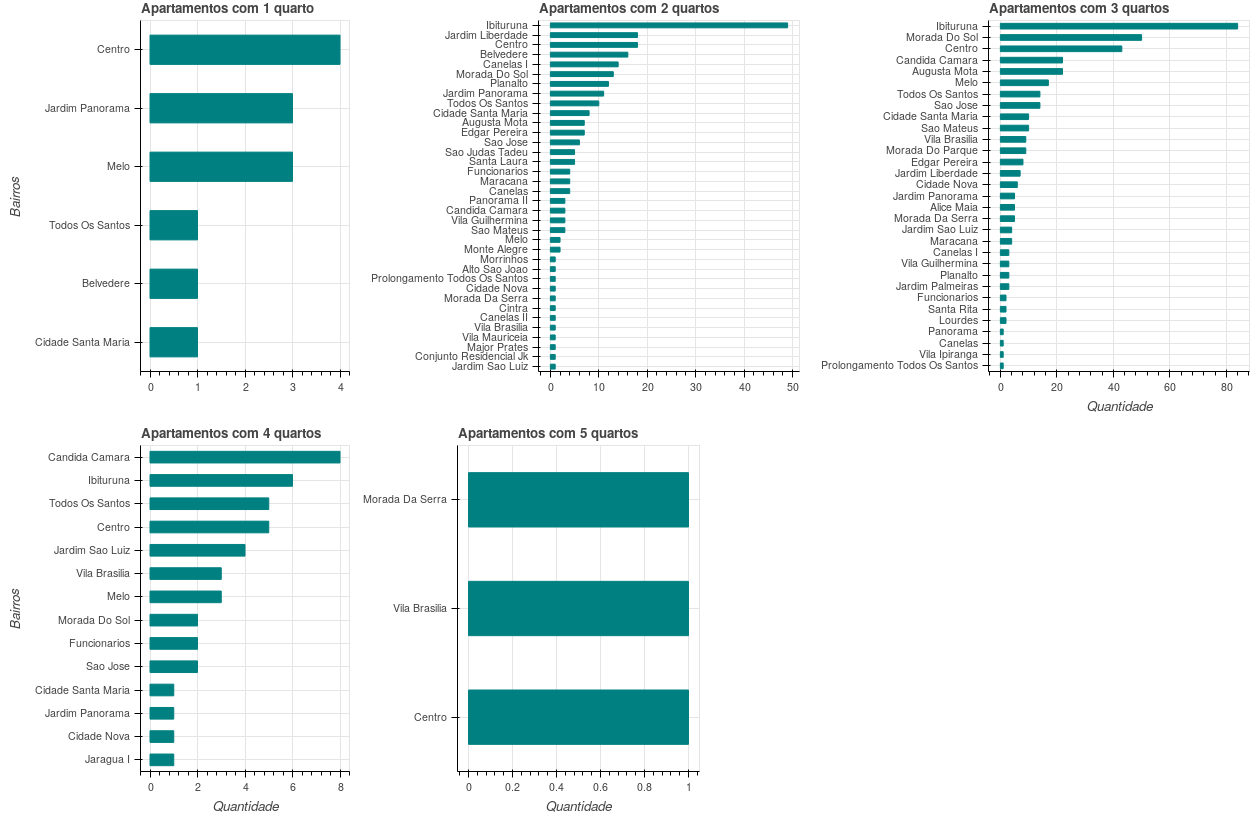


FIGURA 60: Quantidade de imóveis, por bairro, com base na quantidade de quartos.

Fonte: O Autor

Como esperado a quantidade de imóveis com três quartos é predominante e a de apartamentos de cinco quartos muito pequena. Outro ponto importante a se observar, na figura 60, é que somente o bairro Centro, possui imóveis em todas as representações.

E por fim na figura 61, são apresentadas as porcentagens de banheiros e vagas de garagem com relação ao número de apartamentos:

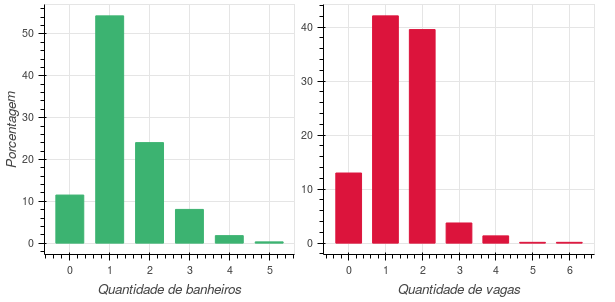


FIGURA 61: Porcentagem de banheiros e vagas de garagem.

Fonte: O Autor

Conforme observado na figura 61, a porcentagem de imóveis sem banheiro é expressiva e muito provável esta falha foi ocasionada por erro humano ao catalogar os imóveis.

Merece destaque também, a pequena diferença entre as quantidades de apartamentos com uma e duas vagas de garagem.

Sabendo-se do funcionamento correto do regressor e de que o bairro Centro possui imóveis com todas as configurações, algumas perguntas foram feitas:

1. Qual a estimativa de preço para o menor apartamento, no bairro Centro, com os valores obtidos através dos dados?
2. E para o apartamento com a maior área?
3. Quanto o valor de um apartamento aumenta se aumentarmos a área?
4. E se aumentarmos a quantidade quartos, banheiros e vagas?

Nesta etapa foi efetuado o pré-processamento dos dados da mesma forma que para o regressor linear.

Para a primeira pergunta a figura 62, apresenta o resultado da questão:





FIGURA 62: Estimativa para o menor apartamento.

Fonte: O Autor

Na linha 7 é obtido o valor da menor área de um apartamento do conjunto de dados, na linha 8 feita a estimativa e por fim o resultado é apresentado, um valor estimado de R$ 153.197,37.

A lógica para a segunda pergunta é a mesma, só obtendo o valor máximo de cada característica, e o resultado é apresentado na figura 63:

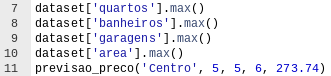




FIGURA 63: Estimativa para o maior apartamento.

Fonte: O Autor

Conforme apresentado na figura 63, um imóvel com estas características, teria um valor estimado de R$ 688.400,00. A predição deste valor pode ser respondida ao se observar que no conjunto de dados os valores de imóveis de 5 quartos variam de R$ 400.000,00 a R$ 650.000,00.

Na terceira pergunta a estimativa tomou como base as configurações do menor apartamento e aumentou-se a área em 10m².





FIGURA 64: Estimativa com área aumentada em 10m².

Fonte: O Autor

A figura 64, mostra que o acréscimo de 10m² na área, aumentou o valor do imóvel em R$ 1.880,00. Valor explicado pelo preço equivalente entre apartamentos com áreas entre 20m² e 35m² observada no conjunto de dados.

Abaixo a figura 65, apresenta o aumento da área em 10m² até 103.5m² e a diferença do preço em relação ao valor imediatamente anterior:

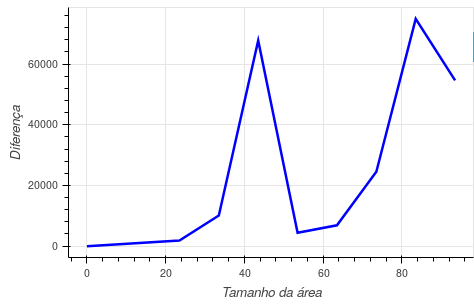


FIGURA 65: Diferença no valor do preço do imóvel a cada aumento.

Fonte: O Autor

Se observa na figura 65 que a cada vez que o imóvel dobra de tamanho o aumento é mais acentuado.

A resposta a última pergunta foi baseada no mesmo modelo de imóvel e abaixo são exibidos os resultado.

A figura 66, representa a diferença de valor, baseada a quantidade de quartos:

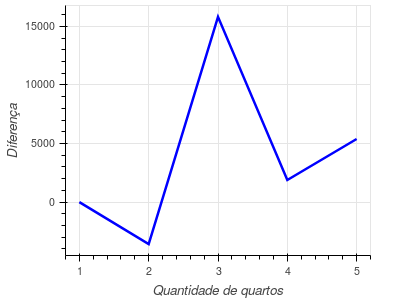


FIGURA 66: Diferença de preços baseada no

aumento do número de quartos.

Fonte: O Autor

O que se observa na figura 66 é uma redução do preço entre 1 e 2 quartos, o que pode ser explicado pela pequena demanda de apartamentos de 1 quarto com preços equivalentes a apartamentos de 2 quartos na região. Tem-se então que o valor médio por quarto adicionado é de R$ 4.872,39.

A figura seguinte apresenta o resultado da diferença baseada no aumento do número de banheiros:

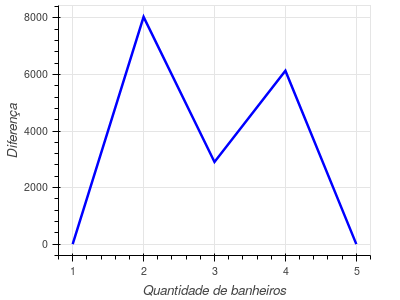


FIGURA 67: Diferença de preços baseada

no aumento do número de banheiros.

Fonte: O Autor

Conforme observado na figura 67, a grande diferença de preços ocorre quando se aumenta para 2 o número de banheiros e outro ponto importante a se observar é que a não há diferença de preço em um apartamento de 4 e 5 quartos. A média de preço no valor do apartamento ao se acrescentar um banheiro é de R$ 4.270,59.

E por fim, na figura 68, tem-se a diferença de preço em um apartamento ao se acrescentar vagas de estacionamento:

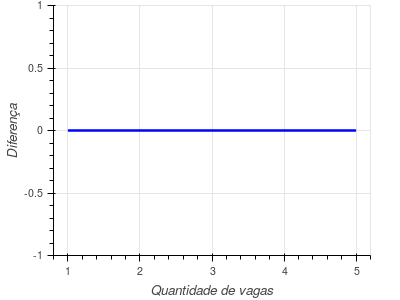


FIGURA 68: Diferença de preços baseada

no aumento do número de vagas.

Fonte: O Autor

A figura 68, mostra que independente da quantidade de vagas, o valor do apartamento se mantém constante. E este efeito pode ser explicado

Os testes e resultados apresentados comprovam o funcionamento do Sistema Agredador de Informações sobre Imóveis.

**CAPÍTULO 4 APLICAÇÃO**

Neste capítulo serão indicadas algumas áreas em que este estudo pode ser aplicado.

Após a análise dos resultados obtidos e descritos no capítulo 3, em que foram listadas as particularidades, a abordagem e o funcionamento do sistema, constatou-se que este deve ser aplicado às empresas e/ou corretoras imobiliárias possiblitando-as a obterem aspectos relevantes sobre a situação dos imóveis na cidade no que diz respeito à oferta, preço, localização, dentre outros.

Pode ser aplicado também aos usuários que desejam realizar a compra de um apartamento e para tal desejam possuir uma ideia inicial sobre o valor do imóvel baseado em suas características.

**CONSIDERAÇÕES FINAIS**

O aproveitamento da internet para uso comercial se prova, a cada ano, de extrema importância para a economia, sobretudo em empresas ligadas ao setor imobiliário, em que o crescente aumento da população reflete a necessidade constante por moradia. Sabendo-se também da grande quantidade de dados digitais gerados por esse meio tecnológico, existem atualmente ferramentas capazes de, através das informações disponibilizadas, aprender e predizer padrões. Por meio desta perspectiva, este estudo procurou desenvolver um sistema que fornecesse ao usuário aspectos e ideias novas sobre um determinado imóvel, auxiliando-o na decisão de efetivar a compra ou até mesmo realizar a busca do apartamento ideal.

O resultado obtido por meio deste trabalho foi possível com a utilização de técnicas de rastreamento de dados web, armazenamento e manipulação dos dados para interpretação do computador, aprendizado de máquina e análise exploratória. Para que este fosse conquistado, houve uma pesquisa sobre as tecnologias que seriam utilizadas, tais como a linguagem de programação Python, ambiente de desenvolvimento VSCode e servidor de banco de dados MySQL, além de uma extensa pesquisa bibliográfica para entender os aspectos matemáticos e estatísticos que envolvem os algoritmos de aprendizagem de máquina e a busca de informações por meio do rastreamento de dados em sites.

O rastreador web foi desenvolvido com o objetivo de resgatar as características dos apartamentos, bairro em que se encontra, quantidade de quartos, banheiros, vagas de estacionamento, tamanho da área e valor do imóvel, disponibilizadas nos sites imobiliários.

Essas características, obtidas por meio do rastreador, foram armazenadas em um servidor de banco de dados MySQL, corretamente configurado para recebê-las, sendo possível resgatá-las para uso posterior.

Após obtenção e armazenamento dos dados, para interpretabilidade destes pelo computador, ocorreu uma manipulação, a fim de retirar inconsistências que prejudicassem o processamento das técnicas de aprendizagem de máquina.

A técnica de aprendizado de máquina, para regressão, foi utilizada neste projeto objetivando estimar o preço de um imóvel com base em suas características se mostrando eficiente após avaliação de suas métricas e resultados adquiridos por meio de testes. A técnica de aprendizado para recomendação, objetivou buscar e/ou recomendar, apartamentos de características semelhantes ao imóvel que um usuário tenha interesse sendo a métrica mais eficiente para busca e/ou recomendação definida por meio de um questionário, no qual os usuários eram apresentados a alternativas e indicavam as mais relevantes, tornando sua funcionalidade mais adaptada à realidade dos entrevistados.

Por fim, a análise exploratória, utilizou-se dos dados obtidos e do sistema de regressão para apresentar características sobre a disponibilidade e distribuição dos apartamentos na cidade e responder perguntas relativas a inferência no preço de um imóvel dadas condições específicas de quantidade de quartos, banheiros, vagas de garagem e tamanho da área.

Todos os procedimentos propostos, sendo eles, a criação de um rastreador web, armazenamento e manipulação correta dos dados, desenvolvimento de um sistema de regressão e recomendação, além da análise exploratória dos dados foram alcançados com sucesso.

Como melhorias futuras sugere-se a abrangência de outros tipos de imóveis, tais como casas, sítios, condomínios e outras condições de transação financeira, além de compra. Criação de um sistema web para disponibilização das informações e a partir de uma quantidade suficiente de dados durante meses, índices de valorização ou desvalorização de um bairro.

**REFERÊNCIAS**

AGGARWAL, Charu C. *Recommender Systems.* New York: Springer International Publishing, 2016.

ALPAYDIN, Ethem. *Introduction to Machine Learning*. 2.ed. Massachusetts: MIT Press, 2010.

BARREIRA, Elisa da Conceição Marques. *População e Enriquecimento de Ontologias através de Web Scraping.* Porto, 2014. Tese (Engenharia Informática). Instituto Superior de Engenharia do Porto.

BERNARD, Benoit. *Web Scraping and Crawling Are Perfectly Legal, Right?.* 2017. Disponível em: <https://benbernardblog.com/web-scraping-and-crawling-are-perfectly-legal-right/>. Acesso em: 23 mar. 2018.

BISHOP, Chistopher M. *Pattern Recognition and Machine Learning.* New York: Springer Science + Business Media, 2006.

BOKEH. *About.* 2017. Disponível em: <https://bokehplots.com/pages/about-bokeh.html>. Acesso em: 5 abr. 2018.

BORGES, Luiz Eduardo. *Python para Desenvolvedores*. 2.ed. Rio de Janeiro: Edição do Autor, 2010.

BUSSAB, Wilton de O.; MORETTIN, Pedro A. *Estatística Básica.* 6.ed. São Paulo: Saraiva, 2010.

CASTRO, Leonardo Nunes de; FERRARI, Daniel Gomes. *Introdução à Mineração de Dados: Conceitos Básicos, Algoritmos e Aplicações.* São Paulo: Saraiva, 2016.

COLLINS. *Definição de Parser.* 2018. Disponível em <https://www.collinsdictionary.com/pt/dictionary/english/parser>. Acesso em: 22 mar. 2018.

CRUZ, Felipe. *Python: escreva seus primeiros programas*. 1.ed. São Paulo: Casa do Código, 2015.

DATA FOR THOUGHTS. *Passo-a-passo: Data Scraping - Aliceweb.* 2014. Disponível em: <https://dataforthoughts.wordpress.com/2014/01/29/passo-a-passo-data-scraping-aliceweb/>. Acesso em: 17 mar. 2018.

EMC. *Executive Summary: Data Growth, Business Opportunities, and IT Imperatives.* 2014. Disponível em: <https://www.emc.com/leadership/digital-universe/2014iview/executive-summary.htm>.Acesso em: 21 jan. 2018.

EXAME. *Perspectivas do mercado imobiliário em 2018.* 2017. Disponível em: <https://exame.abril.com.br/negocios/dino/perspectivas-do-mercado-imobiliario-em-2018/>. Acesso em: 19 jan. 2018.

FACELI, et al. *Inteligência Artificial: Uma Abordagem de Aprendizado de Máquina.* Rio de Janeiro: LTC, 2011.

FERNANDES, Anita Maria da Rocha. *Inteligência Artificial: noções gerais*. 1.ed. Florianópolis: Visual Books, 2003.

FERREIRA, Aurélio Buarque de Holanda. *Miniaurélio: o dicionário da língua portuguesa*. 6.ed. Curitiba: Positivo, 2005.

GERHARDT, Tatiana Engel; SILVEIRA, Denise Tolfo. *Métodos de Pesquisa*. 1.ed. Porto Alegre: Editora da UFRGS, 2009.

GORAKALA, Suresh Kumar. *Building Recommendation Engines.* Birmingham: Packt Publishing, 2016.

HEMENWAY, Kevin; CALISHAIN, Tara. *Spidering Hacks: 100 Industrial-Strength Tips & Tools.* Sebastopol: O’Reilly, 2003.

INTERNET WORLD STATS. Internet Users in the World by Regions. 2017. Disponível em: <https://www.internetworldstats.com/stats.htm>. Acesso em: 19 mar. 2018.

JAMES et al. *An Introduction to Statistical Learning: with Applications in R.* New York: Springer Science + Business Media, 2013.

LATTIN, James; CARROLL, J. Douglas; GREEN, Paul E.. *Análise de Dados Multivariados.* São Paulo: Cengage Learning, 2011.

LENTE, Caio. *O Fluxo do Web Scraping.* 2018. Disponível em: <http://curso-r.com/blog/2018/02/18/2018-02-18-fluxo-scraping/>. Acesso em: 26 mar. 2018.

MATPLOTLIB. *Matplotlib*. 2017. Disponível em: <https://matplotlib.org>. Acesso em 10 fev. 2018.

MITCHELL, Ryan. *Web Scraping with Python: Colleting Data from the Modern Web.* 1.ed. Sebastopol: O’Reilly Media, 2015.

MUELLER, John Paul; MASSARON Luca. *Machine Learning for Dummies*. New Jersey: John Wiley & Sons, 2016.

NUMPY. *Numpy*. 2017. Disponível em: <http://www.numpy.org>. Acesso em: 12 fev. 2018.

PANDAS. *Pandas.* 2018. Disponível em: <http://pandas.pydata.org/index.html>. Acesso em: 5 mar. 2018.

PEDREGOSA, *et al*. *Scikit-learn: machine learning in python*. Journal of Machine Learning Research, n.12, p. 2825-2830, 2011. Disponível em: <http://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf>. Acesso em: 12 fev. 2018.

RASCHKA, Sebastian; MIRJALILI, Vahid. *Python Machine Learning*. 2.ed. Birminghan: Packt Publishing, 2017.

REAMAT. *Tipos de Erros.* 2018. Disponível em: <https://www.ufrgs.br/reamat/CalculoNumerico/livro-sci/rdneadm-tipos\_de\_erros.html>. Acesso em: 10 mai. 2018.

RICCI, Francesco; ROKACH, Lior; SHAPIRA, Bracha. *Recommender Systems Handbook.* 2.ed. New York: Springer Science + Business Media, 2015.

RUSSELL, Stuart J.; NORVIG, Peter. *Inteligência Artificial*. 3.ed. Rio de Janeiro: Elsevier, 2013.

SEGARAN, Toby. *Programming Collective Intelligence: Building Smart Web 2.0 Applications.* 1.ed. Sebastopol: O’Reilly Media, 2007.

SMITH, Gina. *The State of the Internet 2017: All Statistics Here.* 2017. Disponível em: <http://anewdomain.net/2017-internet-statistics-the-state-of-the-internet-web-growth/>.Acesso em: 22 mar. 2018.

SMOLA, Alex; VISHWANATHAN, S. V. N. *Introduction to Machine Learning*. 1.ed. United Kingdom: Cambridge University Press, 2008.

W3C. *Document Object Model (DOM).* 2005. Disponível em: <https://www.w3.org/DOM/>. Acesso em: 16 fev. 2018.