UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MODELOS PREDITIVOS COMPORTAMENTAIS PARA OTIMIZAÇÃO DE DESPACHO DE TÁXIS

BRENO JOSÉ BUENO OTSUKA

ORIENTADORA: PROFA. DRA. KELEN CRISTIANE TEIXEIRA VIVALDINI

São Carlos – SP
16 de maio de 2019

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MODELOS PREDITIVOS COMPORTAMENTAIS PARA OTIMIZAÇÃO DE DESPACHO DE TÁXIS

BRENO JOSÉ BUENO OTSUKA

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação, área de concentração: Inteligência Artificial

Orientadora: Profa. Dra. Kelen Cristiane Teixeira Vivaldini

São Carlos – SP

16 de maio de 2019

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

Folha de Aprovação

Assinaturas dos membros da comissão examinadora que avaliou e aprovou a Defesa de Dissertação de Mestrado do candidato Breno José Bueno Otsuka, realizada em 12/04/2019:

Profa. Dra. Kelen Cristiane Teixeira Vivaldini
UFSCar

Prof. Dr. Pedro Augusto Munari Junior
UFSCar

Prof. Dr. Moacir Antonelli Ponti
ICMC/USP

Certifico que a defesa realizou-se com a participação à distância do(s) membro(s) Pedro Augusto Munari Junior e, depois das arguições e deliberações realizadas, o(s) participante(s) à distância está(ao) de acordo com o conteúdo do parecer da banca examinadora redigido neste relatório de defesa.

Profa. Dra. Kelen Cristiane Teixeira Vivaldini

AGRADECIMENTOS

Agradeço a todas as pessoas e instituições que de alguma forma contribuíram para a realização deste trabalho, em especial à minha orientadora Profa. Dra. Kelen Cristiane Teixeira Vivaldini, ao Dr. Rafael Coronel Bueno Sampaio, ao Departamento de Computação da UFSCar e ao CNPq, pela bolsa concedida.

RESUMO

OTSUKA, B. J. B (2019). Modelos preditivos comportamentais para otimização de despacho de táxis. 60p Dissertação de Mestrado – Departamento de Computação, Universidade Federal de São Carlos, São Carlos, 2019.

Com a popularização dos aparelhos *smartphones*, diversos aplicativos de chamada de táxi *online* surgiram como uma forma mais eficiente de requisitar um táxi. Estes aplicativos intermedeiam a comunicação entre os passageiros e os taxistas, reduzindo o tempo de espera dos passageiros e aumentando a confiabilidade do serviço. Esta intermediação consiste no Problema de Atribuição Táxi-Passageiro, cuja meta é selecionar o melhor taxista para cada passageiro, e para resolver este problema utiliza-se um método de despacho de táxis. Neste contexto, surge o desafio de adequar este método aos interesses e às necessidades dos usuários. Assim, neste trabalho, propôs-se uma abordagem para múltiplos passageiros utilizando a predição de comportamento dos usuários com a meta de maximizar a taxa de atribuições bem sucedidas. Para tanto, treinou-se dois modelos preditivos utilizando métodos de Aprendizagem de Máquina Supervisionada (Regressão Logística e Árvores de Decisão Impulsionadas por Gradiente), um para estimar a probabilidade de um taxista aceitar a oferta de uma requisição e outro para estimar a probabilidade de uma requisição ser atendida por um taxista. Além disso, implementou-se duas funções objetivos, uma linear e outra não-linear, para avaliar as distribuições de ofertas. Uma heurística foi utilizada para cada uma das funções, sendo que para a linear garantiu-se a solução ótima e para a não-linear não. Também implementou-se dois critérios de seleção, um baseado em cancelamento de requisições por passageiros e outro por tempo de espera dos usuários. Em simulações numéricas com dados da capital de São Paulo disponibilizados pela *Easy Taxi*, testou-se o método de despacho de táxis proposto. Os resultados indicaram que o modelo preditivo de aceite foi superior ao de atendimento com a função objetivo não-linear e inferior com a linear, além disso constatou-se que o critério de seleção baseado em cancelamento foi levemente superior ao baseado em tempo de espera. A melhor taxa de atendimento foi de 74,76% com o modelo preditivo de aceite com a função não-linear e o critério de seleção por cancelamento. Por fim, testou-se a repetição de requisições não ofertadas ou não aceitas, obtendo 76,59% de atendimento.

Palavras-chave: Sistema de transporte inteligente, problema de atribuição táxi-passageiro, despacho de táxi, aprendizagem de máquina, otimização combinatória.

ABSTRACT

OTSUKA, B.J.B (2019). Behavioral Predictive Models on the Optimization of Taxi Dispatching. 60p M.Sc Dissertation (Master) - Departamento de Computação, Universidade Federal de São Carlos, São Carlos, 2019.

The popularization of smartphones has given rise to several online taxi-booking applications as a more efficient way to call for a taxi. These applications mediate communication between passengers and taxi drivers, reducing the waiting time of passengers and increasing the reliability of the service. This intermediation consists of the Taxi-Passenger Matching Problem, whose goal is to select the best taxi driver for each passenger, and to solve this problem a taxi dispatch method is used. In this context, the challenge arises to adapt this method to the interests and needs of users. Thus, in this work, a multi-passenger approach was proposed that used users' prediction of behavior with the goal of maximizing the rate of successful assignments. For this, two predictive models were trained using Supervised Machine Learning methods (Logistic Regression and Gradient Boosted Decision Trees), one to estimate the probability of a taxi driver accepting the offer of a request and another to estimate the probability of a request being answered by a taxi driver. In addition, two objective functions, one linear and one non-linear, were implemented to evaluate offer distributions. A heuristic was used for each of the functions, and for the linear it was guaranteed the optimal solution and for the non-linear one not. We also implemented two selection criteria, one based on cancellation of requests by passengers and another by waiting time of users. In numerical simulations with data from the capital of São Paulo made available by Easy Taxi, the proposed taxi dispatching method was tested. The results indicated that the predictive model of acceptance was superior to that of attendance with the objective function non-linear and inferior to linear, in addition it was verified that the selection criterion based on cancellation was slightly superior to the one based on waiting time. The best attendance rate was 74.76% with the predictive model of acceptance with the nonlinear function and the criterion of selection by cancellation. Finally, it was tested the repetition of requests not offered or not accepted, obtaining 76.59% of attendance.

Keywords: Intelligent transportation system, taxi-passenger matching problem, taxi dispatching, machine learning, combinatorial optimization.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

Capítulo 1

INTRODUÇÃO

Neste capítulo são apresentadas a contextualização e motivação deste trabalho, bem como o Problema de Atribuição Táxi-Passageiro, os objetivos geral e específicos e a estrutura do texto.

1.1 Contextualização e motivação

O serviço de táxi está presente em vários centros urbanos oferecendo transporte rápido, conveniente e acessível. Até meados de 2011, para se solicitar um táxi era necessário entrar em contato com uma central para fazer um agendamento, ligar diretamente para um taxista ou acenar para um táxi livre na rua. Entretanto, com a popularização dos aparelhos *smartphones*, os aplicativos de chamada de táxis *online*, tais como EasyTaxi, 99Taxis, Safer Taxi, TaxiBeat, dentre outros, surgiram como uma forma mais eficiente de requisitar um táxi.

Estes aplicativos são comumente disponibilizados para os passageiros e os taxistas. De forma genérica, os passageiros buscam ir a algum lugar gastando "pouco", enquanto os taxistas visam obter um retorno financeiro levando pessoas aos seus respectivos locais de destino. Para tanto, nestes aplicativos, as operações básicas de um passageiro são requisitar um táxi e cancelar uma requisição que realizou, enquanto as de um taxista são aceitar a oferta de uma requisição e cancelar uma requisição que aceitou atender.

Neste contexto, os passageiros requisitam um táxi a qualquer momento, informando o ponto de embarque (origem), o destino, a forma de pagamento pretendida, entre outras informações que variam de aplicativo para aplicativo. Por outro lado, os taxistas recebem ofertas destas requisições e optam por atendê-las ou aguardarem a próxima oferta. Com isso, o papel destes aplicativos é intermediar a comunicação entre os usuários cadastrados, reduzindo o tempo de espera dos passageiros e aumentando a confiabilidade do serviço.

Esta intermediação consiste no Problema de Atribuição Táxi-Passageiro, cuja meta é selecionar o melhor taxista para cada passageiro que realizou uma requisição. Em outras palavras, a essência deste problema é conciliar as tolerâncias e as preferências dos dois grupos de usuários, taxistas e passageiros, a fim de maximizar a taxa de requisições atendidas, ou seja, a porcentagem de atribuições bem sucedidas. Para tanto, este problema é tratado pelo método de despacho de táxis, cuja tarefa é justamente selecionar e enviar um táxi para cada requisição recebida.

Há duas abordagens para este método. A primeira é executar o método de despacho de táxis para cada requisição recebida pelo sistema. Já a segunda é acumular um conjunto de requisições pendentes e executar periodicamente o método de despacho de táxis para múltiplos passageiros. Assim, enquanto a primeira estratégia reduz o tempo de resposta do sistema para os usuários, a segunda visa distribuir melhor as ofertas das requisições pendentes entre os taxistas disponíveis. Além disso, como os taxistas são usuários destes aplicativos, é comum dividir o método de despacho de táxis em duas partes. Uma para distribuir as ofertas aos taxistas, a fim de verificar se eles aceitam ou não atendê-las, e outra para remover os possíveis conflitos de interesse dos taxistas por uma mesma requisição, com o intuito de fazer as atribuições táxi-passageiro.

Além de facilitarem a comunicação entre taxistas e passageiros, estes aplicativos têm o seu processo de atribuição automatizado e monitorado da requisição de um táxi ao atendimento de um passageiro. Assim, os dados armazenados nestes aplicativos contêm informações valiosas sobre o comportamento dos usuários e as características das cidades que atuam, permitindo o desenvolvimento de pesquisas nesta área. Como por exemplos, Hoque, Hong e Dixon (2012) analisaram os padrões de mobilidade urbana dos usuários de táxi na cidade de São Francisco, Estados Unidos, Yuan et al. (2016) propuseram um modelo de detecção de táxis não-licenciados, Zhang et al. (2016) propuseram um algoritmo de agrupamento de dados paralelo, *Grid and Kd-Tree for DBSCAN* (GD-DBSCAN), para detectarem regiões com demanda por táxis e Dong et al. (2016) fizeram um estudo sobre o comportamento operacional de taxistas de Nova York, Estados Unidos.

Dentre as publicações, destaca-se Zhang et al. (2017), que propuseram um modelo preditivo para estimar a probabilidade de um taxista aceitar a oferta uma requisição, a fim de maximizar a taxa de aceite de ofertas por taxistas. Para tanto, testaram Regressão Logística e Árvores de Decisão Impulsionadas por Gradiente para treinar o modelo preditivo e uma heurística de otimização combinatória para maximizar a taxa de aceite. Assim, obtiveram uma taxa de aceite de ofertas de 84%, uma taxa de requisições canceladas de 24% e, consequentemente, uma taxa de atendimento de aproximadamente 60%, demonstrando resultados superiores a dois outros modelos, um baseado em arquitetura multiagente e outro em tempo de espera dos passageiros.

Neste trabalho foi proposto um método de despacho de táxis para múltiplos passageiros validado por meio de simulações numéricas utilizando dados reais. Para tanto, adotou-se a estratégia de usar predição de comportamento dos usuários combinado com otimização. Além disso, considerou-se não somente as operações de aceite de ofertas por taxistas, mas também as de cancelamento de requisições pelos usuários. Assim, com técnicas de Aprendizagem de Máquina Supervisionada, treinou-se um modelo preditivo de atendimento que combinava a predição de aceite de ofertas por taxistas e predição de cancelamento de requisições por taxistas e passageiros, com a meta de maximizar a taxa de requisições atendidas, ou seja, a porcentagem de atribuições bem sucedidas. Com isso, verificou-se o impacto na taxa de atribuições bem sucedidas ao utilizar a combinação destes modelos preditivos, bem como o uso de heurísticas de otimização e critérios de seleção que também foram propostos neste trabalho.

1.2 Problema de Atribuição Táxi-Passageiro

Nesta seção é descrito de maneira formal o Problema de Atribuição Táxi-Passageiro para múltiplos passageiros, ou seja, que considera um conjunto de requisições pendentes e não apenas uma requisição por vez. Para tanto, inicia-se com a Subseção 1.2.1, onde é apresentada a estrutura geral do problema com as variáveis envolvidas e suas etapas. Em seguida, nas Subseções 1.2.2 e 1.2.3 são explicadas, respectivamente, as etapas de distribuição de ofertas aos taxistas e de atribuição táxi-passageiro. Vale ressaltar que a formulação do problema dada nesta seção é genérica, porém, devido à grande diversidade de aplicativos de chamada de táxi *online* e às particularidades de cada cidade que operam, podem ser necessárias pequenas adaptações.

1.2.1 Estrutura do problema

Sejam os conjuntos $T = \{t_1, t_2, ..., t_m\}$ de m taxistas disponíveis e $R = \{r_1, r_2, ..., r_n\}$ de n requisições pendentes. Cada taxista $t_i \in T$ é representado por (tid, lat, lon), onde tid é o identificador do taxista e (lat, lon) é a posição [1] atual do taxista. Já as requisições $r_j \in R$ são representadas por $(rid, pid, lat_{origem}, lon_{origem}, lat_{dest}, lon_{dest}, h_{req})$, onde rid e pid, são os identificadores da requisição e do passageiro, $(lat_{origem}, lon_{origem})$ é a posição atual do passageiro, (lat_{dest}, lon_{dest}) indica o destino do passageiro e h_{req} o horário da requisição.

A relação entre os conjuntos T e R pode ser vista como um grafo bipartido completo e ponderado, como é mostrado na Figura 1.1. Matematicamente, este grafo pode ser representado por uma matriz $S_{m \times n}$, onde o índice das linhas é dado pelo índice i dos taxistas disponíveis, o

[1] Posição: coordenada geográfica (latitude e longitude) do usuário.

índice das colunas é dado pelo índice j das requisições pendentes e cada elemento $s_{i,j}$ indica a relevância do par (t_i, r_j). Esta relevância é medida segundo algum critério de pontuação p, como por exemplo o tempo de chegada do taxista t_i até o ponto de embarque da requisição r_j ou a probabilidade do taxista t_i atender a requisição r_j. Os valores m e n devem ser números naturais não nulos, ou seja, é necessário que haja pelo menos 1 taxista disponível e uma requisição pendente para instanciar-se o problema. Além disso, a matriz $S_{m\times n}$ não necessariamente é quadrada, logo pode haver escassez de taxistas ou requisições.

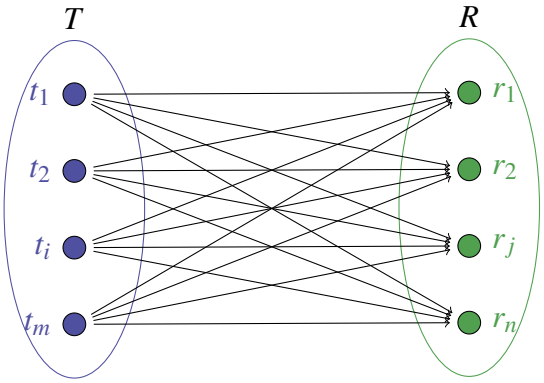

FIGURA 1.1 – Representação do problema de atribuição táxi-passageiro em um grafo bipartido completo. Os vértices $t_i \in T$ representam os taxistas disponíveis, os vértices $r_j \in R$ representam as requisições pendentes e as arestas (t_i, r_j) representam as possíveis atribuições táxi-passageiro. Para cada aresta do grafo tem-se um peso associado $s_{i,j}$ indicando o valor da relação entre t_i e r_j. Fonte: Do autor.

Considerando que, no contexto de aplicativos de chamada de táxi *online*, os taxistas também são usuários do sistema, não é possível atribuir um táxi a um passageiro sem antes consultar o interesse dos taxistas em atendê-lo. Neste sentido, o Problema de Atribuição Táxi-Passageiro é dividido em duas partes. A primeira é ofertar as requisições pendentes aos taxistas disponíveis e a segunda é, com a resposta dos taxistas em relação às ofertas recebidas, fazer a atribuição táxi-passageiro mais adequada de acordo com algum critério de seleção c.

1.2.2 Distribuição de ofertas

A primeira parte do Problema de Atribuição Táxi-Passageiro consiste em encontrar uma matriz de distribuição de ofertas $X_{m\times n}$ que maximize o ganho ou minimize os custos do método de despacho de táxi. Para tanto, faz-se necessário o uso de uma função objetivo f para avaliar a qualidade de uma distribuição de ofertas $X_{m\times n}$ com base na matriz de pontuação $S_{m\times n}$. Assim, o problema de distribuição de ofertas pode ser formulado como um problema de otimização combinatória (Eq. 1.1).

$$\begin{aligned} \max_{X_{m\times n}} (\min) \quad & f(S_{m\times n}, X_{m\times n}) \\ \text{sujeito à} \quad & \sum_{j=1}^{n} x_{i,j} = 1; \forall i \in [1,m]; \\ & x_{i,j} \in \{0,\ 1\} \end{aligned} \tag{1.1}$$

onde:

- m é a quantidade de taxistas disponíveis;
- n é a quantidade de requisições pendentes;
- $S_{m\times n}$ é a matriz de pontuação dos pares (t_i, r_j);
- $X_{m\times n}$ é a matriz de ofertas, em que $x_{i,j} = 1$, se t_i for receber a oferta de r_j; e
- f é a função objetivo.

A restrição do problema de distribuição de ofertas indica que cada taxista disponível receberá exatamente uma única oferta. Isto se dá pois, cada taxista deve ter a chance de atender um passageiro, mas, devido aos problemas inerentes à comunicação entre o sistema e os usuários, não é possível manter uma lista de ofertas ativas para cada taxista. Vale observe que não há restrições em relação ao número de ofertas por requisição pendente, logo uma mesma requisição pode ser ofertada a vários taxistas e também pode haver requisições ofertadas a nenhum taxista. Daí a relevância do uso de uma função objetivo que reduza o risco destes casos ocorrerem. Já os valores possíveis das variáveis $x_{i,j}$ faz com que o problema de distribuição de ofertas seja classificado como um problema de otimização combinatória binário e como a função objetivo f é genérica, o problema pode ser linear ou não-linear.

1.2.3 Atribuição táxi-passageiro

Uma vez construída a matriz $X_{m\times n}$ as ofertas são emitidas aos taxistas, que por sua vez aceitam ou não atender a requisição. Como, geralmente, há mais taxistas disponíveis do que requisições pendentes, as requisições comumente são ofertadas para mais de um taxista. Isto ocorre pois, enquanto as requisições surgem de forma estocástica no sistema, os taxistas ficam ativos por horas diariamente, apenas alternando seu estado entre disponível, em trânsito e ocupado. Por este motivo, existe a possibilidade de mais de um taxista receber a oferta de uma

mesma requisição e, consequentemente, mais de um taxista se interessar em atendê-la. Nestes casos, é utilizado um critério de seleção c para escolher o táxi que será atribuído a cada passageiro, como é mostrado na Equação 1.2.

$$Y_m = c(X_{m\times n}, A_m) \tag{1.2}$$

onde:

- $X_{m\times n}$ é a matriz de ofertas;
- A_m é o vetor de interesse dos taxistas pela oferta recebida;
- Y_m é o vetor de atribuição táxi-passageiro com m elementos (t_i, r_j); e
- c é o critério de desempate.

Uma vez que o critério de seleção foi aplicado o vetor de atribuição táxi-passageiro Y_m é construído. Este vetor contém os pares (t_i, r_j), indicando qual táxi atenderá qual passageiro. Assim, os usuários (taxistas e passageiros) são notificados das atribuições realizadas pelo método de despacho de táxi. No entanto, após esta etapa, tanto o passageiro quanto o taxista podem cancelar a qualquer momento a requisição, seja porque o tempo de espera é muito grande ou o passageiro não estava no ponto de embarque ou por outros motivos. Logo, se a requisição não for cancelada e o passageiro for levado até o ponto de desembarque, esta requisição é considerada bem sucedida, ou seja, atendida.

1.3 Objetivo

Este trabalho teve como objetivo geral propor e validar um método de despacho de táxis para múltiplos passageiros, cuja meta foi maximizar a taxa de requisições atendidas. Para tanto, buscou-se um equilíbrio entre as preferências e tolerâncias dos dois tipos de usuários, taxistas e passageiros. Com isso, para direcionar essa pesquisa, foram estudadas três ações dos usuários: aceite de ofertas e cancelamento de requisições por taxistas e cancelamento de requisições por passageiro. Vale ressaltar que, na extração de características, a operação do passageiro de requisitar um táxi também foi estudada. Assim, os seguintes objetivos específicos foram cumpridos:

- Definir critérios de pontuação, funções objetivo e critérios de seleção necessários para resolver o Problema de Atribuição Táxi-Passageiro;

- Preprocessar os dados e extrair as características relevantes para os modelos preditivos;
- Treinar e validar os modelos preditivos de aceite de ofertas e cancelamento de requisições; bem como treinar e validar o modelo preditivo de atendimento de requisições que combinasse as estimativas dos três primeiros;
- Simular o método de despacho de táxis proposto.

1.4 Estrutura do trabalho

A dissertação está dividida em 6 capítulos. No Capítulo 1, contextualiza-se, expõe-se as motivações para o desenvolvimento deste trabalho e é formulado o Problema de Atribuição Táxi-Passageiro. No Capítulo 2 expõe-se os trabalhos relacionados, evidenciando os métodos e os resultados obtidos.

Em seguida, no Capítulo 3, é apresentada a fundamentação teórica sobre aprendizagem de máquina. Também são descritos os métodos que foram comparados nos experimentos, bem como os métodos e métricas de validação dos modelos preditivos.

No Capítulo 4 é apresentado o método de despacho de táxis geral com a descrição dos critérios de pontuação, das funções objetivo e dos critérios de seleção propostos e testados neste trabalho.

Já no Capítulo 5 são apresentados os resultados dos experimentos realizados para concluir esta pesquisa. Para tanto, são expostos os recursos utilizados, tais como linguagem de programação, bibliotecas e a base de dados, a metodologia e os resultados atingidos para cada etapa do trabalho. Por fim, as conclusões do trabalho e trabalhos futuros são apresentadas no Capítulo 6.

Capítulo 2

REVISÃO BIBLIOGRÁFICA

Com o intuito de explorar de forma abrangente o problema investigado, neste trabalho foi realizada uma revisão bibliográfica na área de despacho de táxis. O objetivo deste processo foi encontrar os métodos adotados para resolver o Problema de Atribuição Táxi-Passageiro e os resultados obtidos nesta área. O capítulo foi dividido em cinco seções: na primeira são descritos os trabalhos de despacho de táxis que aplicaram técnicas de otimização para resolver o problema de atribuição táxi-passageiro, na segunda são apresentadas abordagens que utilizaram modelos preditivos para tomada de decisão, na terceira aborda-se despacho de táxis com compartilhamento de veículos e na quarta diversas soluções para o problema abordado. Por fim, a última seção é reservada para as considerações finais do capítulo.

2.1 Despacho de táxis baseados em otimização combinatória

Nesta seção estão descritos os trabalhos que trataram o despacho de táxis como um problema de otimização combinatória, em que buscou-se encontrar uma atribuição táxi-passageiro ótima para múltiplos passageiros. Assim, foram observadas as características utilizadas na função objetivo e os algoritmos utilizados, bem como as estratégias de separação de dados para reduzir os espaço de soluções admissíveis, nos trabalhos que abordaram esta questão.

Em Wong e Bell (2006) foi proposto um método de despacho de táxis considerando previsão de requisições para minimizar o tempo de espera total dos passageiros até o táxi chegar. Os autores trataram a atribuição táxi-passageiro como um problema de programação dinâmica e utilizaram decomposição de Bellman com uma heurística para reduzir o tempo de processamento. Os experimentos simulados indicaram que a performance da predição de requisições futuras melhorou com o número de estágios de antecipação, mas o tempo de processamento tornou-se um gargalo para aplicações práticas.

Kümmel, Busch e Wang (2016) utilizaram um algoritmo de pareamento estável para otimizar o despacho de táxis. Com experimentos simulados, avaliaram o modelo segundo o ganho dos taxistas, o número de passageiros atendidos, o tempo de espera dos passageiros e as distâncias percorridas com táxi desocupado. Os resultados indicaram que o modelo proposto obteve melhor desempenho em todas as métricas de validação comparado com uma estratégia de fila de taxistas disponíveis para atendimento.

Em Gao, Xiao e Zhao (2016) foi proposto um modelo de despacho de táxis que atribui proativamente os passageiros aos taxistas disponíveis, ou seja, sem verificar se os taxistas têm interesse em atender as requisições que lhes foram atribuídas. Os autores trataram a atribuição táxi-passageiro como um problema de correspondência bipartido ponderado com otimização multi-objetivo. Assim, definiu-se uma função objetivo, considerando o tempo de espera dos usuários, o lucro total dos taxistas, diferentes configurações de veículo e custo associado aos táxis vazios e aplicou-se um algoritmo adaptado de Kuhn-Munkres para otimizá-la. As simulações demonstraram que o método teve uma performance superior às estratégias de atribuir o táxi mais próximo ao passageiro e de maximizar o lucro dos táxis.

Um método de despacho de táxis robusto considerando as incertezas da demanda estimada foi proposto em Miao et al. (2016) para minimizar a distância percorrida com táxi desocupado e manter a integridade do serviço em toda a cidade. Para tanto, foram modeladas as correlações espaço-temporais destas incertezas. A formulação matemática do modelo minimiza o custo do pior caso em detrimento da performance média do sistema. Avaliando quatro anos de dados de táxi em Nova York constatou-se que com um nível de garantia probabilística de 75% e comparado com soluções não robustas, o erro médio da relação demanda-oferta foi reduzido em 31,7% e a distância total percorrida sem passageiro em 10,13%.

Situ et al. (2017) e Liu et al. (2017) lidaram com o grande espaço de soluções admissíveis utilizando uma estratégia de divisão e conquista por regiões geográficas. No primeiro trabalho propuseram um algoritmo de colônia de formigas e no segundo um algoritmo genético, que considerava a tolerância dos usuários e o tempo médio de espera dos passageiros. Ambos métodos foram paralelizados pelos autores. Os experimentos demonstraram que o algoritmo de colônia de formigas foi eficaz, eficiente e extensível, superando, em termos de precisão, uma solução gulosa que seleciona as atribuições táxi-passageiro mais rentáveis. Além disso, constatou-se a eficiência e eficácia do algoritmo genético, em relação à velocidade de execução, qualidade de resposta e escalabilidade.

Zhao et al. (2019) propuseram um método de despacho de táxi *online*, chamado de *Online Stable Matching under Known Identical Independent Distributions* (OSM-KIID), para abordar

o problema de atribuição táxi-passageiro com reconhecimento de preferências dos usuários em aplicativos de chamada de táxi *online*. O modelo é caracterizado por dois objetivos: maximizar o lucro total e minimizar a insatisfação geral em relação às preferências dos taxistas e dos passageiros. Para tanto, formularam o Problema de Atribuição Táxi-Passageiro como um Problema de Programação Linear e resolveram-no com duas estratégias: uma de Pareamento Estável *Online* e outra Gulosa. Os resultados experimentais demonstraram que a estratégia de Pareamento Estável foi superior à Gulosa quando o número de passageiros é maior que o número de taxistas disponíveis.

2.2 Despacho de táxis com modelos preditivos

Nesta seção são descritos os trabalhos que utilizaram modelos preditivos para auxiliarem no processo de tomada de decisão automático em relação às atribuições táxi-passageiro. O uso destes modelos em métodos de despacho de táxis é relevante, pois podem ser treinados para reconhecer padrões relevantes relacionados ao Problema de Atribuição Táxi-Passageiro. Especificamente, estes modelos são úteis para os critérios de pontuação de pares táxi-passageiro de um método de despacho de táxis, ou seja, úteis para computar a matriz de pontuação $S_{m \times n}$.

Meenakshi e Senthilkumar (2017) também adotaram previsão de requisições para minimizar o tempo de espera total dos passageiros e reduzir o desbalanceamento de demanda e oferta. Para tanto, utilizaram o *framework* Hadoop para reduzir o tempo de processamento dos dados. Assim, extraiu-se, de maneira mais eficiente, informações sobre a demanda dos passageiros e os padrões de mobilidade, o que contribuiu no processo de atribuição táxi-passageiro.

Verma e Vo (2015) incorporaram estimativas de tempo de corrida dos taxistas ocupados para considerar, no método de despacho de táxis, aqueles que ficariam disponíveis em um futuro próximo e perto de passageiros que os requisitaram. Desta forma, as chances dos passageiros serem atendidos aumentaram, juntamente com a utilização da frota de táxi e o rendimento dos taxistas. Os resultados dos experimentos com dados reais demonstraram que a abordagem proposta lida eficientemente com períodos de demanda alta e reduz o número de requisições repetidas pelos passageiros.

Um método de despacho de táxis para reduzir a divergência entre demanda e oferta foi proposto por Gelda, Jagannathan e Raina (2016). Os autores utilizaram série temporal para prever com antecedência a distribuição espaço-temporal das requisições e táxis na cidade de Bengaluru, Índia. Os modelos preditivos foram utilizados no método de despacho para redistribuir a frota de táxis entre regiões vizinhas com problema de desbalanceamento de demanda e oferta.

A acurácia dos modelos preditivos aumentou mais de 15% segmentando as regiões da cidade em *geohashes* [1] de precisão 5 e 6. A estratégia proposta, em horários de pico, foi capaz de prever a demanda e a oferta de táxi na cidade com até uma hora de antecedência.

Em Zhang et al. (2017) foi apresentado um método de despacho de táxis, que utilizou aprendizagem de máquina supervisionada para extrair do histórico de corridas os padrões de aceite dos taxistas. Assim, compararam Regressão Logística com Árvores de Decisão Impulsionadas por Gradiente, tendo obtido resultados levemente superiores com o primeiro algoritmo. Em seguida, os autores buscaram maximizar a taxa de aceite global estimada aplicando uma técnica de otimização combinatória *Hill Climbing* modificada. Nos experimentos comparou-se o método proposto com outras duas estratégias: atribuir o táxi com menor tempo de espera até o passageiro e uma abordagem multiagente. Para tanto, utilizaram 1 milhão de dados de requisições e 80 mil táxis das cidades de Pequim e Xangai, China. Os resultados demonstraram que o método atingiu uma taxa de aceite global de 84% (4% superior aos demais). Além disso, melhorou outras métricas, tais como tempo de espera do cliente e distância percorrida até o passageiro. No entanto, a taxa de cancelamento manteve-se elevada (cerca de 24%).

2.3 Despacho de táxis com compartilhamento de veículos

Nesta seção descreve-se os trabalhos em que foram propostos métodos de despacho de táxis com compartilhamento de veículos, com o intuito de melhorar a utilização da frota e atender um número maior de passageiros. Esta abordagem não foi utilizada neste trabalho, mas é uma linha que vem demonstrando resultados satisfatórios, utilizando com mais intensidade a frota de táxi disponível e, assim, reduzindo o tempo de espera dos passageiros.

Golpayegani et al. (2018) propuseram um método de despacho de táxi com compartilhamento de veículos utilizando uma abordagem multiagentes. Para tanto, ao implementarem a estratégia de despacho de táxi consideraram as preferências dos passageiros, tais como tempo e custo, e dos taxistas, como horas de trabalho e rendimento mínimo diário. Assim, tanto os passageiros, quanto os taxistas foram modelados como agentes autônomos com múltiplas preferências conflitantes. Esta abordagem foi comparada com outras duas, uma competitiva e outra colaborativa, em simulações numéricas. Os resultados demostraram que a estratégia proposta foi mais efetiva que as demais abordagens em relação ao atendimento das preferências dos usuários. Além disso, também permitiu que os passageiros adaptassem suas preferências quando nenhuma atribuição perfeita fosse possível.

[1] *Geohash* é uma cadeia de caracteres que indexa uma região retangular do planeta. Quanto maior a *geohash*, maior a precisão, pois menor será a região indexada.

Meng et al. (2010) propuseram um sistema de despacho de táxis utilizando programação de rede genética com compartilhamento de veículos, ou seja, os táxis podiam ser ocupados por mais de um passageiro com origem e destino distintos ao mesmo tempo. Com esta abordagem buscou-se reduzir o tempo de espera e de viagem dos passageiros. Os resultados simulados demonstraram robustez em relação ao crescimento gradual da densidade de clientes e que o modelo é escalável de acordo com o aumento das dimensões do mapa utilizado nos experimentos. Schreieck et al. (2016) também propuseram um modelo de compartilhamento de veículos, visando rotas curtas dentro de uma cidade. Para tanto, utilizaram um índice invertido para buscar as rotas dos táxis ativos no sistema, e assim compartilhar o trajeto com outros passageiros que estavam no caminho. Para validação do modelo, realizou-se uma avaliação da performance do sistema e constatou-se que para até 10 mil veículos ativos, gastou-se em média menos de 0.4 segundos para retornar uma resposta aos usuários.

Chen et al. (2017) propuseram um método de despacho de táxis hierárquico com compartilhamento de veículos. O intuito desta pesquisa foi atender à crescente demanda de passageiros, a oferta limitada de veículos e o serviço ineficiente de mobilidade. Para tanto, o primeiro nível desta abordagem subdivide a cidade em regiões retangulares e, considerando a demanda por táxi prevista pelo sistema, efetua um rebalanceamento dos veículos entre estas regiões, a fim de reduzir a quilometragem da frota de táxi percorrida sem passageiros. Já no segundo nível, para cada região, os agendamentos de coleta e entrega para solicitações em tempo real são obtidos para cada veículo com Programação Linear Inteira Mista, visando minimizar o atraso devido ao compartilhamento, enquanto atende o maior número de requisições possível. Por meio de simulações numéricas em um conjunto de dados de Nova York verificou-se o desempenho desta estrutura. Constatou-se que com este esquema de compartilhamento os veículos atenderam em média 2,18 passageiros por veículo. Além disso, observou-se que pelo menos 90% das requisições foram atendidas a qualquer momento do dia, cerca de 19% superior a uma abordagem sem compartilhamento de veículos.

Outros estudos sobre compartilhamento de veículos foram realizados por Hanna et al. (2016) e Ma et al. (2017), ambos utilizando veículos autônomos. No primeiro, os autores utilizaram o algoritmo *Scalable Collision-avoiding Role Assignment with Minimal-makespam* (SCRAM) para minimizar a distância até o passageiro de cada veículo, evitando, assim, que os passageiros esperassem um tempo maior do que o necessário. Constatou-se em experimentos simulados com mais de 36 mil requisições, que o método SCRAM obteve distância média percorrida com veículo desocupado, tempo médio de espera e quantidade de passageiros em espera menores, comparado com os modelos centralizado, em grade e húngaro. Já no segundo trabalho, os pedidos eram redirecionados aos veículos, que construíam sequências de embarque

e desembarque de passageiros utilizando Programação Linear. Os experimentos indicaram que a taxa de utilização da frota aumentou e que a distância percorrida pelos veículos se manteve em relação a sistemas de táxi convencionais, indicando que o compartilhamento de veículos autônomos pode melhorar a mobilidade e sustentabilidade do sistema de transporte urbano.

2.4 Despacho de táxis com soluções diversas

Nesta seção estão descritos os trabalhos relacionados à despacho de táxis que trataram o problema com diversas soluções, tais como lógica *fuzzy*, arquitetura multiagentes, teoria de jogos, aprendizagem por reforço e monitoramento das condições de tráfego. As estratégias apresentadas nesta seção também não foram utilizadas neste trabalho. No entanto, são descritas neste trabalho para mostrar uma visão geral das demais soluções exploradas nesta área de pesquisa, bem como citar outras características interessantes que foram utilizadas para realizar as atribuições táxi-passageiros.

Shrivastava et al. (1997) utilizaram lógica *fuzzy* para selecionar os taxistas visando reduzir o tempo de espera dos passageiros e aumentar a utilização da frota. Para tanto, os veículos mais próximos e menos utilizados eram escolhidos. Os resultados dos experimentos simulados demonstraram que foi possível reduzir o tempo de espera e manter a carga de trabalho uniforme entre os motoristas. Ngo, Seow e Wong (2004) também utilizaram lógica *fuzzy* em despacho de táxis considerando a distância, a utilização da frota e o tempo gasto para atingir o destino de acordo com as condições de tráfego. Os experimentos simulados demonstraram que o método superou a estratégia em que considerava-se apenas distância.

Seow, Dang e Lee (2010) propuseram um método de despacho de táxis baseado em uma arquitetura multiagente e fizeram um estudo sobre seu desempenho em relação à disponibilidade dos táxis. Esta abordagem visou melhorar a satisfação do passageiro, mantendo um controle de filas de taxistas e requisições por região. Desta maneira, agentes colaborativos de *software* negociavam as requisições entre si em nome dos taxistas. Experimentos simulados foram realizados para constatar que esta estratégia reduziu o tempo de espera dos passageiros e de viagem de táxis vazios em até 41,8% e 41,2%, respectivamente, se comparado com uma estratégia de fila de taxistas disponíveis para atendimento.

Oda e Joe-Wong (2018) buscaram reduzir o tempo de espera dos passageiros despachando proativamente veículos para locais onde as requisições foram previstas. Para tanto, no trabalho foi proposta uma estrutura baseada em *Deep Q-network* (DQN), que aprende por reforço as políticas de despacho de veículos. Como os DQNs escalam mal com um grande número de

soluções possíveis, treinou-se independentemente uma política de despacho própria para cada veículo, garantindo a escalabilidade ao custo de uma menor coordenação entre os veículos. Além disso, formulou-se uma política centralizada de Controle de Horizonte Recuado (CHR) comparando-a com as políticas de DQN. Estas comparações foram realizadas através de simulações numéricas com cerca de 15 milhões de registros de viagens de táxis, e mostraram que a política de despacho adotando DQN reduz o número de requisições não atendidas em 20% comparada com a abordagem CHR.

Qiao et al. (2015) propuseram um sistema centralizado de reserva de carona, onde os passageiros requisitavam um táxi informando origem, destino e horário de saída. As requisições eram ofertadas aos taxistas próximos, que selecionavam as requisições de interesse. Aplicou-se teoria de jogos para encontrar as atribuições que atendessem o maior número de passageiros, respeitando as preferência de ambos usuários. Dados sobre táxi disponibilizados pela Comissão Municipal de Transporte de Shenzhen, China, foram utilizados para avaliar a performance do sistema. Os resultados demonstraram uma redução de até 46% no tempo de espera dos passageiros e um aumento no lucro dos taxistas de pelo menos 18%, comparado com métodos *offline* de mineração de dados e recomendação *online*.

Ramezani e Nourinejad (2017) constataram que o aumento do uso de táxis contribuiu para a intensificação do congestionamento do tráfego, pois os táxis disponíveis não eram bem distribuídos dentro das cidades e, consequentemente, inviabilizavam o atendimento eficaz dos passageiros. Para reduzir este impacto, propuseram um método de despacho de táxi baseado no monitoramento das condições de tráfego. Assim, segmentou-se a cidade em regiões, sendo cada uma representada por um diagrama fundamental macroscópico, a fim de avaliar a evolução dinâmica das condições de tráfego. Em simulações numéricas com dados de Nova York, constataram que a abordagem melhorou a performance do sistema de despacho em relação ao atraso total dos veículos, o tempo de espera dos passageiros e a utilização da frota de táxi.

Billhardt et al. (2019) propuseram uma heurística para atribuições táxi-passageiro, em que consideravam a possibilidade de realizar reatribuições de táxis caso, globalmente, resultassem em soluções melhores. As reatribuições eram sugeridas quando houvesse a possibilidade de dois taxistas trocarem de passageiros atribuídos, pois ambos estariam mais próximo do local de embarque um do outro. Além disso, como essas novas atribuições poderiam reduzir as receitas esperadas dos taxistas individuais, foi proposto um esquema de compensação econômica para que os taxistas que concordassem com as modificações propostas em seus passageiros designados. Os experimentos comparando a heurística com outras comumente usadas, *First-Come/First-Served* (FCFS) e *Nearest-Taxi/Nearest-Request* (NTNR). Os resultados indicaram

que a heurística reduziu o tempo de espera dos passageiros, além de beneficiar economicamente os taxistas com requisições mais vantajosas.

2.5 Considerações finais

Na revisão bibliográfica, constatou-se que todos os métodos de despacho de táxis propostos resolveram o Problema de Atribuição Táxi-Passageiro em duas etapas. Uma para ofertar as requisições aos taxistas, a fim de verificar o interesse deles nelas antes de os atribuí-las, e outra para remover os conflitos de interesse entre os taxistas e realizar de fato a atribuição táxi-passageiro. As únicas exceções para foram Hanna et al. (2016) e Ma et al. (2017) que propuseram o uso de veículos autônomos, e Gao, Xiao e Zhao (2016) e Oda e Joe-Wong (2018), que utilizaram uma abordagem proativa para as atribuições táxi-passageiro.

De acordo com os artigos investigados, destacaram-se as seguintes características: tempo de espera dos passageiros, distância dos taxistas até os passageiros, utilização da frota de táxis, rendimento dos taxistas, demanda e oferta. Outras características detectadas foram: taxa de requisições repetidas, taxa de aceite, taxa de atendimento, condições do tráfego, tolerância dos passageiros em relação ao tempo de espera e preferência dos usuários. Estas informações orientaram a extração de características para os modelos preditivos. Além disso, destacam-se os trabalhos Liu et al. (2017), Golpayegani et al. (2018), Meng et al. (2010), Zhao et al. (2019), por estudarem as preferências e as tolerâncias e o comportamento dos usuários, e, principalmente, Zhang et al. (2017) por treinarem modelos preditivos de comportamento dos taxistas utilizando Aprendizagem de Máquina Supervisionada.

Com base nestas evidências, optou-se, nesta pesquisa, pelo uso de Aprendizagem de Máquina Supervisionada para treinar os modelos preditivos de comportamento dos usuários combinado com métodos de otimização. Para tanto, testou-se os algoritmos Regressão Logística e Árvores de Decisão Impulsionadas por Gradiente. Estes algoritmos foram selecionados, pois ambos apresentaram bons resultados em Zhang et al. (2017). Em acréscimo, retornam valores entre 0 e 1, indicando a probabilidade de um evento ocorrer, unificando e facilitando a utilização dos modelos preditivos no método de despacho de táxis proposto. Especificamente, no caso desta dissertação, os eventos estimados estão relacionados aos comportamentos de aceitar ofertas e cancelar requisições. Além disso, a Regressão Logística é um algoritmo de aprendizagem de máquina linear, enquanto as Árvores de Decisão Impulsionadas por Gradiente são não-lineares, permitindo analisar o desempenho dos modelos preditivos sobre o viés de algoritmos com diferentes níveis de complexidade.

Já em relação ao tratamento dos dados, optou-se por subdividir os dados em *geohashes* de precisão 5, como feito em Gelda, Jagannathan e Raina (2016) para melhorar a acurácia dos modelos. Com isso, disponibilizando informações mais precisas sobre diferentes regiões de uma cidade para os modelos preditivos. Por fim, verificou-se o comportamento dos modelos preditivos testados com diferentes funções objetivos e critérios de seleção em relação à taxa de atendimento de requisições.

Capítulo 3

APRENDIZAGEM DE MÁQUINA

Este capítulo foi dividido em 5 seções. Na Seção 3.1 têm-se uma breve visão geral sobre aprendizagem de máquina. Em seguida, na Seção 3.2 são apresentados especificamente os conceitos sobre aprendizagem de máquina supervisionada para problemas de classificação binária. Depois, na Seção 3.3 são descritos os dois algoritmos de aprendizagem testados neste trabalho, Regressão Logística e Árvores de Decisão Impulsionadas por Gradiente. Adiante, na Seção 3.4 são mostrados os métodos e as métricas de validação utilizados para treinar e avaliar os modelos preditivos. Por fim, na Seção 3.5 são dadas as considerações finais do capítulo. Os conceitos apresentados nas Seções 3.1 e 3.2 foram extraídos de Luxburg e Schölkopf (2011).

3.1 Visão Geral

Aprendizagem de máquina é o campo da computação que se estuda como uma máquina, seguindo um algoritmo, pode aprender uma tarefa específica por meio de observações. Este processo é denominado treinamento e tem como meta gerar regras genéricas o suficiente para "explicar" tanto os exemplos previamente vistos, quanto os futuros. Neste contexto, existem duas categorias: as técnicas de aprendizagem supervisionadas e as não-supervisionadas.

As técnicas supervisionadas recebem este nome, pois os exemplos são rotulados com a resposta esperada, permitindo que a máquina avalie seu desempenho durante o treinamento. Já nas não-supervisionadas, os exemplos não possuem tal rótulo. Com isso, a meta dos algoritmos desta categoria se resume a encontrar relações de similaridade entre os dados.

A seguir, o problema de aprendizagem de máquina supervisionada é apresentada formalmente. Em particular, o texto está focado em classificação binária, ou seja, em problemas que possuem apenas duas respostas possíveis. Isto se deve pois os problemas de predição de aceite

e cancelamento de requisições são binários e os conceitos explicados aqui podem ser facilmente estendidos aos demais problemas de classificação.

3.2 Aprendizagem de máquina supervisionada

Em aprendizagem supervisionada, têm-se um espaço de entrada X (espaço de instâncias ou objetos) e um espaço de saída Y (espaço de rótulos ou classes). No caso de classificação binária, cada objeto pode pertencer à uma de duas classes possíveis, geralmente, rotuladas por -1 e 1. De maneira geral, aprendizagem de máquina supervisionada se resume a estimar uma função da forma $f : X \to Y$. Tal mapeamento f é chamado de classificador. Para fazer isso, é apresentado à máquina um conjunto de exemplos de treinamento (pontos ou dados de treinamento) $(x_1, y_1), ..., (x_n, y_n) \in X \times Y$. Um algoritmo de aprendizagem é um procedimento que recebe dados de treinamento como entrada e retorna um classificador f.

Nenhuma suposição é feita sobre os espaços X e Y, que podem inclusive possuir ruídos e sobreposição entre as classes. No entanto, assume-se que existe uma distribuição de probabilidade conjunta P em $X \times Y$ desconhecida, e os exemplos de treinamento (x_i, y_i) são amostrados de forma idêntica e independentemente distribuída.

Para medir o desempenho de um classificador f utiliza-se uma função de perda (função de custo), que mede o erro entre a saída esperada e a saída estimada por f. Com esta ferramenta é possível direcionar os algoritmos de classificação aos classificadores que erram menos. No geral, uma perda igual a 0 representa uma classificação perfeita, e maiores que 0 representam divergência entre a resposta estimada pelo classificador e a esperada.

Enquanto a função de perda mede o erro de uma função sobre uma observação individual, o risco de uma função, representado por $R(f)$, é a perda média sobre todos os pontos gerados de acordo com a distribuição P, como é mostrado na Equação 3.1, sendo $loss$ a função de perda.

$$R(f) = E(loss(Y, f(X))) \tag{3.1}$$

Isto é, o risco de uma função é a perda esperada da função f para todos os pontos $x \in X$. Intuitivamente, o risco "conta" quantos elementos em X são erroneamente classificados por f. Logo, a função f é um classificador melhor que outra função g, se seu risco for menor, ou seja, $R(f) < R(g)$. Para encontrar um classificador ótimo é necessário buscar por uma função f cujo $R(f)$ seja o menor possível.

O espaço de funções admissíveis $\mathscr{F}_{all}$ é o espaço que contém todas as funções da forma

$f : X \rightarrow Y$, ou seja, $\mathscr{F}_{all} = \{f : X \rightarrow Y\}$. Neste caso, é possível encontrar o classificador ótimo dada a distribuição de probabilidade P (Eq. 3.2). Esta função é chamada de classificador de Bayes (f_{Bayes}). Teoricamente, este é o melhor classificador possível, pois é capaz de classificar os exemplos de entrada com o mínimo de chance de falhar.

$$f_{Bayes}(x) = \begin{cases} 1 & \text{se } P(Y = 1|X = x) \geq 0.5; \\ -1 & \text{caso contrário.} \end{cases} \tag{3.2}$$

Na prática, é impossível computar diretamente o classificador de Bayes, pois a distribuição de probabilidade é desconhecida. Logo, não é possível avaliar $P(Y = 1|X = x)$. Com base nas definições discutidas nesta seção, pode-se formular o problema padrão de classificação binária como: dado os pontos de treinamento $(x_1, y_1), ..., (x_n, y_n) \in X \times Y$ independentemente coletados de alguma distribuição de probabilidade P desconhecida, e dado uma função de perda *loss*, o objetivo dos algoritmos de classificação é construir funções $f : X \rightarrow Y$, tal que $R(f)$ seja o mais próximo possível de $R(f_{Bayes})$, ressaltando que $R(f)$ comumente não é computável.

3.2.1 Generalização e consistência

Dado um classificador f_n construído a partir de um conjunto de treinamento com n exemplos, é possível medir os erros cometidos por f_n nos pontos de treinamento. Para isso, efetua-se a média da divergência entre a classe real e estimada computada pela função de perda *loss* dos n exemplos de treinamento, como é mostrado na Equação 3.3. Esta medida é denominada risco empírico $R_{emp}(f_n)$.

$$R_{emp}(f_n) = \frac{1}{n} \sum_{i=1}^{n} loss(y_i, f_n(x_i)) \tag{3.3}$$

Geralmente, o risco empírico $R_{emp}(f_n)$ é relativamente baixo, caso contrário, o algoritmo de aprendizagem não foi capaz sequer de explicar os exemplos de treinamento. No entanto, não necessariamente se a função f_n cometer poucos erros no treinamento, também cometerá poucos erros no restante do espaço de entrada X. Neste sentido, um classificador generaliza bem se a diferença $|R(f_n) - R_{emp}(f_n)|$ é pequena. Vale ressaltar que um classificador que generaliza bem não necessariamente possui um risco empírico baixo. Isto significa apenas que $R_{emp}(f_n)$ é um bom estimador do risco $R(f_n)$.

Os algoritmos de aprendizagem comumente restringem a busca pelo classificador em um espaço de funções admissíveis $\mathscr{F} \subset \mathscr{F}_{all}$. Teoricamente, o melhor classificador em $\mathscr{F}$, $f_{\mathscr{F}}$, é

o classificador que minimiza o risco, como é mostrado na Equação 3.4. Vale ressaltar que não necessariamente $f_{Bayes} \in \mathscr{F}$, e portanto, é possível que $R(f_{\mathscr{F}}) > R(f_{Bayes})$.

$$f_{\mathscr{F}} = \underset{f \in \mathscr{F}}{\operatorname{argmin}} R(f) \tag{3.4}$$

Já a consistência, ao contrário da noção de generalização, não é uma propriedade de uma função individual, mas sim do espaço de funções admissíveis $\mathscr{F}$. Logo, dado uma sequência infinita de pontos de treinamento e a margem de erro ε. Para cada $n \in \mathbb{N}$, f_n é um classificador baseado nos primeiros n pontos de treinamento e determinado pela Equação 3.5.

$$f_n = \underset{f \in \mathscr{F}}{\operatorname{argmin}} R_{emp}(f) \tag{3.5}$$

1. O algoritmo de aprendizagem é chamado consistente em relação à $\mathscr{F}$, se o risco $R(f_n)$ converge para $R(f_{\mathscr{F}})$, para todo $\varepsilon > 0$, ou seja, $P(R(f_n) - R(f_{\mathscr{F}}) > \varepsilon) \to 0$ para $n \to \infty$;

2. O algoritmo de aprendizagem é chamado Bayes-consistente em relação à $\mathscr{F}$, se o risco $R(f_n)$ converge para $R(f_{Bayes})$, para todo $\varepsilon > 0$, ou seja, $P(R(f_n) - R(f_{Bayes}) > \varepsilon) \to 0$ para $n \to \infty$.

3.2.2 Dilema viés-variância

Para o classificador aprender com sucesso é necessário trabalhar com um $\mathscr{F}$ reduzido, ou seja, aumentar o viés e diminuir a variância de espaço de funções admissíveis. Dados classificadores f_{Bayes} (Eq. 3.2), $f_{\mathscr{F}}$ (Eq. 3.4) e f_n (Eq. 3.5) e a Bayes-consistência, a diferença $R(f_n) - R(f_{Bayes})$ pode ser reescrita como apresentado na Equação 3.6.

$$R(f_n) - R(f_{Bayes}) = (R(f_n) - R(f_{\mathscr{F}})) + (R(f_{\mathscr{F}}) - R(f_{Bayes})) \tag{3.6}$$

Os dois termos à direita da Equação 3.6 entre parênteses são chamados de erro de estimativa e erro de aproximação, respectivamente. O primeiro lida com a incerteza introduzida pelo processo de amostragem, ou seja, o erro de estimar um classificador a partir de um conjunto de treinamento finito. Já o segundo é o erro adicionado ao restringir o espaço de funções $\mathscr{F}$ e, assim, lida com o erro intrínseco nas limitações das funções admissíveis.

Com isso, se o algoritmo de aprendizagem utilizado buscar um classificador em um espaço de funções admissíveis muito grande, o erro de aproximação será pequeno e o de estimativa

alto, pois haverá uma grande variedade de funções complexas. Como consequência disso, a variância do algoritmo será alta, aumentando a probabilidade de *overfitting*. Já no caso oposto, haverá um número restritivo de funções mais simples. Desta forma, o algoritmo sofre com o viés das funções, aumentando a probabilidade de *underfitting*. Em outras palavras, o dilema viés-variância consiste na escolha mais adequada do algoritmo de aprendizagem e seus parâmetros, a fim de balancear os erros de estimativa e aproximação, evitando os casos de *overfitting* e *underfitting*.

3.3 Algoritmos de aprendizagem

Há diversos algoritmos de aprendizagem de máquina supervisionada. Alguns exemplos são: Redes Neurais, *Support Vector Machine* (SVM), *K-Nearest Neighbors* (KNN), dentre outros. Nesta seção são apresentados dois algoritmos de aprendizagem, Regressão Logística Binária e Árvores de Decisão Impulsionadas por Gradiente.

3.3.1 Regressão Logística Binária

Regressão Logística Binária, descrito em Cox (1958), é um modelo estatístico que mede a probabilidade de uma variável categórica binária dependente y dado um vetor de características x utilizando uma função logística. Como é mostrado no exemplo ilustrativo da Figura 3.1, a função logística é uma sigmoide, ou seja, tem uma curva característica em formato de ”S”. Sendo assim, esta função mapeia o vetor de características $x \in \mathbb{R}^d$, em que d é o número de características, para valores de saída $p \in [0,1]$, indicando a probabilidade de x ser da classe positiva ($y = 1$), ou seja, $p = prob(y = 1|x)$. Em contrapartida, por se tratar de um modelo binário, a probabilidade de x ser da classe negativa ($y = 0$) é dada pelo complemento $1 - p$.

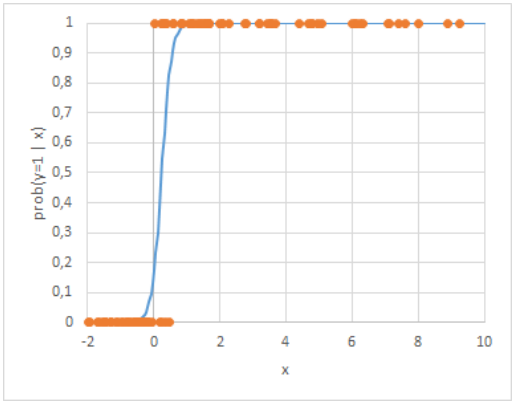

FIGURA 3.1 – Exemplo de uma função logística, em que a curva representa função em S e os pontos os exemplos de treinamento. Fonte: Do autor.

Regressão Logística Binária também pode ser utilizada para classificar objetos. Para tanto, utiliza-se um limiar de classificação θ, geralmente fixado em 0.5. Assim, a classe estimada $\hat{y}$ é positiva, se $p \geq \theta$; e negativa, caso contrário. Com isso em conjunto com a característica sigmoide da função logística, os exemplos negativos concentram-se próximo de $p = 0$, enquanto os positivos concentram-se próximo de $p = 1$. Esta característica é relevante em casos em que a probabilidade p é utilizada para tomada de decisão, como diagnóstico de doenças.

Logo, o espaço de funções admissíveis F deste algoritmo de aprendizagem é constituído pelo conjunto de funções logísticas descritas pela Equação 3.7 parametrizadas por w.

$$h_w(x) = \frac{1}{1 + e^{-w^T x}} \tag{3.7}$$

Por simplificação, $prob(y|x)$ pode ser rescrito como mostrado na Equação 3.8. Isto se dá pois $prob(y = 1|x) = h_w(x)$ e $prob(y = 0|x) = 1 - h_w(x)$. Sendo assim, o treinamento deste modelo se reduz a encontrar o vetor de parâmetros w que melhor se ajusta à base de treinamento.

$$prob(y|x) = h_w(x)^y (1 - h_w(x))^{(1-y)} \tag{3.8}$$

Para tanto, parte-se de uma solução candidata qualquer e otimiza o modelo iterativamente utilizando gradiente descendente até ocorrer a estagnação do processo de aprendizagem. Quando isto ocorre, diz-se que o algoritmo de aprendizagem convergiu. A função de custo utilizada para direcionar o treinamento é mostrada na Equação 3.9.

$$loss(y, h_w(x)) = -log(prob(y|x)) \tag{3.9}$$

Esta função de custo é utilizada para garantir a convergência do algoritmo de aprendizagem para o erro mínimo global, devido sua convexidade. Assim, o gradiente descendente direcionará o treinamento para a solução ótima de acordo com a base de treinamento utilizada.

3.3.2 Árvores de Decisão Impulsionadas por Gradiente

Árvores de Decisão Impulsionadas por Gradiente, descrito em Friedman (2001), é um modelo de *Gradient Boosted Trees* (GBT), ou seja, um modelo baseado em um conjunto de Árvores de Decisão construídas sequencialmente, corrigindo os erros da estrutura anterior utilizando um algoritmo de gradiente descendente, daí o nome do modelo. Assim, somando-se as estimativas das Árvores em um único modelo têm-se um classificador mais preciso do que as Árvores,

como é mostrado na Equação 3.10.

$$f(x) = h_0(x) + \sum_{i=1}^{M} \gamma_i h_i(x) \tag{3.10}$$

onde,

$h_0(x)$ é a Árvore de Decisão inicial;

$h_i(x)$ são as Árvores de Decisão Impulsionadas por Gradiente; para $i = 1, 2, ..., M$;

$\gamma_i(x)$ são os pesos dados às estimativas de $h_i(x)$; para $i = 1, 2, ..., M$.

Uma Árvore de Decisão é um modelo em que a função aprendida $h(x)$ é representada por uma estrutura de árvore binária. Os nós internos indicam expressões lógicas relacionadas ao vetor de características x e cada nó externos representa um valor possível para a variável de saída y. Sendo assim, árvores de decisão são de fácil compreensão, pois sua estrutura pode ser interpretada como um conjunto de regras *if-then*, em que a conjunção dos nós internos de cada um de seus ramos representa uma condição para a ocorrência do nó externo que atinge. O algoritmo de aprendizagem deste modelo consiste em partir de uma árvore vazia e buscar recursivamente a variável em x que melhor divide os exemplos de treinamento em relação à y, medindo-se o ganho de informação com base na entropia dos subconjuntos gerados pela divisão até cada nó externo da árvore representar apenas uma saída esperada para y.

Para treinar um modelo GBT por gradiente, precisa-se de um conjunto de dados de treinamento $\{(x_i, y_i)\}_{i=1}^{n}$, uma função de perda diferenciável L e uma quantidade M de Árvores de Decisão Impulsionadas por Gradiente. Inicialmente, treina-se uma Árvore de Decisão inicial $f_0(x) = h_0(x)$ com os dados de treinamento. Em seguida, para cada m de 1 à M, computa-se os erros do modelo anterior $f_{m-1}(x)$, denominado resíduos $r_{i,m}$, de acordo com a Equação 3.11.

$$r_{i,m} = -\frac{\partial\, L(y_i, f_{m-1}(x))}{\partial\, f_{m-1}(x)} \tag{3.11}$$

De posse dos resíduos $r_{i,m}$, treina-se uma Árvore de Decisão Impulsionada por Gradiente $h_m(x)$ utilizando a base de treinamento adaptado $\{(x_i, r_{i,m})\}_{i=1}^{n}$ e computa-se γ_m de acordo com a Equação 3.12, utilizando um algoritmo de gradiente descendente. Por fim, o classificador $f_m(x)$ é dado pela Equação 3.13.

$$\gamma_m = \underset{\gamma \in \mathbb{R}}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, f_{m-1}(x) + \gamma\, h_m(x)) \tag{3.12}$$

$$f_m(x) = f_{m-1}(x) + \gamma_m \, h_m(x) \tag{3.13}$$

Assim, o algoritmo termina com $f_M(x) = h_0(x) + \gamma_1 \, h_1(x) + ... + \gamma_M \, h_M(x) = f(x)$ (Eq 3.10). Este modelo, devido à composição de funções, é mais complexo se comparado à Regressão Logística Binária e outros classificadores. Por este motivo, GBTs por gradiente são mais suscetíveis à sofrerem *overfitting*. No entanto, há estratégias para evitar este problema, tais como, somar termos de regularização à função de custo L ou aplicar técnicas de poda das Árvores de Decisão para controlar a complexidade do modelo.

3.4 Métodos e métricas de validação

Os algoritmos de aprendizagem possuem duas etapas, uma de treinamento, para construir o classificador baseado nos exemplos de treinamento, e outra de validação, para medir os erros do classificador com exemplos não vistos no treinamento e verificar se o algoritmo generalizou bem. Sendo assim, dois métodos de validação são discutidos nesta seção, ambos retirados de Kohavi et al. (1995).

O primeiro, o método simples de validação consiste basicamente em separar a base de dados em dois conjuntos, um de treinamento e outro de validação. Não é obrigatório que os conjuntos de dados tenham o mesmo tamanho, mas precisam ser mutuamente exclusivos. Já o segundo, o método de validação cruzada, consiste em separar os dados em K grupos mutuamente exclusivos e de mesmo tamanho. Em seguida, é reservado um grupo para teste e os outros $K-1$ são utilizados para treinamento do classificador. Isto se repete K vezes, de tal forma que cada subconjunto de dados seja usado uma vez como base de teste. Esta abordagem obtêm resultados mais confiáveis que a validação simples, por aproveitar melhor a base de dados.

Os métodos de validação são utilizados em conjunto com as métricas de desempenho. Enquanto o método de validação é o processo de treinamento e teste, as métricas de desempenho são as medidas de avaliação de um classificador de acordo com os exemplos de teste. Estas métricas medem a similaridade entre as classes reais associadas aos exemplos e as classes estimadas pelo classificador, como é ilustrado na Figura 3.2.

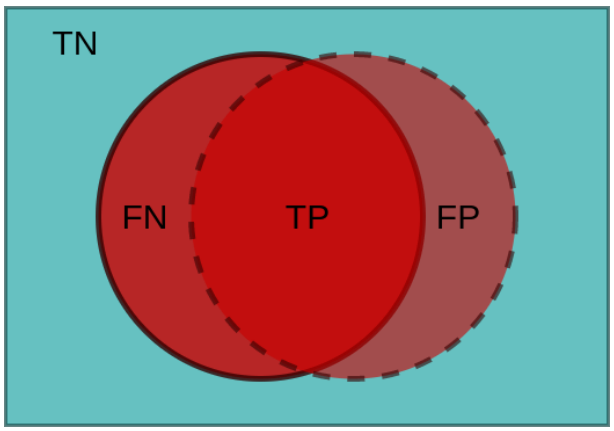

FIGURA 3.2 – Ilustração de um espaço de dados, onde a circunferência contínua representa a classe positiva real e a circunferência segmentada a classe positiva estimada por um classificador. Logo, TP (*True Positive*) e TN (*True Negative*) representam, respectivamente, as quantidades de verdadeiros positivos e verdadeiros negativos, ou seja, que o classificador rotulou corretamente. Enquanto FP (*False Positive*) e FN (False Negative) representam, respectivamente, as quantidades de falsos positivos e falsos negativos. Fonte: Do autor.

Observando-se a Figura 3.2, nota-se os quatro resultados possíveis de uma classificação binária, TP, TN, FP e FN. De posse destes valores, algumas métricas de desempenho podem ser computadas, tais como Precisão (Eq. 3.14), *Recall* (Eq. 3.15), Especificidade (Eq. 3.16) e F_β-score (Eq. 3.17). As três primeiras são usadas para medir o desempenho individual de um classificar, enquanto a última (F_β-score) é comumente utilizada como uma medida de comparação entre dois ou mais classificadores.

A Precisão, Equação 3.14, mede a porcentagem de exemplos corretamente rotulados como positivos pelo classificador. Isto é, observando a Figura 3.2, a Precisão é a porcentagem do círculo segmentado, classe positiva estimada, que faz interseção com o círculo contínuo, classe positiva real. Logo, a Precisão estima a taxa de sucesso de um classificador e, consequentemente, o seu nível de confiabilidade, ou seja, mede as chances de uma resposta positiva deste classificador ser correta.

$$\text{Precisão} = \frac{TP}{TP+FP} \tag{3.14}$$

Já a medida *Recall*, Equação 3.15, enfoca na classe positiva real e mede a porcentagem de exemplos positivos que foram detectados pelo classificador, ou seja, mede a capacidade do classificador identificar um exemplo positivo. Isto é, enquanto a Precisão mede a confiabilidade, a *Recall* mede a sensitividade de um classificador.

$$\text{Recall} = \frac{TP}{TP+FN} \tag{3.15}$$

A Especificidade, Equação 3.16, é a medida equivalente à *Recall* para a classe negativa, ou seja, mede a capacidade de um classificador detectar exemplos da classe negativa.

$$\text{Especificidade} = \frac{TN}{TN+FP} \tag{3.16}$$

A F_β-score, Equação 3.17, é a média harmônica entre a Precisão e o *Recall*, ponderada pelo parâmetro β. Isto se dá pois a Precisão e o *Recall* medem, respectivamente, a confiabilidade e a sensitividade de um classificador e a relevância destas duas métricas variam de acordo com o problema de classificação. Assim, é atribuído ao parâmetro β dos valores 0.5, 1.0 ou 2.0. Se β for igual à 0.5, a Precisão tem maior prioridade, se for igual à 2.0, a *Recall* tem maior prioridade e se for igual à 1.0, ambas as medidas têm a mesma prioridade.

$$F_\beta\text{-score} = (1+\beta^2) * \frac{\text{Precisão} * Recall}{(\beta^2 * \text{Precisão}) + Recall} \tag{3.17}$$

Outra ferramenta importante é a curva ROC (*Receiver Operating Characteristics*). Segundo Fawcett (2006), a curva ROC é uma técnica de visualização, organização e seleção de classificadores baseado em sua performance, comparando as taxas de verdadeiro positivo e falso positivo. Na Figura 3.3 é mostrado um exemplo de curva ROC. Quanto mais a curva ROC se aproximar do canto superior esquerdo do gráfico, melhor o classificador, pois indica a capacidade do classificador acertar mais a classe positiva gerando menos falso positivos.

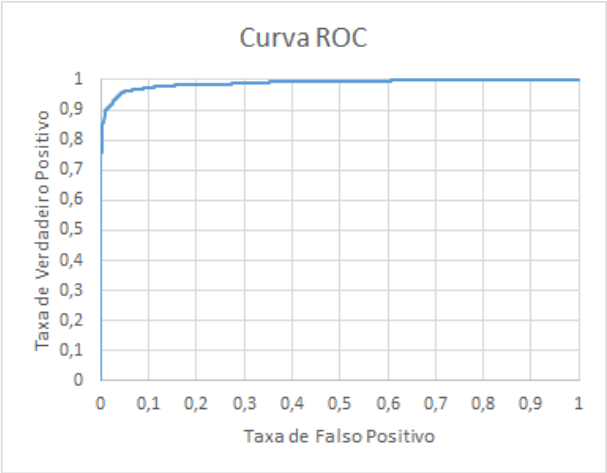

FIGURA 3.3 – Exemplo de uma curva ROC. Fonte: Do autor.

Por último, a métrica área sob a curva ROC (AUC, do inglês *Area Under an ROC Curve*) é uma medida de desempenho que converte a informação gráfica bidimensional da curva ROC em um valor escalar. Como a curva ROC é desenhada em uma área quadrada de lado 1, os valores da métrica AUC variam entre 0 e 1. Com isso, é possível comparar numericamente

os classificadores, ao invés de graficamente. Desta forma, esta métrica permite analisar qual classificador, em relação aos demais, estima um objeto com maior confiança, ou seja, assim como a F_β-score, a AUC é uma medida de comparação entre classificadores.

3.5 Considerações finais

Neste capítulo foi descrita a fundamentação teórica sobre aprendizagem de máquina supervisionada para problemas de classificação binária, apresentando o objetivo geral destes algoritmos de aprendizagem, os conceitos de generalização e consistência, bem como as definições de *underfitting* e *overfitting*. Além disso, foram explicados os dois algoritmos, Regressão Logística e Árvores de Decisão Impulsionadas por Gradiente, os métodos e as métricas de validação para treinar e avaliar os modelos preditivos.

Capítulo 4

DESPACHO DE TÁXIS

Neste capítulo é apresentado o método de despacho de táxis proposto para solucionar o Problema de Atribuição Táxi-Passageiro descrito na Seção 1.2 do Capítulo 1. Para tanto, na Seção 4.1 mostra-se o método de despacho de táxis geral. Em seguida, nas Seções 4.2, 4.3 e 4.4 são explicados respectivamente os critérios de pontuação, funções objetivo e critérios de seleção propostos. Por fim, na Seção 4.5 são dadas as considerações finais do capítulo.

4.1 Método de despacho de táxis geral

Uma visão geral do método de despacho é mostrada no Algoritmo 4.1, onde precisa-se de um critério de pontuação p para computar a matriz de pontuação $S_{m\times n}$, uma função objetivo f para avaliar a matriz de ofertas $X_{m\times n}$ e um critério de seleção c. Em acréscimo, o parâmetro t que determina o intervalo de tempo em segundos que o sistema aguarda para acumular requisições.

Algoritmo 4.1: DESPACHO DE TÁXIS GERAL

Entrada: p, f, c, t
Saída: Y_m

```
início
    aguardar t segundos
    T_m ← buscar conjunto de taxistas disponíveis
    R_n ← buscar conjunto de requisições pendentes
    S_{m×n} ← p(T_m, R_n)
    X_{m×n} ← buscar solução f(S_{m×n})
    realizar as ofertas X_{m×n}
    I_m ← verificar interesse dos taxistas nas ofertas
    Y_m ← c(X_{m×n}, I_m)
    retorna Y_m
fim
```

De forma genérica, passageiros realizam requisições a qualquer momento. Estas requisições são armazenadas no sistema como pendentes até que a próxima iteração do método de despacho seja executada. Quando o tempo t expira as requisições pendentes e os taxistas disponíveis são buscados no sistema, as ofertas são distribuídas e o interesse dos taxistas pelas requisições é verificado. Em seguida, aplica-se o critério de seleção nos casos necessários e realiza-se as atribuições táxi-passageiro. Por fim, os passageiros esperam a chegada do taxista, embarcam e são levados até o seus respectivos destinos.

Vale ressaltar que seria possível fazer uma segmentação dos dados, a fim de reduzir o espaço de busca inerente do problema de distribuição de ofertas. Por exemplo, segmentar por regiões da cidade, por *geohashes* ou por demanda. No entanto, esta estratégia não foi adotada neste trabalho e, portanto, todos os taxistas disponíveis na cidade recebem uma oferta por iteração. Também é possível manter pendentes por algumas iterações as requisições que não foram atribuídas à nenhum taxista, por não terem sido ofertadas a nenhum taxista ou por terem sido recusadas por todos os taxistas que receberam a oferta delas. Esta estratégia foi utilizada em um dos cenários de teste dos experimentos realizados neste trabalho.

4.2 Critérios de pontuação

Dois modelos preditivos foram implementados como critérios de pontuação neste trabalho. O primeiro (Eq. 4.1) estima a probabilidade do taxista t_i aceitar a oferta da requisição r_j, enquanto o segundo (Eq. 4.4) estima a probabilidade de t_i atender r_j. Em ambos os casos treinou-se os modelos preditivos utilizando aprendizagem de máquina supervisionada e as probabilidades estimadas variam entre 0 e 1. Como os modelos preditivos estimam a probabilidade somente de um par (t_i, r_j), para construir a matriz de pontuação $S_{m\times n}$ bastou-se estimar cada elemento $s_{i,j}$ da matriz aplicando os modelos preditivos à cada par (t_i, r_j).

$$s_{i,j} = p_{acc}(t_i,\ r_j) = \text{prob}(y\ =\ \text{`aceitar'} \mid \vec{x_1}) \tag{4.1}$$

O modelo preditivo dado pela Equação 4.1 foi baseado no proposto em Zhang et al. (2017). Assim, p_{acc} mede a probabilidade condicional de t_i aceitar a oferta da requisição r_j. Para tanto, y é uma variável categórica binária, cujos valores possíveis são 'aceitar' ou 'não aceitar'. Já o vetor de características $\vec{x_1}$ representa os dados de entrada do classificador treinado pelo algoritmo de aprendizagem de máquina supervisionada. A construção deste vetor é baseada nas informações disponíveis sobre os taxistas t_i, as requisições r_j e a relação entre eles, tais como tempo estimado de chegada do taxista ao ponto de embarque, tolerâncias e preferências dos

usuários, demanda e oferta, dentre outras.

$$s_{i,j} = p_{can\_t}(t_i,\ r_j) = \text{prob}(y\ =\ \text{‘taxista cancelar’} \mid \vec{x_2}) \tag{4.2}$$

$$s_{i,j} = p_{can\_p}(t_i,\ r_j) = \text{prob}(y\ =\ \text{‘passageiro cancelar’} \mid \vec{x_3}) \tag{4.3}$$

$$s_{i,j} = p_{att}(t_i,\ r_j) = \text{prob}(y\ =\ \text{‘atender’} \mid \vec{x_4}) \tag{4.4}$$

Já o modelo preditivo dado pela Equação 4.4 foi proposto neste trabalho e consiste em incrementar o modelo p_{acc} com outros dois modelos preditivos, p_{can_t} (Eq. 4.2) e p_{can_p} (Eq. 4.3). O primeiro, $pcan\_t$, estima a probabilidade do taxista t_i cancelar a requisição r_j e o segundo, $pcan\_p$, estima a probabilidade do passageiro cancelar a requisição r_j dado que foi atribuída ao taxista t_i. Assim como p_{acc}, os vetores de características $\vec{x_2}$ e $\vec{x_3}$ dos respectivos modelos p_{can_t} e p_{can_p} são baseados nas informações disponíveis sobre os taxistas t_i, as requisições r_j e a relação entre eles. Por outro lado, o vetor de características $\vec{x_4}$ de p_{att} são as saídas dos modelos p_{acc}, p_{can_t} e, p_{can_p}, ou seja, p_{att} computa a probabilidade de t_i atender r_j ponderando as estimativas dos demais modelos preditivos p_{acc}, p_{can_t} e p_{can_p}. Por fim, as variáveis y dos modelos preditivos também são categóricas e seus valores possíveis são “cancelar”e “não cancelar”para p_{can_t} e p_{can_p}, e “atender”e “não atender”para p_{att}.

4.3 Funções objetivo

Nesta pesquisa, duas medidas foram avaliadas como função objetivo para o Problema de Atribuição Táxi-Passageiro (Seção 1.2.1), uma não-linear e outra linear.

4.3.1 Função objetivo não-linear

A função f_{non}, Equação 4.5, foi adaptada da proposta em Zhang et al. (2017). Nesta estratégia busca-se reduzir a probabilidade de fracasso do método de despacho de táxis. Isto se dá pois, os critérios de pontuação estudados retornam valores entre 0 e 1, o que indicam a probabilidade de ocorrer um evento específico, aceite de ofertas ou atendimento de requisições. Com isso, ao multiplicar os complementos destas probabilidades têm-se a probabilidade de todas as ofertas de uma requisição falharem. Assim, soma-se estas probabilidades para obter-se um valor escalar.

$$f_{non} = \sum_{j=1}^{n} \prod_{i=1}^{m} (1 - s_{i,j})^{x_{i,j}} \tag{4.5}$$

Assim como a função objetivo f_{non}, a heurística de otimização, Algoritmo 4.2, utilizada para minimizar esta medida também foi adaptada da proposta em Zhang et al. (2017). Esta estratégia foi dividida em duas etapas. Primeiramente, para cada taxista selecionar a requisição com maior complemento da pontuação. Depois, para cada requisição verificar se a pontuação melhora ao ofertá-la para outros taxistas. Este método não garante uma solução ótima para a função objetivo f_{non}.

Algoritmo 4.2: BUSCAR SOLUÇÃO (NÃO LINEAR)

Entrada: f
Saída: $X_{m \times n}$
início
  $X_{m \times n} \leftarrow$ Matriz de zeros
  para $i = 1, m$ **faça**
    $x_{i,\text{argmin}_{j \in [1,n]}\, 1 - s_{i,j}} \leftarrow 1$
  fim
  $p \leftarrow f(X)$
  para $j = 1, n$ **faça**
    para $\forall u_i \in \{t_i \mid \forall i \in [1,m] \text{ e } x_{i,j} = 0\}$ **faça**
      se *substituir a oferta de* u_i *por* r_j *melhorar* p **então**
        fazer a substituição em X e recalcular $p \leftarrow f(X)$
      fim
    fim
  fim
  retorna $X_{m \times n}$
fim

4.3.2 Função objetivo linear

A medida f_{lin}, Equação 4.6, proposta neste trabalho, é uma linearização da função f_{non}. Para linearizar a função f_{non} aplicou-se logaritmo nas pontuações das requisições. Isto se dá pois as propriedades do logaritmo permite converter a exponenciação em multiplicação e a multiplicação em adição. Assim, adiciona-se um erro na f_{non}, a fim de facilitar a busca pela solução ótima.

$$f_{lin} = \sum_{i=1}^{m} \sum_{j=1}^{n} log(1 - s_{i,j}) x_{i,j} \tag{4.6}$$

A heurística de otimização, Algoritmo 4.3, utilizada para construir a matriz de ofertas $X_{m\times n}$ em relação à função f_{lin}. Com esta estratégia, para cada taxista selecionar a requisição com maior complemento da pontuação. A segunda etapa do Algoritmo 4.2 não é necessário, pois com a primeira etapa garante a solução ótima.

Algoritmo 4.3: BUSCAR SOLUÇÃO (LINEAR)

Entrada: f
Saída: $X_{m\times n}$
início
  $X_{m\times n} \leftarrow$ Matriz de zeros
  para $i = 1, m$ **faça**
    $x_{i,\text{argmin}_{j\in[1,n]}\, 1-s_{i,j}} \leftarrow 1$
  fim
  retorna $X_{m\times n}$
fim

4.4 Critérios de seleção

Nesta seção são apresentados os dois critérios de seleção implementados neste trabalho. O primeiro, c_{can} (Eq. 4.7), escolhe o taxista t_i com a menor probabilidade do passageiro cancelar a requisição r_j. Esta medida foi proposta neste trabalho e busca atender as preferências do passageiro. Isto se dá pois o critério de seleção é aplicado na segunda etapa do problema de atribuição táxi-passageiro, ou seja, depois de verificar o interesse dos taxistas e antes de atribuir o táxi ao passageiro. Também vale ressaltar que os taxistas interessados em atender a requisição não são ofertados ao passageiro. Logo, o critério c_{can} procura prever qual táxi seria escolhido por cada passageiro, com o intuito de satisfazer a ausência desta etapa.

$$c_{can}(X_{m\times n}, A_m) \;=\; \{(t_w,\; r_j) \mid w \;=\; \underset{x_{i,j}a_i=1}{\text{argmin}}\; p_{can\_p}(t_i, r_j)\; e\; i \in [1,m]\}_{j=1}^{n} \tag{4.7}$$

Já o critério de seleção c_{eta} (Eq. 4.8) atribui para cada passageiro o taxista interessado mais próximo do ponto de embarque. Assim, reduz-se o tempo de espera dos passageiros e os gastos com a locomoção de um táxi desocupado, como mencionado. Além disso, o tempo de espera do passageiro (eta) é uma das características mais relevantes para os modelos preditivos, ou seja, uma variável fortemente levada em consideração pelos usuários que avaliam se compensa ou não aceitar ou cancelar uma requisição.

$$c_{eta}(X_{m\times n}, A_m) \;=\; \{(t_w,\; r_j) \mid w \;=\; \underset{x_{i,j}a_i=1}{\operatorname{argmin}}\; eta(t_i, r_j)\; e\; i \in [1,m]\}_{j=1}^{n} \tag{4.8}$$

Note que os critérios de seleção atribuem aos passageiros a somente um taxista que recebeu a oferta da requisição e aceitou atendê-la. Desta forma, as requisições que não foram ofertadas a nenhum taxista ou foram recusadas por todos os taxistas que receberam sua oferta são consideradas não atendidas. No entanto, os passageiros têm uma certa tolerância de espera por uma resposta do sistema de despacho de táxis, ou seja, é possível que estas requisições sejam mantidas no sistema como pendentes e participem das próximas iterações até um limite de tempo. Caso todas as tentativas falharem, considera-se que o tempo da requisição expirou.

4.5 Considerações finais

Neste capítulo foi apresentado o algoritmo geral proposto de despacho de táxis, bem como os critérios de pontuação, funções objetivo e critérios de seleção testados neste trabalho. Para cada uma destas medidas foram abordadas duas estratégias, totalizando um total de 8 combinações possíveis. Vale ressaltar que os critérios de pontuação são modelos preditivos que foram treinados com um algoritmo de aprendizagem supervisionada. Logo, foi necessário avaliar o desempenho dos modelos preditivos utilizando os métodos e métricas de validação discutidos no Capítulo 3. Além disso, em relação as funções objetivo, a f_{non} é não linear e utilizou-se uma heurística de otimização para encontrar uma solução aproximada para o problema de distribuição de ofertas. Já a f_{lin} é uma função linear, cujo método de otimização garante uma solução ótima. Também têm-se que o critério de seleção c_{can} foi proposto para verificar o impacto desta abordagem em comparação com o critério c_{eta}. Por fim, discutiu-se brevemente a possibilidade de manter como pendentes, por um número limitado de tentativas, as requisições que não foram ofertadas aos taxistas ou foram recusadas por eles, a fim de aumentar as chances destas serem atendidas.

Capítulo 5

RESULTADOS

Neste capítulo são apresentados os resultados dos experimentos realizados neste trabalho. Para tanto, divide-se em 5 seções. Na primeira (Seção 5.1) são expostos os recursos utilizados para manipular a base de dados e implementar os experimentos. As Seções 5.2, 5.3 e 5.4 são, respectivamente, sobre o preprocessamento de dados, o treinamento e a validação dos modelos preditivos e os testes simulados com o método de despacho de táxis. Por fim, são dadas as considerações finais do capítulo na Seção 5.5.

5.1 Recursos utilizados

Para o desenvolvimento deste projeto foi escolhida a linguagem de programação *Python* versão 2.7.12. Esta linguagem é multi-paradigma, com sintaxe simplificada e recursos que facilitam a manipulação de dados e o fluxo de trabalho com aprendizagem de máquina. Além disso, foram utilizadas as bibliotecas *pandas* versão 0.19.2 e *numpy* versão 1.15.1 para tratar a base de dados, pois possuem procedimentos que simplificam a leitura e escrita de arquivos, bem como seleção, transformação, junção e geração de dados.

Em relação aos modelos preditivos, usou-se a classe *xgboost.XGBClassifier()* da biblioteca *xgboost* versão 0.7 para analisar as Aŕvores de Decisão Impulsionadas por Gradiente e a classe *sklearn.linear_model.LogisticRegression()* da biblioteca *sklearn* versão 0.19.2 para a Regressão Logística. Em ambas as técnicas os dados de entrada dos modelos preditivos foram normalizados com a métrica *preprocessing.Normalizer()* também da biblioteca *sklearn*.

Para otimizar os parâmetros dos algoritmos de aprendizagem de máquina, separar os dados de treinamento e teste e medir o desempenhos dos modelos preditivos foram utilizados os respectivos módulos *grid_search*, *model_selection* e *metrics* também da biblioteca *sklearn*. Para

tanto, o método *GridSearchCV()* foi utilizado para otimizar os parâmetros em que, dado uma lista de valores para os parâmetros dos algoritmos, testa-se todas as combinações possíveis. E para a separação dos dados utilizou-se o *train_test_split()*, que separa uma porcentagem dos dados para teste e o restante para treinamento. Este método tem a opção de manter a proporção dos exemplos de cada classe tanto na base de treinamento, quanto na base de teste, reduzindo o impacto negativo nos resultados causado pelo desbalanceamento das classes.

Em relação à base de dados, uma parceria foi firmada com a *Easy Taxi*, empresa desenvolvedora e mantenedora do aplicativo de chamada de táxi *online* de mesmo nome, para auxiliar no desenvolvimento desta pesquisa. Assim, disponibilizaram uma base de dados sobre as operações efetuadas na capital de São Paulo no período de 01 de Fevereiro à 15 de Agosto de 2017. Esta base de dados é constituída com cerca de 4,5 milhões de registros de requisições realizadas por passageiros e 212 milhões de exemplos sobre ofertas das requisições aos taxistas.

5.2 Preprocessamento dos dados

Para a realização dos experimentos realizou-se um preprocessamento dos dados. Assim, removeu-se os dados com registros nulos ou inconsistentes, tais como coordenadas geográficas inválidas ou sequências de datas e horários incompatíveis com a realidade, causadas por imprecisão na geolocalização dos aparelhos *smartphones* ou perda de sinal de rede. Em seguida, delimitou-se um perímetro de interesse em torno da cidade, as requisições foram agrupadas em *geohashes* de precisão 5 e removeu-se os registros pertencentes às *geohashes* com menos de 1000 exemplos. Além disso, selecionou-se os registros pertencentes ao período de 01 de Junho à 15 de Agosto de 2017. Isto se dá pois, neste intervalo de tempo, todos os registros continham o atributo *tempo estimado de chegada ao passageiro* preenchidos e fora deste intervalo somente os registros dos taxistas atribuídos às requisições possuíam este atributo. Com isso, resultou-se em 613275 exemplos de requisições e em 18355839 exemplos de ofertas.

Destes dados, extraiu-se as características relevantes para os modelos preditivos p_{acc}, p_{can_t} e p_{can_p}, que estão mostradas na Tabela 5.1. Esta extração de características foi baseada nas informações coletadas na Revisão Bibliográfica (Cap.2) e nas orientações do especialista da *Easy Taxi*. Vale ressaltar que as análises foram restritas aos dados disponibilizados pela empresa e pelas limitações do projeto, como por exemplo, não computou-se o número de requisições repetidas por um passageiro. Esta característica foi relevante para o modelo preditivo p_{can_p}, mas não foi possível simular este dado nos experimentos.

Tabela 5.1: Características selecionadas para os modelos preditivos

Atributo	p_{acc}	p_{can_t}	p_{can_p}
Tempo de chegada ao passageiro estimado (ETA)	*	*	*
Tolerância do taxista ao ETA	*	*	
Diferença entre tol. do taxista e ETA	*	*	
Tolerância do passageiro ao ETA			*
Diferença entre tol. do passageiro e ETA			*
Localização do taxista	*	*	*
Ponto de embarque	*	*	*
Ponto de desembarque			*
Regiões de preferência	*	*	
Preferência pelo tipo de serviço	*	*	*
Qtde de taxistas ativos por região e horário	*	*	*
Qtde de requisições por região e horário	*	*	*
Tipo de serviço	*	*	*
Dia da semana	*	*	*
Horário	*	*	*

Nota-se que os vetores de características dos modelos preditivos p_{acc} e p_{can_t} são idênticos. No entanto, este fato se justifica pois os dados de saída (rótulos) associados aos exemplos de cada um destes modelos preditivos possuem distribuições independentes entre si, ou seja, assume-se que o comportamento dos taxistas em relação ao cancelamento de requisições não tem vínculo algum com os padrões de aceite de ofertas. Logo, os algoritmos de aprendizagem retornam classificadores completamente distintos para estes comportamentos. O mesmo se diz para o modelo preditivo p_{can_p}, que possui um vetor de características similar aos demais modelos, mas os dados de saída associados aos exemplos disponíveis são distinto o suficiente para o algoritmo de aprendizagem resultar em um classificador independente dos demais.

5.3 Treinamento e validação dos modelos preditivos

Os modelos preditivos p_{acc} (probabilidade de aceite de ofertas por taxista), p_{can_t} (probabilidade de cancelamento de requisições por taxista), p_{can_p} (probabilidade de cancelamento de requisições por passageiro) e p_{att} (probabilidade de atendimento de requisições por taxista) foram treinados com Regressão Logística Binária (RLB) e Árvores de Decisão Impulsionadas por Gradiente (do inglês *Extreme Gradient Boosted Trees*) (XGB). Os parâmetros de ambas as técnicas foram otimizados, com validação cruzada com 5 grupos de teste, usando o método *GridSearch()* da bibliotéca *sklearn*. Além disso, para cada modelo preditivo e cada técnica mediu-se as métricas AUC, $F_{0.5}$_score, $F_{2.0}$_score, precisão, *recall* e especificidade, utilizando validação simples. A seguir, são mostradas na Tabela 5.2 as quantidades de dados para treinamento e teste dos modelos preditivos.

Tabela 5.2: Quantidade de dados utilizados para treino e teste dos modelos preditivos

Modelo	Treinamento	Teste
p_{acc}	157498	3149956
p_{can_t}	78184	78185
p_{can_p}	83614	83614
p_{att}	71750	30751

Na Tabela 5.2, nota-se que o modelo p_{acc} foi treinado com uma base quase 20 vezes menor que a base de teste. Isto se dá pois, observou-se que o desempenho do modelo se manteve constante em relação à quantidade de dados de treinamento. Como o tempo de processamento dos dados cresce proporcionalmente com o número de dados, optou-se por utilizar uma quantidade menor de dados para o treinamento deste modelo. Já os modelos p_{can_t} e p_{can_p} foram treinados e testados com a base de dados dividida meio a meio. E o modelo p_{att} foi treinado com 70% dos dados, pois a base de dados é melhor balanceada que as demais bases de dados, como é mostrado na Tabela 5.3.

Tabela 5.3: Frequência de exemplos positivos e negativos de cada operação na base de dados

Operação	Negativo	Positiva
Taxista aceitar oferta	92,2%	07,8%
Taxista cancelar requisição	88,7%	11,3%
Passageiro cancelar requisição	70,8%	29,2%
Requisição ser atendida	38,0%	62,0%

Na Tabela 5.3 é mostrada a frequência de exemplos de cada classe para cada operação estudada, ou seja, a porcentagem de exemplos das classes positiva e negativa das operações de aceite de ofertas, cancelamento e atendimento de requisições. Observa-se que a base de dados sobre requisições atendidas tem mais exemplos da classe positiva do que negativa. No entanto, a quantidade de exemplos da classe negativa nas de aceite de ofertas e cancelamento de requisições é muito maior que da positiva. O desbalanceamento das classes, nesses casos, dificulta o treinamento dos modelos preditivos, pois os algoritmos tendem a classificar negativamente todos os exemplos. Para contornar este problema, com o RLB, optou-se, no treinamento, por ponderar os exemplos da classe positiva, ou seja, multiplicar a função de custo *loss* dos exemplos positivos por um peso. Este peso foi calculado dividindo-se a quantidade de exemplos da classe negativa pela quantidade de exemplo da classe positiva. Já o XGB foi capaz de generalizar os padrões sem adotar esta estratégia de ponderar os exemplos da classe positiva. Na Tabela 5.4 são mostrados os resultados dos modelos preditivos com RLB.

Tabela 5.4: Resultados utilizando Regressão Logística Binária (RLB)

modelo	p_{acc}		p_{can_t}		p_{can_p}		p_{att}	
-	treino	teste	treino	teste	treino	teste	treino	teste
AUC	73.00	72.68	63.34	63.26	66.97	66.50	58.38	58.25
$F_{0.5}$_score	20.63	20.49	20.26	20.21	47.81	47.35	72.47	72.48
$F_{2.0}$_score	45.68	45.33	43.09	43.01	63.11	62.50	67.28	67.67
Precisão	17.44	17.33	17.22	17.17	44.23	43.81	74.39	74.24
Recall	76.73	76.06	69.01	68.93	70.64	69.97	65.71	66.21
Especificidade	69.27	69.29	57.67	57.59	63.30	63.03	51.05	50.28

Observa-se na Tabela 5.4 que as métricas medidas tanto com a base de treinamento, quanto com a base de teste obtiveram, em todos os casos, uma diferença menor do que 1%, ou seja, foram praticamente equivalentes. Isto demonstra a capacidade de generalização do algoritmo avaliado. Ao analisarmos a precisão e o *recall* dos modelos, nota-se claramente que os três primeiros modelos possuem um *recall* pelo menos 26% maior que a precisão. Isto demonstra que o algoritmo gera para estes casos um grande resíduo de falsos positivos para detectar uma taxa superior de verdadeiros positivos. Este impacto da relação entre precisão e *recall* é constatado nas métricas AUC e nas duas F_β*_scores*. Por último, o modelo p_{att} possuem os melhores resultados relacionados à precisão e o *recall*, com aproximadamente 74% e 66%, respectivamente. Logo, as F_β*_scores* também são melhores. Entretanto possui a menor AUC, o que pode ser justificado pela frequência maior de exemplos positivos. Na Tabela 5.5 são mostrados os resultados dos modelos preditivos com XGB.

Tabela 5.5: Resultados utilizando Árvores de Decisão Impulsionadas por Gradiente (XGB)

modelo	p_{acc}		p_{can_t}		p_{can_p}		p_{att}	
-	treino	teste	treino	teste	treino	teste	treino	teste
AUC	84.79	82.75	74.88	73.80	79.62	79.10	67.98	68.35
$F_{0.5}$_score	34.87	33.53	79.82	77.88	79.39	78.66	80.82	81.00
$F_{2.0}$_score	63.07	60.40	55.34	53.21	67.36	66.50	93.88	93.83
Precisão	30.35	29.20	93.63	92.12	84.41	83.77	77.24	77.46
Recall	86.35	82.42	50.20	48.13	64.12	63.24	99.23	99.07
Especificidade	83.23	83.09	99.56	99.47	95.12	94.95	36.73	37.64

Na Tabela 5.5 são mostrados os resultados dos modelos preditivos com Árvores de Decisão Impulsionadas por Gradiente (XGB). Nota-se que, assim como com o RLB, o XGB tem as métricas medidas no treinamento são similares às medidas no teste. Um resultado interessante foi que os modelos preditivos p_{acc} e p_{can_t}, possuem precisão e *recall* complementares, ou seja, enquanto a precisão do p_{acc} é baixa, a precisão do p_{can_t} é alta, e vice-versa em relação ao *recall*. Como consequência disso, o *recall* do p_{att} é praticamente 100%. Já em relação aos modelos p_{can_t} e p_{can_p}, observa-se que atingiram uma precisão alta, com cerca de 92% e 83%, respectivamente. Entretanto, o *recall* destes modelos foi baixo, com cerca de 48% e 63% cada. Além disso, o modelo p_{acc} ainda possui uma precisão muito baixa, com menos de 30%. Isto pode ser justificado pelo grande desbalanceamento das classes deste modelo.

Tabela 5.6: Comparação entre os algoritmos XGB e RLB. Os campos coloridos na Tabela destacam as métricas utilizadas para comparar os algoritmos. Os campos coloridos de verde indicam resultados melhores que os campos coloridos em amarelo. Relembrando que a F_β_score é a média harmônica entre a Precisão e o *Recall*, onde $\beta = 0.5$ valoriza-se a Precisão e $\beta = 2.0$ o *Recall*.

modelo	p_{acc}		p_{can_t}		p_{can_p}		p_{att}	
-	XGB	RLB	XGB	RLB	XGB	RLB	XGB	RLB
AUC	82.75	72.68	73.80	63.26	79.10	66.50	68.35	58.25
$F_{0.5}$_score	33.53	20.49	77.88	20.21	78.66	47.35	81.00	72.48
$F_{2.0}$_score	60.40	45.33	53.21	43.01	66.50	62.50	93.83	67.67
Precisão	29.20	17.33	92.12	17.17	83.77	43.81	77.46	74.24
Recall	82.42	76.06	48.13	68.93	63.24	69.97	99.07	66.21
Especificidade	83.09	69.29	99.47	57.59	94.95	63.03	37.64	50.28

Por último, em relação aos modelos preditivos têm-se a Tabela 5.6, em que são mostrados lado a lado os resultados dos algoritmos XGB e RLB. Claramente percebe-se que a técnica

XGB obteve resultados melhores do que a RLB em todos os casos, como estão em destaque as métricas $F_{0.5}$_score e $F_{2.0}$_score. Os únicos casos em que a técnica XGB obtiveram resultados inferiores foram com a métrica *recall* com os modelos p_{can_t} e p_{can_p} e a especificidade com o modelo p_{att}, devido ao ponderamento das classes com o RLB e as classes balanceadas dos dados de treinamento do p_{att}. No entanto, nos três casos, a precisão dos modelos compensaram tais resultados inferiores.

5.4 Simulação do método de despacho de táxis

O método de despacho de táxis foi simulado usando os dados reais das ofertas de requisições realizadas no período de 23 à 29 de Julho de 2017. Este intervalo equivale a uma semana de dados, de Domingo à Sábado. Estes dados foram selecionados de forma arbitrária, mas teve-se o cuidado de verificar se os dados estavam dentro dos padrões detectados nos dados, ou seja, se não tinham eventos atípicos ou fora do comum de acordo com o restante dos dados. Assim, obteve-se 54010 exemplos de requisições e 8776 taxistas ativos nestes dias. Na Tabela 5.7 são mostradas as quantidades de taxistas ativos e requisições registradas por dia da semana. Nota-se que a quantidade de taxistas foi menor que a quantidade de requisições. No entanto, como as requisições são operações pontuais e os taxistas ficam ativos por horas ao longo do dia, constatou-se que a disponibilidade de taxistas foi suficiente para atender as requisições.

Tabela 5.7: Quantidade de taxistas e requisições por dia da semana

Data	Dia	Qtde taxistas	Qtde requisições
2017-07-23	Domingo	2886	4125
2017-07-24	Segunda-feria	5701	6945
2017-07-25	Terça-feira	6209	9624
2017-07-26	Quarta-feira	6300	10768
2017-07-27	Quinta-feira	6250	9946
2017-07-28	Sexta-feira	6204	8499
2017-07-29	Sábado	3993	4103

Os dados das requisições e dos taxistas foram agrupados em intervalos de 12 segundos, a fim de simular os dados de entrada do método de despacho de táxis, ou seja, os conjuntos de taxistas disponíveis $T_{m\times n}$ e requisições pendentes $R_{m\times n}$. Este intervalo de tempo foi utilizado para o parâmetro t. Assim, cada *slot* de tempo possui os conjuntos $T_{m\times n}$ e $R_{m\times n}$ necessários para a simulação do método de despacho. Vale ressaltar que foram utilizados somente os taxistas

detectados pelo método de despacho de táxis da *Easy Taxi*, pois não foram disponibilizados os dados das localizações dos taxistas ativos.

As simulações consistiam em executar o método de despacho de táxis para cada *slot* de tempo t da semana de dados selecionada para os experimentos. Logo, os vetores de características $\vec{x_1}$, $\vec{x_2}$ e $\vec{x_3}$ dos modelos preditivos p_{acc}, p_{can_1} e p_{can_p} eram extraídos dos conjuntos $T_{m\times n}$ e $R_{m\times n}$ e o método de despacho de táxis executado. Para simular o comportamento dos usuários, treinou-se outro conjunto de modelos preditivos (p'_{acc}, p'_{can_1} e p'_{can_p}). Com isso, de posse das ofertas e das atribuições táxi-passageiro realizadas pelo método de despacho de táxis, foi possível simular quais requisições foram aceitas, quais foram canceladas e, consequentemente, quais foram atendidas. Na Tabela 5.8 são mostradas as métricas dos modelos preditivos utilizados para simular o comportamento dos usuários. O modelo p_{att} não foi treinado, pois foi possível inferir as requisições atendidas com base nas estimativas dos demais modelos.

Tabela 5.8: Métricas dos modelos preditivos utilizados para simular o comportamento dos usuários

modelo	p'_{acc}		p'_{can_1}		p'_{can_p}	
-	treino	teste	treino	teste	treino	teste
AUC	82.93	80.71	74.67	73.77	79.40	78.94
F2-score	54.64	51.97	54.88	53.11	66.29	65.63
Precisão	22.25	21.14	95.24	95.05	85.85	84.64
Recall	85.91	81.79	49.62	47.83	62.72	62.14
Especificidade	79.96	79.63	99.71	99.71	96.09	95.74

Os experimentos foram divididos em três casos. Nos dois primeiros as requisições não ofertadas e as recusadas pelos taxistas foram diretamente consideradas não atendidas, ou seja, não eram mantidas no sistema como pendentes por mais de uma iteração. Já o terceiro cenário as requisições tinham 5 chances para serem atendidas, pois o intervalo de execução t do método de despacho de táxis proposto foi de 12 segundos, totalizando um total de no máximo 1 minuto para o sistema concluir as requisições; caso contrário, o tempo expira e estas requisições eram consideradas não atendidas. No caso 1 foi testada a função objetivo não-linear f_{non} com as 4 combinações possíveis entre os critérios de pontuação p_{acc} e p_{att} e os de seleção c_{can} e c_{eta}. Já no caso 2 foi testada a função objetivo linear f_{lin} também com as 4 combinações. Por último, no caso 3 foi testado somente o método de despacho de táxis com a função objetivo f_{non} e o critério de seleção c_{can} e os dois critérios de pontuação p_{acc} e p_{att}. No total analisou-se 10 configurações distintas para o método de despacho de táxis proposto.

Tabela 5.9: Caso 1 - Resultados com a função objetivo não linear

modelo	p_{acc}, f_{non}, c_{can}		p_{att}, f_{non}, c_{can}	
(%)	**média**	σ	**média**	σ
não ofertadas	3.76	1.57	3.54	1.66
recusadas	0.11	0.03	0.05	0.04
aceitas	96.12	1.60	96.39	1.70
canceladas por taxistas	6.43	0.32	7.23	0.51
canceladas por passageiros	16.15	2.74	18.03	2.08
canceladas total	21.36	2.64	23.71	2.13
atendidas	74.76	4.12	72.68	3.53
modelo	p_{acc}, f_{non}, c_{eta}		p_{att}, f_{non}, c_{eta}	
(%)	**média**	σ	**média**	σ
não ofertadas	3.75	1.55	3.55	1.67
recusadas	0.13	0.06	0.04	0.04
aceitas	96.10	1.61	96.40	1.69
canceladas por taxistas	6.49	0.52	6.92	0.52
canceladas por passageiros	16.77	2.45	18.68	1.79
canceladas total	22.06	2.67	24.14	1.64
atendidas	74.04	4.17	72.25	3.07

De acordo com o que é apresentado na Tabela 5.9, nota-se que a taxa de requisições atendidas, na média, foi maior com o critério de pontuação p_{acc} (74,76%) do que com o critério p_{att} (72,68%), contrariando as expectativas do modelo p_{att} de reduzir as taxas de requisições canceladas por taxistas e passageiros. Além disso, verificou-se que o modelo preditivo p_{acc} obteve a taxa de aceite menor do que o p_{att}. Uma possível explicação para estes resultados é a reutilização da heurística de otimização proposta em Zhang et al. (2017), mesmo trabalho em que também propôs-se o uso do modelo preditivo p_{acc}. Logo, a heurística pode ter favorecido o critério baseado em aceite de ofertas (p_{acc}). Já em relação aos critérios de seleção, constatou-se que o c_{can} obteve uma porcentagem de requisições atendidas levemente superior ao c_{eta}.

Tabela 5.10: Caso 2 - Resultados com a função objetivo linear

modelo	p_{acc}, f_{lin}, c_{can}		p_{att}, f_{lin}, c_{can}	
(%)	**média**	σ	**média**	σ
não ofertadas	4.84	1.12	3.53	1.65
recusadas	1.01	0.29	0.08	0.07
aceitas	94.13	0.95	96.38	1.72
canceladas por taxistas	8.99	1.03	7.93	0.62
canceladas por passageiros	30.40	2.37	19.87	2.38
canceladas total	36.21	2.77	26.08	2.22
atendidas	57.92	2.92	70.29	3.55
modelo	p_{acc}, f_{lin}, c_{eta}		p_{att}, f_{lin}, c_{eta}	
(%)	**média**	σ	**média**	σ
não ofertadas	4.83	1.11	3.55	1.66
recusadas	1.02	0.23	0.05	0.05
aceitas	94.13	1.00	96.39	1.71
canceladas por taxistas	9.27	0.91	7.54	0.66
canceladas por passageiros	30.04	2.52	20.37	2.15
canceladas total	36.10	2.66	26.27	2.08
atendidas	58.02	2.99	70.11	3.32

Por outro lado, como é mostrado na Tabela 5.10, a porcentagem de requisições atendidas do critério de pontuação p_{att} foi cerca de 12% superior ao p_{acc}. Entretanto, o p_{att} com a função objetivo f_{lin} ainda foi cerca de 4% inferior ao p_{acc} com a função objetivo f_{non}. Estes resultados podem ser justificados com a heurística que garante a solução ótima, mas que a linearização da função objetivo adicionou um erro significativo na estimativa da taxa de atendimento. Além disso, neste caso de teste, o modelo p_{att} obteve uma taxa de aceite maior e as taxas de cancelamento menores que o p_{acc}. Também constatou-se que o critério de seleção c_{can} foi superior ao c_{eta} com o critério p_{att}, porém 0,1% inferior ao c_{eta} com o p_{acc}. De maneira geral, o critério de seleção c_{can} obtêm resultados melhor que o c_{eta}.

Tabela 5.11: Caso 3 - Resultados com a função objetivo não linear com repetição de requisições

modelo	p_{acc}, f_{non}, c_{can}		p_{att}, f_{non}, c_{can}	
(%)	**média**	σ	**média**	σ
expiradas	0.11	0.11	0.21	0.16
recusadas	0.03	0.04	0.02	0.01
aceitas	99.85	0.15	99.76	0.17
canceladas por taxistas	7.24	0.82	7.67	0.52
canceladas por passageiros	17.42	3.20	20.15	2.96
canceladas total	23.26	3.59	26.19	2.88
atendidas	76.59	3.72	73.56	3.03

Por último, testou-se o impacto de se manter pendente as requisições não ofertadas ou recusadas pelos taxistas por até 5 tentativas. Neste caso de teste, avaliou-se os critério de pontuação p_{acc} e p_{att} com a função objetivo f_{non} e o critério de seleção c_{can}. Nota-se, na Tabela 5.11, que com as repetições das requisições a taxa de aceite foi de quase 100% e um acréscimo de aproximadamente 2% da taxa de atendimento para ambos os critérios de pontuação.

5.5 Considerações finais

Neste capítulo foram apresentados os resultados obtidos com os experimentos realizados desta pesquisa. Este projeto foi desenvolvimento em *Python* versão 2.7.12. Além disso, uma parceria foi firmada com a *Easy Taxi*, empresa desenvolvedora e mantenedora do aplicativo de chamada de táxi *online* de mesmo nome, que contribuiu com a pesquisa disponibilizando uma base de dados, um especialista e dois computadores.

Para tanto, removeu-se os dados com registros nulos ou inconsistentes, as requisições foram agrupadas em *geohashes* de precisão 5 e removeu-se os registros pertencentes à *geohashes* com menos de 1000 exemplos. Além disso, selecionou-se os registros pertencentes ao período de 01 de Junho à 15 de Agosto de 2017. Com isso, resultou-se em uma tabela de requisições com 613275 exemplos e uma tabela de ofertas com 18355839 exemplos. Destes dados, extraiu-se as características relevantes para os modelos preditivos p_{acc}, p_{can_1} e p_{can_p}, tais como: tolerâncias e preferências dos usuários, demanda e oferta, tempo de chegada ao passageiro (eta). Os modelos preditivos p_{acc}, p_{can_1}, p_{can_p} e p_{att} foram treinados com Regressão Logística Binária (RLB) e Árvores de Decisão Impulsionadas por Gradiente (XGB). Constatou-se claramente que a técnica XGB obteve resultados melhores do que a RLB em todos os casos.

Para testar o método de despacho de táxis proposto selecionou-se arbitrariamente os registros de uma semana de dados, resultando em 54010 exemplos de requisições e 8776 taxistas ativos nestes dias. Os dados das requisições e dos taxistas foram encadeados em intervalos de 12 segundos. Para simular o comportamento dos usuários, treinou-se outro conjunto de modelos preditivos (p'_{acc}, p'_{can_t} e p'_{can_p}). Os experimentos foram divididos em três casos, totalizando 10 configurações distintas para o método de despacho de táxis proposto.

Os resultados demonstraram que o critério de seleção c_{can} obteve uma porcentagem de requisições atendidas levemente superior ao c_{eta}. Já em relação aos modelos preditivos, nota-se que o critério p_{acc} foi mais eficiente que o p_{att} em relação ao atendimento das requisições. Como a heurística busca uma solução aproximada em um subespaço reduzido do espaço de soluções admissíveis, não é possível afirmar que o modelo p_{acc} seja superior ao p_{att}. No entanto, com esta heurística os resultados favorecem o p_{acc}. O modelo preditivo p_{acc} obteve uma taxa de aceite menor do que o p_{att} e o modelo p_{att} obteve taxas de cancelamento por taxistas e passageiros foram menores que o p_{acc}. Por outro lado, a porcentagem de requisições atendidas do critério de pontuação p_{att} foi cerca de 12% superior ao p_{acc}. Entretanto, o p_{att} com a função objetivo f_{lin} ainda foi cerca de 4% inferior ao p_{acc} com a função objetivo f_{non}. Além disso, o modelo p_{att} obteve uma taxa de aceite maior e as taxas de cancelamento menores que o p_{acc}.

Por último, testou-se o impacto de se manter pendente as requisições não ofertadas ou recusadas pelos taxistas por até 5 tentativas. Neste caso de teste, avaliou-se os critério de pontuação p_{acc} e p_{att} com a função objetivo f_{non} e o critério de seleção c_{can}. Notou-se que com as repetições das requisições a taxa de aceite foi de quase 100% e um acréscimo de aproximadamente 2% da taxa de atendimento para ambos os critérios de pontuação.

Capítulo 6

CONCLUSÕES E TRABALHOS FUTUROS

Nesta dissertação de mestrado formulou-se o Problema de Atribuição Táxi-Passageiro como um problema de otimização combinatória para encontrar uma distribuição de ofertas, seguido de um problema de escolha, para remover conflitos de interesse entre os taxistas. Desta forma, propôs-se um método de despacho de táxis genérico para múltiplos passageiros que resolve o problema de atribuição formulado definido por três componentes. O critério de pontuação, que é utilizado para estimar a relevância de cada par táxi-passageiro; a função objetivo, para avaliar a qualidade das distribuições de ofertas; e o critério de seleção, para de fato escolher os pares táxi-passageiro. Assim, foram comparadas duas estratégias para cada um destes componentes, ou seja, oito combinações possíveis. Além disso, testou-se as duas combinações com os melhores resultados com o acréscimo de que as requisições não ofertadas ou recusadas pelos taxistas se mantivessem no sistema por até cinco tentativas, totalizando 10 conjuntos de testes.

Os critérios de pontuação foram modelos preditivos treinados usando Árvores de Decisão Impulsionadas por Gradiente da biblioteca *xgboost* versão 0.7, um algoritmo de aprendizagem de máquina que, de forma geral, constrói um conjunto de árvores de decisão. Esta técnica foi comparada com Regressão Logística, uma solução mais simples e menos efetiva. O primeiro modelo preditivo estimava a probabilidade dos taxistas aceitarem a oferta de uma requisição, enquanto que com o segundo buscou-se estimar a probabilidade dos taxistas atenderem uma requisição. Sabendo que para uma requisição ser atendida precisa ser aceita por um taxista e não ser cancelada pelo taxista ou passageiro, treinou-se outros dois modelos preditivos, um para cancelamento de requisições por taxistas e outro por passageiros. De posse dos três modelos, treinou-se um quarto modelo preditivo que ponderava as estimativas dos três primeiros e estimava a probabilidade de atendimento das requisições. Os resultados obtidos foram, respectivamente, de 82,75% e 68,35% de AUC, além da precisão e do *recall* serem de 29,20% e 82,42%% para a primeira abordagem e 77,46% e 99,07% para a outra.

Duas funções objetivo foram propostas neste trabalho. A primeira foi uma função não-linear baseada na função objetivo citada em Zhang et al. (2017), que garante uma solução aproximada, mas não ótima, e a segunda foi uma linearização da primeira, que garante a solução ótima, mas, devido à linearização, adicionou-se um erro em relação a função objetivo não-linear. A principal diferença entre ambas as estratégias é o método de otimização adotado.

Os resultados demonstraram que cada função objetivo favoreceu um critério de pontuação distinto, ou seja, a não-linear favoreceu o modelo preditivo de aceite de ofertas e a linear favoreceu o de atendimento de requisições. No entanto, de forma geral, a função não-linear com a heurística de otimização aproximada obteve resultados superiores, com cerca de 74% de requisições atendidas com o modelo preditivo de aceite e 72% com o modelo de atendimento, enquanto que com a linear os resultados foram 58% e 70%, respectivamente. A justificativa mais provável é que a heurística de otimização foi ajustada para o modelo preditivo de aceite, ou seja, o subespaço do espaço de soluções admissíveis favorece o primeiro critério de pontuação. Além disso, a linearização da função objetivo também adicionou um erro que degradou o desempenho do método de despacho de táxis.

Já em relação aos critérios de seleção testou-se duas abordagens. A primeira, considerando que o processo de negociação entre os usuários não possui uma etapa para verificar o interesse dos passageiros nos taxistas que aceitaram atendê-los, seleciona o taxista com a menor probabilidade do passageiro cancelar a requisição. Assim, permite que o sistema antecipe a ação do usuário e busque atender as necessidades e preferências do passageiro. Já o segundo critério foi escolher o taxista mais próximo do passageiro. Para tanto, estimou-se o tempo de chegada estimada de cada taxista e optou-se pelo que levasse menos tempo para chegar ao ponto de embarque. Assim, de acordo com os resultados, constatou-se que o critério de seleção baseado na probabilidade do passageiro cancelar a requisição foi cerca de 1% superior ao baseado em tempo estimado de chegada.

Em relação à repetição de requisições não ofertadas e recusadas pelos taxistas, testou-se os dois critérios de pontuação, com a função objetivo não-linear e o critério de seleção baseado em cancelamento, pois estas combinações obtiveram os melhores resultados nos experimentos anteriores. Nestes experimentos, a taxa de requisições aceitas foi praticamente 100% para os dois casos, mas a taxa de cancelamento também aumentou, reduzindo o ganho da taxa de aceite sobre a taxa de atendimento. Como resultado, obteve-se uma taxa de atendimento de 76,59% com o modelo preditivo de aceite de ofertas e 73,56% com o de atendimento de requisições, o que indica, respectivamente, um acréscimo de 1,83% e 0,88% em relação às mesmas combinações sem as repetições.

No futuro, pretende-se estudar a arquitetura do sistema, a fim de estabelecer uma forma mais eficiente e escalável de se estabelecer a comunicação entre os servidores e os usuários, tratar as falhas e exceções, bem como otimizar a eficiência de resposta do sistema e, principalmente, estudar o fluxo dos dados, para simplificar a mineração de dados e desenvolvimento de novas pesquisas. Em relação aos modelos preditivos, faz-se necessário uma extração de características mais refinada, como por exemplo, características de pontos de interesse, padrões recentes de comportamento dos usuários ou o impacto de eventos nos padrões de mobilidade de uma cidade. Constatou-se também que a função objetivo influencia drasticamente na taxa atendimento de requisições. Logo, utilizar um método de otimização para resolver a função objetivo não-linear e aplicar alguma técnica de linearização mais eficiente, que reduza o erro na destorção da função, ou até mesmos a proposta de outras funções objetivo. Por fim, testar o método em cenário real, para remover o viés da simulação nos resultados da pesquisa.

REFERÊNCIAS

BILLHARDT, H. et al. Taxi dispatching strategies with compensations. *Expert Systems with Applications*, Elsevier, v. 122, p. 173–182, 2019.

CHEN, X. et al. Hierarchical data-driven vehicle dispatch and ride-sharing. In: IEEE. *Decision and Control (CDC), 2017 IEEE 56th Annual Conference on*. [S.l.], 2017. p. 4458–4463.

COX, D. R. The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B (Methodological)*, JSTOR, p. 215–242, 1958.

DONG, Y. et al. Revealing new york taxi drivers' operation patterns focusing on the revenue aspect. In: IEEE. *Intelligent Control and Automation (WCICA), 2016 12th World Congress on*. [S.l.], 2016. p. 1052–1057.

FAWCETT, T. An introduction to roc analysis. *Pattern recognition letters*, Elsevier, v. 27, n. 8, p. 861–874, 2006.

FRIEDMAN, J. H. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, JSTOR, p. 1189–1232, 2001.

GAO, G.; XIAO, M.; ZHAO, Z. Optimal multi-taxi dispatch for mobile taxi-hailing systems. In: IEEE. *Parallel Processing (ICPP), 2016 45th International Conference on*. [S.l.], 2016. p. 294–303.

GELDA, R.; JAGANNATHAN, K.; RAINA, G. Taxi dispatches using supply forecasting: A time-series based approach. In: IEEE. *High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), 2016 IEEE 18th International Conference on*. [S.l.], 2016. p. 1333–1340.

GOLPAYEGANI, F. et al. Co-ride: Collaborative preference-based taxi-sharing and taxi-dispatch. In: IEEE. *2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)*. [S.l.], 2018. p. 864–871.

HANNA, J. P. et al. Minimum cost matching for autonomous carsharing. *IFAC-PapersOnLine*, Elsevier, v. 49, n. 15, p. 254–259, 2016.

HOQUE, M. A.; HONG, X.; DIXON, B. Analysis of mobility patterns for urban taxi cabs. In: IEEE. *Computing, Networking and Communications (ICNC), 2012 International Conference on*. [S.l.], 2012. p. 756–760.

KOHAVI, R. et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In: MONTREAL, CANADA. *Ijcai*. [S.l.], 1995. v. 14, n. 2, p. 1137–1145.

KÜMMEL, M.; BUSCH, F.; WANG, D. Z. Taxi dispatching and stable marriage. *Procedia Computer Science*, Elsevier, v. 83, p. 163–170, 2016.

LIU, Y.-W. et al. A parallel genetic algorithm with region division strategy to solve taxi-passenger matching problem. In: IEEE. *Computational Intelligence (SSCI), 2017 IEEE Symposium Series on*. [S.l.], 2017. p. 1–7.

LUXBURG, U. V.; SCHÖLKOPF, B. Statistical learning theory: Models, concepts, and results. In: *Handbook of the History of Logic*. [S.l.]: Elsevier, 2011. v. 10, p. 651–706.

MA, J. et al. Designing optimal autonomous vehicle sharing and reservation systems: A linear programming approach. *Transportation Research Part C: Emerging Technologies*, Elsevier, v. 84, p. 124–141, 2017.

MEENAKSHI, S.; SENTHILKUMAR, R. Efficient taxi dispatching system in distributed environment. In: IEEE. *Information, Communication, Instrumentation and Control (ICICIC), 2017 International Conference on*. [S.l.], 2017. p. 1–6.

MENG, Q. et al. A novel taxi dispatch system integrating a multi-customer strategy and genetic network programming. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, Fuji Technology Press Ltd., v. 14, n. 5, p. 442–452, 2010.

MIAO, F. et al. Data-driven robust taxi dispatch under demand uncertainties. *arXiv preprint arXiv:1603.06263*, 2016.

NGO, M. N.; SEOW, K. T.; WONG, K. W. Fuzzy linear assignment problem: An approach to vehicle fleet deployment. In: IEEE. *Fuzzy Systems, 2004. Proceedings. 2004 IEEE International Conference on*. [S.l.], 2004. v. 2, p. 1197–1202.

ODA, T.; JOE-WONG, C. Movi: A model-free approach to dynamic fleet management. *arXiv preprint arXiv:1804.04758*, 2018.

QIAO, C. et al. An efficient dispatch and decision-making model for taxi-booking service. In: IEEE. *Ubiquitous Intelligence and Computing and 2015 IEEE 12th Intl Conf on Autonomic and Trusted Computing and 2015 IEEE 15th Intl Conf on Scalable Computing and Communications and Its Associated Workshops (UIC-ATC-ScalCom), 2015 IEEE 12th Intl Conf on*. [S.l.], 2015. p. 392–398.

RAMEZANI, M.; NOURINEJAD, M. Dynamic modeling and control of taxi services in large-scale urban networks: A macroscopic approach. *Transportation Research Procedia*, Elsevier, v. 23, p. 41–60, 2017.

SCHREIECK, M. et al. A matching algorithm for dynamic ridesharing. *Transportation Research Procedia*, Elsevier, v. 19, p. 272–285, 2016.

SEOW, K. T.; DANG, N. H.; LEE, D.-H. A collaborative multiagent taxi-dispatch system. *IEEE Transactions on Automation Science and Engineering*, IEEE, v. 7, n. 3, p. 607–616, 2010.

SHRIVASTAVA, M. et al. Taxi despatch: a fuzzy rule approach. In: IEEE. *Intelligent Transportation System, 1997. ITSC'97., IEEE Conference on*. [S.l.], 1997. p. 978–982.

SITU, X. et al. A parallel ant colony system based on region decomposition for taxi-passenger matching. In: IEEE. *Evolutionary Computation (CEC), 2017 IEEE Congress on*. [S.l.], 2017. p. 960–967.

VERMA, S. K.; VO, H. T. A predictive taxi dispatching system for improved user satisfaction and taxi utilization. In: IEEE. *Smart City/SocialCom/SustainCom (SmartCity), 2015 IEEE International Conference on*. [S.l.], 2015. p. 175–182.

WONG, K.; BELL, M. G. The optimal dispatching of taxis under congestion: A rolling horizon approach. *Journal of advanced transportation*, Wiley Online Library, v. 40, n. 2, p. 203–220, 2006.

YUAN, W. et al. An unlicensed taxi identification model based on big data analysis. *IEEE Transactions on Intelligent Transportation Systems*, IEEE, v. 17, n. 6, p. 1703–1713, 2016.

ZHANG, L. et al. Exploiting taxi demand hotspots based on vehicular big data analytics. In: IEEE. *Vehicular Technology Conference (VTC-Fall), 2016 IEEE 84th*. [S.l.], 2016. p. 1–5.

ZHANG, L. et al. A taxi order dispatch model based on combinatorial optimization. In: ACM. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. [S.l.], 2017. p. 2151–2159.

ZHAO, B. et al. Preference-aware task assignment in on-demand taxi dispatching: An online stable matching approach. 2019.

GLOSSÁRIO

AUC – *Area Under an ROC curve*

CHR – *Controle de Horizonte Recuado*

DQN – *Deep Q-network*

ETA – *Tempo Estimado de Chegada ao Passageiro*

FN – *False Negative*

FP – *False Positive*

GBT – *Gradient Boosted Trees*

GD-DBSCAN – *Grid and Kd-Tree for DBSCAN*

KNN – *K-Nearest Neighbors*

OSM-KIID – *Online Stable Matching under Known Identical Independent Distributions*

RLB – *Regressão Logística Binária*

ROC – *Receiver Operating Characteristics*

SCRAM – *Scalable Collision-avoiding Role Assignment with Minimal-makespam*

SVM – *Support Vector Machine*

TN – *True Negative*

TP – *True Positive*

XGB – *Extreme Gradient Boosted Trees*