UNIVERSIDADE FEDERAL FLUMINENSE
POLO UNIVERSITÁRIO DE RIO DAS OSTRAS
INSTITUTO DE CIÊNCIA E TECNOLOGIA
CIÊNCIA DA COMPUTAÇÃO

DAVID LUCAS CAMPOS FAIAL

# PREVISÃO DE DEMANDA DE TÁXI BASEADA EM APRENDIZADO DE MÁQUINA ONLINE

Rio das Ostras
2019

DAVID LUCAS CAMPOS FAIAL

# PREVISÃO DE DEMANDA DE TÁXI BASEADA EM APRENDIZADO DE MÁQUINA ONLINE

Trabalho de conclusão de curso apresentado ao curso de Ciência da Computação da Universidade Federal Fluminense, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientadora:
Prof.ª Dr.ª, Flávia Cristina Bernardini

Rio das Ostras
2019

Ficha catalográfica automática - SDC/BRO
Gerada com informações fornecidas pelo autor

C198p Campos faial, David Lucas
Previsão de Demanda de Táxi Baseada em Aprendizado de Máquina Online / David Lucas Campos faial ; Flávia Cristina Bernardini, orientadora. Niterói, 2019.
44 f. : il.

Trabalho de Conclusão de Curso (Graduação em Ciência da Computação)-Universidade Federal Fluminense, Instituto de Ciência e Tecnologia, Rio das Ostras, 2019.

1. Aprendizado de Máquina. 2. Previsão de Demanda. 3. Cidades Inteligentes. 4. Aprendizado em Fluxo de Dados. 5. Produção intelectual. I. Bernardini, Flávia Cristina, orientadora. II. Universidade Federal Fluminense. Instituto de Ciência e Tecnologia. III. Título.

CDD -

Bibliotecária responsável: Monnique São Paio de Azeredo Esteves Veiga - CRB7/6921

DAVID LUCAS CAMPOS FAIAL

# PREVISÃO DE DEMANDA DE TÁXI BASEADA EM APRENDIZADO DE MÁQUINA ONLINE

Trabalho de conclusão de curso apresentado ao curso de Ciência da Computação da Universidade Federal Fluminense, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Aprovada em 19 de julho de 2019.

BANCA EXAMINADORA

_____

Profª Drª Flávia Cristina Bernardini (Orientadora) – IC/UFF

_____

Prof. Dr. Dalessandro Soares Vianna – ICT/UFF

_____

Prof. Dr. Edwin Benito Mitacc Meza – ICT/UFF

Rio das Ostras
2019

# Lista de tabelas

Tabela 1 – Diferenças entre processamentos de dados tradicionais e em fluxo. . . . . 14
Tabela 2 – Conjuntos de Dados relativos a corridas de táxis amarelos. . . . . 27
Tabela 3 – Exemplo do mapeamento segmentado por hora. . . . . 31
Tabela 4 – Exemplo do mapeamento segmentado por dia da semana. . . . . 32
Tabela 5 – Resultados do aprendizado online, utilizando mudança de conceito, variando o tamanho da janela de avaliação. Modelo treinado com todos os distritos. . . . . 34
Tabela 6 – Influência causada pela variação da taxa de aprendizado do modelo online. Modelo treinado com todos os distritos . . . . . 34
Tabela 7 – Performance do modelo online em cada distrito, separadamente. . . . . 35
Tabela 8 – Resultado do aprendizado em lote, variando o learning rate. Todos os distritos considerados . . . . . 35
Tabela 9 – Resultado do aprendizado em lote, para cada distrito separadamente. . . . . 36

# Lista de ilustrações


Figura 1 – Diagrama de resumo do fluxo do trabalho. . . . . 9
Figura 2 – Ilustração da interação entre diversos tipos de sensores e tipos de dados enviados por eles. Fonte: https://bit.ly/2loOaAB . . . . . 11
Figura 3 – Diagrama com alta abstração, mostrando os passos de criação de um modelo em lote vs modelo online. Elaborado pelo autor. . . . . 13
Figura 4 – Ilustração gráfica dos tipos de mudança de conceito (*concept drift*). Fonte: (LEMAIRE; SALPERWYCK; BONDU, 2015) . . . . . 16
Figura 5 – Exemplo de uma árvore de decisão e suas partes: (a) nós internos; (b) nós folhas representando um rótulo cada um, ou seja, uma classe . . . . . 20
Figura 6 – Perceptron Artificial. . . . . 21
Figura 7 – Neurônio biológico . . . . . 21
Figura 8 – Rede de Perceptrons multicamadas. . . . . 22
Figura 9 – Ilustração do funcionamento da softmax. Cada input é mapeado para uma probabilidade. . . . . 23
Figura 10 – Brooklyn, vista geral. . . . . 27
Figura 11 – Brooklyn detalhado. . . . . 27
Figura 12 – Número de embarques por dia (Junho de 2014). . . . . 30
Figura 13 – Bloco residencial - Brooklyn . . . . . 30
Figura 14 – Bloco vizinho ao Madison Square Garden, em Manhattan . . . . . 30
Figura 15 – Número de embarques por dia da semana. . . . . 31
Figura 16 – Erro do algoritmo online vs algoritmo em lote. Modelo treinado para todos os distritos. Quanto menor, melhor. . . . . 37
Figura 17 – Erro do algoritmo online vs algoritmo em lote, por distrito. Modelo treinado para cada distrito separadamente. Em azul: modelo online. Em laranja: modelo em lote . . . . . 37

# Sumário


**1 INTRODUÇÃO . . . . . 8**
**1.1 Objetivo e Metodologia . . . . . 9**

**2 REFERENCIAL TEÓRICO E TRABALHOS RELACIONADOS . . . 10**
**2.1 Cidades Inteligentes . . . . . 10**
**2.2 Aprendizado de Máquina . . . . . 11**
**2.3 Aprendizado de Máquina em Fluxo de Dados . . . . . 13**
**2.4 Mudança de Conceito . . . . . 14**
**2.5 Classificação de Modelos com detecção de mudança de conceito 16**
2.5.1 Conjuntos Adaptativos . . . . . 16
**2.6 Detectores de Mudança de Conceito . . . . . 17**
2.6.1 DDM - Drift Detection Method . . . . . 17
2.6.2 EDDM - Early Drift Detection Method . . . . . 18
2.6.3 HDDM - Hoeffding's based Drift Detection Method . . . . . 18
2.6.4 FHDDM - Fast Hoeffding's based Drift Detection Method . . . . . 18
**2.7 Exemplos de Algoritmos de Aprendizado . . . . . 19**
2.7.1 Árvores de Decisão . . . . . 19
2.7.2 Perceptron . . . . . 21
2.7.3 Rede Neural de Perceptrons Multicamada . . . . . 22
**2.8 Trabalhos Relacionados . . . . . 24**

**3 DESENVOLVIMENTO DA METODOLOGIA . . . . . 26**
**3.1 Natureza dos dados do problema . . . . . 26**
**3.2 Pré-Processamento dos dados geográficos . . . . . 27**
**3.3 Análise dos dados . . . . . 30**
**3.4 Ferramentas e Métodos . . . . . 32**

**4 RESULTADOS . . . . . 33**
**4.1 Modelo Online . . . . . 34**
**4.2 Modelo em Lote . . . . . 35**
**4.3 Visualização dos resultados . . . . . 37**

**5 CONCLUSÕES E TRABALHOS FUTUROS . . . . . 38**

**REFERÊNCIAS . . . . . 39**

# Resumo

Sistemas inteligentes de apoio a transportes tem causado um grande impacto na mobilidade urbana das pessoas. Em grandes centros urbanos, serviços de transporte ainda precisam de maneiras de otimizar o fornecimento de veículos em determinadas áreas, de acordo com a demanda em cada uma delas. Uma distribuição otimizada de serviços de táxis por demanda pode fazer parte de um plano inteligente de mobilidade urbana, causando impactos diretos no tráfego urbano, melhoria na acessibilidade ao transporte, maior segurança nos pontos de espera dos táxis pela diminuição do tempo de espera, redução da tarifa de transporte etc. Atualmente muitos sensores instalados em veículos geram informações em tempo real, as quais não são aproveitadas para processamento e geração de informação e valor. Este trabalho propõe um modelo de Previsão de demanda de táxis usando aprendizado de máquina online e detecção de mudança de conceito, usando fluxo contínuo de dados. Uma fonte de dados real disponibilizada na plataforma aberta de Nova Iorque alimenta um modelo de aprendizado online e outro em lote, criados com o auxílio da ferramenta MOA - *Massive Online Analysis* - um framework para mineração de fluxos de dados. O modelo online se mostra promissor na previsão de demanda de táxis, atingindo precisão de 78%, ao contrário do modelo em lote, que não apresentou boa performance. Apesar de utilizar dados de uma cidade específica, a metodologia e resultados deste trabalho podem contribuir para uma gestão mais proativa de demanda em outras cidades. **Palavras-chave**: Mobilidade Urbana. Mineração de Dados Online. Big Data. Algoritmo Preditivo.

# Abstract

Intelligent transport support systems have had a major impact on people's urban mobility. In large urban centers, transportation services still need ways to optimize vehicle supply in certain areas, according to the demand in each of them. Optimized distribution of on-demand taxi services can be part of an intelligent urban mobility plan, causing direct impacts on urban traffic, improving transport accessibility, improving safety at taxi standpoints by reduced waiting times, reduce transportation fare etc. Many vehicle-mounted sensors currently generate real-time information that is not used for processing and generating information with value. This paper proposes a Taxi Demand Forecasting model using online machine learning and concept drift detection usingdata stream. A real data source made available on the New York open platform feeds one online and one batch learning model, both created using the Massive Online Analysis (MOA) tool - a framework for data stream mining. The online model shows promising results in forecasting taxi demand, reaching 78% accuracy, unlike the batch model, which did not perform well. Despite using data from a specific city, the methodology and results of this work can contribute to a more proactive demand management in other cities.**Keywords**: Urban Mobility. Online Data Mining. Big Data. Predictive Algorithm.

# 1 Introdução

Existem aproximadamente 13.437 táxis na cidade de Nova Iorque prontos para uma corrida. Será que é fácil conseguir um táxi na saída de um dia cansativo do trabalho ou após a saída de um evento que ocorre frequentemente em determinado lugar? Muitas vezes o desbalanceamento entre demanda e oferta de veículos de transporte causa transtornos muitas vezes evitáveis para o motorista e para o possível passageiro de uma corrida.

Aplicativos de táxi como Lyft, Cabify e 99táxi começaram a transformar o cenário do transporte, mudando drasticamente a forma de mobilidade urbana da população (HOFFMANN; IPEIROTIS; SUNDARARAJAN, 2016), (RAYLE et al., 2014). Essas novas experiências de transporte começaram a crescer e ganhar espaço nos últimos anos, conseguindo preencher lacunas que o transporte público e os serviços tradicionais de táxi deixavam para trás. Em 2011, com falhas na linha vermelha de metrô em Boston e problemas com ônibus que demoravam a chegar, "deixando 35.000 pessoas esperando no frio", o Uber aproveitou para oferecer corridas grátis e promoções relâmpago nas áreas afetadas pelos problemas citados (UBER, 2018). O serviço dos tradicionais táxis amarelos de Nova Iorque precisa adotar uma estratégia mais imperativa e proativa, afim de se manter competitivo com os aplicativos de transporte, maximizando os ganhos dos motoristas, oferecendo um serviço melhor e mais atrativo para os passageiros, e possivelmente reduzindo o congestionamento.

Um tipo de problema em específico ainda chama atenção quanto à demanda de táxis em certas regiões: regiões com alta demanda e baixo número de carros disponíveis, assim como regiões com muitos motoristas ociosos e baixa demanda de passageiros. (HUANG; POWELL, 2012) apresentou um framework para detecção do desequilíbrio em regiões de serviço de táxis, uma possível solução para passageiros que esperam tempo demais em pontos de táxi e motoristas que faturam cada vez menos, mesmo com demandas em outras regiões. (LIU et al., 2017) usou um algoritmo genético paralelo (PGA) com estratégia de divisão de regiões para atacar o problema entre demanda e oferta de passageiros e táxis. O problema de desequilíbrio entre demanda e oferta de táxis foi abordado em (ZHAO; KHRYASHCHEV; VO, 2017).

Atualmente, a TLC (Taxi and Limousine Commission) é a agência responsável por manter uma comunicação direta com os taxistas na cidade de Nova Ioque. A agência alerta os motoristas de onde existem carência de táxis, mas apenas depois desta carência ter sido identificada e aparente. O modelo de previsão de demanda deste trabalho poderia ajudar em uma estratégia de comunicação proativa, em vez de uma estratégia reativa. O conhecimento de onde táxis ficarão disponíveis ou de que áreas terão maior demanda de passageiros pode ajudar a resolver grandes desbalanceamentos entre demanda/procura, motoristas ociosos, maior gasto de tempo e combustível, maior custo para buscar passageiro do que para levá-lo ao seu destino, elevado tempo de espera de passageiros, engarrafamentos etc.

## 1.1 Objetivo e Metodologia

Este trabalho tem por objetivo medir a contribuição do aprendizado online na previsão de demanda de táxis, usando uma abordagem com fluxo contínuo de dados sob mudança de conceito (*concept drift*). A constante mudança na demanda de táxis observada em (ZHAO; KHRYASHCHEV; VO, 2017) pode ser melhor representada por um modelo que perceba mudanças no contexto do ambiente do qual os dados são provenientes. As mudanças podem se originar de diversas características dos distritos (bairros) ou de eventos que acontecem em cada um deles.

Dois conjuntos de dados (datasets) foram utilizados neste trabalho, ambos descritos mais detalhadamente no capítulo 3. O primeiro contém todos os embarques de táxis da cidade de Nova Iorque, do mês de Julho de 2014. O segundo conjunto contém as estruturas geográficas detalhadas do terreno da cidade, em arquivos *shape, extensão shp*. Afim de associar cada coordenada dos embarques ao seu respectivo bloco (quadra) no conjunto geográfico, um primeiro mapeamento foi feito. Logo em seguida, foram realizados três agrupamentos (segmentações) da frequência de embarque: embarques por hora, por dia da semana e por dia do mês. Os três conjuntos resultantes dos agrupamentos serviram de *input* para o algoritmo preditor usado neste trabalho e descrito na seção 3.4. O algoritmo preditor foi construído com base em uma rede neural, usando duas abordagens: 1. Abordagem com aprendizagem em fluxo de dados e com detecção de mudança de conceito e 2. Abordagem de aprendizado em lote.

A imagem abaixo resume o fluxo do trabalho:

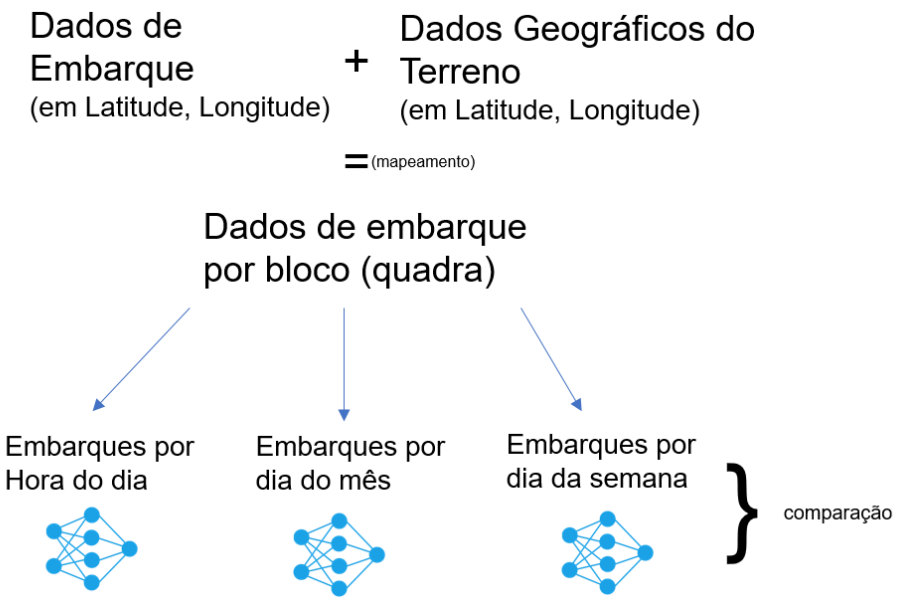


Figura 1 – Diagrama de resumo do fluxo do trabalho.

# 2 Referencial Teórico e Trabalhos Relacionados

## 2.1 Cidades Inteligentes

O termo cidade inteligente se refere ao uso de tecnologias para coletar, analisar e integrar informações provenientes de diversos sistemas e sensores nos núcleos urbanos. Ao passo que dados são coletados e processados, respostas cada vez mais inteligentes (em formas de serviços) podem surgir para necessidades cotidianas das pessoas, e também para atividades comerciais e industriais. Em cidades inteligentes, aplicações produzem dados em alta velocidade em contextos e ambientes que mudam com o passar do tempo, produzindo o que chamamos de fluxo contínuo de dados. Por causa desta nova característica de disponibilidade dos dados, novas necessidades surgiram para o aprendizado de máquina. Algumas delas são: (i) novas técnicas de amostragem de dados, (ii) novos algoritmos de aprendizado, (iii) técnicas para processar os dados na velocidade em que são disponibilizados e capacidade de esquecer dados desatualizados (FACELI et al., 2011).

A abordagem usual para tratamento dos dados gerados era, até pouco tempo: (i) Escolha de um conjunto de dados finito e (ii) Geração de um modelo estático. Esta estratégia pode ser ótima para previsão dos dados daqui a alguns meses, dias ou algumas horas; mas a performance do modelo preditivo pode começar a cair drasticamente conforme o tempo passa e necessidade e costume das pessoas mudam, sendo necessário um novo treinamento. Sistemas que mantinham armazenamento de exemplos de treinamento, treinados com conjuntos de dados limitados e modelos estatísticos estáticos não estavam preparados para o alto volume de dados gerados pelos sensores modernos (dispositivos de IoT, por exemplo), logo, não conseguiam manter um modelo preditivo atualizado que consistisse com o estado atual dos cenários da natureza a qual estava tentando modelar.

Estudos feitos por (GANTZ; REINSEL, 2012) estimam que até 2012 apenas 0.5% dos dados disponíveis no "mundo digital"foi analisado, de um total de 2.8 Zettabytes. Deste total, estima-se que apenas 3% dos dados estão rotulados, o que pode enfatizar ainda mais os três principais problemas do Big Data (velocidade, variedade e volume) e trazer algumas perguntas à tona: Qual o tamanho do desafio ao tentar extrair informação útil de tamanha quantidade de dados gerados? Quanto tempo se dever gastar rotulando ou minerando esses dados para análise? Seria melhor investir em algoritmos mais inteligentes, em dados mais estruturados ou um pouco dos dois? Certamente, afim de encontrar respostas para estas perguntas, a abordagem de coleta e processamento de dados utilizada até pouco tempo precisaria de alguns ajustes e novidades para se encaixar no ecossistema moderno das cidades inteligentes (CI).

Mais do que simples grandes repositórios de dados, existem bases que crescem de modo

ilimitado, onde milhões de registros são criados diariamente. Sensores distribuídos em redes de fornecimento de energia, água, tratamento de esgoto e GPS de veículos produzem dados a todo momento, e fornecem serviços ao ambiente em que estão instalados. Os serviços oferecidos podem ser classificados como produtores de dados, consumidores de dados ou uma combinação dos dois. Por exemplo, um conjunto de sensores que marcam quantidades de vagas livres nas ruas disponibilizam um serviço produtor de dados, neste caso o número de vagas disponíveis e talvez a sinalização de onde elas se encontram. Por um outro lado um carro equipado com sensores que buscam por uma vaga podem consumir esse serviço, logo, os dados que ele disponibiliza, podendo ser classificado então como consumidor de dados. O terceiro exemplo pode vir de um táxi equipado com um serviço que avisa outros veículos das condições do trânsito, alertando sobre possíveis perigos e rotas a serem evitadas.

O número de sensores nestas novas redes pode aumentar com o tempo, além da possibilidade de eles mandarem dados em velocidades, escalas e granularidades diferentes.

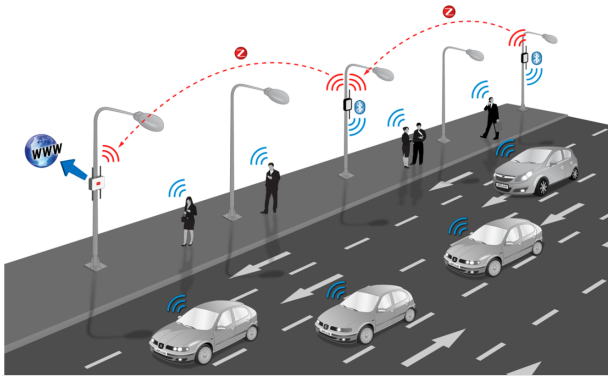


Figura 2 – Ilustração da interação entre diversos tipos de sensores e tipos de dados enviados por eles. Fonte: https://bit.ly/2loOaAB

Entre os principais objetivos das cidades inteligentes, a construção de um sistema de transporte eficiente pode influenciar significativamente a vida dos cidadãos. Com isso a distribuição de veículos pela malha de transporte urbana é um dos principais fatores a se considerar para garantir qualidade na mobilidade das pessoas e diminuir a ocorrência de alguns problemas como: transportes lotados, trânsito caótico, pontos lotados com pessoas esperando por condução, pouco transporte circulando em zonas de alta demanda, acidentes, alta emissão de poluentes etc.

## 2.2 Aprendizado de Máquina

A ideia de se criar máquinas inteligentes sempre esteve presente no desenvolvimento humano; hoje, com o nome de Inteligência Artificial (IA), expõe um enorme potencial criando sistemas autônomos com comportamentos inteligentes e interação com o ambiente ao redor. A inteligência artificial é construída a partir de alguns componentes, entre eles: aprendizado de máquina, processamento de linguagem natural, processamento de imagens e robótica. É

comum atualmente a presença de modelos computacionais utilizados para detecção de câncer de mama, fraudes em cartões de crédito, projeção de vendas, consumo e detecção de atividade suspeitas em um ambiente. Todos esses modelos podem ser derivados de um dos subcampos da IA utilizado neste trabalho: o Aprendizado de Máquina (ML).

(SIMON, 2013) definiu o aprendizado de máquina como "o campo de estudo que dá à computadores a habilidade de aprender sem ser explicitamente programado", com origens na otimização matemática e estatística, cobrindo técnicas de previsão, prescrição, diagnostico e planejamento. Os problemas de ML são tipicamente classificados em três categorias amplas, dependendo da natureza do *input* ou *feedback* disponível para o sistema. Essas três categorias são:

- Aprendizagem supervisionada: As instâncias (exemplos) apresentadas ao modelo de aprendizado têm "etiquetas" disponíveis em cada uma. O objetivo é aprender as regras ou modelo que mapeia o input aos outputs, ou etiquetas.

- Aprendizagem não-supervisionada: Nenhum rótulo ou etiqueta é apresentada junto com as instancias de entrada. Nesse tipo de aprendizado, o modelo usado pode descobrir padrões nos dados e agrupá-los de acordo com características encontradas.

- Aprendizado por reforço: Um algoritmo interage com um ambiente dinâmico no qual ele precisa atingir um objetivo ou maximizar a performance em um conjunto de passos. O programa é penalizado caso ele tome decisões ruins, ou recompensado caso tome decisões boas.

Entre o aprendizado supervisionado e o não supervisionado está o semi-supervisionado, onde o conjunto de dados possui instâncias (geralmente muitas) sem rótulo.

O aprendizado de máquina possui ainda uma categorização referente ao tipo do problema a ser tratado. Se o objetivo do problema é relacionar cada instância de entrada a um elemento de um conjunto discreto, então esse é um problema de classificação. Um exemplo de um problema de classificação seria classificação de e-mails entre "spam" ou "não spam", duas classes discretas. Se o *output* consiste no mapeamento de uma instância à um elemento de um conjunto contínuo, então esse problema é de regressão. Um exemplo de problema de regressão seria a previsão do preço de casas baseada na características do imóvel. Já um problema de clusterização (ou agrupamento) consiste no agrupamento das instâncias em grupos diferentes, baseado nas características de cada instância analisada. Comumente um problema de agrupamento é tratado com aprendizado não-supervisionado (BISHOP, 2006), ou inversamente, uma abordagem não-supervisionada é comumente derivada de um problema de agrupamento.

## 2.3 Aprendizado de Máquina em Fluxo de Dados

O aprendizado de máquina em lote coloca todo o conjunto de dados em uma estrutura de dados para a realização da aprendizagem, e a primeira previsão só pode ser feita quando todas os exemplos de aprendizagem tiverem sido analisados. Ao contrario deste, o aprendizado online (em fluxo de dados) atualiza o modelo a cada $n$ exemplos de treinamento visitados; o conjunto desses $n$ exemplos é chamado de janela (window), e $n$ por sua vez é o tamanho da janela de aprendizado.

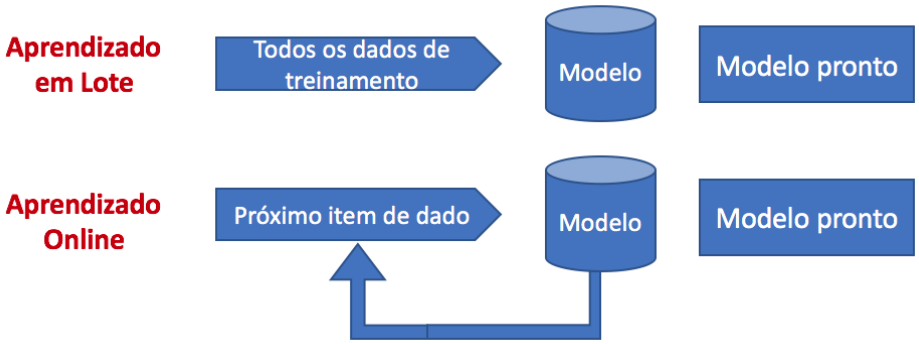


Figura 3 – Diagrama com alta abstração, mostrando os passos de criação de um modelo em lote vs modelo online. Elaborado pelo autor.

Vamos tomar como exemplo a aplicação de um algoritmo de agrupamento (*clustering*) em uma matriz $M$ de dados de fluxo contínuo, para a criação de um modelo preditivo. Para o aprendizado deste modelo é preciso transpor $M$, o que não será possível visto que o operador de transposição é um operador bloqueante (DANIEL, 2002), ou seja, precisa da matriz de entrada completa antes de executar quaisquer ações sobre os dados.

De acordo com (DOMINGOS; HULTEN, 2003), há propriedades desejáveis nesse tipo de aprendizado:

- Curto tempo de acesso a cada instância de treinamento, devido ao grande número de instâncias que chegam a todo momento. Um longo tempo de acesso pode levar o algoritmo à falha;
- Quantidade de memória usada é limitada, independentemente da quantidade de exemplos;
- Única leitura do exemplo de treinamento para a construção do modelo, pois:
  - Pode não haver tempo de revisitar antigos exemplos;
  - Exemplos podem não estar disponíveis em armazenamento secundário para consultas futuras;

- Disponibilidade de um modelo de dados utilizável a qualquer momento de uma requisição, uma vez que, em oposição ao aprendizado em lote, o modelo é atualizado a cada exemplo de aprendizado visitado;
- Em um cenário ideal, deve ser capaz de produzir um modelo equivalente ao modelo que seria produzido pelo algoritmo correspondente no aprendizado em lote;
- O modelo deve ser capaz de ser atualizado mesmo quando houver mudança de conceito, ou seja, mesmo quando houver mudanças nas relações entre variáveis de entrada e de saída.

A tabela 1 mostra algumas diferenças entre processamento de dados normais e em fluxo:

Tabela 1 – Diferenças entre processamentos de dados tradicionais e em fluxo.

| | Processamento Tradicional | Processamento em Fluxo |
|---|---|---|
| Número de visitas | Multiplas | Única |
| Tempo de Processamento | Ilimitado | Limitado |
| Uso de Memória | Ilimitado | Limitado |
| Tipo de Resultado | Preciso | Aproximado |

## 2.4 Mudança de Conceito

Um problema ao se modelar dados reais é que o relacionamento entre as variáveis pode mudar com o tempo, tornando os modelos usados para análise rapidamente obsoletos. Em aprendizagem de máquina o fenômeno que compreende a mudança na distribuição de probabilidades dos dados é conhecido como *Concept Drift* (WIDMER; KUBAT, 1996), ou Mudança de Conceito em português. Neste trabalho, os dias da semana, período do mês ou condições climáticas podem fazer com que as regras usadas no modelo de previsão mudem, tornando o aprendizado de máquina mais desafiador.

Com o passar do tempo o contexto no qual o aprendizado é feito muda. Atualmente a entrega de uma encomenda em três semanas é considerada demorada, quando a algum tempo atrás era comum o mesmo tempo de espera. Um tempo considerado fresco no Rio Grande do Sul pode ser inverno para os cariocas. Uma mudança de conceito, contexto ou ambiente gera impactos diretos ao preditor utilizado, por isso, um modelo sensível a estas alterações e resistente à ruídos - muitas vezes confundidos com o primeiro - é considerado um modelo robusto.

A mudança de conceito pode acontecer de muitas formas, sendo mais fácil rastrear quando se dá de forma temporal, fazendo com que dados coletados em intervalos específicos

de tempo revelem um padrão de mudança nos dados. Algumas formas de mudança nos dados incluem:

- Mudança brusca e repentina do relacionamento;
- Mudança suave, de forma temporal;
- Mudança cíclica, onde não se sabe quando um determinado padrão voltará a se repetir;
- Mudança incremental.

Cada tipo de mudança é baseada em um tipo de configuração. Considere a partir deste momento duas possíveis origens (fontes) de dados: A e B.

O primeiro tipo de mudança é a *Mudança Repentina, ou Brusca*, quando em um momento do tempo t uma fonte de dados A é repentinamente substituída por uma fonte de dados B.

Uma *Mudança Gradual, ou Suave* é outro tipo encontrada na literatura. Ela ocorre de duas formas, a primeira quando há inicialmente duas fontes de dados presentes na leitura das instâncias (NARASIMHAMURTHY; KUNCHEVA, 2007), (WIDMER; KUBAT, 1996). Com o tempo a probabilidade da amostra vir de A diminui, enquanto a probabilidade de leitura de instâncias de B aumenta.

A segunda forma de mudança gradual dos dados ocorre quando a diferença entre as duas fontes A e B é muito pequena, tanto que só é percebida conforme o preditor aprende com o tempo. (DELANY et al., 2005), (TSYMBAL et al., 2008), (BAENA-GARCıA et al., 2005). Esta forma de mudança gradual é também chamada de *Mudança Incremental*

Existe outro grande tipo de mudança. Chamada de *Mudança de Contexto Recorrente*, ela acontece quando contextos que estiveram ativos anteriormente tornam a aparecer. Se difere de sazonalidade, pois o período de um suposto ciclo não é conhecido.

Comumente em trabalhos de aprendizagem com fluxo de dados, o fluxo é infinito, logo, pode haver um enorme número de padrões de mudanças relacionadas a troca de contexto. Acima foram definidas as principais mudanças de conceito, e a figura 4 ilustra essas mudanças, com dados unidimensionais.

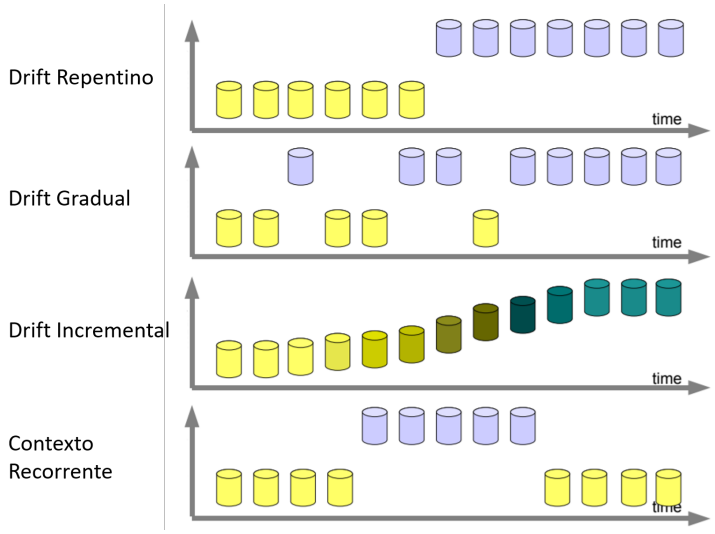


Figura 4 – Ilustração gráfica dos tipos de mudança de conceito (*concept drift*). Fonte: (LEMAIRE; SALPERWYCK; BONDU, 2015)

## 2.5 Classificação de Modelos com detecção de mudança de conceito

Desde que (SCHLIMMER; GRANGER, 1986) propôs o termo "concept drift"pela primeira vez em 1986, o tema ganhou relevância e outros trabalhos surgiram, Houveram alguns picos de publicações sobre o tema em determinados períodos no passado. Um primeiro estava relacionado a um problema de aprendizado de máquina com mudança de contexto (WIDMER, 1998). Já a partir de 2007, uma série de trabalhos ligados ao uso de streaming de dados começou a aparecer: (BIFET, 2009), (NISHIDA, 2008), (WIDYANTORO, 2003), (CASTILLO, 2008).

Os algoritmos de aprendizado podem ser divididos em dois grupos distintos, diferenciados a partir da adaptatividade à mudança de conceito estar "ligada"ou não. O primeiro grupo é formado por algoritmos baseados em gatilho, que recebem um sinal quando uma uma mudança do modelo é necessária, devido à mudança de conceito. O segundo, é constituído por algoritmos evolutivos, os quais, ao contrário do primeiro grupo, não mantém uma relação explicita entre a evolução dos dados e a construção do modelo e geralmente não conseguem detectar mudanças. Cada algoritmo do segundo grupo guarda um conjunto de modelos alternativos durante seu funcionamento, e os seleciona de acordo com o momento temporal e performance atual.

### 2.5.1 Conjuntos Adaptativos

A técnica mais comum para lidar com a mudança de conceito é o conjunto classificador (*classification ensemble*). Saídas de classificação de vários modelos são combinadas para obter uma decisão final. As regras de combinação ou seleção são geralmente chamadas de regras de fusão. Elas são criadas a partir de pesos atribuídos para saída de cada classificador, indicando

a "contribuição"deste determinado classificador para o conjunto e para a classificação final. O peso da saída de cada classificador depende de como este classificador performou historicamente (KOLTER; MALOOF, 2007), (STANLEY, 2003a), (STREET; KIM, 2001), (WANG et al., 2003), (Karnick et al., 2008), (BECKER; ARIAS, 2007), ou de estimativas de performance dos próprios classificadores. Geralmente a melhor estratégia para combinar classificadores é ter um classificador treinado para cado tipo de conceito diferente, preparando o modelo para as mudanças de contexto que possam ocorrer.

Existem grupos de conjuntos classificadores usados para detecção de mudança que dependem do algoritmo base de classificação e os que independem dele. Alguns estudos do primeiro grupo: (TSYMBAL et al., 2008), (STANLEY, 2003b), (STREET; KIM, 2001), (WANG et al., 2003), (SCHOLZ; KLINKENBERG, 2007), (ZHANG; ZHU; SHI, 2008). Há também conjuntos que estão ligados ao algoritmo base. Neles a formação dos conjuntos classificadores e suas regras dependem do classificador base. Alguns estudos com: SVM - *Support Vector Machine* (KLINKENBERG; JOACHIMS, 2000), (KLINKENBERG, 2004), Perceptrons (RAUDYS; MITASIUNAS, 2007), kNN - k-Nearest Neighbours (LAW; ZANIOLO, 2005).

## 2.6 Detectores de Mudança de Conceito

Devido à realidade que este trabalho se propõe a modelar, a abordagem precisa envolver modelos adaptativos. Os métodos apresentados trabalham com dois tipos de sinalização (ou alarme): *warning* e *drift* (ATTAR et al., 2012). O alarme do tipo drift sinaliza que de fato ocorreu uma mudança de conceito, ao contrário do warning, o qual dá um alerta ao classificador, o qual por sua vez cria uma nova instância de si e a mantém em paralelo com o a instância antiga. Se o nível de drift for alcançado, a instância antiga do classificador é exluída e a nova mantida como atual.Esta sessão apresenta os principais detectores de mudança de conceito apresentados por (HIDALGO, 2017).

### 2.6.1 DDM - Drift Detection Method

O Drift Detection Method (GAMA et al., 2004) assume que, de acordo com o modelo PAC (Probably Approximately Correct Learning) (MITCHELL, 1997), se um determinado algoritmo preditor estiver recebendo entradas provenientes de uma distribuição estacionária, a taxa de erro deste algoritmo decai quando o número de exemplos aumenta. A partir desta análise, caso os erros de um classificador comecem a subir significativamente, uma mudança na distribuição de probabilidades dos dados pode estar ocorrendo, prejudicando o aprendizado do preditor, e sugerindo que o estado do modelo atual de aprendizado pode não ser mais o ideal.

O algoritmo de detecção DDM gerencia duas variáveis durante o treinamento: $\mathrm{p_{min}}$ e $\mathrm{s_{min}}$, onde as variáveis $\mathrm{p}$ e $\mathrm{s}$ representam respectivamente a taxa de erro e o desvio padrão do algoritmo. Ao se iniciar, os valores das duas variáveis são propositalmente muito altos, e a cada instância $\mathrm{i}$ avaliada, a verificação $\mathrm{p_i + s_i < p_{min} + s_{min}}$ é executada. Caso a verificação seja

verdadeira, $p_{min}$ e $s_{min}$ recebem os valores de $p_i + s_i$. Como mencionado anteriormente, os algoritmos de detecção de mudança trabalham com os níveis de alarme *warning* e *drift*, sendo neste caso do DDM: *warning* quando $p_i + s_i \geqslant p_{min} + w.s_{min}$ e *drift* quando $p_i + s_i \geqslant p_{min} + d.s_{min}$, onde $w$ e $d$ são parâmetros configuráveis pelo usuário, sendo seus valores padrão iguais a 2 e 3, respectivamente. O método DDM também mantém controle sobre uma variável $n$, que guarda o número de instâncias mínimas que devem ser observadas antes que as comparações acima sejam realizadas.

### 2.6.2 EDDM - Early Drift Detection Method

O EDDM é semelhante ao DDM, e foi criado com o objetivo de melhoria em comparação ao algoritmo anterior. Ao contrário do DDM, em vez de considerar o número medio de erros do classificador em aprendizado, ele considera as distancias entre os erros. O EDDM calcula a distância média entre os erros $p'_i$ e seu desvio padrão $s'_i$. As variáveis $p'_{max}$ e $s'_{max}$ são mantidas afim de se realizar comparações, e inicializadas com valores bem baixos. A seguinte comparações é feita pelo método na chegada de cada erro vindo do classificador: $p'_i + 2.s'_i > p'_{max} + 2.s'_{max}$. Se a comparação for verdadeira $p'_{max}$ recebe o valor de $p'_i$ e $s'_{max}$ recebe o valor de $s'_i$.

O EDDM atinge o nível de warning quando $p'_i + 2.s'_i / p'_{max} + 2.s'_{max} < \alpha$, para o nível de drift quando $p'_i + 2.s'_i / p'_{max} + 2.s'_{max} < \beta$. Os valores padrão de para $\alpha$ e $\beta$ são respectivamente 0,95 e 0,90.

### 2.6.3 HDDM - Hoeffding's based Drift Detection Method

(Frías-Blanco et al., 2015) apresentou o HDDM como um grupo de métodosque tem por objetivo sinalizar uma mudança de conceito quando uma variação significativa é detectada nas métricas de desempenho. Duas versões dos testes são:

- **ATest** - Um teste estatístico que detecta mudanças na média da população, e por conseguinte permite monitorar a diferença entre as médias abordadas do teste utilizando uma sequência de variáveis aleatórias. O método tem um melhor comportamento na detecção de mudanças de conceito abruptas.

- **WTest** - Teste mais indicado para mudanças de conceitos graduais, usando médias móveis ponderadas, onde os valores mais recentes do fluxo de dados têm mais peso que os valores mais antigos.

### 2.6.4 FHDDM - Fast Hoeffding's based Drift Detection Method

O FHDDM foi proposto para tentar reduzir a taxa de falsos positivos e falsos negativos na detecção de mudança de conceito. O algoritmo desliza uma janela com um tamanho de $n =$

200 (valor padrão) nos resultados da classificação. Posteriormente, insere um 1 na janela se o resultado da previsão for verdadeiro, 0 caso contrário.

## 2.7 Exemplos de Algoritmos de Aprendizado

### 2.7.1 Árvores de Decisão

Um modelo baseado em árvores de decisão pode cobrir tanto a classificação quanto a regressão. Na análise de decisão, uma árvore de decisão pode ser usada para representar visualmente e explicitamente as decisões e a tomada de decisões. Com o nome já diz, é um modelo de decisões em forma de árvore, amplamente usado no aprendizado de máquina. A árvore de decisão é uma forma de representação muito popular devido a sua simplicidade e transparência (ROKACH et al., 2014). Trata-se de uma representação autoexplicativa, não havendo a necessidade de ser um especialista para entendê-la e podendo inclusive lidar com dados multidimensionais (HAN; KAMBER; PEI, 2012).

As etapas de aprendizado de classificação de árvores de decisão são simples e rápidas, e em geral têm boa acurácia, dependendo dos dados e problemas disponíveis. A indução de algoritmos de árvore de decisão são usadas em diversas áreas de aplicação, como na medicina, manufatura e produção, análise financeira e astronomia. Uma árvore de decisão é composta por:

- Nós, os quais representam um teste em um valor de atributo;
- Ramos, os quais representam os resultados dos testes dos nós;
- Folhas, as quais representam um rótulo final, uma classe resultante.

O critério de onde serão feitas as divisões dos nós de uma árvore de decisão vem a partir de um atributo chamado *Ganho de Informação*, representado aqui pela letra G, o qual quantifica a "impureza"do nó, de acordo com o atributo escolhido para a divisão deste nó. Se um nó N foi o escolhido para sofrer uma divisão utilizando um atributo A, então significa que o G deste nó N é **menor** que o G dos nós candidatos. Um G pequeno significa que o nó escolhido minimiza a informação necessária para classificar os exemplos que virão posteriormente, ou seja, que a informação em N é mais pura. Como não é objetivo deste trabalho discutir matematicamente os conceitos apresentados nesta seção, a equação abaixo apresenta brevemente o cálculo do ganho de informação de um nó N:

$$\mathtt{Ganho}(A) = \mathtt{Info}_{\text{original}}(D) - \mathtt{Info}_{\text{depois}}(D) \tag{2.1}$$

Onde:

- $\mathtt{Info}_{\text{original}}(D)$: é a informação necessária para classificar um exemplo de dados (tupla de dados) de um conjunto de dados D antes da divisão do nó, no estado atual da árvore;

- $\mathtt{Info}_{\text{depois}}(\mathtt{D})$: é a informação necessária para classificar um exemplo de dados (tupla de dados) de um conjunto de dados D depois da divisão do nó;
- D: Conjunto de dados;
- A: Atributo utilizado para fazer a divisão do nó.

Dois algoritmos conhecidos para classificação são o C4.5 e seu predecessor, o ID3. Ambos os algoritmos utilizam o ganho de informação como heurística de divisão dos nós.

A figura 5 exibe os testes que são realizados nos nós internos da árvore, utilizando um atributo, representado em (a). Cada nó folha representa um resultado de classe do atributo classe em questão, representado por (b). Considerando o exemplo exibido na árvore de decisão abaixo, o atributo classe indica se uma criança poderá brincar de acordo com o tempo, podendo expelir as seguintes classes como saída: S (sim), N (não) e T (talvez). Dessa forma, por exemplo, se o tempo estiver chuvoso e houver relâmpagos, a árvore de decisão indica que a criança não poderá brincar.

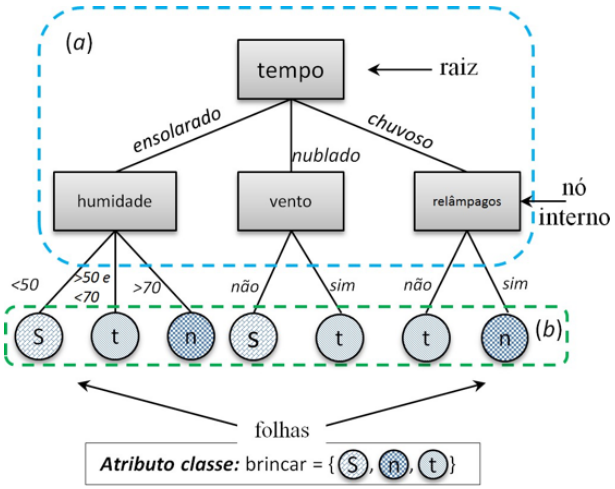


Figura 5 – Exemplo de uma árvore de decisão e suas partes: (a) nós internos; (b) nós folhas representando um rótulo cada um, ou seja, uma classe

Ao utilizar aprendizado em lote, os dados são todos carregados para a memória de uma vez, para então serem feitas as etapas de leitura e análise. Esse processo constrói um modelo de árvore estático de forma gulosa (YANG; FONG; SI, 2012) . Quando novos dados são adicionados ao conjunto analisado, todos os dados (os novos e os anteriormente existentes) são recarregados para a atualização do modelo. No contexto de fluxo de dados as árvores são construídas de forma incremental, se adaptando conforme a chegada e o processamento de novos dados.

### 2.7.2 Perceptron

O perceptron foi um modelo desenvolvido por (ROSENBLATT, 1958) na década de 50, inspirado no modelo neural biológico, humano. Ele recebe entradas $x_1$,$x_2$,..$x_n$ e produz uma saída binária: 0 ou 1, inspirado no ato de disparar ou não um impulso elétrico de um neurônio biológico. Abaixo uma representação simples e resumida de um perceptron (neurônio) artificial e de um neurônio biológico:

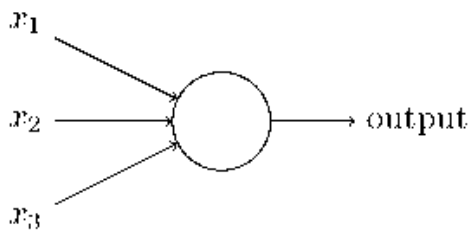


Figura 6 – Perceptron Artificial.

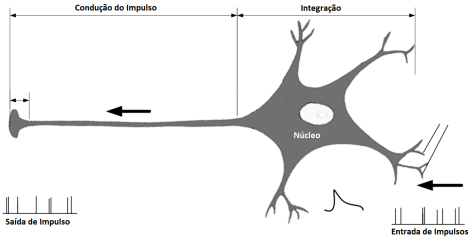


Figura 7 – Neurônio biológico

Indo um pouco além, ao receber inputs, o perceptron utiliza pesos $w_1$,$w_2$,...,$w_n$ em cada um dos inputs, com o objetivo de expressar a importância de cada uma das entradas para a saída. A saída do perceptron é determinada assim por dois fatores principais: a soma ponderada das entradas com os pesos $\sum w.x$, e do resultado dessa soma ser menor ou maior que algum valor limite T. Matematicamente, o modelo básico:

$$\text{output} = \begin{cases} 0, & \text{se } \sum_j w_j x_j \leqslant \text{ threshold} \\ 1, & \text{se } \sum_j w_j x_j > \text{ threshold} \end{cases} \tag{2.2}$$

Um exemplo seria a utilização do perceptron para determinar se uma casa será vendida ou não baseando-se nas características (atributos) do imóvel. As entradas podem ser o numero de quartos, numero total de cômodos, tamanho do terreno, área construída e idade do imóvel.

- $x_1$ = numero de quartos;
- $x_2$ = numero total de cômodos;
- $x_3$ = tamanho do terreno;
- $x_4$ = área construída;
- $x_5$ = idade do imóvel

Uma imobiliária decide alimentar esse perceptron com um histórico de venda de imóveis, contendo todas as características acima, bem como valor final que o imóvel foi vendido (rótulo). Os pesos do perceptron se ajustarão conforme o perceptron for sendo alimentado desse conjunto de treinamento, podendo ser observado posteriormente. Ao final do treinamento, pode se ter algo como mostrado no modelo abaixo:

$$3x_1 + 2x_2 + 9x_3 + x_4 + x_5 \tag{2.3}$$

O peso $w_3$, do tamanho do terreno, é destacadamente maior que os outros, significando que o perceptron treinado pela equipe de TI da imobiliária priorizará o tamanho do terreno para predizer se a casa será vendida. O raciocínio se repete para os outros atributos.

### 2.7.3 Rede Neural de Perceptrons Multicamada

Um perceptron sozinho é bastante limitado quanto ao seu potencial de tomada de decisão, logo, uma rede de perceptrons interligados deve ter muito mais a oferecer quanto a gama de decisões a serem tomadas, recomendações a serem feitas, trabalhar com diferentes tipos de input e output.

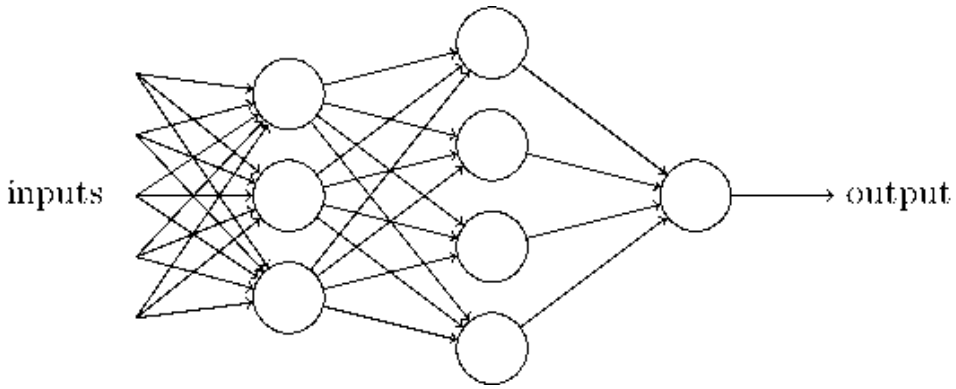


Figura 8 – Rede de Perceptrons multicamadas.

Uma rede de perceptrons (ou neurônios) é construída com ligações entre neurônios entre camadas, nos modelos mais comuns, ou ainda intra-camadas em outros modelos A saída de um neurônio serve a entrada dos outros ao qual ele estiver ligado. É importante ressaltar que cada neurônio na rede acima continua tendo uma única saída; a representação de várias saídas na imagem serve somente para deixar claro que a saída de um neurônio pode servir de entrada para vários outros.

Para adaptar a notação matemática para uma rede, a condição da soma ponderada $\sum w.\mathrm{x} >$ T pode ser escrita em forma de um produto vetorial entre os dois vetores, W e X, não esquecendo do valor limite, threshold, ou bias. O modelo fica representado então por:

$$\mathrm{output} = \begin{cases} 0, & \text{se } w.\mathrm{x} + \mathrm{b} \leqslant 0 \\ 1, & \text{se } w.\mathrm{x} + \mathrm{b} > 0 \end{cases} \tag{2.4}$$

A arquitetura pode ser também representada por uma matriz W = $[w_{\mathrm{ij}}]$, onde $w_{\mathrm{ij}}$ denota o peso da conexão do neurônio i com o j; não havendo esta se $w_{\mathrm{ij}}$ = 0.

Funções de ativação são geralmente usadas em redes neurais para computar a soma ponderada das entradas e do bias, decidindo se um neurônio será disparado ou não, além de introduzir propriedades não-lineares nas redes. A função principal dela é converter o sinal de input de um neurônio de uma rede neural em um sinal de output, o qual é usado na camada seguinte da rede. Uma função de ativação também pode ser conhecida como função de transferência. Alguns tipos de funções de ativação são mais usados e conhecidos. Entre eles:

- **Função Step:** Também conhecida como função step binária, boa para usar com classificação binária, pois seu valor é: 0 se $0 > x$ ou 1 se $x \geqslant 0$. A desvantagem desta função é que ela não pode ser usada para problemas de multi-classificação.

- **Função Linear:** Seu modelo é $Y = a * x$, ideal para classificações simples, onde é desejável que a interpretabilidade seja alcaçada facilmente, pois seu gráfico é uma linha reta com inclinação igual a $a$.

- **Função Sigmoid:** Também chamada de função logística. O gráfico é em forma de S, e o modelo matemático é descrito por:

$$\alpha = \frac{1}{1 + e^{-z}} \tag{2.5}$$

  Esta função sigmoid é continuamente diferenciável, fazendo com o que as derivadas usadas no cálculo do gradiente sejam realizadas sem mais problemas. É uma função não-linear, logo, o output dela também não é.

- **Função ReLU:** A função ReLU tem valor 0 se $x < 0$ e $x$ se $x \geqslant 0$. Comparada com a função sigmoid, converge mais rapidamente para um mínimo ou máximo.

- **Função Softmax:** Ao contrário de outras funções que produzem um único output, a softmax pode ser usada para construir redes neurais que consegue classificar mais de uma classe, ao invés de uma classificação binária. Por esse motivo, ela é a função de ativação utilizada no modelo deste trabalho. O modelo matemático da softmax é:

$$\alpha(x_j) = \frac{e^{x_j}}{\sum_i e^{x_i}} \tag{2.6}$$

  Como output, a softwax aponta as probabilidas de cada saída:

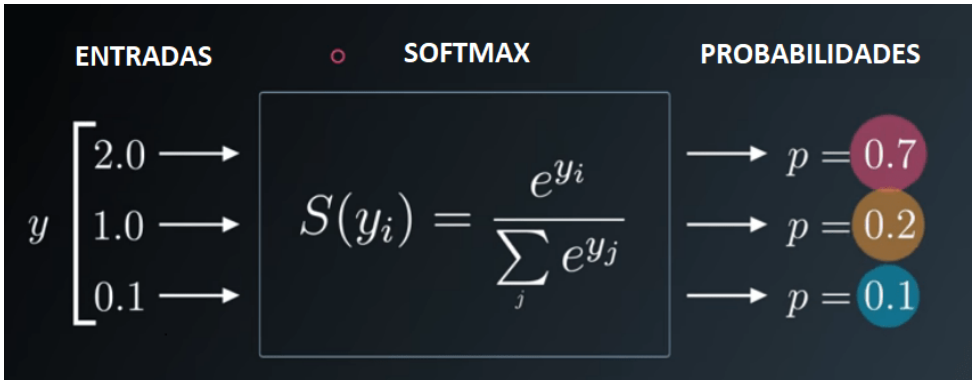


Figura 9 – Ilustração do funcionamento da softmax. Cada input é mapeado para uma probabilidade.

## 2.8 Trabalhos Relacionados

(SONG et al., 2010) foi pioneiro ao definir o conceito de *previsibilidade máxima*($\Pi^{max}$), ao analizar o grau de previsibilidade do comportamento humano, mais especificamente sua mobilidade. Para este trabalho, a previsibilidade de uma região será definida pelo potencial (grau) de previsão de demanda de táxis dessa região. Para conseguir selecionar um preditor com melhor performance, balanceando precisão e tempo de computação, será utilizado o conceito de previsibilidade máxima $\Pi^{max}$. Considerando uma sequência de demandas de táxis $S_n$, onde S é o nome da sequência e $n$ é o número de demandas de táxis desta sequência, $\Pi^{max}$ é definida de acordo "bagunça"ou "ordem"desta sequência de demandas, ou seja, de acordo com a entropia de $S_n$.

A previsibilidade máxima consegue mostrar o grau de correlação entre demandas de táxi na sequência $S_n$, medindo a regularidade das demandas tanto em eventos cotidianos (e.g. inicio e fim de expedientes) quanto em eventos específicos ou inesperados. Dada a $\Pi^{max}$ para um bloco (quadra), $\Pi^{max}$ é o limite máximo de previsibilidade que um algoritmo preditivo pode obter em tal região, não importa o quão bom este algoritmo seja. Um exemplo seria uma região com $\Pi^{max}$ = 0.4, o que indica que em 40% do tempo esta região tem demanda previsível, e 60% do tempo a demanda parece ser aleatória.

(ZHAO; KHRYASHCHEV; VO, 2017) Usou quatro algoritmos preditivos para medir a previsibilidade (previsibilidade) da demanda: Preditor de Markov (um método probabilístico), Lempel-Ziv-Welch(método baseado em sequência), ARIMA(um método série-temporal) e Rede Neural (método baseado em aprendizagem de máquina). Os quatro algoritmos foram utilizados em lote, onde o preditor é gerado a partir do aprendizado do conjunto inteiro de dados; em oposição ao aprendizado online.

(COVIENSKY, 2017) usou modelos preditivos baseados em árvore para prever demanda horária de táxis no aeroporto de Laguardia. O trabalho também apresenta uma rede neural com acurácia de aproximadamente 50 por cento na previsão de demanda. Ao final foi produzida uma aplicação web que permitia os usuários usassem o modelo treinado para fazer pequenas predições de demanda para táxis no aeroporto. Apesar dos resultados, o ambiente proporcionado no aeroporto fazia com que muitos passageiros ficassem em fila esperando os táxis para transporte. O problema disso é que a base disponibilizada pela plataforma de dados de Nova Iorque considera embarques apenas de passageiros que efetivamente entram nos táxis, e não daqueles que também demandam o veículo e esperam nas filas do aeroporto.

(TONG et al., 2017)discute um modelo espaço-temporal para previsão de demanda por unidade de tempo, em regiões de grande demanda. É proposto um modelo de regressão linear n-dimensional apelidado de LinUOTD. O trabalho de (KAMGA; YAZICI; SINGHAL, 2013) analisa as mudanças temporais e relacionadas com o clima no equilibrio da demanda e oferta de taxis. O estudo chega a conclusão de que sob quaisquer condições chuvosas, sem discrição da intensidade, os taxistas conseguem mais viagens. É interessante a conclusão do mesmo estudo de que condições de neve quase não afetam (ou afetam pouco) a demanda de embarques.

(Anwar; Volkov; Rus, 2013) apresenta uma ferramenta chamada ChangiNOW, usada para otimizar as filas de táxis no aeroporto de Singapura. O modelo estatístico desenvolvido é baseado em duas filas de passageiros e veículos por terminal do aeroporto. O trabalho modela a distribuição de passageiros por uma distribuicão de Poisson, usando dados históricos dos números de passageiros chegando em cada terminal, com intervalos de 15 minutos. O trabalho também estima o horário de chegada baseado nas posições dos GPS dos táxis rodando por perto, enviadas em tempo real para o sistema. Isso mostra que a agência de táxis TLC tem muito potencial com toda a coleção de dados em mãos, ainda faltando estratégia de como utilizar esses dados.

# 3 Desenvolvimento da Metodologia

## 3.1 Natureza dos dados do problema

Existem aproximadamente 13.437 táxis amarelos na cidade de Nova Iorque, prontos para realizarem 485 mil viagens diárias. O primeiro conjunto de dados é proveniente ***NYC Trip Record Data***. Este conjunto é público e contém: data, hora, local de embarque e desembarque dos passageiros, distância da viagem, tarifas, tipos de pagamento e número de passageiros. Estes dados foram fornecidos à Comissão de Táxis e Limousines (TLC - *táxi and Limousine Comission*) de Nova Iorque por provedores de tecnologia autorizados pelo programa *táxicab and Livery Passenger Enhancement*(COMISSION, 2014). Cada registro da base era gerado por medidores instalados em cada táxi, um total de 13.237 táxis amarelos da cidade de Nova Iorque fizeram 13.813.031 embarques de passageiros no mês de Julho de 2014. Entre os 18 atributos presentes neste conjunto de dados, os principais são:

1. ID do motorista: Chave de identificação do motorista do táxi;
2. Data e hora do embarque: Correspondentes ao embarque do passageiro;
3. Data e hora do desembarque: Correspondentes ao desembarque do passageiro;
4. Número de passageiros;
5. Distância da viagem: em quilômetros;
6. Latitude do embarque e desembarque: Formato WGS 84 (*World Geodetic System 1984)*
7. Longitude do embarque e desembarque: Formato WGS 84 (*World Geodetic System 1984)*
8. Bandeira utilizada;
9. Tipo de pagamento;
10. Preço da corrida;

O segundo conjunto de dados (**NYC PLUTO -** ***Primary Land Use Tax Lot Output***) é o conjunto dos dados geográficos, mostrando informações sobre cada lote, medidas de cada construção dentro de um determinado lote e informações sobre endereço, donos, uso e localização geográfica expressas no sistema de coordenadas de Nova Iorque-Long Island. Este conjunto de dados é disponibilizado online pelo Departamento de Planejamento da Cidade de Nova Iorque (*NYC Department of City Planning*).

Tabela 2 – Conjuntos de Dados relativos a corridas de táxis amarelos.

| Tipo de Serviço | Táxi Amarelo |
|---|---|
| Fonte dos Dados | GPS |
| Número de Instâncias | 13.813.031 |
| Quando | Junho de 2014 |
| Precisão(metros) | 3m |
| Número de táxis participantes | 13.237 |

## 3.2 Pré-Processamento dos dados geográficos

A cidade de Nova Iorque é formada por cinco distritos (bairros): Bronx, Brooklin, Queens, State Island e Manhattan, distrito com maior densidade demográfica dos EUA e atual centro econômico da cidade. Os dados geográficos dos distritos de Nova Iorque, disponibilizados pelo Departamento de Planejamento da Cidade de Nova Iorque, foram tratados com algoritmos em Python, manipulações com Excel e com o software QGIS (QGIS Development Team, 2009), usado para manipulações de arquivos em formato *shape* e tradução das coordenadas geográficas. As figuras 10 e 11 são imagens vetoriais dos mapas geográficos do distrito de Brooklin, ambas foram alaboradas neste trabalho:

Figura 10 – Brooklyn, vista geral.

Figura 11 – Brooklyn detalhado.

O arquivo geográfico original usa o sistema de coordenadas *New York Long Island - EPSG:2263*, incompatível com o padrão usado no conjunto de arquivos de embarques dos táxis, padrão Longitude e Latitude - EPSG:4326 utilizado na maioria dos sistemas atuais de geoposicionamento. A conversão para o padrão Latitude, Longitude de cada coordenada do arquivo geográfico, para todos os distritos, foi feita utilizando o software QGIS. Ao longo deste trabalho, a nomenclatura bloco foi utilizada como sinônimo de quadra, e distrito como sinônimo de bairro.

Para fazer o mapeamento entre cada ponto de embarque e a tupla (bloco,distrito) correspondente a esse ponto, um algoritmo em Python foi desenvolvido e utilizado. O conjunto dos pontos de embarque e o conjunto geográfico foram carregados em dataframes, estruturas de dados da biblioteca Pandas (MCKINNEY, 2010). O Pandas é uma biblioteca *open source*

e oferece alguns tipos de estruturas de dados de alta performance, disponibilizando métodos otimizados para manipulações de dataframes com grande volume de dados. Ela facilita a importação e processamento de arquivos xlsx e csv, amplamente utilizados neste trabalho.

Um ponto de embarque $\mathtt{p}$ é associado a um bloco $\mathtt{b}$ quando a distância entre o $\mathtt{p}$ e $\mathtt{b}$ é menor que a distancia entre $\mathtt{p}$ e qualquer outro bloco do $\mathtt{b}_1, \mathtt{b}_2, \mathtt{b}_3, ..., \mathtt{b}_n$.. Para o calulo da distância entre dois pontos em uma esfera (planeta terra), foi usada a formula de haversine, muito importante na navegação. A equação de haversine é um caso de trigonometria esférica (SILVA; EDSON, 2014), cuja prova matemática é fora do escopo deste trabalho.
Seja $\theta$ um ângulo em radianos:

$$\mathtt{haversine}(\theta) = \sin^2(\frac{\theta}{2}) \tag{3.1}$$

Para cada ponto de embarque $\mathtt{p}$:
Sejam $\mathtt{pLat}$,$\mathtt{pLon}$ as coordenadas do ponto de latitude e longitude do embarque;
Sejam $\mathtt{bLat}$,$\mathtt{bLon}$ as coordenadas de um bloco $\mathtt{b}$ em um distrito $\mathtt{dx}$ ;

$$1)\ \mathtt{Raio}_{\mathtt{terra}}\ \mathtt{R} = 6.672,8\mathtt{Km} \tag{3.2}$$

$$2)\ \mathtt{dLat} = \mathtt{Radianos}(\mathtt{pLat} - \mathtt{bLat}) \tag{3.3}$$

$$3)\ \mathtt{dLon} = \mathtt{Radianos}(\mathtt{BLon} - \mathtt{pLon}) \tag{3.4}$$

$$4)\ \mathtt{pLat} = \mathtt{Radianos}(\mathtt{pLat}) \tag{3.5}$$

$$5)\ \mathtt{bLat} = \mathtt{Radianos}(\mathtt{bLat}) \tag{3.6}$$

A distância $\mathtt{d}$ entre os pontos de embarque e o bloco (quadra) é dada por:

$$\mathtt{d} = 2\mathtt{R}\arcsin\sqrt{(\sin(\frac{\mathtt{dLat}}{2})^2 + \cos(\mathtt{pLat})\cos(\mathtt{bLat})\sin(\frac{\mathtt{dLon}}{2})^2)} \tag{3.7}$$

Ou simplesmente:

$$\mathtt{d} = 2\mathtt{R}\arcsin\sqrt{\mathtt{haversine}(\mathtt{dLat}) + \cos(\mathtt{pLat})\cos(\mathtt{bLat})\mathtt{haversine}(\mathtt{dLon})} \tag{3.8}$$

Para o armazenamento dos registros do mapeamento, um arquivo csv (comma-delimited values) foi utilizado. Cada linha do arquivo contem os seguintes campos: id do mapeamento, nome do distrito de embarque, id do bloco de embarque, data do embarque, hora do embarque.

O algoritmo 1 trata do mapeamento entre pontos de embarque e seus respectivos blocos de cada distrito:

**Algorithm 1:** Mapeamento entre ponto de embarque, bloco e distrito

```
Data: Dataframe blocos, Dataframe embarques, String idMenor, Float d, String distrito,
      Int contador
Result: d
contador = 0;
for e in embarques do
    idMenor = 0;
    d = 9999;
    for b in blocos do
        pLat = e.Latitude;
        pLon = e.Longitude;
        bLat = b.Latitude;
        bLon = b.Longitude;
        aux = distancia(pLat,pLon,bLat,bLon);
        if aux < d then
            d = aux;
            idMenor = b.Id;
            distrito = b.Distrito;;
            EscreveLinhaSaida(contador,distrito, idMenor, d, e.DataHora);
    contador + +
```

O mapeamento durou 3 meses, devido ao alto volume de dados tanto no conjunto de embarques quanto no conjunto geográfico do terreno de Nova Iorque. A infraestrutura final utilizada foi uma máquina virtual no Google Cloud, disponibilizada durante período de avaliação da plataforma. A estrutura contava com processador de 8 núcleos e 16GB de memória RAM, rodando Ubuntu Linux.

## 3.3 Análise dos dados

A fim de se conhecer mais sobre o conjunto de dados, algumas análises foram feitas com as instâncias. O gráfico abaixo mostra a relação do número de embarques por dia do mês.

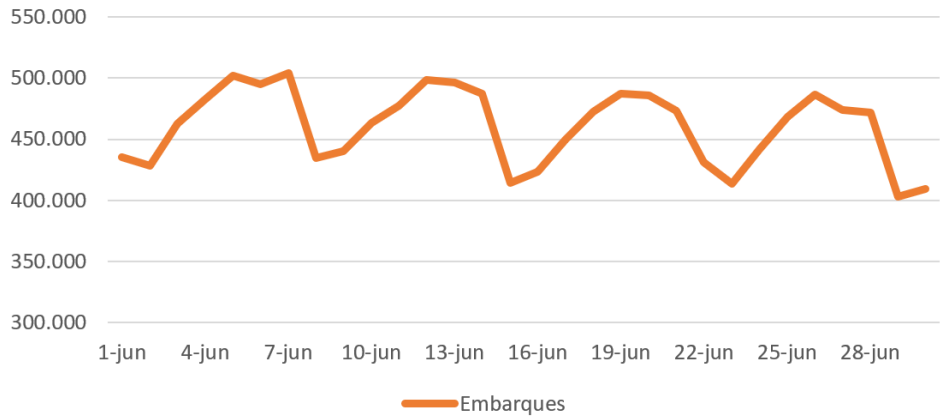


Figura 12 – Número de embarques por dia (Junho de 2014).

É possível notar que apesar do número de embarques cair ao longo do mês, um comportamento periódico geral acontece. A periodicidade no comportamento geral das demandas não necessariamente reflete a demanda em granularidades menores (em distritos ou blocos, por exemplo). Nas figuras abaixo, são mostradas demandas horárias (número de embarques por hora) de dois blocos, o primeiro, de um bloco residencial tranquilo no Brooklyn, o segundo, do bloco vizinho ao Madison Square Garden (Manhattan).

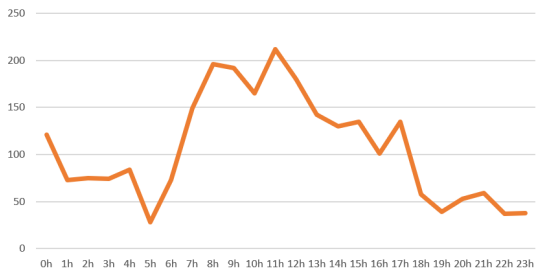


Figura 13 – Bloco residencial - Brooklyn

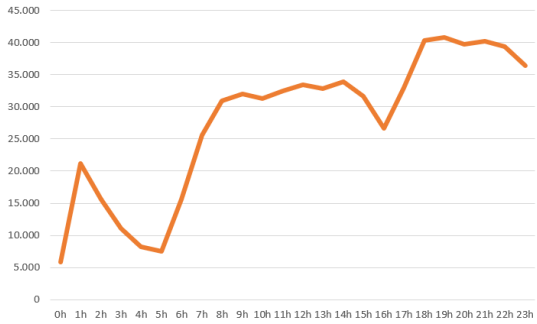


Figura 14 – Bloco vizinho ao Madison Square Garden, em Manhattan

Nota-se que a demandas são completamente diferentes, tanto em volume, quanto em horário e variação. A primeira, um bloco residencial, apresenta demandas mil vezes menores do que o bloco vizinho ao Madison Square Garden. O bloco no Brooklyn tem demanda decrescente após as 18hrs, enquanto o bloco em Manhattan tem pico de demanda a partir das 18h, quando a maioria dos diversos eventos no Madison Square Garden começa. O gráfico abaixo mostra a quantidade de embarques por dia da semana na cidade de Nova Iorque.

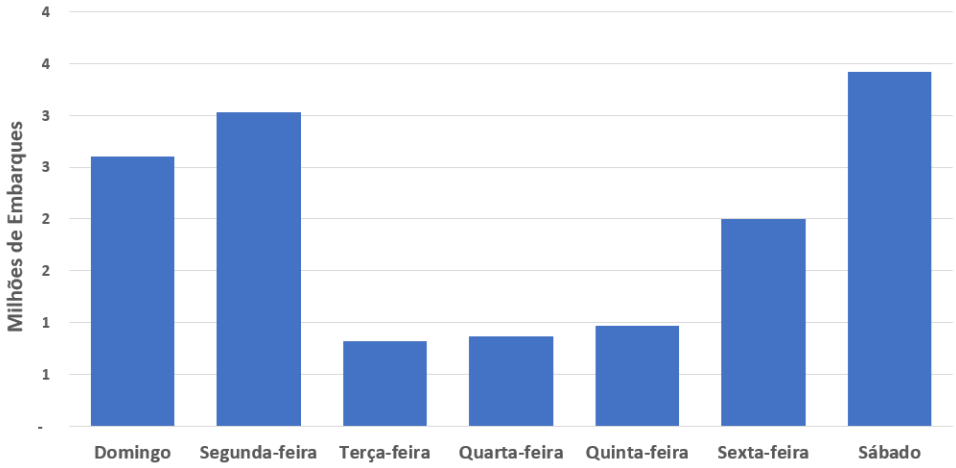


Figura 15 – Número de embarques por dia da semana.

Três tipos de agrupamentos foram feitos no conjunto inicial de dados, agrupamento por hora, dia do mês e por dia da semana. É importante lembrar que o conjunto inicial contava com 13,8 milhões de instâncias (embarques). Nova Iorque possui aproximadamente 41 mil blocos, e cada agrupamento resultou na seguinte quantidade de instâncias (exemplos):

- Por hora: 986.808 exemplos resultantes;
- Por dia do mês: 1.233.340 exemplos resultantes;
- Por dia da semana: 287.700 exemplos resultantes.

A tabela abaixo mostra um exemplo do mapeamento agrupado por hora. A coluna frequência significa o número de embarques:

Tabela 3 – Exemplo do mapeamento segmentado por hora.

| Distrito | ID Bloco | Hora | Frequência |
|---|---|---|---|
| MN | 1 | 0h | 68 |
| MN | 623 | 18h | 13 |
| BK | 34 | 3h | 2 |
| SI | 99 | 17h | 20 |

Exemplo de agrupamento por dia da semana:

Tabela 4 – Exemplo do mapeamento segmentado por dia da semana.

| Distrito | ID Bloco | Dia da Semana | Frequência |
|---|---|---|---|
| MN | 623 | Sábado | 508 |
| MN | 623 | Segunda | 127 |

## 3.4 Ferramentas e Métodos

Uma abordagem utilizando mudança de conceito foi utilizada para o treinamento de uma rede neural, usando a função de ativação Softmax. Para fins de comparação, um modelo de rede neural em lote também foi desenvolvido. O motivo da utilização da rede neural é devido a boa performance (melhor) aferida em (ZHAO; KHRYASHCHEV; VO, 2017), a frente de outros três preditores também analisados pelo mesmo trabalho. A frequência de demanda (número de demandas) é o atributo classe.

O software MOA (Massive Online Analysis) foi utilizado para o treinamento dos dois modelos. Os seguintes parâmetros foram variados no modelo online: taxa de aprendizado (learning rate), tamanho da janela de avaliação. Para o modelo em lote apenas a taxa de aprendizado foi variada. A métrica de erro utilizada foi a MAPE - Mean Absolute Percentage Error (MEADE; ARMSTRONG, 1986). Os valores de erro são apresentados em números decimais, logo, 0,49 representa 49%.

# 4 Resultados

Esta seção apresentará os resultados dos experimentos propostos por este trabalho.

Primeiro, os resultados do modelo de rede neural online (com fluxo de dados) são apresentados na tabela 5. O objetivo desta tabela é representar a variação do erro de acordo com o tamanho da janela de avaliação usada no algoritmo online. Por serem as mais adotadas na literatura, janelas com 1000 e 200 instâncias foram utilizadas. Além de variar o tamanho da janela, a tabela também mostra os resultados segmentados por hora, dia do mês e dia da semana, afim de encontrar o melhor agrupamento a ser trabalhado.

Definido o melhor agrupamento, a tabela 6 apresenta a variação dos resultados com a variação da taxa de aprendizado (learning rate). As taxas testadas foram: 0,01, 0,05 e 0,1, as mais comumente encontradas na literatura pesquisada.

A tabela 7 traz resultados interessantes. Nela é possível ver a taxa de erro para preditores treinados para cada distrito separadamente. É interessante ver que alguns distritos são menos "previsíveis" que outros. Os resultados apresentados nesta tabela também são submetidos à variação do tamanho da janela de avaliação.

Os dados da tabela 8 mostram os resultados da previsão de demanda utilizando modelo de rede neural em lote. A variação da performance sob influência da taxa de aprendizado é uma análise mostrada.

Assim como feito na tabela 7, a tabela 9 mostra a performace de modelos em lote criados para cada distrito separadamente.

## 4.1 Modelo Online

Para fins de comparação com o trabalho de (ZHAO; KHRYASHCHEV; VO, 2017), o agrupamento (segmentação) das demandas por hora deve ser considerado. Abaixo a tabela de resultados do modelo online, considerando o tipo de agrupamento e tamanho da janela de avaliação:

Tabela 5 – Resultados do aprendizado online, utilizando mudança de conceito, variando o tamanho da janela de avaliação. Modelo treinado com todos os distritos.

| Aprendizado | Learning Rate | Training Time | Erro | Tamanho da Janela | Segmentação |
|---|---|---|---|---|---|
| Fluxo | 0,01 | 4:30h | 0,52 | 1000 | Dia do mês |
| Fluxo | 0,01 | 27min | 0,48 | 1000 | Dia da Semana |
| Fluxo | 0,01 | 2:09h | 0,23 | 1000 | Hora |
| Fluxo | 0,01 | 6:02h | 0,51 | 200 | Dia do mês |
| Fluxo | 0,01 | 41min | 0,44 | 200 | Dia da Semana |
| Fluxo | 0,01 | 3:01h | 0,22 | 200 | Hora |

A última linha da tabela mostra que o agrupamento da frequência por hora e uma janela de avaliação de 200 exemplos produzem um erro de 22% na previsão de demanda de táxis, quase duas vezes menor que os erros observados nos agrupamentos por dia do mês e por dia da semana. Por este resultado e pela recomendação do trabalho de (TONG et al., 2017) é possível afirmar que o mapeamento segmentado por hora conduz os resultados ao um mínimo encontrado nos experimentos. As próximas análises também serão segmentadas por hora, com tamanho de janela de 200 instâncias.

A escolha da taxa de aprendizado no modelo online(*learning rate*) usada produziu pequenas variações no agrupamento dos dados por hora, conforme tabela abaixo:

Tabela 6 – Influência causada pela variação da taxa de aprendizado do modelo online. Modelo treinado com todos os distritos

| Aprendizado | Learning Rate | Erro | Tamanho da Janela | Segmentação |
|---|---|---|---|---|
| Fluxo | 0,01 | 0,22 | 200 | Hora |
| Fluxo | 0,05 | 0,25 | 200 | Hora |
| Fluxo | 0,1 | 0,34 | 200 | Hora |

(ZHAO; KHRYASHCHEV; VO, 2017) utilizou ainda como dados de entrada, informações históricas sobre temperatura e condições climáticas, as quais podem ter contribuído para maximização dos resultados. Contudo, esses dados temporais não parecem ser facilmente disponibilizados para análise, sendo necessária uma extração manual, com periodicidade diária das condições climáticas e da temperatura.

Abaixo, a tabela mostra a performance do modelo online treinado para cada distrito separadamente. A análise das performances pode revelar o quão previsível pode ser a demanda em cada um, de acordo com suas características.

Tabela 7 – Performance do modelo online em cada distrito, separadamente.

| Aprendizado | Distrito | Learning Rate | Trainning Time | Erro | Window Size | Segmentação |
|---|---|---|---|---|---|---|
| Fluxo | MN | 0,01 | 57min | 0,69 | 1000 | Hora |
| Fluxo | MN | 0,01 | 1:02min | 0,40 | 200 | Hora |
| Fluxo | SI | 0,01 | 21min | 0,27 | 1000 | Hora |
| Fluxo | SI | 0,01 | 28min | 0,11 | 200 | Hora |
| Fluxo | BK | 0,01 | 32min | 0,31 | 1000 | Hora |
| Fluxo | BK | 0,01 | 38min | 0,30 | 200 | Hora |
| Fluxo | BX | 0,01 | 20min | 0,22 | 1000 | Hora |
| Fluxo | BX | 0,01 | 24min | 0,25 | 200 | Hora |
| Fluxo | QN | 0,01 | 18min | 0,53 | 1000 | Hora |
| Fluxo | QN | 0,01 | 22min | 0,45 | 200 | Hora |

Manhattan mostra ser o distrito com a menor previsibilidade, e State Island com a maior. A janela de avaliação com 200 exemplos se mostrou mais eficaz em quase todas as avaliações, com exceção do Bronx. O distrito de State Island se mostra grande candidato a implantação de um projeto piloto para gerência e recomendação da demanda de táxis. A performance neste distrito chega a 89% de previsibilidade na demanda, um resultado promissor se comparado a alguns dos trabalhos relacionados.

## 4.2 Modelo em Lote

Os resultados do modelo em lote são revelados abaixo. É possível notar que, considerando todos os distritos, o modelo em lote possui performance inferior na previsão de demanda, se comparado com o modelo online. A menor taxa de erro foi de 49%, mais que o dobro da menor taxa encontrada no modelo online, de 22%. O tempo de treinamento do modelo em lote não foi medido neste experimento, por isso não é mostrado.

Tabela 8 – Resultado do aprendizado em lote, variando o learning rate. Todos os distritos considerados

| Aprendizado | Learning Rate | Erro | Segmentação |
|---|---|---|---|
| Lote | 0,01 | 0,58 | Hora |
| Lote | 0,05 | 0,60 | Hora |
| Lote | 0,1 | 0,49 | Hora |

Usando o learning rate que produziu a melhor performance (0,1), um modelo em lote para cada distrito foi gerado:

Tabela 9 – Resultado do aprendizado em lote, para cada distrito separadamente.

| Aprendizado | Learning Rate | Distrito | Erro | Segmentação |
|---|---|---|---|---|
| Lote | 0,1 | MN | 0,58 | Hora |
| Lote | 0,1 | SI | 0,60 | Hora |
| Lote | 0,1 | BK | 0,49 | Hora |
| Lote | 0,1 | BX | 0,24 | Hora |
| Lote | 0,1 | QN | 0,30 | Hora |

Ao comparar com a tabela 7 dos resultados do algoritmo online por distrito, o aprendizado em lote ainda menor performance na previsão da demanda por distrito. Tanto no modelo online quanto modelo em lote, o distrito Bronx apresenta boa previsibilidade, sendo um bom candidato para projetos pilotos de implantação de um sistema de recomendação de demanda de táxis, assim como o já mencionado distrito de State Island.

## 4.3 Visualização dos resultados

Os gráficos a seguir trazem uma visualização resumida dos resultados das duas seções anteriores, comparando a performance do algoritmo online vs em lote. Os gráficos retratam os erros, logo, quanto menor, melhor a performance.

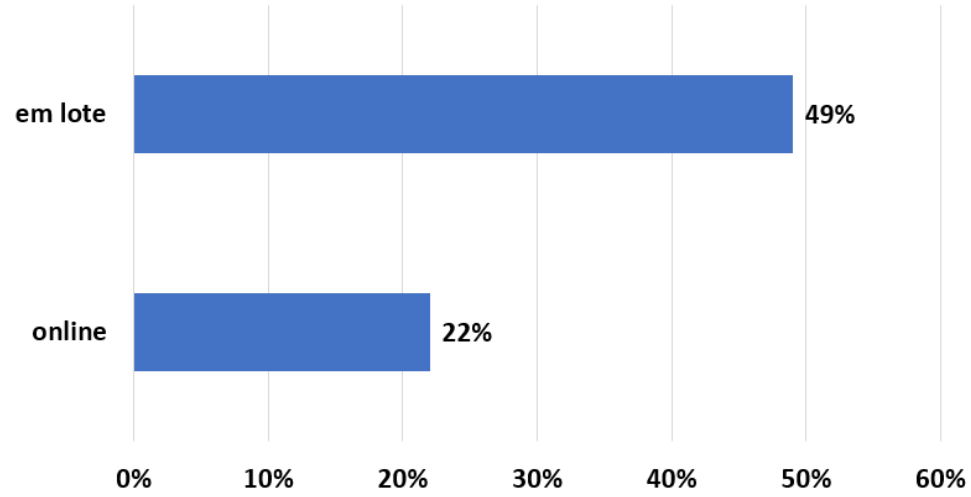


Figura 16 – Erro do algoritmo online vs algoritmo em lote. Modelo treinado para todos os distritos. Quanto menor, melhor.

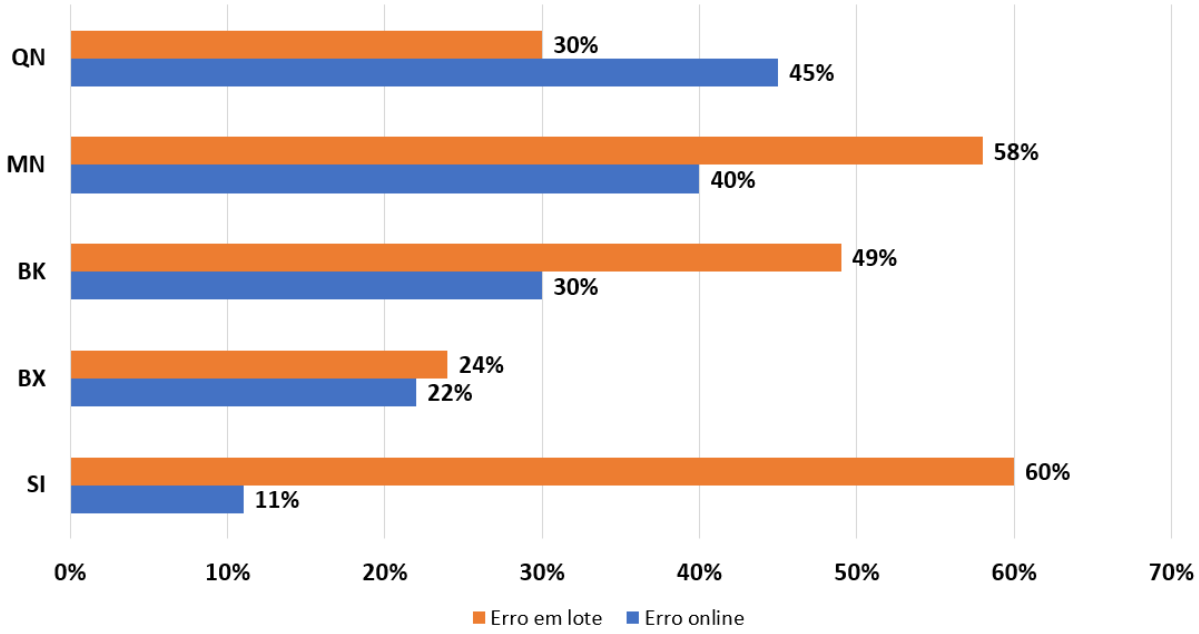


Figura 17 – Erro do algoritmo online vs algoritmo em lote, por distrito. Modelo treinado para cada distrito separadamente. Em azul: modelo online. Em laranja: modelo em lote

# 5 Conclusões e Trabalhos Futuros

Este trabalho apresentou uma proposta para um modelo de previsão de demanda de táxis na cidade de Nova Iorque, usando um conjunto de dados de treinamento de Julho de 2014 disponibilizado na plataforma de dados aberto da cidade.

Mesmo sem dados sobre o clima, o modelo online final apresentou performance considerável na previsão de demanda, sendo promissor também quando avaliado separadamente por distrito. Futuramente, a incorporação de fatores climáticos no conjunto de dados pode afetar positivamente a performance do modelo, pois é um fator que influencia diretamente na mobilidade urbana das pessoas.

Conforme observado em (COVIENSKY, 2017), a demanda de táxis em um ponto pode não retratar muito bem a realidade em alguns casos, pois o conjunto de dados disponibilizado considera uma demanda o fato do passageiro já ter embarcado no veículo, desconsiderando outras pessoas que possivelmente estejam na fila esperando um táxi livre. Caso houvessem dados sobre os passageiros esperando na fila, tempo de espera dos passageiros em um ponto e tempo de espera dos táxis parados no ponto, o modelo poderia medir a demanda mais precisamente. Para trabalhos futuros pode-se citar:

- Análise estatística mais profunda dos dados;
- Incorporação de dados climáticos;
- Incorporação de dados sobre o trânsito;
- Uso de outros algoritmos de previsão;
- Aplicação da metodologia em transportes públicos.

Uma limitação encontrada foi o tempo de mapeamento entre o conjunto de embarques dos táxis e o conjunto geográfico do terreno da cidade. Os dois conjuntos apresentavam um número consideravelmente grande de exemplos a serem mapeados, conforme mencionado no capítulo 3. O mapeamento entre esses dois conjuntos levou alguns meses. A primeira abordagem foi utilizar um notebook padrão, o que não funcionou a longo prazo, pois depois de um longo tempo ligado a máquina apresentou alguns problemas durante a execução do algoritmo. A segunda abordagem foi realizada utilizando uma máquina virtual no Google Cloud, onde a execução durou cerca 1 mês e meio, devido a configuração não tão robusta oferecida pelo período de avaliação da plataforma.

Os conjuntos de dados utilizados neste trabalho será disponibilizado online pelo GitHub, usuário *dlfaial*. O algoritmo construído para mapeamento também ficará disponível no mesmo ambiente.

# Referências


Anwar, A.; Volkov, M.; Rus, D. Changinow: A mobile application for efficient taxi allocation at airports. In: *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*. [S.l.: s.n.], 2013. p. 694–701. ISSN 2153-0009.

ATTAR, V. et al. An instance-window based classification algorithm for handling gradual concept drifts. In: CAO, L. et al. (Ed.). *Agents and Data Mining Interaction*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. p. 156–172. ISBN 978-3-642-27609-5.

BAENA-GARCıA, M. et al. Early drift detection method. In: . [S.l.: s.n.], 2005.

BECKER, H.; ARIAS, M. Real-time ranking with concept drift using expert advice. In: *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 2007. (KDD '07), p. 86–94. ISBN 978-1-59593-609-7. Disponível em: <http://doi.acm.org/10.1145/1281192.1281205>.

BIFET, A. Adaptive learning and mining for data streams and frequent patterns. *SIGKDD Explor. Newsl.*, ACM, New York, NY, USA, v. 11, n. 1, p. 55–56, nov. 2009. ISSN 1931-0145. Disponível em: <http://doi.acm.org/10.1145/1656274.1656287>.

BISHOP, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN 0387310738.

CASTILLO, G. Adaptive learning algorithms for bayesian network classifiers. *AI Commun.*, IOS Press, Amsterdam, The Netherlands, The Netherlands, v. 21, n. 1, p. 87–88, jan. 2008. ISSN 0921-7126. Disponível em: <http://dl.acm.org/citation.cfm?id=1365980.1365984>.

COMISSION, N. T. *Taxicab Passenger Enhancements Project (TPEP)*. 2014. Disponível em: http://www.nyc.gov/html/tlc/html/industry/taxicab$_s$er$v_e$nh$_a$rchive.shtml.Acessoem : 01deabrilde2018.

COVIENSKY, A. Estimating demand for taxis at laguardia airport. *Taxi and Limousine Comission*, p. 1–14, 2017.

DANIEL, B. Requirements for clustering data streams. *SIGKDD Explor. Newsl.*, ACM, New York, NY, USA, v. 3, n. 2, p. 23–27, 2002. ISSN 1931-0145. Disponível em: <http://doi.acm.org/10.1145/507515.507519>.

DELANY, S. J. et al. A case-based technique for tracking concept drift in spam filtering. *Know.-Based Syst.*, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 18, n. 4-5, p. 187–195, ago. 2005. ISSN 0950-7051. Disponível em: <http://dx.doi.org/10.1016/j.knosys.2004.10.002>.

DOMINGOS, P.; HULTEN, G. A general framework for mining massive data streams. *Journal of Computational and Graphical Statistics*, Taylor Francis, v. 12, n. 4, p. 945–949, 2003.

FACELI, K. et al. *Inteligência Artificial. Uma Abordagem de Aprendizado de Máquina (Em Portuguese do Brasil)*. [S.l.]: LTC, 2011. ISBN 8521618808.

Frías-Blanco, I. et al. Online and non-parametric drift detection methods based on hoeffding's bounds. *IEEE Transactions on Knowledge and Data Engineering*, v. 27, n. 3, p. 810–823, March 2015. ISSN 1041-4347.

GAMA, J. et al. Learning with drift detection. In: BAZZAN, A. L. C.; LABIDI, S. (Ed.). *Advances in Artificial Intelligence – SBIA 2004*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 286–295.

GANTZ, J.; REINSEL, D. The digital universe in 2020: Big data big digital shadows and biggest growth in far east. In: *IDC IVIEW*. Berlin, Heidelberg: EMC Corporation, 2012. p. 1–16.

HAN, J.; KAMBER, M.; PEI, J. - classification: Basic concepts. In: HAN, J.; KAMBER, M.; PEI, J. (Ed.). *Data Mining (Third Edition)*. Third edition. Boston: Morgan Kaufmann, 2012, (The Morgan Kaufmann Series in Data Management Systems). p. 327 – 391. ISBN 978-0-12-381479-1. Disponível em: <https://www.sciencedirect.com/science/article/pii/B9780123814791000083>.

HIDALGO, J. I. G. *Experiências com variações prequential para avaliação da aprendizagem em fluxo de dados*. Tese (Thesis), 2017. Disponível em: <https://repositorio.ufpe.br/handle/123456789/26725>.

HOFFMANN, K.; IPEIROTIS, P. G.; SUNDARARAJAN, A. Ridesharing and the use of public transportation. *Digital Innovation At The Crossroads*, v. 18, n. 6, p. 11 – 14, 2016. ISSN 0968-090X. Special issue on Transportation SimulationAdvances in Air Transportation Research. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0968090X1000029X>.

HUANG, Y.; POWELL, J. W. Detecting regions of disequilibrium in taxi services under uncertainty. In: *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*. New York, NY, USA: ACM, 2012. (SIGSPATIAL '12), p. 139–148. ISBN 978-1-4503-1691-0. Disponível em: <http://doi.acm.org/10.1145/2424321.2424340>.

KAMGA, C.; YAZICI, M. A.; SINGHAL, A. Hailing in the rain: Temporal and weather-related variations in taxi ridership and taxi demand-supply equilibrium. 01 2013.

Karnick, M. et al. Learning concept drift in nonstationary environments using an ensemble of classifiers based approach. In: *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. [S.l.: s.n.], 2008. p. 3455–3462. ISSN 2161-4393.

KLINKENBERG, R. Learning drifting concepts: Example selection vs. example weighting. *Intell. Data Anal.*, IOS Press, Amsterdam, The Netherlands, The Netherlands, v. 8, n. 3, p. 281–300, ago. 2004. ISSN 1088-467X. Disponível em: <http://dl.acm.org/citation.cfm?id=1293831.1293836>.

KLINKENBERG, R.; JOACHIMS, T. Detecting concept drift with support vector machines. In: *Proceedings of the Seventeenth International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000. (ICML '00), p. 487–494. ISBN 1-55860-707-2. Disponível em: <http://dl.acm.org/citation.cfm?id=645529.657791>.

KOLTER, J. Z.; MALOOF, M. A. Dynamic weighted majority: An ensemble method for drifting concepts. *J. Mach. Learn. Res.*, JMLR.org, v. 8, p. 2755–2790, dez. 2007. ISSN 1532-4435. Disponível em: <http://dl.acm.org/citation.cfm?id=1314498.1390333>.

LAW, Y.-N.; ZANIOLO, C. An adaptive nearest neighbor classification algorithm for data streams. In: JORGE, A. M. et al. (Ed.). *Knowledge Discovery in Databases: PKDD 2005*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. p. 108–120. ISBN 978-3-540-31665-7.

LEMAIRE, V.; SALPERWYCK, C.; BONDU, A. A survey on supervised classification on data streams. *Lecture Notes in Business Information Processing*, 03 2015.

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

MCKINNEY, W. Data structures for statistical computing in python. In: WALT, S. van der; MILLMAN, J. (Ed.). *Proceedings of the 9th Python in Science Conference*. [S.l.: s.n.], 2010. p. 51 – 56.

MEADE, N.; ARMSTRONG, J. Long range forecasting: From crystal ball to computer (2nd edition). *The Journal of the Operational Research Society*, v. 37, p. 533, 05 1986.

MITCHELL, T. *Machine Learning*. McGraw-Hill, 1997. (McGraw-Hill International Editions). ISBN 9780071154673. Disponível em: <https://books.google.com.br/books?id=EoYBngEACAAJ>.

NARASIMHAMURTHY, A.; KUNCHEVA, L. I. A framework for generating data to simulate changing environments. In: *Proceedings of the 25th Conference on Proceedings of the 25th IASTED International Multi-Conference: Artificial Intelligence and Applications*. Anaheim, CA, USA: ACTA Press, 2007. (AIAP'07), p. 384–389. Disponível em: <http://dl.acm.org/citation.cfm?id=1295303.1295369>.

NISHIDA, K. Learning and detecting concept drift. In: . [S.l.: s.n.], 2008.

QGIS Development Team. *QGIS Geographic Information System*. [S.l.], 2009. Disponível em: <http://qgis.osgeo.org>.

RAUDYS, S.; MITASIUNAS, A. Multi-agent system approach to react to sudden environmental changes. In: *Proceedings of the 5th International Conference on Machine Learning and Data Mining in Pattern Recognition*. Berlin, Heidelberg: Springer-Verlag, 2007. (MLDM '07), p. 810–823. ISBN 978-3-540-73498-7. Disponível em: <http://dx.doi.org/10.1007/978-3-540-73499-4_61>.

RAYLE, L. et al. App-based, on-demand ride services: Comparing taxi and ridesourcing trips and user characteristics in san francisco. *Transportation Sustainability Research*, v. 18, n. 6, p. 0 – 18, 2014. ISSN 0968-090X. Disponível em: <http://tsrc.berkeley.edu/node/797>.

ROKACH et al. *Data Mining With Decision Trees: Theory and Applications*. 2nd. ed. River Edge, NJ, USA: World Scientific Publishing Co., Inc., 2014. ISBN 9789814590075, 981459007X.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, v. 65, n. 6, p. 386–408, 1958. ISSN 1939-1471(Electronic),0033-295X(Print).

SCHLIMMER, J. C.; GRANGER, R. H. Incremental learning from noisy data. *Machine Learning*, v. 1, n. 3, p. 317–354, Sep 1986. ISSN 1573-0565. Disponível em: <https://doi.org/10.1007/BF00116895>.

SCHOLZ, M.; KLINKENBERG, R. Boosting classifiers for drifting concepts. *Intell. Data Anal.*, IOS Press, Amsterdam, The Netherlands, The Netherlands, v. 11, n. 1, p. 3–28, jan. 2007. ISSN 1088-467X. Disponível em: <http://dl.acm.org/citation.cfm?id=1367489.1367491>.

SILVA, F.; EDSON, P. d. A. A trigonometria esférica e o globo terrestre. In: CEARA, U. F. do (Ed.). [S.l.: s.n.], 2014.

SIMON, P. *Too Big to Ignore: The Business Case for Big Data*. 1st. ed. [S.l.]: Wiley Publishing, 2013. ISBN 1118638174, 9781118638170.

SONG, C. et al. Limits of predictability in human mobility. *Science*, American Association for the Advancement of Science, v. 327, n. 5968, p. 1018–1021, 2010. ISSN 0036-8075. Disponível em: <http://science.sciencemag.org/content/327/5968/1018>.

STANLEY, K. O. *Learning Concept Drift with a Committee of Decision Trees*. [S.l.], 2003. Disponível em: <http://nn.cs.utexas.edu/?stanley:utcstr03-302>.

STANLEY, K. O. Learning concept drift with a committee of decision trees. In: . [S.l.: s.n.], 2003.

STREET, W. N.; KIM, Y. A streaming ensemble algorithm (sea) for large-scale classification. In: *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 2001. (KDD '01), p. 377–382. ISBN 1-58113-391-X. Disponível em: <http://doi.acm.org/10.1145/502512.502568>.

TONG, Y. et al. The simpler the better: A unified approach to predicting original taxi demands based on large-scale online platforms. In: . [S.l.: s.n.], 2017. p. 1653–1662.

TSYMBAL, A. et al. Dynamic integration of classifiers for handling concept drift. *Inf. Fusion*, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 9, n. 1, p. 56–68, jan. 2008. ISSN 1566-2535. Disponível em: <http://dx.doi.org/10.1016/j.inffus.2006.11.002>.

UBER, N. by. *Red Line Fail? Ride for Free on Boston's Uber Line*. 2018. Disponível em: https://www.uber.com/newsroom/red-line-fail-introducing-the-u-line/. Acesso em: 18 de março de 2018.

WANG, H. et al. Mining concept-drifting data streams using ensemble classifiers. In: *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 2003. (KDD '03), p. 226–235. ISBN 1-58113-737-0. Disponível em: <http://doi.acm.org/10.1145/956750.956778>.

WIDMER, G. *Special Issue on Context Sensitivity and Concept Drift*. Kluwer, 1998. (Machine learning). Disponível em: <https://books.google.com.br/books?id=8KU_vwEACAAJ>.

WIDMER, G.; KUBAT, M. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, v. 23, n. 1, p. 69–101, Apr 1996. ISSN 1573-0565. Disponível em: <https://doi.org/10.1007/BF00116900>.

WIDYANTORO, D. H. Concept drift learning and its application to adaptive information filtering. In: . [S.l.: s.n.], 2003.

YANG, H.; FONG, S.; SI, Y.-W. Multi-objective optimization for incremental decision tree learning. In: CUZZOCREA, A.; DAYAL, U. (Ed.). *Data Warehousing and Knowledge Discovery*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. p. 217–228.

ZHANG, P.; ZHU, X.; SHI, Y. Categorizing and mining concept drifting data streams. In: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 2008. (KDD '08), p. 812–820. ISBN 978-1-60558-193-4. Disponível em: <http://doi.acm.org/10.1145/1401890.1401987>.

ZHAO, K.; KHRYASHCHEV, D.; VO, H. Predicting taxi and uber demand in cities. 2017.