DSA – Formação Cientista de Dados

8. Business Analytics

8.1.Introdução

O que é Business Analytics (BA)?

Business Analytics é a prática de analisar dados para suportar a tomada de decisões!

Estudo dos dados através de Data Science e análise de operações.

Business Analytics exige a aplicação de métodos quantitativos e baseados em evidências, para modelagem de problemas de negócios e suporte à tomada de decisão.

Quando as empresas analisam o Big Data com métodos baseados em Data Science, elas estão usando o Business Analytics para obter os insights necessários para tomar melhores decisões de negócio e movimentos estratégicos.

Empresas de classe mundial tendem a alavancar a análise 5 vezes mais do que outras, quando aplicam Business Analytics.

Business Analytics e Data Science não são a mesma coisa.

Business Analytics = Business + Big Data + Data Science

Fluxo do BA:

Dados Brutos → Dados Agregados → Inteligência → Insights → Decisões → Impacto Operacional → Resultados Financeiros → Criação De Valor

Todos os dias 2.5 quintilhões de bytes de dados são criados. E como criamos valor a partir de todos esses dados? Com Business Analytics.

O Que é Data Mining?

Data Mining é uma das técnicas usadas em Business Analytics!

De forma simplificada, a mineração de dados pode ser definida como um processo automático ou semi-automático de explorar analiticamente grandes bases de dados, com a finalidade de descobrir padrões relevantes que ocorrem nos dados e que sejam importantes para embasar a assimilação de informação, suportando a geração de conhecimento.

Knowledge Discovery in Databases KDD – Descoberta de Conhecimento em Bancos de Dados

Mineração de dados é o processo de exploração de grandes quantidades de dados com o objetivo de encontrar anomalias, padrões e correlações para suportar a tomada de decisões e proporcionar vantagens estratégicas.

A Mineração de Dados tornou-se uma ferramenta de apoio com papel fundamental na gestão da informação dentro das organizações.

Para gerar resultados significativos, a mineração de dados exige um processo constante de coleta, processamento e análise de informações.

Ferramentas de Data Mining:

- Licenciadas: • Oracle Data Miner • SAS Text Miner • IBM Intelligent Miner • KNIME • Rapid Miner • Weka
- Open source: • Python • R

Metodologia Analítica

Característica	**Data Warehouses**	**Big Data (Data Lakes)**
Dados	Estruturados	Não Estruturados ou Semi-Estruturados
Volume	Grande	Muito Grande
Geração	Dados Transacionais	Todos os Tipos de Dados
	Business Intelligence	Business Analytics

Business Analytics:

- Business Intelligence
  - Análise Descritiva
  - Análise Diagnóstica
- Análise Preditiva
- Análise Prescritiva

O resultado de uma previsão é uma estimativa e a ciência que fundamenta a metodologia é a Estatística.

Geração de Dados → Codificação de Dados → Captação de Dados → Transmissão de Dados → Armazenagem de Dados → Organização dos Dados → Processamento de Dados → Definição de Métricas → Construção de Modelos Preditivos → Aplicação

Definindo um Problema de Negócio

Identificar e definir um problema não é difícil, mas requer paciência, repetição e análise.

3 passos para identificar e definir um problema:

1. Explore o cenário atual
2. Explique o problema
3. Questione a si mesmo

Uma técnica para definição de problema é o Toyota 5 Why’s.

Mas por que a definição do problema de negócio é importante?

Porque tudo depende desta definição. Existem dezenas de linguagens de programação, diversos bancos de dados, diversos sistemas, ferramentas e técnicas. Cada qual mais apropriado para determinado tipo de problema. A escolha da melhor solução depende da clara definição do problema!

Business Intelligence (BI) x Business Analytics (BA)

Principais ferramentas em BI:

- QlikView
- SAP BusinessObjects
- IBM Cognos
- Microstrategy

Principais ferramentas em BA:

- SAP Business Analytics Suite
- Pentaho BA
- Birst BI
- Tableau Big Data Analytics

8.2. Trabalhando com Analytics – Parte 1

O Processo de Business Analytics

Enquanto as aplicações estatísticas tradicionais se concentram em conjuntos de dados relativamente pequenos, a Ciência de Dados envolve quantidades muito grandes de dados, normalmente o que chamamos de Big Data.

Business Analytics deve ser visto como um processo.

O processo de Business Analytics envolve várias etapas inter-relacionadas:

1. O armazenamento de dados eficiente e os passos de pré-processamento de dados são muito críticos para o sucesso da análise.
2. É preciso selecionar as variáveis de resposta apropriadas e decidir sobre o número de variáveis que devem ser investigadas.
3. Os dados precisam ser rastreados em busca de outliers e os valores faltantes (missing) precisam ser endereçados (com valores ausentes omitidos ou apropriadamente imputados através de um dos vários métodos disponíveis).
4. Antes de aplicar modelos preditivos e métodos sofisticados, os dados precisam ser visualizados e resumidos. Costuma-se dizer que uma imagem vale mais que 1000 palavras.
5. Resumo dos dados envolve estatísticas típicas de resumo como média, percentis e mediana, desvio padrão e correlação, bem como resumos mais avançados, tais como componentes principais.
6. Métodos apropriados de modelagem preditiva precisam ser aplicados. Dependendo do problema, isso pode envolver regressão linear, regressão logística, árvores de regressão / classificação, métodos de vizinho mais próximo, clustering, entre outros.
7. Finalmente, os insights da análise precisam ser implementados. É preciso agir sobre os resultados. Isto é o que W.E. Deming tinha em mente quando falou sobre melhoria de processo: "planejar, fazer, verificar e agir”.

A Importância da Escolha das Variáveis

Observações x Variáveis:

É importante encontrar as variáveis que importam e as que não são relevantes para o problema em questão. A maioria das variáveis coletadas pode ser irrelevante e representar apenas ruído!

A mineração de dados explora e analisa grandes quantidades de dados para descobrir padrões significativos.

A escala de uma aplicação típica de mineração de dados, com seu grande número de casos e muitas variáveis, excede a de uma investigação estatística padrão. É por isso que os pesquisadores se referem à mineração de dados como estatísticas em escala e velocidade.

A mineração de dados tem ampla aplicabilidade, com aplicações em inteligência e análise de segurança, genética, ciências sociais e naturais e negócios.

Processos e Práticas em Business Analytics

Knowledge Discovery in Databases (KDD)

Dados → Seleção (Dados relevantes) → Pré-processamento (Dados pré-processados) → Transformação (Dados transformados) → Data Mining (Padrões) → Interpretação (Conhecimento)

Práticas MAD (Magnetic / Agile / Deep)

Magnetic (Atrair Dados)
Agile (Agilidade do Processo e Integração/Alteração)
Deep Analytics

BPM (Business Process Management) – Ciclo de vida BPM:

Análise e Desenho → Modelagem → Implementação → Monitoramento e Controle → Otimização → Mapeamento e Classificação → Análise e Desenho…

O Que é Análise Descritiva?

Este tipo de análise é muitas vezes relacionado com Business Intelligence (BI), pois fornece o conhecimento necessário sobre o passado, a fim de ajudar tomadores de decisão a fazer previsões futuras.

Essa análise de dados do passado é feita através de agregação de dados e técnicas de mineração de dados para determinar o que pode estar ocorrendo até o presente momento. Isso pode ser usado então para determinar o que provavelmente irá acontecer no futuro. Usando várias técnicas de mineração e processamento de dados nós podemos transformar tais dados em fatos e números, que até então não estavam claros para as pessoas, permitindo dessa forma, utilizar esses dados para planejar ações e estratégias futuras.

Análise Descritiva nos permite aprender a partir dos eventos passados, bem como usar esses dados para antecipar como eles podem influenciar o comportamento futuro. Por exemplo, se estamos cientes do número médio de vendas de produtos que fizemos por mês nos últimos três anos e somos capazes de ver tendências como o aumento ou queda dos números, nós podemos antecipar como essas tendências irão influenciar as vendas futuras ou até mesmo podemos ver quais números tendem a cair com o passar dos anos. Isso significa que nós podemos mudar algo para que as vendas aumentem, quer se trate de uma remodelagem, a troca de marca, expansão da equipe ou até mesmo a criação de um novo produto.

A maioria das estatísticas de uso de negócios em suas operações cotidianas caem na categoria de análise descritiva. O que os estatísticos fazem é justamente coletar os dados descritivos do passado e depois convertê-los em uma linguagem compreensível para a gerência e os funcionários. Usando a

análise descritiva as empresas sabem quanto estão gastando em média em relação as despesas, quanto do percentual de vendas dos produtos está associado as despesas e quanto é realmente lucro líquido. Todas essas informações irão permitir aparar as arestas e obter mais lucros, que é o objetivo de qualquer empresa.

Estatísticas Para Descrever os Dados

Há duas maneiras principais de descrever dados: as medidas de tendência central e medidas de variabilidade ou dispersão.

Quando falamos em medidas de tendência central, estamos nos referindo a medir dados e encontrar o valor médio ou média de um determinado conjunto de dados.

A média é determinada somando-se todos os dados e dividindo-os pelo número de unidades de dados, obtendo-se um valor médio que pode ser usado de várias maneiras.

Outra unidade utilizada na medição da tendência central - que talvez seja ainda mais útil - é a mediana. Ao contrário da média, a mediana leva em consideração apenas o valor médio de um determinado conjunto de dados. Por exemplo, em uma sequência de nove números, o número 5 é considerado a mediana. Se colocarmos os números ordenados do menor para o maior, a mediana será muitas vezes um valor mais realista do que a média porque pode haver outliers (valores extremos) em qualquer extremidade do conjunto dados, que dobre a média transformando-a assim em um número errado. Os outliers são números extremamente pequenos ou grandes que naturalmente tornarão a média irrealista, e a mediana será mais útil nos casos em que haja outliers. Daqui a pouco praticaremos isso na linguagem R.

As medidas de dispersão ou variabilidade permite-nos ver como está a difusão dos dados a partir de um valor central ou a média. A variância e desvio padrão são os valores utilizados para medir a dispersão. O intervalo é calculado subtraindo o menor número do maior. Este valor também é muito sensível a outliers, pois você poderia ter um número extremamente pequeno ou grande nas extremidades do seu conjunto de dados.

A variância é a medida do desvio que nos diz a distância média de um ponto de dados da média. A variância é tipicamente usada para calcular o desvio padrão, por seu valor, ela terá pouco propósito.

Desvio padrão é o método de dispersão mais popular, pois fornece a distância média dos pontos de dados a partir da média. Tanto a variância quanto o desvio padrão serão elevados nos casos em que os dados estejam muito espalhados. Você encontrará o desvio padrão calculando a variância e então encontrando sua raiz quadrada. O desvio padrão será um número na mesma unidade que os dados originais, o que torna mais fácil de interpretar do que a variância. Todos os valores utilizados para calcular a tendência central e a dispersão de dados podem ser empregados para fazer várias inferências, o que pode ajudar com as previsões futuras feitas pela análise preditiva.

Análise Descritiva x Análise Diagnóstica

Quando consideramos que a ideia de Big Data Analytics é compreender o que um volume grande de dados pode nos dizer, os caminhos analíticos são infinitos. Há, no entanto, quatro tipos de análises que se destacam pela usabilidade e potencialidade de seus resultados: análise descritiva, análise diagnóstica, análise preditiva e análise prescritiva. Já falamos sobre a análise descritiva e agora vamos diferenciá-la da análise diagnóstica. Veremos as análises preditiva e prescritiva mais a frente.

Análise Descritiva

Este tipo de análise responde à pergunta: “O que aconteceu?”. Análise descritiva é baseada em dados históricos e dados atuais. Compreensão em tempo real dos acontecimentos é o que define a análise descritiva. É a mineração de dados na base da cadeia de Big Data. Um exemplo da sua aplicação é a análise de crédito feita por instituições financeiras. Elas analisam as informações de um indivíduo, de uma empresa ou de um grupo social para compreender os riscos envolvidos na concessão de crédito, tudo com base em informações que estão lá, colhidas conforme o tempo. Dessa análise vem a definição de taxas de juros em financiamentos. É uma maneira de visualizar os dados, entender como um conjunto de dados se organiza e o que significa para o presente, sem necessariamente relacioná-la com padrões. Apenas descrevemos os dados.

Análise Diagnóstica

Usa-se esse tipo de análise para encontrar respostas para a pergunta: “por que algo específico aconteceu?” Ou “o que deu errado?” Análises de diagnóstico são úteis para deduzir e inferir o sucesso ou o fracasso de subcomponentes de qualquer iniciativa centrada em dados. Enquanto a análise descritiva busca detalhar uma base de dados, a análise diagnóstica tem como objetivo compreender de maneira causal (Quem, Quando, Como, Onde e Por que) todas as suas possibilidades. Se uma empresa executa uma ação de marketing, por exemplo, a análise diagnóstica é o caminho mais curto e eficiente para que os profissionais avaliem os impactos e o alcance dessa ação após sua realização.

Como uma espécie de relatório expandido, quando feita em uma base de dados volumosa, esse tipo de análise permite ainda entender a razão de cada um dos desdobramentos das ações adotadas e, a partir disso, mudar estratégias ineficazes ou reforçar as funcionais. Big Data Analytics opera de várias formas diferentes. Os dados, sozinhos, não podem fazer nada. Quem os analisa é responsável por aplicá-los da melhor maneira possível. Cada tipo de análise tem seu próprio escopo e sua própria finalidade. Por isso é preciso compreendê-las bem.

O Que é Análise Preditiva?

Análise Preditiva é a prática de extrair informações de conjuntos de dados, a fim de determinar padrões e resultados futuros. Perceba que não existe mágica: a análise preditiva não prevê o que vai acontecer exatamente no futuro. Ela prevê o que pode acontecer no futuro com um nível aceitável de confiabilidade e inclui cenários hipotéticos e avaliação de riscos.

Prever o futuro é um desejo comum entre as pessoas. Se você fosse capaz de saber com 6 meses de antecedência que uma grande crise econômica irá assolar o seu país, o que faria? Será que você conseguiria criar um plano para prevenir ou diminuir o impacto do problema? Poderia mudar o rumo da história? Com a ajuda de estratégias de análise preditiva, sua resposta pode ser “sim”. Como diz o ditado, não é magia, é tecnologia! Análise preditiva não é bola de cristal, mas sim o trabalho de analisar um cenário específico e traçar possíveis tendências e mudanças capazes de afetar seu planejamento estratégico.

Análise preditiva é realizada através de uma série de técnicas analíticas e estatísticas, utilizadas para o desenvolvimento de modelos que podem prever eventos futuros a partir de comportamentos diários, incluindo análise de séries temporais ou modelos de regressão. Existem diferentes formas de modelos preditivos, que variam de acordo com o evento ou comportamento que está sendo

previsto. Quase todos os modelos preditivos produzem uma pontuação; uma pontuação mais elevada indica que um dado evento ou comportamento é muito provável que ocorra.

É óbvio que, muitas das vezes, esse tipo de trabalho lida com volumes gigantescos de dados e, por isso, exige o uso de ferramentas de Big Data Analytics para ser executado. Os resultados, no entanto, são inestimáveis: não há nada mais precioso para um bom gestor do que conseguir prever tendências do mercado.

Aplicada aos negócios, análise preditiva é usada para analisar dados históricos, a fim de compreender melhor o comportamento futuro de clientes, produtos e parceiros e para identificar riscos e oportunidades potenciais para uma empresa. Utiliza-se uma série de técnicas, incluindo a mineração de dados, modelagem estatística e Machine Learning para ajudar os analistas a realizarem previsões.

Soluções para construção de modelos de análise preditiva (Gratuitas):

• Linguagem R – sem dúvida uma das mais utilizadas atualmente e a base para muitas outras soluções (inclusive as proprietárias).
• NumPy e SciPy – pacotes de computação científica em Python.
• Scikit-Learn – conjunto de pacotes para Machine Learning.
• Orange – ferramenta de visualização e análise. O data mining pode ser feito utilizando scripts em Python.
• Weka – conjunto de algoritmos para Machine Learning e data mining.
• Octave – o Octave é muito parecido com o Matlab.
• Data Science Studio (DSS Community Edition) – plataforma com todas as ferramentas necessárias para análise de Big Data e geração rápida de resultados de análise.
• Apache Spark MLLib – engine para processamento em larga escala, com diversos algoritmos poderosos para análise de regressão, classificação e muito mais.
• TensorFlow – biblioteca para construção de modelos avançados usados principalmente em Inteligência Artificial.

Soluções para construção de modelos de análise preditiva (Proprietárias):

• Oracle Data Miner (ODM)
• SAS Predictive Analytics
• IBM Predictive Analytics
• SAP Predictive Analytics
• STATISTICA
• MATLAB
• Minitab
• RapidMiner
• GraphLab Create
• TIBCO Analytics
• Data Science Studio

Análise Preditiva x Análise Prescritiva

O grande diferencial do Business Analytics em relação ao Business Intelligence, é exatamente a possibilidade de criarmos modelos preditivos capazes de realizar previsões com alto nível de precisão. Enquanto o BI tradicional tem o foco em explicar o passado, o Business Analytics tem o

foco em prever o futuro. E claro, o BI pode ser usado como ferramenta de apoio para isso. A análise preditiva e prescritiva são os 2 tipos de Analytics voltados para fazer previsões. Vamos defini-las.

Análise Preditiva

Embora este tipo de análise seja baseado em dados históricos e atuais, a análise preditiva vai um passo além do que análises descritivas. A análise preditiva envolve a construção de modelos complexos de análise, a fim de prever um evento futuro ou tendência. Essas análises seriam realizadas pelo Cientista de Dados. A partir da identificação de padrões passados em sua base dados, esse tipo de análise permite aos gestores o mapeamento de possíveis acontecimentos futuros em seus campos de atuação. A ideia é deixar de tomar decisões baseadas unicamente na intuição, conseguindo estabelecer um prognóstico mais sólido para cada ação. Conhecida por “prever” o futuro, a análise preditiva usa mineração de dados, modelos estatísticos e dados históricos para conhecer as futuras tendências. A análise preditiva é o tema principal aqui do nosso curso.

Análise Prescritiva

Esse tipo de análise tem como objetivo otimizar processos, estruturas e sistemas através de ações baseadas em análise preditiva – essencialmente dizendo o que você deve fazer com base em uma estimativa do que vai acontecer. Ambos os Analistas de Negócios e Cientistas de Dados podem gerar análises prescritivas, mas os seus métodos e fontes de dados podem ser diferentes. Muito confundida com a análise preditiva, a análise prescritiva trabalha com a mesma lógica, porém com objetivos diferentes.

Enquanto a análise preditiva identifica tendências futuras, a prescritiva traça as possíveis consequências de cada ação. É uma forma de definir qual escolha será mais efetiva em determinada situação. Mesmo assim é pouco utilizada, na maioria das vezes, por causa de desconhecimento – segundo o Gartner, apenas 3% das empresas fazem uso dessa análise.

Dentro de uma indústria ou setor, o valor dessa análise se dá pela capacidade de numerar determinados padrões e filtrá-los por especificidades, obtendo um cenário bastante fiel da situação e como cada intervenção responderá. Na área de saúde, por exemplo, os gestores dos planos podem traçar padrões de determinados pacientes e doenças e, com isso, analisar possíveis impactos de ações sobre esse grupo, analisando qual a melhor opção de gestão.

Estatística Elementar em R / Exercício / Análise Descritiva em R

```
# Lab - Estatística Elementar em R

# Obs: Caso tenha problemas com a acentuação, consulte este link:
# https://support.rstudio.com/hc/en-us/articles/200532197-Character-Encoding

# Configurando o diretório de trabalho
# Coloque entre aspas o diretório de trabalho que você está usando no seu computador
# Não use diretórios com espaço no nome
setwd("~/Dropbox/DSA/Business-Analytics2.0/Cap02/R")
getwd()


# Dataset
install.packages("mlbench")
library(mlbench)
data(PimaIndiansDiabetes)
View(PimaIndiansDiabetes)
```

```
# Melhorando o nome do dataset
dataset <- PimaIndiansDiabetes
View(dataset)

# Dimensões e tipos dos dados
?dim
dim(dataset)
str(dataset)

# Sumário dos dados
summary(dataset)
summary(dataset$age)

# A média de uma variável é uma medida numérica da localização central dos valores dos dados.
# É a soma de seus valores dividido pela contagem de dados.
mean(dataset$age)
mean(dataset$age, na.rm = TRUE)
is.na(dataset$age)

# A mediana de uma variável é o valor no meio quando os dados são classificados em ordem crescente.
# É uma medida ordinal da localização central dos valores de dados.
median(dataset$age)

# Existem vários quartis de uma variável. O primeiro quartil, ou quartil inferior, é o valor
# que corta os primeiros 25% dos dados quando é classificado em ordem crescente.
# O segundo quartil, ou mediana, é o valor que corta os primeiros 50%.
# O terceiro quartil, ou quartil superior, é o valor que corta os primeiros 75%.
quantile(dataset$age)
quantile(dataset$age, c(.32, .57, .98))

# O intervalo (range) de uma variável é a diferença de seus maiores e menores valores de dados.
# É uma medida de quão distante os dados se espalham.
range(dataset$age)

# A variação interquartil de uma variável é a diferença de seus quartis superior e inferior.
# É uma medida de quão distante a parte média dos dados se espalha.
IQR(dataset$age)

# A variância é uma medida numérica de como os valores dos dados estão dispersos em torno da média.
var(dataset$age)

# O desvio padrão de uma variável é a raiz quadrada de sua variância.
# O desvio padrão mede a dispersão dos seus dados (quão os dados estão distantes da média).
sd(dataset$age)

# A covariância de duas variáveis x e y em um conjunto de dados mede como as duas variáveis
# estão linearmente relacionadas.
# Uma covariância positiva indicaria uma relação linear positiva entre as variáveis, e uma
# covariância negativa indicaria o contrário.
cov(dataset$age, dataset$glucose)

# O coeficiente de correlação de duas variáveis em um conjunto de dados é igual a sua covariância
# dividida pelo produto de seus desvios-padrão individuais. É uma medida normalizada de como os
# dois estão linearmente relacionados. Os valores vão de -1 a +1. Valores próximos de zero indicadm
# que não há correlação. Valor de -1 indica forte correlação negativa e +1 forte correlação positiva.
cor(dataset$age, dataset$glucose)
correlacoes <- cor(dataset[,1:8])
print(correlacoes)

install.packages("corrplot")
library(corrplot)
corrplot(correlacoes, method = "circle")


# Pacote para as funções skewness e kurtosis
install.packages("e1071")
library(e1071)

# Em estatística, a assimetria (skewness) é uma medida da distorção da distribuição de
# probabilidade de uma variável aleatória sobre sua média. Em outras palavras, a assimetria
# informa a quantidade e a direção da inclinação (partida de simetria horizontal).

# O valor de assimetria pode ser positivo ou negativo ou ainda indefinido. Se a assimetria é 0,
# os dados são perfeitamente simétricos, embora seja bastante improvável para dados do mundo real.

# Via de regra:

# Se a assimetria é menor que -1 ou maior que 1, a distribuição é altamente distorcida.
```

```
# Se a assimetria é entre -1 e -0,5 ou entre 0,5 e 1, a distribuição é enviesada moderadamente.
# Se a assimetria é entre -0,5 e 0,5, a distribuição é aproximadamente simétrica.

# A medida positiva indicaria que a média dos valores dos dados é maior do que a mediana e
# a distribuição dos dados é desviada para a direita.
# Uma medida negativa indica que a média dos valores dos dados é menor que a mediana e a
# distribuição dos dados é inclinada para a esquerda.
?skewness
skewness(dataset$age)
hist(dataset$age)

skewness(dataset$pressure)
summary(dataset$pressure)
hist(dataset$pressure)

# A curtose informa a altura e a nitidez do pico central, em relação a uma curva de sino padrão.
# A distribuição normal tem kurtosis igual a zero.
kurtosis(dataset$age)
hist(dataset$age)

kurtosis(dataset$pressure)
hist(dataset$pressure)

# Exercício
# Considere os dados de exemplo a seguir.
# Calcule as estatísticas elementares e interprete o resultado da assimetria e da curtose.
# A distribuição dos dados é normal?

# Dados de exemplo
dados <- rnorm(n = 10000, mean = 55, sd = 4.5)
View(dados)

# Solução na próxima aula!
```

```
# Estatística Elementar em R - Exercício

# Obs: Caso tenha problemas com a acentuação, consulte este link:
# https://support.rstudio.com/hc/en-us/articles/200532197-Character-Encoding

# Configurando o diretório de trabalho
# Coloque entre aspas o diretório de trabalho que você está usando no seu computador
# Não use diretórios com espaço no nome
setwd("~/Dropbox/DSA/Business-Analytics2.0/Cap02/R")
getwd()

# Considere os dados de exemplo a seguir.
# Calcule as estatísticas elementares e interprete o resultado da assimetria e da curtose.
# A distribuição dos dados é normal?

# Dados de exemplo
set.seed(140)
dados <- rnorm(n = 10000, mean = 55, sd = 4.5)
View(dados)
summary(dados)
mean(dados)

# Converte os dados para um dataframe para facilitar a manipulação
dataset <- data.frame(dados)

# Renomeia a coluna
colnames(dataset) <- c("Medida")
View(dataset)

# Estatísticas Elementares
mean(dataset$Medida)
median(dataset$Medida)
var(dataset$Medida)
sd(dataset$Medida)

# Moda
table(as.vector(dataset$Medida))
table(as.vector(round(dataset$Medida)))
moda <- table(as.vector(round(dataset$Medida)))
names(moda)[moda == max(moda)]

# Sumário
summary(dataset)
```

```
# Assimetria e Curtose
library(e1071)

skewness(dados)
kurtosis(dados)

# Usando a função describe
library(psych)
describe(dataset)

########## Assimetria ##########

# Assimetria é o grau de afastamento de uma distribuição da unidade de assimetria.
# Uma Distribuição é Simétrica quando seus valores de Média, Mediana e Moda coincidem.
# A Assimetria, dá, portanto, uma indicação da inclinação da distribuição.

# O coeficiente de assimetria (skewness) permite distinguir as distribuições assimétricas.
# Um valor negativo indica que a cauda do lado esquerdo da função densidade de probabilidade
# é maior que a do lado direito. Um valor positivo para a assimetria indica que a cauda do
# lado direito é maior que a do lado esquerdo. Um valor nulo indica que os valores são
# distribuídos de maneira relativamente iguais em ambos os lados da média, mas não implica
# necessariamente, uma distribuição simétrica.

# A assimetria dos dados simulados é negativa e próxima de zero.
# Isso conclui que os dados estão próximos de uma distribuição gaussiana (formato de sino),
# mas ligeiramente inclinados para a esquerda.


########## Curtose ##########

# Curtose é  uma medida de dispersão que caracteriza o "achatamento" da curva da função
# de distribuição. O que significa analisar um conjunto quanto à Curtose?

# Significa apenas verificar o “grau de achatamento da curva”. Ou seja, saber se a Curva de
# Frequência que representa o conjunto é mais “afilada” ou mais “achatada” em relação a uma
# Curva Padrão, chamada de Curva Normal!

# Uma curva (um conjunto) poderá ser, quanto à sua Curtose:

# Mesocúrtica: ou de curtose média!
# Será essa a nossa Curva Normal. “Meso” lembra meio! Esta curva está no meio termo:
# nem muito achatada, nem muito afilada;

# Platicúrtica: é a curva mais achatada. Seu desenho lembra o de um prato emborcado.
# Então “prato” lembra “plati” e “plati” lembra “platicúrtica”;

# Leptocúrtica: é a curva mais afilada!

# Quando se trata de Curtose, não há como extrairmos uma conclusão sobre qual será a situação
# da distribuição – se mesocúrtica, platicúrtica ou leptocúrtica – apenas conhecendo os valores
# da Média, Moda e Mediana.

# Histograma
library(ggplot2)

# Plot
ggplot(dataset, aes(x = dataset$Medida), binwidth = 20) +
  geom_histogram(aes(y = ..density..), fill = 'red', alpha = 0.5) +
  geom_density(colour = 'blue') + xlab(expression(bold('Dados de Exemplo'))) +
  ylab(expression(bold('Densidade')))
```

```
# Análise Descritiva em R

# Obs: Caso tenha problemas com a acentuação, consulte este link:
# https://support.rstudio.com/hc/en-us/articles/200532197-Character-Encoding

# Configurando o diretório de trabalho
# Coloque entre aspas o diretório de trabalho que você está usando no seu computador
# Não use diretórios com espaço no nome
setwd("~/Dropbox/DSA/Business-Analytics2.0/Cap02/R")
getwd()

# Carregando os dados
carros <- read.csv("dados/carros.csv", stringsAsFactors = FALSE)

# Sumário da estrutura do dataset
```

```
str(carros)
names(carros) <- c('Ano', 'Modelo', 'Preco', 'Kilometragem', 'Cor', 'Transmissao')
View(carros)

##### Explorando Variáveis Numéricas #####

# Sumário
summary(carros$Ano)
str(carros)
carros$Ano <- as.character(carros$Ano)
summary(carros$Ano)
summary(carros[c("Preco", "Kilometragem")])

# Calculando a média
(36000 + 44000 + 56000) / 3
mean(c(36000, 44000, 56000))
mean(carros$Preco)

# Mediana
median(c(36000, 44000, 56000))
median(carros$Preco)

# Range - Min e Max
range(carros$Preco)

# Diferença do Range (Max -  Min)
diff(range(carros$Preco))

# IQR - Interquantile Range
IQR(carros$Preco)

# Quartis
quantile(carros$Preco)

# Percentil 1% e 99%
quantile(carros$Preco, probs = c(0.01, 0.99))

# Percentis com intervalo de 20
quantile(carros$Preco, seq(from = 0, to = 1, by = 0.20))

# Boxplot
boxplot(carros$Preco, main = "Boxplot Preços", ylab = "Preço (R$)")
boxplot(carros$Kilometragem, main = "Boxplot Kilometragem", ylab = "Kilometragem (Km)")

# Histograma
mean(carros$Preco)
hist(carros$Preco, main = "Histograma Preços", xlab = "Preço (R$)")

mean(carros$Kilometragem)
hist(carros$Kilometragem, main = "Histograma Kilometragem", xlab = "Kilometragem (Km)")

# Assimetria e Curtose da variável Preço
library(e1071)
skewness(carros$Preco)
kurtosis(carros$Preco)

# Variância e Desvio Padrão
mean(carros$Preco)
var(carros$Preco)
sd(carros$Preco)

mean(carros$Kilometragem)
var(carros$Kilometragem)
sd(carros$Kilometragem)

# Explorando relacionamento entre as variáveis numéricas

# Scatter Plot
plot(x = carros$Kilometragem, y = carros$Preco,
     main = "Scatterplot Kilometragem x Preço",
     xlab = "Kilometragem (Km)",
     ylab = "Preço (R$)")

# Calculando o coeficiente de correlação
cor(carros$Kilometragem, carros$Preco)

# Agregação
?aggregate
```

```
# Média de preços dos carros por ano
str(carros)
aggregate(carros$Preco ~ carros$Ano, FUN = mean, data = carros)


#####  Explorando Variáveis Categóricas #####

# Tabela de Frequência
str(carros)
?table
table(carros$Ano)
table(carros$Modelo)
table(carros$Cor)

# Proporções da Tabela de Frequência
model_table <- table(carros$Modelo)
prop.table(model_table)

# Ajuste do resultado das proporções
color_table <- table(carros$Cor)
color_pct <- prop.table(color_table) * 100
round(color_pct, digits = 1)

# Resumo gráfico e relação entre as variáveis categóricas
library(ggplot2)

# Total de veículos por tipo de transmissão
ggplot(data = carros, aes(x = as.factor(Transmissao))) +
  geom_bar(aes(y = (..count..))) +
  labs(x = "Transmissao", y = "Contagem de Carros Por Tipo de Transmissao")

str(carros)
carros$Transmissao <- as.factor(carros$Transmissao)
str(carros)

# Proporção de veículos por tipo de transmissão e por cor
ggplot(carros, aes(x = as.factor(Transmissao))) +
  geom_bar(aes(y = (..count..)/sum(..count..))) +
  xlab("Transmissao") +
  scale_y_continuous(labels = scales::percent, name = "Proporção") +
  facet_grid(~ Cor) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

O Que é Probabilidade?

Probabilidade é o estudo da aleatoriedade e incerteza. É a quantificação do conhecimento que temos sobre um particular evento.

Probabilidade, que mede a possibilidade de que um evento venha a ocorrer, representa uma parte importante da estatística. Representa a base da estatística inferencial, que por sua vez é a base do aprendizado de máquina.

Na estatística inferencial, tomamos decisões em condições de incerteza. A teoria da probabilidade é utilizada para avaliar a incerteza envolvida nessas decisões. Por exemplo, a estimativa das vendas para o próximo ano, para uma empresa, é baseada em muitas premissas, algumas das quais podem vir a ser verdadeiras, enquanto outras não. A teoria da probabilidade irá nos ajudar a tomar decisões em tais condições de informações imperfeitas e incertas.

A combinação entre probabilidade e distribuições de probabilidades com a estatística descritiva irá nos ajudar a tomar decisões sobre populações com base em informações obtidas a partir de amostras.

Probabilidade é um valor numérico que indica a chance, ou probabilidade, de um evento específico ocorrer. Este valor numérico vai estar entre 0 e 1.

• Se um evento não possui chance de ocorrer, sua probabilidade é 0 (ou 0%).
• Se temos certeza sobre a ocorrência do evento, sua probabilidade é 1 (ou 100%).

Experimento, Espaço da Amostra e Evento

Experimento é o processo de medir ou observar uma atividade com o propósito de coletar dados. Exemplo: jogar um dado.

Espaço da Amostra representa todos os possíveis resultados de um experimento. Exemplo: ao jogar um dado, todos os resultados possíveis são {1, 2, 3, 4, 5, 6}.

Experimento	**Espaço da Amostra**
Jogar uma moeda	{cara, coroa}
Responder uma questão de múltipla escolha	{a, b, c, d, e}
Inspecionar um produto	{defeituoso, não defeituoso}
Puxar uma carta de um baralho padrão	{52 cartas de uma baralho padrão}

Evento representa um ou mais resultados de um experimento. O resultado e/ou resultados são um subconjunto do espaço da amostra. Os eventos podem ser:

• Evento Simples – um único resultado de um experimento.
• Evento Composto – mais de um resultado de um experimento.

Regras Básicas que Regem a Teoria da Probabilidade

A Teoria da Probabilidade possui 5 regras básicas que servem de guia em qualquer trabalho de análise. São elas:

1- A probabilidade de qualquer evento sempre será entre 0 e 1. Probabilidades nunca podem ser negativas ou maior que 1.
2- A soma de todas as probabilidades para um evento simples, em um espaço de amostra, será igual a 1.
3- Se P(A) = 1, então podemos garantir que o evento A ocorrerá.
4- Se P(A) = 0, então podemos garantir que o evento A não ocorrerá.
5- P(A) = 1 – P(A’), onde P(A’) é o complemento do evento A

O Que São Distribuições de Probabilidade?

Uma distribuição de probabilidade descreve como os valores de uma variável aleatória são distribuídos.

Por exemplo, a coleção de todos os resultados possíveis de uma sequência de lançamento de moeda é conhecida por seguir a distribuição de probabilidade binomial.

Médias de amostras suficientemente grandes de uma população de dados são conhecidos por se assemelhar a distribuição de probabilidade normal. E assim por diante.

Como as características dessas distribuições teóricas são bem compreendidas, elas podem ser usadas para fazer inferências estatísticas em toda a população de dados como um todo. Muitos algoritmos de Machine Learning são baseados em distribuições de probabilidade.

As distribuições podem ser criadas para variáveis aleatória discretas (que assumem valores inteiros, como 1, 2, 3, etc...) ou para variáveis aleatórias contínuas (que assumem valores decimais, como 23.89, 19.87, 45.3, etc...).

Distribuições de Probabilidade e Função de Densidade

Probabilidade é um valor numérico que representa uma chance, uma eventualidade ou uma possibilidade de que um determinado evento venha a acontecer.

Existem três tipos de probabilidade:

- Probabilidade clássica: usada quando cada resultado no espaço amostral tem a mesma probabilidade de ocorrer. A probabilidade é baseada no conhecimento prévio do processo envolvido.
- Probabilidade empírica: baseia-se em observações obtidas de experimentos aleatórios. Os resultados são baseados em dados observados e não no conhecimento prévio do processo. De acordo com a Lei dos grandes números, a medida que um experimento é repetido mais e mais vezes, a probabilidade empírica (frequência relativa) de um evento tende à sua probabilidade real.
- Probabilidade subjetiva: intuição estimativa ou palpite. Normalmente baseada em experiência no passado, opinião pessoal ou análise de algum indivíduo. Pode ser útil, quando não há possibilidade de utilização da probabilidade clássica ou empírica.

Definições:

Experimento ou Fenômeno Aleatório:

São aqueles que, mesmo repetidos várias vezes sob condições semelhantes, apresentam resultados imprevisíveis.

Espaço Amostral:

É o conjunto de possíveis resultados de um experimento ou fenômeno aleatório, representado por S.

Evento:

É qualquer subconjunto do espaço amostral S de um evento aleatório.

Probabilidade:

Dado um experimento aleatório, sendo S o seu espaço amostral, admitindo que todos os elementos de S tenham a mesma chance de acontecer, ou seja, que S é um conjunto equiprovável.

Probabilidade de um evento A (A está contido em S):

$$P(A) = \frac{n(A)}{n(S)}$$

Sendo:

n(A) – número de elementos de A;
n(S) – número de elementos de S.

Eventos Complementares

Sabendo que um evento pode ocorrer ou não. Sendo p a probabilidade de que ele ocorra (sucesso) e q a probabilidade de que ele não ocorra (insucesso), para um mesmo evento existe sempre a relação:

$$p + q = 1 \Rightarrow q = 1 - p$$

Eventos Independentes:

Dizemos que dois eventos são independentes quando a realização ou não realização de um dos eventos não afeta a probabilidade da realização do outro e vice-versa.

Assim, sendo p1 a probabilidade de realização do primeiro evento e p2 a probabilidade do segundo evento, a probabilidade de que tais eventos se realizem simultaneamente é dada por:

$$p = p_1 \cdot p_2$$

Também conhecida como regra do "e".

Eventos Mutuamente Exclusivos:

Dizemos que dois ou mais eventos são mutuamente exclusivos quando a realização de um exclui a realização do(s) outro(s).

Assim, no lançamento de uma moeda, o evento “tirar cara” e o evento “tirar coroa” são mutuamente exclusivos, já que, ao se realizar um deles, o outro não se realiza.

Se dois eventos são mutuamente exclusivos, a probabilidade de que um ou outro se realize é igual à soma das probabilidades de cada um deles se realize:

$$p = p_1 + p_2$$

Também conhecida como regra do "ou".

Distribuição Binomial
Aplica-se a experimentos que satisfaçam as seguintes condições:

• O experimento deve ser repetido, nas mesmas condições, um número finito de vezes, n;
• As provas repetidas devem ser independentes, o resultado de uma não afeta o resultado da outra;
• Tem-se apenas dois resultados possíveis: sucesso ou insucesso;
• A probabilidade do sucesso em uma tentativa é p e a do insucesso é: q = 1 - p
• A probabilidade de se obter sucesso k vezes durante n tentativas é determinado por:

$$f(x) = P(x = k) = \frac{n!}{k!(n-k)!} \cdot p^k \cdot q^{n-k}$$

Sendo:

n = número de tentativas
K = número de sucessos
p = probabilidade de sucesso
q = probabilidade de fracasso

Distribuição de Poisson:

Muitos estudos são baseados na contagem das vezes em que um evento ocorre em uma determinada área de oportunidades.

Uma área de oportunidades é uma unidade contínua ou um intervalo de tempo, volume ou área em que possa acontecer mais de uma ocorrência de um evento. Exemplos:

• Defeitos na pintura de uma geladeira nova;
• Número de falhas na rede em um determinado dia;
• Número de pulgas no pelo de um cachorro.

Pode-se utilizar a distribuição de Poisson para calcular probabilidades se:

• O interesse é encontrar o número de vezes em que um evento específico ocorre em uma determinada área de oportunidades.
• A probabilidade de que um evento específico ocorra em uma área de oportunidades é a mesma para todas as áreas de oportunidades.
• número de eventos que ocorrem em uma determinada área de oportunidades é independente do número de eventos que ocorrem em qualquer outra área de oportunidades.
• A probabilidade de que dois ou mais eventos venham a ocorrer em uma determinada área de oportunidades se aproxima de zero à medida em que a área de oportunidades de torna menor.

$$P(x) = \frac{e^{-k} k^x}{X!}$$

Onde:

P(x) = probabilidade de eventos
K = número esperado de eventos
e = constante matemática = 2,71828
X = número de eventos

Distribuição Normal

A distribuição normal pode ser considerada como a mais importante distribuição de probabilidade, pode ser aplicada em vários fenômenos. Também é conhecida como distribuição de Gauss.

A Curva Normal é um modelo teórico ou ideal que resulta muito mais de uma equação matemática do que um real delineamento de pesquisa com posterior coleta de dados.

A distribuição normal é importantíssima para a Estatística pelas seguintes razões:

• Muitas variáveis contínuas possuem distribuição que se aproximam da normal;
• Pode ser utilizada para aproximações de várias distribuições discretas;
• Proporciona a base para a inferência estatística, pois possui relação direta com o Teorema Central do Limite.

Propriedades:

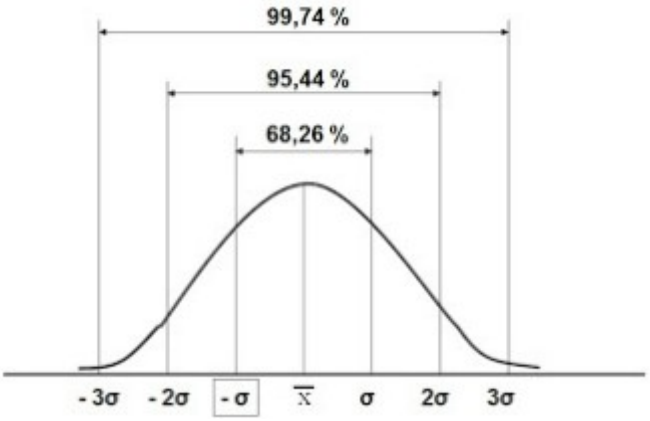
• A variável “x” pode assumir qualquer valor no conjunto dos números reais.
• Seu gráfico possui formato de sino e a curva é totalmente simétrica.
• A média, a moda e a mediana possuem o mesmo valor.
• Em seu gráfico, a curva normal aproxima-se do eixo das abscissas infinitamente, mas sem alcançá-lo.
• Como a curva é simétrica, o valor de sua área é 1 e a probabilidade de ocorrer um valor menor que a média é o mesmo que ocorrer um valor maior que a média (0,5).
• A maioria dos valores agrupa-se em torno do centro (a média);
• Os valores vão se tornado cada vez menos prováveis, quanto mais distantes eles se encontram da média.

Função de Densidade da Probabilidade Normal:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}, -\infty < x < \infty$$

Onde:

μ = média
σ = desvio-padrão
π = constante aproximada por 3,1416
e = constante aproximada por 2,71828
x = qualquer valor da variável



Quando se tem uma variável com distribuição normal, é possível calcular a probabilidade de essa variável assumir um valor em um intervalo.

A Distribuição Normal Padronizada:

A distribuição normal padronizada possui média 0 (zero) e desvio padrão 1. As probabilidades da curva normal padrão são obtidas em tabelas. A vantagem da curva padronizada é que os a média e o desvio padrão, estão definidos para qualquer escala de medida.

Para converter qualquer distribuição normal para a distribuição normal padrão deve-se utilizar a seguinte fórmula:

$$Z = \frac{x - \mu}{\sigma}$$

Tabela da Distribuição Normal Padronizada:

Podemos encontrar vários tipos de tabelas que fornecem as probabilidades sob a curva normal padrão, sendo a mais utilizada a Tabela de faixa Central. Essa tabela fornece a área entre a média (0) e qualquer valor à direita (positivos). Como a curva é simétrica, podem-se obter também as probabilidades à esquerda da média (valores negativos).

A tabela fornece a área (probabilidade) no intervalo entre 0 e z1, ou seja:

$$P(0 \leq z \leq z_1)$$

Distribuição de Probabilidade Para Variável Discreta / Distribuição de Probabilidade Para Variável Contínua

```
# Distribuições de Probabilidade Para Variáveis Discretas

# Obs: Caso tenha problemas com a acentuação, consulte este link:
# https://support.rstudio.com/hc/en-us/articles/200532197-Character-Encoding

# Configurando o diretório de trabalho
# Coloque entre aspas o diretório de trabalho que você está usando no seu computador
# Não use diretórios com espaço no nome
setwd("~/Dropbox/DSA/Business-Analytics2.0/Cap02/R")
getwd()


# Distribuição de Probabilidade Para Variável Discreta - Qual a Probabilidade de Responder Corretamente um Exame Final?


######---------- Distribuição Binomial ----------######

# Na teoria da probabilidade e na estatística, a distribuição binomial com os parâmetros n e p é a distribuição discreta de
# probabilidade do número de sucessos em uma sequência de n experimentos independentes, cada um fazendo uma pergunta sim-não
# e cada um com seu próprio resultado com valor booleano: sucesso / sim / verdadeiro / um (com probabilidade p) ou
# falha / não / falso / zero (com probabilidade q = 1 - p).
```

```
# Um único experimento de sucesso / fracasso também é chamado de Trial de Bernoulli ou Experimento de Bernoulli e
# uma sequência de resultados é chamada de Processo de Bernoulli; para uma única tentativa, ou seja, n = 1, a distribuição
# binomial é uma Distribuição de Bernoulli.

# Portanto, a distribuição binomial é uma distribuição de probabilidade discreta, que descreve o resultado de n experimentos
# independentes. Cada ensaio é suposto ter apenas dois resultados, seja sucesso ou fracasso.

# Em R, temos duas funções principais que nos ajudam a calcular as probabilidades:

# dbinom é uma função de massa de probabilidade da distribuição binomial, enquanto
# pbinom é uma função de distribuição cumulativa dessa distribuição.

# O primeiro diz a você o que é P (X = x) (probabilidade de observar valor igual a x)
# enquanto o segundo, o que é P (X≤x) (probabilidade de observar valor menor ou igual a x)

# Exemplo:

# Suponha que haja 12 perguntas de múltipla escolha em um exame (para ser aprovado precisa acerta 6 ou mais questões).
# Cada pergunta tem 5 respostas possíveis, e apenas 1 delas está correta.

# Encontre a probabilidade de ter 6 respostas corretas se um aluno tentar responder a todas as perguntas aleatoriamente.

# Uma vez que apenas uma entre cinco respostas possíveis está correta, a probabilidade de responder a
# uma pergunta corretamente por aleatoriedade é 1/5 = 0,2. Podemos encontrar a probabilidade de ter
# exatamente 6 respostas corretas por tentativas aleatórias como segue.

# A função dbinom fornece a distribuição de densidade de probabilidade em cada ponto.
# A distribuição binomial requer dois parâmetros extras, o número de tentativas e a probabilidade de sucesso
# de uma única tentativa.
help(Binomial)
?dbinom
dbinom(6, size = 12, prob = 0.2)
dbinom(6, size = 12, prob = 0.2) * 100

# Vamos plotar todas as possibilidades
x <- dbinom(0:12, size = 12, p = 0.2)
barplot(x, names.arg = 0:12, space = 0)

# Qual a probabilidade de ter 5 ou menos perguntas respondidas corretamente de forma aleatória, no questionário
# de 12 questões de múltipla escolha?

# Poderíamos resolver assim:
dbinom(0, size=12, prob=0.2) +
dbinom(1, size=12, prob=0.2) +
dbinom(2, size=12, prob=0.2) +
dbinom(3, size=12, prob=0.2) +
dbinom(4, size=12, prob=0.2) +
dbinom(5, size=12, prob=0.2)

# Ou assim:
# A função pbinom fornece a probabilidade cumulativa de um evento. É um valor único que representa a probabilidade.
?pbinom
pbinom(5, size = 12, prob = 0.2)

# Vamos plotar todas as possibilidades
x <- pbinom(0:12, size = 12, p = 0.2)
barplot(x, names.arg = 0:12, space = 0)


# Distribuição de Probabilidade Para Variável Discreta - Qual a Probabilidade de Ter Um Número Específico de Vendas Por Semana?


######---------- Distribuição Poisson ----------######

# Na teoria das probabilidades e estatística, a Distribuição de Poisson, nomeada em homenagem ao matemático francês
# Siméon Denis Poisson, é uma distribuição de probabilidade discreta que expressa a probabilidade de um determinado número
# de eventos ocorrendo em um intervalo fixo de tempo ou espaço, se esses eventos ocorrerem com uma taxa média constante
# conhecida e independentemente do tempo desde o último evento.

# A Distribuição Poisson também pode ser usada para o número de eventos em outros intervalos especificados, como distância,
# área ou volume.

# A Distribuição de Poisson é a distribuição de probabilidade de ocorrências de eventos independentes em um
# intervalo fixo de tempo ou espaço.

# A função R dpois (x, lambda) é a probabilidade de x sucessos em um período em que o número esperado de eventos é lambda.
# A função R ppois (q, lambda, lower.tail) é a probabilidade cumulativa, sendo
# (lower.tail = TRUE para a cauda esquerda, lower.tail = FALSE para a cauda direita) menor ou igual a q sucessos.
```

```
# Qual é a probabilidade de realizar de 2 a 4 vendas em uma semana se a taxa média de vendas for de 3 por semana?

# Usando a probabilidade exata
?dpois
dpois(x = 2, lambda = 3) + dpois(x = 3, lambda = 3) + dpois(x = 4, lambda = 3)

# Usando probabilidade acumulada (por que a quarta opção é a correta?)
ppois(q = 4, lambda = 3)
ppois(q = 4, lambda = 3, lower.tail = TRUE)
ppois(q = 4, lambda = 3, lower.tail = FALSE)
ppois(q = 4, lambda = 3, lower.tail = TRUE) - ppois(q = 1, lambda = 3, lower.tail = TRUE)

# Vamos plotar todas as possibilidades
x <- ppois(q = 0:10, lambda = 3,  lower.tail = TRUE)
barplot(x, names.arg = 0:10, space = 0)

# Qual a probabilidade de qualquer número de vendas em uma semana se a taxa média de vendas for de 3 por semana?
# Número esperado de vendas = lambda = 3

# Pacotes
library(ggplot2)

# Formata valores decimais
options(scipen = 999, digits = 2)

# Eventos possíveis (número de vendas)
eventos <- 0:10

# Calcula as probabilidades para todos os eventos, ou seja, a distribuição de probabilidades
# para a variável aleatória.
probs <- dpois(x = eventos, lambda = 3)

# Calcula as probabilidades acumuladas para todos os eventos.
prob_acumulada <- ppois(q = eventos, lambda = 3, lower.tail = TRUE)

# Consolidade tudo em um dataframe
df <- data.frame(eventos, probs, prob_acumulada)
df

# Plot (cuidado com a escala do plot)

# Sem probabilidade acumulada
ggplot(df, aes(x = factor(eventos), y = probs)) +
  geom_col() +
  geom_text(aes(label = round(probs,2), y = probs + 0.01), position = position_dodge(0.9), size = 3, vjust = 0) +
  labs(title = "Distribuição Poisson Para Calcular a Probabilidade de Vendas Por Semana",
       x = "Evento (Número de Vendas)",
       y = "Probabilidade")

# Com probabilidade acumulada
ggplot(df, aes(x = factor(eventos), y = probs)) +
  geom_col() +
  geom_text(aes(label = round(probs,2), y = probs + 0.01), position = position_dodge(0.9), size = 3, vjust = 0) +
  labs(title = "Distribuição Poisson Para Calcular a Probabilidade de Vendas Por Semana",
       x = "Evento (Número de Vendas)",
       y = "Probabilidade") +
  geom_line(data = df, aes(x = eventos, y = prob_acumulada))
```

```
# Distribuições de Probabilidade Para Variáveis Contínuas

# Obs: Caso tenha problemas com a acentuação, consulte este link:
# https://support.rstudio.com/hc/en-us/articles/200532197-Character-Encoding

# Configurando o diretório de trabalho
# Coloque entre aspas o diretório de trabalho que você está usando no seu computador
# Não use diretórios com espaço no nome
setwd("~/Dropbox/DSA/Business-Analytics2.0/Cap02/R")
getwd()



######---------- Distribuição Uniforme Contínua ----------######

# Uma distribuição uniforme (geralmente chamada de 'retangular') é aquela em que todos os valores estão
# entre dois limites e ocorrem aproximadamente da mesma forma. Por exemplo, se você rolar um dado de seis
# lados, você terá 1, 2, 3, 4, 5 ou 6. Se o rolar 6.000 vezes, provavelmente obterá aproximadamente
```

```
# 1.000 de cada resultado ou algo aproximado a isso. Os resultados formariam uma distribuição uniforme de 1 a 6.

# A distribuição uniforme é definida em um intervalo [a, b]. A ideia é que qualquer número selecionado no
# intervalo [a, b] tenha uma chance igual de ser selecionado.

# A distribuição uniforme contínua é a distribuição de probabilidade de seleção da variável aleatória
# a partir do intervalo contínuo entre a e b.

# Qual a probabilidade de jogar um dado de 6 faces não viciado e obter o valor 5? E o valor 2?

# Parâmetros para o experimento
num_amostras <- 6000
min <- 1
max <- 6

# Jogando o dado
?sample
sample(min:max, size = 1, replace = TRUE)

# Armazenando 6 mil jogadas

# Cria um vetor vazio
jogadas1 <- vector("numeric")

# Executa 6 mil jogadas e armazena no vetor
for (i in 1:num_amostras) {
  x <- sample(min:max, size = 1, replace = TRUE)
  jogadas1[i] <- x
}
View(jogadas1)
table(jogadas1)

# Outra alternativa é o uso da função runif()

# Agora vamos gerar uma distribuição uniforme
?runif
jogadas2 <- runif(num_amostras, min = 1, max = 6 + 1)
View(jogadas2)

# Vamos arredondar o resultado convertendo para valor inteiro
# A função as.integer() trunca o resultado, convertendo em números inteiros.
# Adicionaremos 1 ao valor max para que o "arredomendamento" possa ser feito para o valor 6 também.
jogadas2 <- as.integer(runif(num_amostras, min = 1, max = 6 + 1))
View(jogadas2)
table(jogadas2)

# Histograma
?hist
hist(jogadas1, main = paste(num_amostras," jogadas de um único dado"), breaks = seq(min-.5, max+.5, 1))
hist(jogadas2, main = paste(num_amostras," jogadas de um único dado"), breaks = seq(min-.5, max+.5, 1))

# Qual a probabilidade de jogar um dado de 6 faces não viciado e obter o valor 5? E o valor 2?
# Já sabemos que a probabilidade de ter a face igual a 5 é igual a 1/6 = 0.1666. Mesmo caso para face 2.
# Isso é o que chamamos de PDF - Probability Density Function
?dunif
dunif(5, min = 1, max = 6 + 1)
dunif(2, min = 1, max = 6 + 1)

# Temos o pacote dice que permite fazer diversas simulações com dados
install.packages("dice")
library(dice)
?getEventProb

# Qual a probabilidade de jogar um dado de 6 faces não viciado e obter o valor 5?
getEventProb(nrolls = 1,
             ndicePerRoll = 1,
             nsidesPerDie = 6,
             eventList = list(5))

# Qual a probabilidade de jogar um dado de 6 faces não viciado e obter o valor 1 ou 2?
# 2/6 = 0.33333

# Isso é o que chamamos de CDF - Cumulative Density Function
# Obs: (punif) - é zero em x = 1 porque é aí que o seu intervalo começa.
?punif
punif(1, min = 1, max = 6 + 1)
punif(2, min = 1, max = 6 + 1)
punif(3, min = 1, max = 6 + 1)
```

```
# Resposta com a função getEventProb()
getEventProb(nrolls = 1,
             ndicePerRoll = 1,
             nsidesPerDie = 6,
             eventList = list(1:2))


# Qual a probabilidade de jogar um dado de 6 faces 10 vezes e obter o valor 5?
# Podemos usar dbinom, pois estamos modelando a chance de sucesso/fracasso em um número n de tentativas.
?dbinom
dbinom(x = 1, size = 10, prob = 1/6)

# Qual a probabilidade de jogar um dado de 6 faces 10 vezes e obter o valor 5 menos de 3 vezes?
# Podemos usar dbinom, pois estamos modelando a chance de sucesso/fracasso em um número n de tentativas, e
# nesse caso de forma acumulada.
pbinom(3, size = 10, prob = 1/6)


######---------- Distribuição Normal ----------######

# A Distribuição Normal é uma das mais importantes distribuições da estatística, conhecida também como
# Distribuição de Gauss ou Gaussiana.

# Além de descrever uma série de fenômenos físicos e financeiros, possui grande uso na estatística inferencial.
# É inteiramente descrita por seus parâmetros de média e desvio padrão, ou seja, conhecendo-se estes valores
# consegue-se determinar qualquer probabilidade em uma distribuição Normal.

# Um interessante uso da Distribuição Normal é que ela serve de aproximação para o cálculo de outras
# distribuições quando o número de observações for muito grande.

# Essa importante propriedade provém do Teorema Central do Limite que diz que:
# "Toda soma de variáveis aleatórias independentes de média finita e variância limitada é aproximadamente Normal,
# desde que o número de termos da soma seja suficientemente grande."

# O Teorema Central do Limite é talvez o conceito mais importante em estatística. Para qualquer distribuição com
# média finita e desvio padrão, as amostras colhidas nessa população tenderão a uma distribuição normal em torno
# da média da população à medida que o tamanho da amostra aumenta. Além disso, à medida que o tamanho da amostra
# aumenta, a variação da média da amostra diminui.

# Uma distribuição é a maneira pela qual um conjunto de valores é distribuído por um possível intervalo de
# valores. Uma maneira comum de visualizar uma distribuição é um histograma que mostra o número de elementos,
# ou frequência, dentro dos intervalos de valores:
numero_vendas_dia = c(3, 5, 2, 3, 3, 6, 3, 10, 5, 5, 5, 7, 8, 7, 1, 5, 5, 4, 4, 7)

?hist
hist(numero_vendas_dia)

mean(numero_vendas_dia)
sd(numero_vendas_dia)

# A escala vertical de um 'histograma de frequência' mostra o número de observações em cada posição.
# Opcionalmente, também podemos colocar rótulos numéricos em cima de cada barra, mostrando quantos indivíduos
# ela representa.
# A contagem de posições pode somar qualquer número inteiro.
hist(numero_vendas_dia,
    main = "Frequência do Número de Vendas Por Dia",
    xlab = "Número de Vendas Por Dia",
    ylab = "Frequência",
    labels = TRUE)

# A escala vertical de um 'histograma de densidade' mostra as unidades que fazem o total de todas as barras ser igual a 1.
# Isso permite mostrar a curva de densidade da população usando a mesma escala vertical.
# As densidades somam um, pois a integração do pdf gera a área da unidade.
hist(numero_vendas_dia,
    main = "Densidade do Número de Vendas Por Dia",
    xlab = "Número de Vendas Por Dia",
    ylab = "Densidade de Probabilidade",
    probability = TRUE)

?lines
lines(density(numero_vendas_dia), col = "blue", lwd = 2)

?dnorm
dnorm(numero_vendas_dia, mean = 4.9, sd = 2.17)

# PDF (Probability Density Function) é usado para especificar a probabilidade da variável aleatória cair dentro de um
# determinado intervalo de valores, em vez de assumir qualquer valor. Essa probabilidade é dada pela integral do PDF
```

```
# dessa variável nesse intervalo - ou seja, é dada pela área sob a função de densidade, mas acima do eixo horizontal e
# entre os valores mais altos e mais baixos do intervalo. Essa definição pode não fazer muito sentido; portanto, vamos
# esclarecer o gráfico da função de densidade de probabilidade para uma distribuição normal.

# Embora a fórmula para a distribuição normal seja complexa, R contém várias funções que permitem a análise
# de dados. O histograma suavizado associado à distribuição normal é conhecido popularmente como curva de sino:
x = seq(-3, 3, 0.1)
plot(x = x, y = dnorm(x), type = "l", bty = "n")

# A curva de sino é uma curva de densidade e a área sob a curva de sino entre um conjunto de valores
# representa a porcentagem de números na distribuição entre esses valores.

# Em teoria das probabilidades e estatística, a função densidade de probabilidade, ou densidade de uma variável
# aleatória contínua, é uma função que descreve a probabilidade relativa de uma variável aleatória tomar um valor
# dado. A probabilidade da variável aleatória cair em uma faixa particular é dada pela integral da densidade dessa
# variável sobre tal faixa - isto é, é dada pela área abaixo da função densidade mas acima do eixo horizontal
# e entre o menor e o maior valor dessa faixa. A função densidade de probabilidade é não negativa sempre, e sua
# integral sobre todo o espaço é igual a um. A função densidade pode ser obtida a partir da função distribuição
# acumulada a partir da operação de derivação (quando esta é derivável).

# Para variáveis aleatórias contínuas, as probabilidades são representadas pelas áreas sob a curva.

# O valor médio de uma distribuição normal é a média, e a largura da curva de sino é definida pelo desvio padrão.

# Regra 68-95-99 para a Distribuição Normal

# 68,2% dos valores estão dentro de 1 desvio padrão da média
# 95,4% dos valores estão dentro de 2 desvios padrão da média
# 99,7% dos valores estão dentro de 3 desvios padrão da média

# O número de desvios padrão dos quais um valor se afasta da média é chamado de escore z.
# O escore z da média é zero. Por exemplo, se a média de uma distribuição for 7 e o desvio padrão for 2,
# um valor de 4 terá um escore z de -1,5.

valor_media = 7
valor_desvio = 2
x = 3:11
z = (x - valor_media) / valor_desvio
data.frame(x, z)

# Podemos generalizar o exemplo anterior do dado para o caso de amostras de tamanhos variados retiradas
# de uma distribuição contínua que varia de 0-1. Esta simulação mostra a distribuição de amostras
# dos tamanhos 1, 2, 4, ... 32 retiradas de uma distribuição
 uniforme. Observe que, para cada amostra,
# estamos descobrindo o valor médio da amostra, e não a soma que estávamos fazendo no caso dos dados.

# Parâmetros para o experimento
num_amostras <- 10000
min <- 0
max <- 1
n_vezes <- 6

# Ajustando a área de plotagem
par(mar = c(1,1,1,1))
op <- par(mfrow = c(n_vezes, 1))
i2 <- 1

# Aumentando o tamanho de cada amostra e criando um histograma
# Comprovando um dos fundamentos do Teorema Central do Limite
for (i in 1:n_vezes)
  {  sample = rep(0, num_amostras)
     k = 0
  for (j in 1:i2)
  {
    sample <- sample + runif(num_amostras, min, max)
    k <- k+1  }
  x <- sample/k
  saida <- c(k, mean(x), sd(x))

  hist(x, xlim = range(0,1), prob = T, main = paste( "Amostras de tamanho",  k ), col = "blue")
  i2 <- 2*i2
}

# Outro exemplo do Teorema Central do Limite:

# Reset da área de plotagem (se necessário, feche o RStudio e abra novamente)
par(mfrow = c(1,1))
```

```
# Dataset de idades de segurados
idades <- read.csv("dados/idades.csv")
head(idades)
tail(idades)

# Histograma
hist(idades$idade, right = FALSE)

# Vamos melhorar este histograma
hist(idades$idade,
     right = FALSE,
     breaks = seq(0,102,2),
     col = "blue",
     las = 1,
     xlab = "Idade do Segurado (anos)",
     ylab = "Frequência",
     main = "")

# Vamos coletar várias amostras de dados
n <- 4
resultados <- vector()

for(i in 1:10000) {
  valores_idades <- sample(idades$idade, size = n, replace = FALSE)
  resultados[i] <- mean(valores_idades)
}

# Histograma do resultado
hist(resultados,
     right = FALSE,
     breaks = 50,
     col = "firebrick",
     las = 1,
     xlab = "Média de Idade do Segurado",
     ylab = "Frequency",
     main = "")

# A influência do desvio padrão

# Massa de dados
x = seq(-8, 8, length = 500)

# Desvio padrão igual a 1
y1 = dnorm(x, mean = 0, sd = 1)
plot(x, y1, type="l", lwd=2, col="red")

# Desvio padrão igual a 2
y2 = dnorm(x, mean = 0, sd = 2)
lines(x, y2, type="l", lwd=2, col="blue")


# Vamos calcular probabilidades para variável aleatória.

# Suponha que as pontuações dos exames de vestibular se enquadrem em uma distribuição normal.
# Além disso, a nota média do teste é 72 e o desvio padrão é 9.3.
# O exame vai de 0 a 100 pontos possíveis.

# Qual é a probabilidade de alunos conseguirem exatamente 85 pontos no exame?
# Qual é a probabilidade de alunos conseguirem menos de 70 pontos no exame?
# Qual é a probabilidade de alunos conseguirem mais de 90 pontos no exame?

# Reset da área de plotagem (se necessário, feche o RStudio e abra novamente)
par(mfrow = c(1,1))

# Gerando a massa de dados com notas aleatórias de 100 alunos

# Média e desvio
media <- 72
desvio <- 9.3

# Sequência de valores para a massa de dados
?rnorm
notas <- rnorm(100, mean = media, sd = desvio)
min(notas)
max(notas)

# Plot
hist(notas, main = "Notas Para o Exame Vestibular", xlab = "Notas", col = "blue", breaks = 10)
```

```
h <- hist(notas, main = "Notas Para o Exame Vestibular", xlab = "Notas", col = "blue", breaks = 10)
text(h$mids, h$counts, labels = h$counts, adj = c(0.5, -0.5))

# Curva PDF
probabilidades_notas = dnorm(notas, mean = media, sd = desvio)
plot(x = notas, y = probabilidades_notas)

# Curva CDF
probabilidades_notas_cumul = pnorm(notas, mean = media, sd = desvio)
plot(x = notas, y = probabilidades_notas_cumul)


# Qual é a probabilidade de alunos conseguirem exatamente 85 pontos no exame?
?dnorm
dnorm(85, mean = media, sd = desvio)
dnorm(85, mean = media, sd = desvio) * 100


# Qual é a probabilidade de alunos conseguirem menos de 70 pontos no exame?
# Aplicamos a função pnorm da distribuição normal com média 72 e desvio padrão 9.3. Uma vez que
# estamos procurando o percentual de alunos com pontuação inferior a 70, estamos interessados na cauda
# inferior da distribuição normal.
?pnorm
pnorm(70, mean = media, sd = desvio, lower.tail = TRUE)
pnorm(70, mean = media, sd = desvio, lower.tail = TRUE) * 100


# Qual é a probabilidade de alunos conseguirem mais de 90 pontos no exame?
# Aplicamos a função pnorm da distribuição normal com média 72 e desvio padrão 9.3. Uma vez que
# estamos procurando o percentual de alunos com pontuação superior a 90, estamos interessados na cauda
# superior da distribuição normal.
?pnorm
pnorm(90, mean = media, sd = desvio, lower.tail = FALSE)
pnorm(90, mean = media, sd = desvio, lower.tail = FALSE) * 100


# As pontuações de QI das crianças são normalmente distribuídas com um média de 100 e desvio padrão de 15.
# Que proporção de crianças deverá ter um QI entre 80 e 120? Crie um plot para demonstrar seu resultado.

# Reset da área de plotagem
par(mfrow = c(1,1))

# Variáveis
media = 100; desvio = 15
limite_inferior = 80; limite_superior = 120

# Cria uma distribuição de dados
x <- seq(-4, 4, length = 100) * desvio + media

# Calcula a PDF
hx <- dnorm(x, media, desvio)

# Plot
plot(x, hx, type = "n", xlab = "Valores de QI", ylab = "", main = "", axes = FALSE)

# Define os valores entre os limites
i <- x >= limite_inferior & x <= limite_superior

# Adiciona uma linha com valores de x e as probabilidades
lines(x, hx)

# Cria o polygon
?polygon
polygon(c(limite_inferior, x[i], limite_superior), c(0, hx[i] ,0), col = "red")

# Calcula as probabilidades acumuladas entre os limites
area <- pnorm(limite_superior, media, desvio) - pnorm(limite_inferior, media, desvio)
area

# Prepara o título para o gráfico
resultado <- paste("P(", limite_inferior, "< QI <", limite_superior,") =", signif(area, digits = 3))
mtext(resultado, 3)
axis(1, at = seq(40, 160, 20), pos=0)


# Teste de Normalidade

# Gerando 2 datasets
```

```
# O primeiro segue uma distribuição normal e o segundo segue uma distribuição uniforme
df1 = rnorm(100)
df2 = runif(100)

hist(df1)
hist(df2)

# Plot das densidades
plot(density(df1))
plot(density(df2))

# Shapiro Test
# Hipótese Nula (H0): Os dados são normalmente distribuídos.
# Se o valor-p for maior que 0.05 não rejeitamos a hipótese nula e podemos assumir a normalidade dos dados.
# Se o valor-p for menor que 0.05 rejeitamos a hipótese nula e não podemos assumir a normalidade dos dados.
?shapiro.test
shapiro.test(df1)
shapiro.test(df2)

# Teste Visual Usando Normal Q-Q Plot
?qqnorm
?qqline
qqnorm(df1);qqline(df1, col = 2)
qqnorm(df2);qqline(df2, col = 2)

# Testando a variável que criamos anteriormente para o QI
x <- rnorm(100, mean = 100, sd = 15)
qqnorm(x);qqline(x,  col = 2)
shapiro.test(x)


######---------- Distribuição Exponencial ----------######

# A distribuição exponencial descreve o tempo de chegada de uma sequência de eventos independentes,
# aleatoriamente recorrentes.

# Suponha que o tempo médio de checkout de um caixa de supermercado seja de 3 minutos.
# Encontre a probabilidade de uma compra de cliente ser concluída pelo caixa em menos de 2 minutos.

# A taxa de processamento de saída é igual a 1 dividido pelo tempo médio de conclusão do checkout.
# Daí a taxa de processamento é 1/3 checkouts por minuto.
# Aplicamos então a função pexp da distribuição exponencial com taxa = 1/3.

# A probabilidade de terminar um checkout em menos de dois minutos pelo caixa é de 48,7%
?pexp
pexp(2, rate = 1/3)
```

Distribuição Normal e Inferência Estatística

O Teorema Central do Limite é um importante resultado da estatística e a demonstração de muitos outros teoremas estatísticos dependem dele. Em teoria das probabilidades, esse teorema afirma que quando o tamanho da amostra aumenta, a distribuição amostral da sua média aproxima-se cada vez mais de uma distribuição normal. Este resultado é fundamental na teoria da inferência estatística.

Na inferência estatística a utilidade do teorema central do limite vai desde estimar os parâmetros como a média populacional ou o desvio padrão da média populacional, a partir de uma amostra aleatória dessa população, ou seja, da média amostral e do desvio padrão da média amostral até calcular a probabilidade de um parâmetro ocorrer dado um intervalo, sua média amostral e o desvio padrão da média amostral.

O teorema central do limite afirma que a média de uma amostra de n elementos de uma população tende a uma distribuição normal. Pode-se pensar de forma empírica que ao nos distanciarmos da média, a probabilidade de ocorrência diminui, ou seja, é mais provável ocorrer um evento que se encontra próximo da média do que um evento de um dos extremos. Além disso, uma distribuição pode ganhar a forma de curva normal se possuir diferentes combinações para cada resultado

possível do espaço amostral. Isso é válido (em se tratando de amostras discretas), para amostras suficientemente grandes da população. O suficientemente grande, varia de acordo com a população, para populações com distribuição quase simétrica, a amostra pode ser menor do que para populações cuja distribuição seja assimétrica. A curva normal obtida pode então ser convertida em uma curva binomial ou em uma curva de Poisson, e posteriormente pode-se ainda realizar uma correção de continuidade. A precisão da correção de continuidade também pode ser medida.

Assim, é permitido inferir sobre a população através da média amostral e do desvio padrão amostral. Se extraíssemos todos os elementos da população, os dados sobre a amostra seriam exatamente iguais aos da população, mas isso pode ser demasiadamente custoso e/ou lento e/ou impossível (é impossível medir a resistência máxima de qualquer produto para todos os elementos da população).

Quizz

Enquanto as aplicações estatísticas tradicionais se concentram em conjuntos de dados relativamente pequenos, a Ciência de Dados envolve quantidades muito grandes de dados, normalmente referido como Big Data.

A mineração de dados explora e analisa grandes quantidades de dados para descobrir padrões significativos. A escala de uma aplicação típica de mineração de dados, com seu grande número de casos e muitas variáveis, excede a de uma investigação estatística padrão.

Análise Descritiva é um tipo de análise muitas vezes relacionada como Business Intelligence, pois fornece o conhecimento necessário sobre o passado, a fim de ajudar tomadores de decisão a fazer previsões futuras.

A medição da dispersão ou variabilidade permite-nos ver como está a difusão dos dados a partir de um valor central ou a média. A variância e desvio padrão são os valores utilizados para medir a dispersão.

Análise preditiva é realizada através de uma série de técnicas analíticas e estatísticas, utilizadas para o desenvolvimento de modelos que podem prever eventos futuros a partir de comportamentos diários, incluindo análise de séries temporais ou modelos de regressão, por exemplo.

8.3. Trabalhando Com Analytics – Parte 2

O Que é Análise de Regressão?

Você provavelmente já sabe que, sempre que possível, deve tomar decisões baseadas em dados. Mas você sabe analisar todos os dados disponíveis? Não estou falando da Matemática, pois o computador faz isso por nós. Estou falando de interpretar corretamente a análise criada. Um dos tipos mais importantes de análise de dados e um dos mais simples de interpretar, é a regressão.

O Que é Análise de Regressão?

Suponha que você seja um gerente de vendas tentando prever os números do próximo mês. Você sabe que dezenas, talvez até centenas de fatores, desde o clima até a promoção de um concorrente e o boato de um modelo novo e aprimorado do produto podem afetar o número. Talvez as pessoas da sua organização tenham até uma teoria sobre o que terá o maior efeito nas vendas. "Confie em mim. Quanto mais chuva temos, mais vendemos.” ou "Seis semanas após a promoção do concorrente, as vendas aumentam."

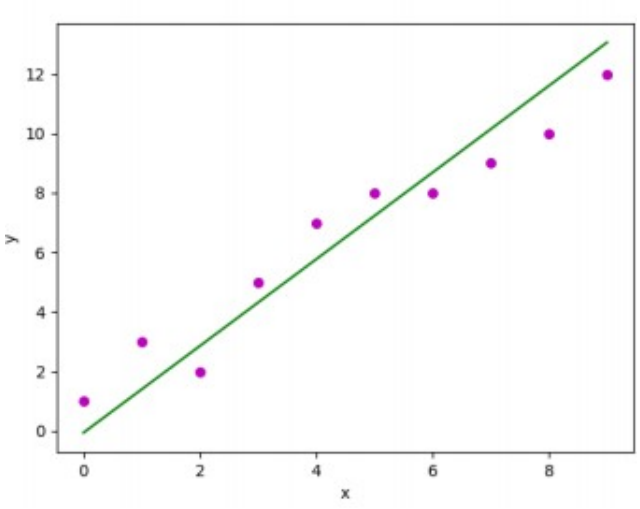
A análise de regressão é uma maneira de classificar matematicamente quais dessas variáveis realmente têm impacto. Responde às perguntas: Quais fatores são mais importantes? O que podemos ignorar? Como esses fatores interagem entre si? E, talvez o mais importante, até que ponto estamos certos sobre todos esses fatores?

Na análise de regressão, esses fatores são chamados de variáveis. Você tem sua variável dependente - o principal fator que você está tentando entender ou prever. No exemplo acima, a variável dependente é o número que representa as vendas mensais. E então você tem suas variáveis independentes - os fatores que você suspeita ter impacto na sua variável dependente.

A análise de regressão permite estudar e prever esse relacionamento.

Como Funciona a Regressão

Para realizar uma análise de regressão, você reúne os dados nas variáveis em questão. Você usa todos os seus números de vendas mensais nos últimos três anos e, por exemplo, os dados das variáveis independentes nas quais você está interessado. Então, neste caso, digamos que você descubra também a precipitação média mensal nos últimos três anos. Em seguida, você plota todas essas informações em um gráfico semelhante a este:



O eixo y é a quantidade de vendas (a variável dependente, a coisa em que você está interessado, está sempre no eixo y) e o eixo x é a precipitação total. Cada ponto rosa representa os dados de um mês - quanto choveu naquele mês e quantas vendas você fez no mesmo mês.

Olhando para esses dados, você provavelmente percebe que as vendas são maiores nos dias em que chove muito. É interessante saber, mas em quanto? Se chover 3 centímetros, você sabe quanto vai vender? E se chover 10 cm?

Agora imagine desenhar uma linha no gráfico acima, que percorre aproximadamente o meio de todos os pontos de dados. Essa linha o ajudará a responder, com certo grau de certeza, quanto você normalmente vende quando chove uma certa quantidade. É o que nos diz a linha verde no gráfico acima.

Isso é chamado de linha de regressão (a linha verde no gráfico acima) e é desenhado para mostrar a linha que melhor se ajusta aos dados. Em outras palavras: “a linha verde é a melhor explicação do relacionamento entre a variável independente e a variável dependente”.

Além de desenhar a linha, seu programa de estatísticas também gera uma fórmula que explica a inclinação da linha e se parece com isso:

Y = 200 + 5x + erro

Mas vamos desconsiderar o erro por enquanto e focar nesta parte da fórmula:

Y = 200 + 5x

O que essa fórmula está dizendo é que, se não houver um "x", então Y = 200. Então, historicamente, quando não choveu, você fez uma média de 200 vendas e pode esperar fazer o mesmo daqui para frente. assumindo que outras variáveis permaneçam iguais. E no passado, para cada centímetro adicional de chuva, você fazia em média mais cinco vendas. “Para cada incremento que x sobe um, y sobe cinco”.

Agora vamos voltar ao termo de erro. Você pode ficar tentado a dizer que a chuva tem um grande impacto nas vendas se, a cada centímetro, você conseguir mais cinco vendas, mas se essa variável vale a pena, sua atenção dependerá do termo de erro. Uma linha de regressão sempre tem um termo de erro porque, na vida real, variáveis independentes nunca são preditores perfeitos das variáveis dependentes. Em vez disso, a linha é uma estimativa com base nos dados disponíveis. Portanto, o termo de erro informa o quanto você pode ter certeza sobre a fórmula. Quanto maior, menor a certeza da linha de regressão.

O exemplo acima usa apenas uma variável para prever o fator de interesse - nesse caso, chuva para prever vendas. Normalmente, você inicia uma análise de regressão que deseja entender o impacto de diversas variáveis independentes. Portanto, você pode incluir não apenas chuva, mas também dados sobre a promoção de um concorrente. “Você continua fazendo isso até que o termo de erro seja muito pequeno”. “Você está tentando obter a linha que melhor se ajusta aos seus dados.” Embora possa haver perigos em tentar incluir muitas variáveis em uma análise de regressão, Cientistas de Dados qualificados podem minimizar esses riscos.

E considerar o impacto de diversas variáveis ao mesmo tempo é uma das maiores vantagens da regressão. Isso é o que chamamos de análise multivariada.

Análise de Regressão em R

```
# Análise de Regressão

# Obs: Caso tenha problemas com a acentuação, consulte este link:
# https://support.rstudio.com/hc/en-us/articles/200532197-Character-Encoding

# Configurando o diretório de trabalho
# Coloque entre aspas o diretório de trabalho que você está usando no seu computador
# Não use diretórios com espaço no nome
setwd("~/Dropbox/DSA/Business-Analytics2.0/Cap03/R")
getwd()

# Explorando Testes "Motor Trend Car Road"

# Neste projeto, trabalhamos para a Motor Trend (uma revista sobre a indústria automobilística)
# Observando um conjunto de dados de uma coleção de carros (que contém 32 observações), eles estão
# interessados em explorar a relação entre um conjunto de variáveis e a autonomia de combustível em milhas
# por galão (mpg), que é a nossa variável target. Estão particularmente interessados nestas duas questões:

# Qual tipo de transmissão consome menos combustível, automática ou manual?
# Quão diferente é a autonomia (mpg) entre as transmissões 'Automática' e 'Manual'?


#####---------------------- Pré-processamento e Transformações de Dados ----------------------------

# Carga de dados
data(mtcars)
View(mtcars)
dim(mtcars)

# mpg       - Milhas/galão
# cyl       - Número de cilindros
# disp      - Deslocamento
# hp        - Horsepower (cavalos de potência)
# drat      - Relação do eixo traseiro
# wt        - Peso (1000 lbs)
# qsec      - 1/4 mile time
# vs        - Motor (0 = turbo, 1 = normal)
# am        - Transmissão (0 = automática, 1 = manual)
# gear      - Número de engrenagens
# carb      - Número de carburadores


#####---------------------- Análise Exploratória ----------------------------

# Exploramos várias relações entre as variáveis de interesse e o resultado (target). Inicialmente, traçamos as
# relações entre todas as variáveis do conjunto de dados conforme gráfico abaixo.
# A partir da matriz de correlação notamos que variáveis como cyl, disp, hp, drat, wt, vs e am parecem ter
# alguma forte correlação com mpg. Usaremos modelos lineares para quantificar isso na próxima seção.

# Além disso, traçamos um boxplot da variável mpg quando am é 'Automático' ou 'Manual'
# Este gráfico mostra que o mpg aumenta quando a transmissão é 'Manual'.

# Pacotes
install.packages("corrplot")
install.packages("plyr")
install.packages("printr")
install.packages("GGally")
library(corrplot)
library(plyr)
library(knitr)
library(printr)
library(GGally)
library(ggplot2)
library(MASS)

# Formatando os dados para posterior documentação
?kable
kable(head(mtcars), align = 'c')
head(mtcars)

# Tipos das variáveis e resumo estatístico
str(mtcars)
summary(mtcars)
summary(mtcars$mpg)
sum(is.na(mtcars))
```

```
# Tabela de frequência
table(mtcars$cyl)
?plyr::count
count(mtcars, 'carb')

# Gráficos

# Percentual de carros por número de cilindros
cyl_freq <- table(mtcars$cyl)
labels <- names(cyl_freq)
percent <- round(cyl_freq/sum(cyl_freq) * 100)
labels <- paste(labels, percent)
labels <- paste(labels, "%", sep = "")
length(labels)
pie(cyl_freq, labels = labels, col = rainbow(length(labels)), main = "% de Carros Por Número de Cilindros")

# Número de carros por HP
count <- table(mtcars$hp)

?barplot
barplot(count,
        main = "Carros Por HP",
        xlab = "HP",
        ylab = "Número de Carros")

?sort
barplot(sort(count, decreasing = TRUE),
        main = "Carros Por HP",
        xlab = "HP",
        ylab = "Número de Carros")

# Scatter Plot
?plot
plot(mtcars$mpg, mtcars$hp, xlab = "MPG (Autonomia)", ylab = "HP (Potência)")

# Histograma da variável alvo
hist(mtcars$mpg)
?hist
hist(mtcars$mpg,
     breaks = 10,
     xlab = "Milhas Por Galão",
     main = "Histograma da Variável MPG",
     xlim = range(10:35))

# Boxplot
?boxplot
boxplot(mtcars$mpg)

# Corrplot
?cor
m_cor <- cor(mtcars)
?corrplot
corrplot(m_cor, method = "circle")

# Pair Plot
pairs(mtcars, panel = panel.smooth, main = "Pair Graphs")

# Também vale a pena verificar como o MPG varia de acordo com a transmissão automática
# versus a manual. Para esse efeito, criamos um gráfico de violino para MPG através de
# transmissões automáticas e manuais. No nosso conjunto de dados, 0 representa uma
# transmissão automática e 1 significa uma transmissão manual.
# Violin Plot
?ggplot
ggplot(mtcars, aes(y = mpg,
                   x = factor(am, labels = c("automatic", "manual")),
                   fill = factor(am))) +
  geom_violin(colour = "black", size = 1) +
  xlab("Transmissão") +
  ylab("MPG")

# Podemos criar uma hipótese clara a partir dessa visualização: parece que os carros
# automáticos têm milhas mais baixas por galão e, portanto, uma menor eficiência de
# combustível do que os carros manuais. Mas é possível que esse padrão aparente tenha
# acontecido por acaso - ou seja, que tenhamos escolhido um grupo de carros automáticos
# com baixa eficiência e um grupo de carros manuais com maior eficiência. Portanto, para
# verificar se é esse o caso, precisamos usar um teste estatístico.

# Função para diversos gráficos estatísticos em um único plot
```

```
cria_plot <- function(data, mapping, method = "loess", ...){
  p <- ggplot(data = data, mapping = mapping) +
    geom_point() +
    geom_smooth(method = method, ...)
  p
}

# Cria o plot
?ggpairs
ggpairs(mtcars, lower = list(continuous = cria_plot))


# Teste Estatístico
# Teste t de duas amostras

# Estamos interessados em saber se uma transmissão automática ou manual é melhor para MPG.
# Então, primeiro testamos a hipótese de que carros com transmissão automática consomem
# mais combustível do que carros com transmissão manual. Para comparar duas amostras e
# verificar se elas têm médias diferentes, usamos o Teste T de duas amostras.

# Existem 3 tipos comuns de Teste t:

# Teste t para duas amostras independentes (ou não pareadas) - usaremos este!
# Teste t para duas amostras dependentes (ou pareadas)
# Teste t para uma amostra
?t.test
View(mtcars)
teste <- t.test(mpg ~ am, data = mtcars, var.equal = FALSE, paired = FALSE, conf.level = .95)
print(teste)

# Um dos testes mais comuns em estatística é o Teste t, usado para determinar se as médias
# de dois grupos são iguais entre si. Este é um teste paramétrico e a suposição para o teste
# é que ambos os grupos são amostrados de distribuições normais e com variâncias iguais.

# Ou seja, o Teste t pode ser implementado para determinar se as amostras são diferentes.

# Vamos testar as suposições?

# Extraindo os registros segmentados pelo tipo de transmissão
str(mtcars)
mtcars$am <- as.factor(mtcars$am)
str(mtcars)
levels(mtcars$am) <- c("Automatic", "Manual")

Automatic <- mtcars[mtcars$am == "Automatic",]
Manual <- mtcars[mtcars$am == "Manual",]
View(Automatic)
View(Manual)

# Médias e Sumário
mean(Automatic$mpg)
mean(Manual$mpg)
summary(mtcars$mpg)

# Teste de Normalidade - Shapiro Test
# Hipótese Nula (H0): Os dados são normalmente distribuídos.
# Hipótese Alternativa (H1): Os dados não são normalmente distribuídos.

# Se o valor-p for maior que 0.05 não rejeitamos a hipótese nula e podemos assumir a normalidade dos dados.
# Se o valor-p for menor que 0.05 rejeitamos a hipótese nula e não podemos assumir a normalidade dos dados.
?shapiro.test
mtcars$mpg[mtcars$am == "Automatic"]
shapiro.test(mtcars$mpg[mtcars$am == "Automatic"])
shapiro.test(Manual$mpg)

# Teste de Homogeneidade das Variâncias - Bartlett's Test
# A hipótese nula (H0) para o teste é que as variâncias são iguais para todas as amostras.
# A hipótese alternativa (H1) (a que você está testando) é que as variâncias não são iguais.

# Se o valor-p for maior que 0.05 não rejeitamos a hipótese nula e podemos assumir que as variâncias são iguais para todas as amostras.
# Se o valor-p for menor que 0.05 rejeitamos a hipótese nula e não podemos assumir que as variâncias são iguais para todas as amostras.
?bartlett.test
bartlett.test(mpg ~ am, data = mtcars)

# Obs: O leveneTest() é usado quando os dados não são normalmente distribuídos.

# O teste t de Student (ou simplesmente teste t) compara duas médias e mostra se as
# diferenças entre essas médias são significativas. Em outras palavras, permite que você
```

```
# avalie se essas diferenças ocorreram por um mero acaso ou não.

# A necessidade de determinar se duas médias de amostras são diferentes entre si é uma
# situação extremamente frequente em pesquisas científicas.

# Por exemplo se um grupo experimental difere de um grupo controle, se uma amostra difere
# da população, se um grupo difere antes e depois de um procedimento. Nessas diversas
# situações, um método bastante comum é a comparação das médias da medida de interesse.

# A hipótese nula é que as duas médias são iguais, e a alternativa é que não são. Sabe-se
# que, sob a hipótese nula, podemos calcular uma estatística t que seguirá uma
# distribuição t com n1 + n2 - 2 graus de liberdade. Há também uma modificação amplamente
# usada do teste t, conhecida como Teste t de Welch, que ajusta o número de graus de
# liberdade quando se pensa que as variações não são iguais umas às outras. Esse é o teste usado em R.

# Como todo teste estatístico, a teste t também tem como produto a medida do valor-p.
# Ou seja, no final das contas, teremos calculado a probabilidade da diferença encontrada
# (entre as médias) terem sido por acaso. Se esse valor for menor que 5% (p < 0.05),
# a tradição científica é de rejeitarmos a hipótese de que as diferenças sejam por acaso
# (rejeitamos a hipótese nula) e alegamos termos encontrado uma diferença estatísticamente
# significativa.

# Teste t
teste <- t.test(mpg ~ am, data = mtcars, var.equal = TRUE, paired = FALSE, conf.level = .95)
print(teste)

# Hipóteses:
# H0 (hipótese nula): a verdadeira diferença das médias é igual a 0
# H1 (hipótese alternativa): a verdadeira diferença das médias não é igual a 0

# O valor-p que mostra a probabilidade de que essa aparente diferença entre os dois grupos
# possa aparecer por acaso é muito baixo.

# Rejeitamos a hipótese nula e portanto
 as médias apresentam diferenças. Conclui-se que
# as médias de consumo de combustível entre os tipos de transmissão são diferentes e não são fruto do acaso.

# Boxplots
?boxplot
boxplot(Automatic$mpg, Manual$mpg)
```

```
# Análise de Regressão

# Obs: Caso tenha problemas com a acentuação, consulte este link:
# https://support.rstudio.com/hc/en-us/articles/200532197-Character-Encoding

# Configurando o diretório de trabalho
# Coloque entre aspas o diretório de trabalho que você está usando no seu computador
# Não use diretórios com espaço no nome
setwd("~/Dropbox/DSA/Business-Analytics2.0/Cap03/R")
getwd()

# O Processo de Análise pode ser dividido em 3 grandes etapas:

# Etapa 1 - Análise Exploratória
# Etapa 2 - Pré-Processamento dos Dados
# Etapa 3 - Análise dos Dados e Modelagem Preditiva


# Explorando Testes "Motor Trend Car Road"

# Neste projeto, trabalhamos para a Motor Trend (uma revista sobre a indústria automobilística)
# Observando um conjunto de dados de uma coleção de carros (que contém 32 observações), eles estão
# interessados em explorar a relação entre um conjunto de variáveis e a autonomia de combustível em milhas
# por galão (mpg), que é a nossa variável target. Estão particularmente interessados nestas duas questões:

# Qual tipo de transmissão consome menos combustível, automática ou manual?
# Quão diferente é a autonomia (mpg) entre as transmissões 'Automática' e 'Manual'?

# Dicionário de Dados

# mpg      - Milhas/galão
# cyl      - Número de cilindros
# disp     - Deslocamento
# hp       - Horsepower (cavalos de potência)
# drat     - Relação do eixo traseiro
# wt       - Peso (1000 lbs)
```

```
# qsec      - 1/4 mile time (velocidade de aceleração)
# vs        - Motor (0 = turbo, 1 = normal)
# am        - Transmissão (0 = automática, 1 = manual)
# gear      - Número de engrenagens
# carb      - Número de carburadores

# Pacotes
install.packages("corrplot")
install.packages("plyr")
install.packages("printr")
install.packages("GGally")
library(corrplot)
library(plyr)
library(knitr)
library(printr)
library(GGally)
library(ggplot2)
library(MASS)

# Carga de dados
data(mtcars)
View(mtcars)
dim(mtcars)


#####---------- Análise de Regressão ----------

# Nesta seção, vamos construir modelos de regressão linear com base nas diferentes variáveis de
# interesse e tentar descobrir o melhor modelo de ajuste. Vamos criar um modelo base e então criar
# versões otimizadas e ao final avaliar e escolher o melhor modelo.

# Modelo de Regressão Linear Simples

# Também podemos ajustar variáveis fatoriais como regressores e criar uma análise de variância
# como um caso especial de modelos de regressão linear. Do ponto de vista das "variáveis fictícias",
# não há nada de especial na análise de variância (ANOVA). É apenas regressão linear no caso especial
# de que todas as variáveis preditivas são categóricas. Nossa variável fator nesse caso é Transmission (am).

# Checando as variáveis
str(mtcars)

# Criando uma variável do tipo fator a partir da variável am
mtcars$amfactor <- factor(mtcars$am, labels = c("automatic", "manual"))
str(mtcars)
View(mtcars)

# Criando o modelo base com a variável am (categórica) sendo usada como numérica
# Fórmula de regressão ==> y = a + bx
?lm
modelo_v1_base <- lm(mpg ~ am, data = mtcars)
summary(modelo_v1_base)

# Criando o modelo base com a variável am com seu tipo correto (categórica)
modelo_v1_base <- lm(mpg ~ amfactor, data = mtcars)
summary(modelo_v1_base)

# A inclinação (slope) de 7.24 é a mudança na média entre transmissão manual e transmissão automática.
# O valor-p de 0,000285 para a diferença média de MPG entre transmissão manual e automática é
# significativo. Portanto, concluímos que, de acordo com este modelo, a transmissão manual é
# mais eficiente em termos de combustível.


# Regressão Linear Múltipla

# Queremos saber qual combinação de preditores melhor vai prever a eficiência do combustível.
# Quais preditores aumentam nossa precisão em uma quantidade estatisticamente significativa?
# Podemos adivinhar algumas das tendências do gráfico, mas realmente queremos realizar um teste
# estatístico para determinar quais preditores são significativos e para determinar a fórmula ideal
# para a previsão.

# A inclusão de variáveis que não deveríamos ter no modelo aumenta os erros padrão reais das
# variáveis de regressão. Portanto, não queremos lançar variáveis no modelo à toa.
# Para confirmar esse fato, você pode ver abaixo que, se incluirmos todas as variáveis, nenhuma
# delas será um preditor significativo de MPG (a julgar pelo valor-p no nível de confiança de 95%).
str(mtcars)
modelo_v2 <- lm(mpg ~ cyl+disp+hp+drat+wt+qsec+factor(vs)+factor(am)+gear+carb, data = mtcars)
summary(modelo_v2)
```

```
# Detectando Colinearidade

# Um grande problema com a regressão multivariada é a colinearidade. Se duas ou mais variáveis
# preditivas são altamente correlacionadas e ambas são inseridas em um modelo de regressão,
# aumenta o verdadeiro erro padrão e você obtém estimativas muito instáveis da inclinação.
# Podemos avaliar a colinearidade pelo Fator de Inflação de Variância (VIF). Vamos analisar os
# fatores de inflação da variação se lançarmos todas as variáveis no modelo.

library(car)
?vif
kable(vif(modelo_v2), align = 'c')

# Valores para o VIF maior que 10 são considerados grandes. Também devemos prestar atenção aos
# valores de VIF entre 5 e 10. Nesse ponto, podemos considerar deixar apenas uma dessas variáveis
# no modelo.


# Método de Seleção Gradual de Atributos (Stepwise Selection Method)

# Entre os métodos disponíveis, vamos realizar a seleção gradual para nos ajudar a selecionar
# um subconjunto de variáveis que melhor explicam o MPG. Observe que também tratamos a variável (vs)
# como uma variável categórica.
?lm
modelo_v3 <- lm(mpg ~ cyl+disp+hp+drat+wt+qsec+factor(vs)+factor(am)+gear+carb, data = mtcars)

# O critério de informação de Akaike (AIC) é um estimador de erro de previsão e, portanto, estima a
# qualidade relativa de modelos estatísticos para um determinado conjunto de dados.

# Dada uma coleção de modelos para os dados, a AIC estima a qualidade de cada modelo, em relação
# a cada um dos outros modelos. Assim, o AIC fornece um meio para a seleção de modelos.

# O Coeficiente AIC é baseado na teoria da informação. Quando um modelo estatístico é usado para
# representar o processo que gerou os dados, a representação quase nunca será exata; portanto,
# algumas informações serão perdidas usando o modelo para representar o processo. A AIC estima
# a quantidade relativa de informações perdidas por um determinado modelo: quanto menos informações
# um modelo perde, maior a qualidade desse modelo.
?stepAIC
step <- stepAIC(modelo_v3, direction = "both", trace = FALSE)
summary(step)
summary(step)$coeff
summary(step)$r.squared


# Isso mostra que, além da transmissão, o peso do veículo (wt) e a velocidade de
# aceleração (qsec) têm a maior relação com a explicação da variação em mpg.
# O R^2 é de 85%, o que significa que o modelo explica 85% da variação
# em mpg, indicando que é um modelo robusto e altamente preditivo.


# Ajustando o modelo final

# Agora, usando as variáveis selecionadas, podemos ajustar o modelo final.

modelo_final <- lm(mpg ~ wt+qsec+factor(am), data = mtcars)
summary(modelo_final)

# Você pode observar que todas as variáveis agora são estatisticamente
# significativas. Este modelo explica 85% da variação em milhas por galão (mpg).
# Agora, quando lemos o coeficiente para a variável am, dizemos que, em média,
# os carros com transmissão manual têm 2,94 MPGs a mais de autonomia que os carros com transmissão
# automática. No entanto, esse efeito foi muito maior do que quando não ajustamos
# wt e o qsec.


# Diagnóstico de Regressão

# Realizaremos agora alguns diagnósticos no modelo de regressão final.

# Desta vez, analisando os fatores de inflação de variação, revelamos que os números
# são razoáveis e não detectamos nenhum sinal de colinearidade.

# Detectando colinearidade
kable (vif(modelo_final), align = 'c')

# Resíduos versus os valores ajustados

# Ao plotar resíduos versus os valores ajustados, procuramos qualquer tipo de padrão.
```

```
# O mesmo ocorre com os valores ajustados versus os padronizados, quando estamos plotando
# uma função dos resíduos padronizados. O gráfico abaixo mostra que não existe um
# padrão específico nos resíduos.
modelo_final$fitted.values
?qqPlot
qqPlot(modelo_final, main = "Normal Q-Q plot")


#####---------- Conclusões ----------

# A partir do resumo (modelo_final) podemos concluir o seguinte:
coefficients(modelo_final)
confint(modelo_final)
summary(modelo_final)

# Milhas por galão (mpg) irá aumentar em 2.93 em carros com transmissão 'Manual'
# em comparação com carros com transmissão 'Automatic' (ajustado por wt e qsec).
# Conclusão para a Motor Trend Magazine é: 'Transmissão manual' é melhor para mpg.

# Milhas por galão (mpg) irá diminuir em 3.9 por cada 1000 lb de aumento em peso.
# Conclusão: Milhas por galão (mpg) diminui com aumento de peso (wt) do veículo.

# Milhas por galão (mpg) irá aumentar em um fator de 1.2 se aumentarmos em 1 ponto a aceleração.
# Conclusão: Milhas por galão (mpg) aumenta com um leve aumento da velocidade de aceleração.
```

Teste de Hipóteses Como Ferramenta de Negócio

Os Testes de Hipóteses são uma importante ferramenta estatística e de negócios. Podemos formular uma hipótese, coletar dados, aplicar um teste e então rejeitar ou não a hipótese, o que pode ser usado para suportar a tomada de decisões.

Muitos testes estatísticos aplicam Testes de Hipóteses aos dados como forma de apresentar os resultados.

Estudo de Caso – Teste de Hipóteses

```
# Teste de Hipóteses

# Obs: Caso tenha problemas com a acentuação, consulte este link:
# https://support.rstudio.com/hc/en-us/articles/200532197-Character-Encoding

# Configurando o diretório de trabalho
# Coloque entre aspas o diretório de trabalho que você está usando no seu computador
# Não use diretórios com espaço no nome
setwd("~/Dropbox/DSA/Business-Analytics2.0/Cap03/R")
getwd()

# O Teste de Hipóteses é uma ferramenta estatística útil que pode ser usada para tirar uma conclusão
# sobre a população a partir de uma amostra.

# Pacotes
install.packages("readxl")
library(readxl)
library(e1071)

# Carregando o Dataset
dados <- read_excel("dados/Bola_Futebol.xlsx")
View(dados)
dim(dados)
str(dados)

##### Análise Exploratória #####

# Verificando valores missing
colSums(is.na(dados))

# Calculando as estatísticas
summary(dados)

# Vetor com o nome das estatísticas
```

```
nomes_stats <- c("Média", "Desvio Padrão", "Variância", "Tipo de Bola")

# Calculando as estatísticas para a Bola com Revestimento Atual
dados_stats_atual <- c(round(mean(dados$Atual), digits = 2),
                       round(sd(dados$Atual), digits = 2),
                       round(var(dados$Atual), digits = 2),
                       "Bola com Revestimento Atual")

# Calculando as estatísticas para a Bola com Revestimento Novo
dados_stats_novo <- c(round(mean(dados$Novo), digits = 2),
                      round(sd(dados$Novo), digits = 2),
                      round(var(dados$Novo), digits = 2),
                      "Bola com Revestimento Novo")

# Combina os resultados para comparação
dados_stats_combined <- rbind(nomes_stats, dados_stats_atual, dados_stats_novo)
View(dados_stats_combined)

# Análise Univariada

# Range
range_atual <- max(dados$Atual) - min(dados$Atual)
range_atual

range_novo <- max(dados$Novo) - min(dados$Novo)
range_novo

# Intervalo Interquartil
summary(dados)

IQR_atual <- IQR(dados$Atual)
IQR_atual

IQR_novo <- IQR(dados$Novo)
IQR_novo

# O intervalo interquartil de ambas as variáveis é equivalente aproximadamente a 12, ou seja, não
# há diferença entre os valores do primeiro ao terceiro quartil.

# Da média, pode-se dizer que as bolas do modelo de revestimento atual cobrem uma distância maior
# que as bolas do novo modelo (em média), mas o desvio padrão é muito parecido.
# Portanto, podemos dizer que não há muita diferença (significativa) nos dois modelos de revestimento da bola.

# Representação Gráfica

# A distribuição normal é a distribuição mais fácil de trabalhar, a fim de obter um entendimento
# sobre estatísticas. As distribuições da vida real geralmente são distorcidas. Muita assimetria
# e muitas técnicas estatísticas não funcionam. Como resultado, são utilizadas técnicas matemáticas
# avançadas, incluindo logaritmos e técnicas de regressão quantílica.
# http://www.econ.uiuc.edu/~roger/research/rq/rq.html

# Ajusta a área de plotagem
par(mfrow = c(2,2))

# Histograma
hist(dados$Atual,
    main = "Distância - Bola Com Revestimento Atual",
    xlab = "Distância (metros)",
    ylab = "Número de Bolas",
    col = "Blue")

hist(dados$Novo,
    main = "Distância - Bola Com Revestimento Novo",
    xlab = "Distância (metros)",
    ylab = "Número de Bolas",
    col = "Green")

# Um histograma é inclinado para a direita se o pico do histograma virar para a esquerda.
# Portanto, a cauda do histograma tem uma inclinação positiva para a direita.
# O raciocínio inverso é o mesmo.

# Boxplot
boxplot(dados$Atual,
       main = "Distância - Bola Com Revestimento Atual",
       xlab = "Distância (metros)",
       ylab = "Número de Bolas",
       col = "Blue",
       horizontal = TRUE)
```

```
boxplot(dados$Novo,
        main = "Distância - Bola Com Revestimento Novo",
        xlab = "Distância (metros)",
        ylab = "Número de Bolas",
        col = "Green",
        horizontal = TRUE)

# Boxplot assimétrico à esquerda
# Se a maior parte das observações estiver na extremidade superior da escala, um boxplot
# será assimétrico à esquerda.
# Consequentemente, o "bigode" esquerdo é mais longo que o "bigode" direito.
# Como resultado, a média é menor que a mediana.

# Boxplot assimétrico à direita
# Se um boxplot estiver assimétrico à direita, a caixa mudará para a esquerda e o
# "bigode" direito ficará mais longo. Como resultado, a média é maior que a mediana.

# Portanto, a variável com revestimento atual é assimétrica à direita, pois a maioria dos
# dados está localizada à direita do "pico". A média é maior que a mediana.

# A variável revestimento novo também é inclinada (assimétrica) à direita, pois a maioria dos dados
# está localizada à direita do gráfico. A média é maior que a mediana.

summary(dados$Atual)
summary(dados$Novo)

# Assimetria

# Se uma cauda é mais longa que a outra, a distribuição é inclinada ou assimétrica.
# Às vezes, essas distribuições são chamadas de distribuições assimétricas, pois não mostram
# nenhum tipo de simetria. Simetria significa que metade da distribuição é uma imagem espelhada
# da outra metade. Por exemplo, a distribuição normal é uma distribuição simétrica sem inclinação.
# As caudas são exatamente as mesmas.

# Uma distribuição inclinada para a esquerda tem uma longa cauda esquerda.
# As distribuições inclinadas para a esquerda também são chamadas de distribuições inclinadas
# negativamente. Isso ocorre porque há uma cauda longa na direção negativa na linha numérica.
# A média está à esquerda do pico.

# Uma distribuição inclinada à direita tem uma longa cauda direita.
# Distribuições com inclinação à direita também são chamadas de distribuições com inclinação positiva.
# Isso ocorre porque há uma cauda longa na direção positiva na linha numérica.
# A média está à direita do pico.

# Em uma distribuição normal, a média e a mediana são o mesmo número, enquanto a média e a mediana
# em uma distribuição assimétrica se tornam números diferentes. Podemos então calcular o
# coeficiente de assimetria.

# Se a assimetria é menor que -1 ou maior que 1, a distribuição é altamente distorcida.
# Se a assimetria é entre -1 e -0,5 ou entre 0,5 e 1, a distribuição é enviesada (assimétrica) moderadamente.
# Se a assimetria é entre -0,5 e 0,5, a distribuição é aproximadamente simétrica.

# A medida positiva indicaria que a média dos valores dos dados é maior do que a mediana e
# a distribuição dos dados é inclinada para a direita.

# Uma medida negativa indica que a média dos valores dos dados é menor que a mediana e a
# distribuição dos dados é inclinada para a esquerda.

skewness(dados$Atual)
summary(dados$Atual)

skewness(dados$Novo)
summary(dados$Novo)

# Curtose

# A curtose informa a altura e a nitidez do pico central, em relação a uma curva de sino padrão.
# A distribuição normal tem kurtosis igual a zero.

# Uma curtose negativa significa que sua distribuição é mais plana que uma curva normal com a
# mesma média e desvio padrão. O raciocínio inverso é o mesmo.
kurtosis(dados$Atual)
kurtosis(dados$Novo)

# Se você estiver executando um teste estatístico paramétrico em seus dados (por exemplo, uma ANOVA),
# o uso de dados altamente inclinados para a direita ou esquerda pode levar a resultados enganosos.
# Portanto, se você deseja executar um teste nesse tipo de dados, execute uma transformação de log e
```

```
# execute o teste nos números transformados.

# Teste de Normalidade - Shapiro Test
# Hipótese Nula (H0): Os dados são normalmente distribuídos.
# Hipótese Alternativa (H1): Os dados não são normalmente distribuídos.

# Se o valor-p for maior que 0.05 não rejeitamos a hipótese nula e podemos assumir a normalidade dos dados.
# Se o valor-p for menor que 0.05 rejeitamos a hipótese nula e não podemos assumir a normalidade dos dados.
?shapiro.test
shapiro.test(dados$Atual)
shapiro.test(dados$Novo)

# Conclusão da Análise Univariada

# A partir do boxplot e demais análises, pode-se dizer que não há discrepâncias.
# A média do modelo de revestimentpo atual é 270,3 e a média do novo modelo é 267,5 e o desvio padrão
# do atual é 8,7529 e o desvio padrão do novo é 9,8969. Não parece haver diferença significativa.

# Análise Bivariada

# Ajusta a área de plotagem
par(mfrow = c(1,1))

# Scatter Plot
plot(dados$Atual, dados$Novo)

# Correlação
cor(dados$Atual, dados$Novo)


# Teste de Hipóteses

# O Teste de Hipóteses é uma forma de inferência estatística que usa dados de uma amostra para tirar
# conclusões sobre um parâmetro populacional ou uma distribuição de probabilidade populacional.

# Primeiro, é feita uma suposição provisória sobre o parâmetro ou distribuição. Essa suposição é chamada
# de hipótese nula e é indicada por H0. Uma hipótese alternativa (denotada H1), é o oposto do que é
# declarado na hipótese nula. O procedimento de teste de hipóteses envolve o uso de dados
# de amostra para determinar se H0 pode ou não ser rejeitada.

# Se H0 for rejeitada, a conclusão estatística é que a hipótese alternativa H1 é verdadeira.

# Formulação da Hipótese

# Hipótese Nula (H0) – Não há diferença significativa entre a distância percorrida das bolas de futebol
# com revestimento atual e novo.
# H0: muAtual - muNovo igual a 0 (zero)

# Hipótese Alternativa (H1) – Há diferença significativa entre a distância percorridad as bolas de futebol
# com revestimento atual e novo.
# H1: muAtual - muNovo diferente de 0 (zero)

# A condição preliminar para se aplicar o teste t é a existência de distribuição normal dos dados
# em ambos grupos de dados.

# Todavia, existe um impasse! Há três tipos de test t:
# teste t de uma amostra
# teste t de amostras independentes
# teste t de amostras relacionadas (pareadas)

# Usamos o teste t de uma amostra para verificar os valores da variável em relação a média conhecida
# de uma população.

# Para realizarmos os testes de igualdade de variâncias e os testes de médias, precisamos que as
# duas populações
 sejam independentes. Esse é um teste de amostras independentes. Por isso paired = F.

?t.test
teste_hipo <- t.test(dados$Atual, dados$Novo, paired = F, conf.level = 0.95, alternative = "t")
teste_hipo

# O teste t pareado é útil para analisar o mesmo conjunto de itens que foram medidos sob duas
# condições diferentes, as diferenças nas medições feitas sobre o mesmo assunto antes e depois
# de um tratamento, ou diferenças entre dois tratamentos dados ao mesmo assunto.

# Interpretação do nosso resultado:

# Falhamos em rejeitar a hipótese nula, pois valor-p é maior que o limite de 0.05.
```

```
# Significa dizer que há uma probabilidade alta de não haver diferença significativa entre
# os tipos de revestimento das bolas de futebol.

# Valor-p

# Probabiblidade de que a estatística do teste assuma um valor extremo em relação ao valor
# observado quando H0 é verdadeira.

# Lembre-se disso (considerando oo valor-p de 0.05):

# Valor-p Baixo: Forte evidência empírica contra H0.
# Valor-p Alto: Pouca ou nenhuma evidência empírica contra H0

# Determinando a Força do Teste e o Tamanho o Ideal de Amostra

# Diferença das médias
delta_mean <- mean(dados$Atual) - mean(dados$Novo)
delta_mean

# Desvio padrão da diferença entre os dados
delta_desvio <- sd(dados$Atual - dados$Novo)
delta_desvio

# Size Effect
size_effect = delta_mean/delta_desvio
size_effect

# Power Test - Força do Teste
library(pwr)
?pwr.t.test
dim(dados)
power_teste <- pwr.t.test(n = 40, d = size_effect, sig.level = 0.05, alternative = "t")
power_teste

# Tamanho ideal da amostra
tamanho_amostra <- pwr.t.test(power = .95, d = 0.5, type = "t", alternative = "t", sig.level = .05)
tamanho_amostra

# A Conclusão da Análise e Recomendações ao Cliente você encontra no manual
# em pdf no próximo item de aprendizagem.
```

Conclusão e Recomendações ao Nosso Cliente

Resultado do Teste de Hipóteses

1. O valor-p é 0,188 e, portanto, falhamos em rejeitar a hipótese nula. Logo, podemos dizer que não há diferença significativa entre a distância percorrida pela bola de futebol com revestimento atual e novo.
2. Deve haver algum fator diferente do revestimento que afeta a distância percorrida pela bola de futebol.
3. A empresa não deveria introduzir o novo modelo no mercado.

Intervalos de confiança

1. Intervalo de confiança de 95% para o modelo atual [273.0743, 267.4757]
2. Intervalo de confiança de 95% para o novo modelo [270.6652, 264.3348]

Assim, a respectiva média populacional deve estar dentro dessa faixa para obter resultados consistentes.

Força do Teste e Tamanho da Amostra

1. A força do teste é 0,144, o que é baixo.
2. Tamanho da amostra: O teste deve ser realizado com um tamanho de amostra maior, pelo menos 105 registros.

3. É sempre melhor ter um tamanho de amostra maior para obter melhores resultados, mas também deve-se levar em consideração outros fatores como custo, praticabilidade e tempo.

Recomendações

1. Estatisticamente, não há diferença significativa entre a distância percorrida entre as bolas de futebol com revestimento atual e novo.
2. A empresa deve continuar seu estudo e considerar outros fatores que afetam a distância percorrida, como o tamanho e o peso da bola, seu material de fabricação - enchimento, tecnologia e número de gomos.
3. A empresa também deve considerar a tecnologia predominante no setor e os produtos lançados por seus concorrentes.
4. A partir de agora, para aumentar sua participação no mercado, a empresa deve se concentrar em outras áreas como marketing, estratégia e vendas e seguir trabalhando em pesquisa e desenvolvimento.

Estudo Dirigido – Simulação de Monte Carlo e Séries Temporais Para Modelagem Financeira

Simulações de Monte Carlo são usadas para modelar a probabilidade de resultados diferentes em um processo que não pode ser facilmente previsto devido à intervenção de variáveis aleatórias. É uma técnica usada para entender o impacto do risco e da incerteza nos modelos de previsão.

A Simulação de Monte Carlo pode ser usada para resolver uma série de problemas em praticamente todos os campos, como finanças, engenharia, cadeia de suprimentos e ciência. A Simulação de Monte Carlo também é conhecida como simulação de múltiplas probabilidades.

Quando confrontada com uma incerteza significativa no processo de fazer uma previsão ou estimativa, em vez de apenas substituir a variável incerta por um único número médio, a Simulação de Monte Carlo pode vir a ser uma solução melhor. Como os negócios e as finanças são afetados por variáveis aleatórias, as simulações de Monte Carlo têm uma vasta gama de aplicações em potencial nesses campos. As simulações são usadas para estimar a probabilidade de exceder os custos em grandes projetos e a probabilidade de um preço de ativo se mover de uma certa maneira. As telecomunicações as utilizam para avaliar o desempenho da rede em diferentes cenários, ajudando-os a otimizar a rede. Os analistas os utilizam para avaliar o risco de uma entidade adiar e analisar derivativos, ou outros produtos financeiros. Seguradoras e perfuradores de poços de petróleo também os utilizam.

As simulações de Monte Carlo têm inúmeras aplicações fora dos negócios e das finanças, como meteorologia, astronomia e física de partículas. As simulações de Monte Carlo têm o nome do hot spot de apostas em Mônaco, pois o acaso e os resultados aleatórios são centrais na técnica de modelagem, assim como em jogos como roleta, dados e caça-níqueis. A técnica foi desenvolvida pela primeira vez por Stanislaw Ulam, um matemático que trabalhou no Projeto Manhattan. Após a guerra, enquanto se recuperava de uma cirurgia no cérebro, Ulam se divertiu jogando inúmeros jogos de paciência. Ele ficou interessado em traçar o resultado de cada um desses jogos, a fim de observar sua distribuição e determinar a probabilidade de vitória. Depois que ele compartilhou sua ideia com John Von Neumann, os dois colaboraram para desenvolver a simulação de Monte Carlo.

Exemplo de simulações de Monte Carlo: Modelagem de Preços de Ativos

Uma maneira de empregar uma simulação de Monte Carlo é modelar possíveis movimentos de preços de ativos. Existem dois componentes nos movimentos de preços de um ativo: drift, que é um movimento direcional constante, e uma entrada aleatória, que representa a volatilidade do mercado. Ao analisar dados históricos de preços, é possível determinar a drift, o desvio padrão, a variação e o movimento médio dos preços de um título. Estes são os blocos de construção de uma simulação de Monte Carlo.

Estudo Dirigido

O Estudo Dirigido é um material de auto estudo. Faça pesquisa adicional se necessário, altere o código, incorpore outras análises, explore as variáveis e mais importante: associe o conteúdo deste material com o que foi estudado nos Capítulos 2 e 3 deste curso.

Neste trabalho vamos usar a Simulação de Monte Carlo para prever a cotação de fechamento das ações da empresa Cedar Realty Trust, Inc. (CDR), listada na bolsa de valores americana. Site da empresa:

http://cedarrealtytrust.com

Para projetar uma possível trajetória de preço, usaremos os dados históricos de preço do ativo para gerar uma série de retornos diários periódicos. Aplicaremos a Simulação de Monte Carlos e faremos previsões sobre a cotação futura das ações.

Os dados serão fornecidos a você junto com o Jupyter Notebook disponível ao final do capítulo. Os dados foram extraídos do portal de finanças do Yahoo:

https://finance.yahoo.com/quote/CDR/history/

Coletamos dados entre 1994 e 2020. Estude o Jupyter Notebook, leia cada comentário e execute cada etapa. Faça mudanças nos parâmetros da simulação e analise os resultados.

```
# <font color='blue'>Data Science Academy</font>
# <font color='blue'>Business Analytics</font>
# <font color='blue'>Capítulo 3</font>
### Estudo Dirigido - Simulação de Monte Carlo e Séries Temporais Para Modelagem Financeira
# ->
# Versão da Linguagem Python
from platform import python_version
print('Versão da Linguagem Python Usada Neste Jupyter Notebook:', python_version())
> Todos os detahes sobre este projeto estão no manual em pdf no item anterior.
# ->
# Para atualizar um pacote, execute o comando abaixo no terminal ou prompt de comando:
# pip install -U nome_pacote
# Para instalar a versão exata de um pacote, execute o comando abaixo no terminal ou prompt de comando:
# pip install nome_pacote==versão_desejada
# Depois de instalar ou atualizar o pacote, reinicie o jupyter notebook.
# Instala o pacote watermark.
# Esse pacote é usado para gravar as versões de outros pacotes usados neste jupyter notebook.
!pip install -q -U watermark
### Carregando os Pacotes
# ->
# Imports para manipulação de dados
import numpy as np
import pandas as pd
# Imports para visualização
import matplotlib.pyplot as plt
import matplotlib as m
import seaborn as sns
# Imports para cálculos estatísticos
import scipy
from scipy.stats import kurtosis, skew, shapiro
import warnings
warnings.filterwarnings("ignore")
# Imports para formatação dos gráficos
```

```
plt.style.use('fivethirtyeight')
m.rcParams['axes.labelsize'] = 14
m.rcParams['xtick.labelsize'] = 12
m.rcParams['ytick.labelsize'] = 12
m.rcParams['text.color'] = 'k'
from matplotlib.pylab import rcParams
rcParams['figure.figsize'] = 20,10
# ->
# Versões dos pacotes usados neste jupyter notebook
%reload_ext watermark
%watermark -a "Data Science Academy" --iversions
### Carregando os Dados
# ->
# Carrega o dataset
# Obs: O valor das ações foi ajustado para facilitar a criação dos gráficos de forma didática
dados = pd.read_csv("dados/dataset.csv", parse_dates = True, index_col = "Data")
# ->
# Visualizando registros
# Cada coluna representa o valor da ação em cada dia da série.
# valor de abertura, fechamento, máximo, mínimo e volume.
# A coluna Mudanca(%) representa a variação diária.
dados.head()
# ->
# Tipos de Dados
dados.dtypes
# ->
# Shape
dados.shape
# ->
# Sumário estatístico
dados.describe()
## Visualizando o Preço Diário de Fechamento das Ações no Tempo
# ->
# Plot
plt.plot(dados["Fechamento"])
plt.title("Preço Diário de Fechamento das Ações", size = 14)
plt.show()
Vamos calcular o retorno diário da série.
# ->
# Calculando o percentual de mudança na cotação de fechamento diário das ações
# Ou seja, quanto o valor de fechamento varia de um dia para outro, o retorno diário da ação
retorno_diario = dados["Fechamento"].pct_change().dropna()
retorno_diario.head()
Vamos calcular o retorno acumulado da série.
# ->
# Retorno acumulado
retorno_diario_acumulado = (1 + retorno_diario).cumprod() - 1
retorno_diario_acumulado.max()
### Análise Exploratória e Estatística Descritiva
Vamos usar a estatística para calcular o retorno médio e a variação (desvio padrão).
# ->
# Média do fechamento diário da cotação das ações
media_retorno_diario = np.mean(retorno_diario)
# ->
# Desvio padrão do fechamento diário da cotação das ações
desvio_retorno_diario = np.std(retorno_diario)
# ->
# Média e desvio padrão
print("Média do Retorno de Fechamento:", media_retorno_diario)
print("Desvio Padrão do Retorno de Fechamento:", desvio_retorno_diario)
Vamos considerar o ano com 252 dias de funcionamento da bolsa americana.
# ->
# Média e desvio padrão no ano (considerando 252 dias úteis de operações na bolsa americana)
print("Retorno Médio Anualizado de Fechamento:", (1 + media_retorno_diario) ** 252 - 1)
print("Desvio Padrão Anualizado de Fechamento:", desvio_retorno_diario*np.sqrt(252))
Embora nos últimos anos a performance das ações tenha sido boa, na média o ganho tem sido baixo, embora positivo. No longo prazo o investidor não perdeu dinheiro. Vamos criar um plot com o Retorno Diário.
# ->
# Plot
plt.plot(retorno_diario)
plt.title("Retorno Diário", size = 14)
plt.show()
Com apenas duas grandes variações, o retorno diário tem sido constante ao longo do tempo. Vamos criar um histograma com a dstribuição do retorno diário.
# ->
# Plot
plt.hist(retorno_diario, bins = 75)
plt.title("Histograma do Retorno Diário", size = 14)
```

```
plt.show()
```
Os valores estão bem próximos mesmo da média. Mas vamos confirmar isso calculando curtose e assimetria.
```
# ->
# Calculando curtose e assimetria
print("Curtose:", kurtosis(retorno_diario))
print("Assimetria:", skew(retorno_diario))
```
A curtose indica que os registros estão bem próximos da média. Mas a assimetria indica que os dados estão bem distorcidos e distantes de uma distribuição normal. Vamos aplicar o teste de normalidade na série.
Teste de Normalidade Shapiro-Wilk
```
# ->
# Executa o teste de normalidade para a série
teste_normalidade = shapiro(retorno_diario)[1]
# Verifica o retorno com base no valor-p de 0.05
if teste_normalidade <= 0.05:
    print("Rejeitamos a Hipótese Nula de Normalidade dos Dados.")
else:
    print("Falhamos em Rejeitar a Hipótese Nula de Normalidade dos Dados.")
```
Como imaginávamos a distribuição não é normal. Vamos aplicar uma transformação de log à série e então aplicar a técnica de diferenciação para retirar da série os padrões de tendência e deixarmos apenas os dados reais, que nos interessam. Com isso calculamos o retorno diário.
```
# ->
# Transformação de log e diferenciação para cálculo do retorno diário
log_retorno_diario = (np.log(dados["Fechamento"]) - np.log(dados["Fechamento"]).shift(-1)).dropna()
# Calculamos média e desvio padrão após a transformação
log_media_retorno_diario = np.mean(log_retorno_diario)
log_desvio_retorno_diario = np.std(log_retorno_diario)
```
Vamos criar um plot com o retorno diário da série transformada.
```
# ->
# Plot
plt.plot(log_retorno_diario)
plt.title("Retorno Diário (Log Transformation)", size = 14)
plt.show()
# ->
# Plot
plt.hist(log_retorno_diario, bins = 75)
plt.title("Histograma do Retorno Diário (Log Transformation)", size = 14)
plt.show()
# ->
# Calculando curtose e assimetria
print("Curtose:", kurtosis(log_retorno_diario))
print("Assimetria:", skew(log_retorno_diario))
# ->
# Executa o teste de normalidade para a série
teste_normalidade = shapiro(log_retorno_diario)[1]
# Verifica o retorno com base no valor-p de 0.05
if teste_normalidade <= 0.05:
    print("Rejeitamos a Hipótese Nula de Normalidade dos Dados.")
else:
    print("Falhamos em Rejeitar a Hipótese Nula de Normalidade dos Dados.")
```
Os dados ainda não são normais, porém reduzimos a distorção dos dados. Poderíamos aplicar outras transformações, mas para o objetivo este estudo isso é suficiente. Seguimos com a série transformada.
Valor Histórico
Vamos calcular o valor histórico do preço da ação.
```
# ->
# Nível de variância
var_level = 95
var = np.percentile(log_retorno_diario, 100 - var_level)
print("Certeza de que as perdas diárias não excederão o VaR% em um determinado dia com base em valores históricos.")
print("VaR 95%:", var)
# ->
# Var para os próximos 5 dias
var * np.sqrt(5)
```
Valor Histórico Condicional
```
# ->
# Nível de variância
var_level = 95
var = np.percentile(log_retorno_diario, 100 - var_level)
cvar = log_retorno_diario[log_retorno_diario < var].mean()
print("Nos piores 5% dos casos, as perdas foram, em média, superiores ao percentual histórico.")
print("CVaR 95%:", cvar)
```
Monte Carlo simulation
```
# ->
# Simulação de Monte Carlo
# Número de dias a frente
dias_posteriores = 252
# Número de simulações
simulacoes = 2500
# Último valor da ação
ultimo_preco = 270.3
```

```
# Cria um array vazio com as dimensões
results = np.empty((simulacoes, dias_posteriores))
# Loop por cada simulação
for s in range(simulacoes):
    # Calcula o retorno com dados randômicos seguindo uma distribuição normal
    random_returns = 1 + np.random.normal(loc = log_media_retorno_diario,
                                          scale = log_desvio_retorno_diario,
                                          size = dias_posteriores)
    result = ultimo_preco * (random_returns.cumprod())
    results[s, :] = result
# ->
# Definindo o índice da série simulada
index = pd.date_range("2020-03-11", periods = dias_posteriores, freq = "D")
resultados = pd.DataFrame(results.T, index = index)
media_resultados = resultados.apply("mean", axis = 1)
## Resultado da Simulação de Monte Carlo
# ->
# Dividindo a área de plotagem em 2 subplots
fig, ax = plt.subplots(nrows = 2, ncols = 1)
# Plot
ax[0].plot(dados["Fechamento"][:"2018-12-31"])
ax[0].plot(resultados)
ax[0].axhline(270.30, c = "orange")
ax[0].set_title(f"Monte Carlo {simulacoes} Simulações", size = 14)
ax[0].legend(["Preço Histórico", "Último Preço = 270.30"])
ax[1].plot(dados["Fechamento"][:"2018-12-31"])
ax[1].plot(resultados.apply("mean", axis = 1), lw = 2)
ax[1].plot(media_resultados.apply((lambda x: x * (1+1.96 * log_desvio_retorno_diario))),
           lw = 2, linestyle = "dotted", c = "gray")
ax[1].plot(media_resultados, lw = 2, c = "orange")
ax[1].plot(media_resultados.apply((lambda x: x * (1-1.96 * log_desvio_retorno_diario))),
           lw = 2, linestyle = "dotted", c = "gray")
ax[1].set_title(f"Resultado Médio Monte Carlo {simulacoes} Simulações", size = 14)
ax[1].legend(["Preço", "Previsão Média", "2x Desvio Padrao"])
plt.show()
A previsão é positiva com os dados simulados e no longo prazo as ações da CDR tendem a valorizar. Mas não espere um grande retorno dessas ações.
# Fim
### Obrigado - Data Science Academy - <a href="http://facebook.com/dsacademybr">facebook.com/dsacademybr</a>
```

Todo Business é Analytics

A análise de dados é a ciência da extração de padrões, tendências e informações acionáveis de grandes conjuntos de dados. Pense em análise de dados como a maneira pela qual as empresas usam dados para melhorar sua gestão e operações. A análise de dados envolve o processo de dar forma e sentido aos dados, antes de agir sobre eles. Mais ainda, você pode fatiar os dados para extrair insights que permitem alavancar os processos internos de qualquer organização, gerando uma vantagem competitiva. Todo Business é Analytics. Não existe um único negócio que não se beneficie de análise de dados e vantagem competitiva está em saber fazer isso melhor e mais rápido que os concorrentes.

Os dados e informações, em geral, estão proliferando em uma taxa exponencial, graças ao rápido crescimento nos três aceleradores digitais que vem explodindo desde os anos 80: largura de banda, armazenamento digital e poder de processamento. Os "grandes dados" (Big Data) que temos hoje não serão nada em comparação com a abundância de dados e informações disponíveis para nós no futuro próximo - dados gerados por centenas de milhões de dispositivos, coisas como tecnologia wearable, smartphones e tudo o que faz parte da Internet das Coisas (IoT). Agora responda: alguma empresa pode imaginar não implementar um processo de Analytics? Isso é razoável?

Ao aproveitar a grande quantidade de dados disponíveis, você pode colocar sua empresa à frente de disrupções em sua área de atuação, alavancando os dados para aumentar sua posição competitiva em relação a outros em seu mercado.

Os dados disponíveis das mídias sociais são um grande exemplo da maneira pela qual as novas tecnologias digitais proporcionam às empresas uma compreensão mais abrangente do consumidor. Meios de comunicação social tornaram-se incorporados em quase todas as esferas da vida. Notícias circulam em canais de mídia social, promoções de diferentes empresas aparecem lado a lado aos sentimentos cotidianos dos usuários. As ações e interações são gravadas e coletadas - você pode usar isso para chegar à frente do seu concorrente.

Qual é o seu público-alvo? Quais são as tendências? Mais especificamente, como eles estão interagindo com sua empresa? Eles estão compartilhando ou gostando do que você postou? A mídia social não apenas expande o alcance da coleta de dados, como torna o processo de coleta mais rápido. Sua empresa pode tomar decisões rápidas com base na comunicação quase instantânea entre seus clientes, o público ou você e seu público.

Mas mídia social é apenas um campo de disrupção digital que está gerando mais dados para que possamos extrair mais insights de. As formas como as transações econômicas ocorrem foram digitalizadas de tal forma que podemos analisar não só a interação do consumidor com o marketing e as notícias, mas também como e quando elas agem depois de gostar ou compartilhar a promoção de uma empresa. O monitoramento e coleta de dados dos sites da empresa permite o rastreamento de consumidores, mapeando seu gostos, preferências, localização e interação social. Esta lista continua.

Com quantidades exponencialmente crescentes de dados acumulando em tempo real, não há nenhuma razão para não criar processo de Business Analytics e alavancar uma vantagem competitiva.

8.4. Marketing Analytics – Projeto 1 – Segmentação de Clientes de Food Delivery

Por Que Marketing Analytics é Importante?

Ao longo dos anos, à medida que as empresas expandiram para novas categorias de marketing, novas tecnologias foram adotadas para apoiá-las. Como cada nova tecnologia era normalmente implantada isoladamente, e o resultado foi uma mistura de ambientes de dados desconectados.

Consequentemente, os profissionais de marketing costumam tomar decisões com base em dados de canais individuais (marketing digital e métricas de sites, por exemplo), sem levar em consideração todo o cenário de marketing. Dados de mídia social por si só não são suficientes. Dados de análise da Web por si só não são suficientes. E ferramentas que analisam apenas um instantâneo do tempo de um único canal são lamentavelmente inadequadas.

Marketing Analytics, por outro lado, considera todos os esforços de marketing em todos os canais ao longo de um período de tempo - o que é essencial para a tomada de decisões corretas e a execução eficiente das campanhas e iniciativas de marketing de uma empresa.

Por Que Segmentar Clientes?

A segmentação permite que os profissionais de marketing adaptem melhor seus esforços de marketing a vários subconjuntos de público-alvo. Esses esforços podem estar relacionados às comunicações e ao desenvolvimento de produtos. Especificamente, a segmentação ajuda uma empresa a:

• Criar e comunicar mensagens de marketing direcionadas que ressoarão com grupos específicos de clientes, mas não com outros (que receberão mensagens personalizadas para suas necessidades e interesses).
• Selecionar o melhor canal de comunicação para o segmento, que pode ser e-mail, publicações em mídias sociais, publicidade em rádio ou outra abordagem, dependendo do segmento.
• Identificar maneiras de melhorar produtos ou novas oportunidades de produtos ou serviços.
• Estabelecer melhores relacionamentos com os clientes.
• Testar as opções de preços.
• Concentrar-se nos clientes mais rentáveis.
• Melhorar o atendimento ao cliente.
• Escolher entre venda por atacado e venda cruzada de outros produtos e serviços.

O Que Deve Ser Feito na Análise Exploratória?

A Análise Exploratória de Dados refere-se ao processo de realização de investigações iniciais sobre dados, a fim de descobrir padrões, detectar anomalias, testar hipóteses e verificar suposições com a ajuda de resumos estatísticos e representações gráficas.

A Análise Exploratória de Dados (EDA), também conhecida como Exploração de Dados, é uma etapa do Processo de Análise de Dados, em que várias técnicas são usadas para entender melhor o conjunto de dados que está sendo usado.

“Compreender o conjunto de dados” pode se referir a várias coisas, incluindo, entre outras:

• Extrair variáveis importantes e deixar para trás variáveis inúteis (nem todas as variáveis no dataset serão relevantes).
• Identificar outliers, valores ausentes ou erro humano.
• Compreender os relacionamentos, ou falta de, entre variáveis.
• Verificar a distribuição e tipo das variáveis.

Isso permite maximizar sua compreensão sobre um conjunto de dados e minimizar possíveis erros que podem ocorrer posteriormente no processo.

Aqui você encontra 7 exemplos de operações que podem ser feitas durante a Análise Exploratória, com exemplos em R:

https://r4ds.had.co.nz/exploratory-data-analysis.html

Métricas de Clusterização – Definição e Interpretação

Abaixo você encontra as principais métricas da análise de cluster e como deve ser a interpretação.

Métrica: Número de observações

O número de observações em cada cluster na partição final.

Interpretação:

Examine o número de observações em cada cluster ao interpretar as medidas de variabilidade, como a distância média e a soma dos quadrados dentro do cluster. A variabilidade de um cluster pode ser afetada por ter um número menor ou maior de observações. Por exemplo, a soma dos quadrados dentro do cluster se torna maior à medida que mais observações são adicionadas.

Examine os clusters que possuem significativamente menos observações que outros clusters. Clusters que têm poucas observações podem conter discrepâncias ou observações incomuns com características únicas.

Métrica: Soma de quadrados dentro do cluster (Within cluster sum of squares)

A soma dos desvios ao quadrado de cada observação e do centróide do cluster.

Interpretação:

A soma dos quadrados dentro do cluster é uma medida da variabilidade das observações dentro de cada cluster. Em geral, um cluster que possui uma pequena soma de quadrados é mais compacto do que um cluster que possui uma grande soma de quadrados. Clusters com valores mais altos exibem maior variabilidade das observações dentro do cluster.

No entanto, semelhante à soma dos quadrados e dos quadrados médios na ANOVA, a soma dos quadrados dentro do cluster é influenciada pelo número de observações. À medida que o número de observações aumenta, a soma dos quadrados se torna maior. Portanto, a soma dos quadrados dentro do cluster geralmente não é diretamente comparável entre os clusters com diferentes números de observações. Para comparar a variabilidade dentro do cluster de diferentes clusters, use a distância média do centróide.

Métrica: Distância média do centróide

A média das distâncias das observações ao centróide de cada cluster.

Interpretação:

A distância média das observações ao centróide do cluster é uma medida da variabilidade das observações dentro de cada cluster. Em geral, um cluster que possui uma distância média menor é mais compacto do que um cluster que tem uma distância média maior. Clusters com valores mais altos exibem maior variabilidade das observações dentro do cluster.

Métrica: Distância máxima do centróide

O máximo das distâncias das observações ao centróide de cada cluster.

Interpretação:

A distância máxima das observações ao centróide do cluster é uma medida da variabilidade das observações dentro de cada cluster. Um valor máximo mais alto, especialmente em relação à distância média, indica uma observação no cluster que fica mais distante do centróide do cluster.

Métrica: Centróide do cluster

O meio de um cluster. Um centróide é um vetor que contém um número para cada variável, e que cada número é a média de uma variável para as observações nesse cluster. O centróide pode ser considerado a média multidimensional do cluster.

Interpretação:

Use o centróide do cluster como uma medida geral da localização do cluster e para ajudar a interpretar cada cluster. Cada centróide pode ser visto como representando a “observação média” dentro de um cluster em todas as variáveis da análise.

Métrica: Distâncias entre centróides de cluster

As distâncias entre os centróides do cluster medem a distância entre os centróides dos clusters na partição final.

Interpretação:

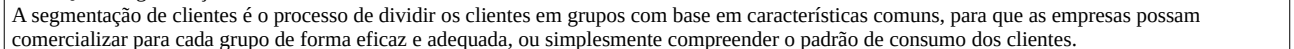
Embora os valores da distância não sejam muito informativos, você pode comparar as distâncias para ver quão diferentes os clusters são. Uma distância maior geralmente indica uma diferença maior entre os clusters.

Referências: Cluster Analysis https://www-users.cs.umn.edu/~kumar001/dmbook/ch8.pdf

Projeto – Segmentação de Clientes de Food Delivery

```
# <font color='blue'>Data Science Academy</font>
# <font color='blue'>Business Analytics</font>
# <font color='blue'>Capítulo 4 - Marketing Analytics</font>
# <font color='blue'>Projeto 1 - Segmentação de Clientes de Food Delivery</font>
# ->
# Versão da Linguagem Python
from platform import python_version
print('Versão da Linguagem Python Usada Neste Jupyter Notebook:', python_version())

![title](imagens/projeto1.png)
## Marketing Analytics
Marketing Analytics compreende os processos e tecnologias que permitem aos profissionais de Marketing avaliar o sucesso de suas iniciativas. Isso é feito medindo o desempenho das campanhas de Marketing, coletando os dados e analisando os resultados. Marketing Analytics utiliza métricas importantes de negócios, como ROI (Retorno Sobre o Investimento), Atribuição de Marketing e Eficácia Geral do Marketing. Em outras palavras, o Marketing Analytics mostra se os programas de Marketing estão sendo efetivos ou não.
Marketing Analytics reúne dados de todos os canais de marketing e os consolida em uma visão de marketing comum. A partir dessa visão comum, você pode extrair resultados analíticos que podem fornecer assistência inestimável para impulsionar os esforços de marketing.
```

Por Que Marketing Analytics é Importante?
Leia o manual em pdf no próximo item de aprendizagem!
O Que Você Pode Fazer com Marketing Analytics?
Com Marketing Analytics, você pode responder a perguntas como estas:
- Como estão as nossas iniciativas de marketing hoje? Que tal a longo prazo? O que podemos fazer para melhorá-las?
- Como nossas atividades de marketing se comparam às de nossos concorrentes? Onde eles estão gastando seu tempo e dinheiro? Eles estão usando canais que não estamos usando?
- O que devemos fazer em seguida? Nossos recursos de marketing estão alocados corretamente? Estamos dedicando tempo e dinheiro aos canais certos? Como devemos priorizar nossos investimentos para o próximo ano?
- Qual o perfil dos nossos clientes? Eles são da mesma área regional? Tem os mesmos gostos e preferências?
- Consigo segmentar meus clientes por similaridade? Tenho como saber os gastos por grupo?
- E muitas outras...
O Que é Segmentação de Clientes?
A segmentação de clientes é o processo de dividir os clientes em grupos com base em características comuns, para que as empresas possam comercializar para cada grupo de forma eficaz e adequada, ou simplesmente compreender o padrão de consumo dos clientes.

Marketing B2B x Marketing B2C
No Marketing Business-to-Business (B2B), uma empresa pode segmentar clientes de acordo com uma ampla variedade de fatores, incluindo:
- Indústria
- Número de empregados
- Produtos comprados anteriormente na empresa
- Localização
No Marketing Business-to-Consumer (B2C), as empresas geralmente segmentam os clientes de acordo com dados demográficos e padrões de consumo, tal como:
- Idade
- Gênero
- Estado civil
- Localização (urbana, suburbana, rural)
- Estágio da vida (sem filhos, aposentado, etc.)
- Produtos comprados
- Valor gasto
- Horário de consumo
Por Que Segmentar Clientes?
Leia o manual em pdf no próximo item de aprendizagem!
Como Segmentar Clientes?
A segmentação de clientes exige que uma empresa colete informações específicas - dados - sobre clientes e analise-as para identificar padrões que podem ser usados para criar segmentos.
Parte disso pode ser obtida a partir de informações de compra - cargo, geografia, produtos adquiridos, por exemplo. Algumas delas podem ser obtidas da forma como o cliente entrou no seu sistema. Um profissional de marketing que trabalha com uma lista de e-mail de inscrição pode segmentar mensagens de marketing de acordo com a oferta de inscrição que atraiu o cliente, por exemplo. Outras informações, no entanto, incluindo dados demográficos do consumidor, como idade e estado civil, precisarão ser adquiridas de outras maneiras.
Os métodos típicos de coleta de informações incluem:
- Entrevistas presenciais ou por telefone
- Pesquisas
- Coleta de informações publicadas sobre categorias de mercado
- Grupos de foco
- Dados de acessos a sistemas ou apps
Usando Segmentos de Clientes
Características comuns nos segmentos de clientes podem orientar como uma empresa comercializa segmentos individuais e quais produtos ou serviços ela promove. Uma pequena empresa que vende guitarras feitas à mão, por exemplo, pode decidir promover produtos com preços mais baixos para guitarristas mais jovens e guitarras premium com preços mais altos para músicos mais velhos, com base no conhecimento do segmento que lhes diz que os músicos mais jovens têm menos renda disponível do que seus colegas mais velhos.
A segmentação de clientes pode ser praticada por todas as empresas, independentemente do tamanho ou setor, e se vendem on-line ou presencialmente. Começa com a coleta e a análise de dados e termina com a atuação nas informações coletadas de maneira apropriada e eficaz, com a entrega das conclusões.
Iniciando o Desenvolvimento do Projeto
```
# ->
# Para atualizar um pacote, execute o comando abaixo no terminal ou prompt de comando:
# pip install -U nome_pacote
# Para instalar a versão exata de um pacote, execute o comando abaixo no terminal ou prompt de comando:
# pip install nome_pacote==versão_desejada
# Depois de instalar ou atualizar o pacote, reinicie o jupyter notebook.
# Instala o pacote watermark.
# Esse pacote é usado para gravar as versões de outros pacotes usados neste jupyter notebook.
!pip install -q -U watermark
```

Carregando os Pacotes
```
# ->
# Imports
# Manipulação e visualização de dados
import time
import sklearn
import datetime
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib as m
import matplotlib.pyplot as plt
```

```
# Machine Learning
from sklearn.cluster import KMeans
from sklearn import metrics
from sklearn.preprocessing import MinMaxScaler
# Formatação dos gráficos
plt.style.use('fivethirtyeight')
plt.figure(1 , figsize = (15 , 6))
%matplotlib inline

# ->
# Versões dos pacotes usados neste jupyter notebook
%reload_ext watermark
%watermark -a "Data Science Academy" --iversions

# ->
# Para instalar uma versão específica do Scikit-Learn, por exemplo:
!pip install -q -U scikit-learn==0.23.1

### Carregando e Compreendendo os Dados
# ->
# Carrega o dataset
df_food_delivery = pd.read_csv("dados/dataset.csv", encoding = 'utf-8')

# ->
# Shape
df_food_delivery.shape

# ->
# Visualiza os dados
df_food_delivery.head()

## Dicionário de Dados
Está disponível no manual em pdf no próximo item de aprendizagem.
### Análise Exploratória
Vamos explorar os dados por diferentes perspectivas e compreender um pouco mais o relacionamento entre as variáveis.
# ->
# Verifica o total de valores únicos por coluna
df_food_delivery.nunique()

# ->
# Tipos de dados
df_food_delivery.dtypes

# ->
# Resumo das colunas numéricas
df_food_delivery.describe()

Começaremos criando uma tabela que nos fornecerá o número de vezes cada item foi solicitado em cada pedido.
# ->
# Lista para receber o total de pedidos
total_pedidos = []

Loop para criar a tabela pivot totalizando os itens por transação.
# ->
%%time
print("\nIniciando o agrupamento para o cálculo do total de pedidos. Seja paciente e aguarde...")
# Extraímos cada id e cada grupo do 'group by' por id_transacao
for k, group in df_food_delivery.groupby('id_transacao'):
    # Extraímos cada id e cada grupo do group by por horario_pedido
    for m, n in group.groupby('horario_pedido'):
        # Extraímos cada item de cada grupo
        id_transacao = k
        horario_pedido = m
        localidade = n['localidade'].values.tolist()[0]
        bebida = 0
        pizza = 0
        sobremesa = 0
        salada = 0
        n = n.reset_index(drop = True)
        # E então contabilizamos os itens pedidos
        for i in range(len(n)):
            item = n.loc[i, 'nome_item']
            num = n.loc[i, 'quantidade_item']
            if item == 'bebida':
                bebida = bebida + num
            elif item == 'pizza':
                pizza = pizza + num
            elif item == 'sobremesa':
```

```
            sobremesa = sobremesa + num
        elif item == 'salada':
            salada = salada + num
    output = [id_transacao, horario_pedido, localidade, bebida, pizza, sobremesa, salada]
    total_pedidos.append(output)
print("\nAgrupamento concluído!")

# ->
# Convertemos a lista para dataframe
df_item_pedidos = pd.DataFrame(total_pedidos)

# ->
# Ajustamos os nomes das colunas
df_item_pedidos.columns = ['id_transacao', 'horario_pedido', 'localidade', 'bebida', 'pizza', 'sobremesa', 'salada']

# ->
# Shape
df_item_pedidos.shape

# ->
# Verifica o total de valores únicos por coluna
df_item_pedidos.nunique()

# ->
# Visualiza os dados originais
df_food_delivery.head()

# ->
# Visualiza o resultado do pivot
df_item_pedidos.head(10)
```

Observe como uma simples mudança nos dados já oferece uma perspectiva completamente diferente dos dados. Na prática, o que fizemos foi criar uma tabela pivot.
Será que temos uma forma mais simples de criar a tabela pivot? Sim.

```
# ->
# Visualiza os dados originais
df_food_delivery.head()

# ->
# Shape
df_food_delivery.shape

# ->
# Visualiza os dados modificados
df_item_pedidos.head()

# ->
# Shape
df_item_pedidos.shape

# ->
# Tipo do objeto
type(df_food_delivery)

# ->
# Vamos criar uma tabela pivot com id_transacao, nome_item e quantidade_item
df_pivot = df_food_delivery.pivot_table(index = ['id_transacao'], columns = ['nome_item'], values = 'quantidade_item')

# ->
# Substituímos possíveis valores NA gerados no pivot, por 0 e transformamos o índice em coluna
df_pivot = df_pivot.fillna(0).reset_index()

# ->
# Tipo do objeto
type(df_pivot)

# ->
# Tipos de dados nas colunas
df_pivot.dtypes

# ->
# Nomes das colunas
df_pivot.columns

# ->
# Visualiza os dados
df_pivot.head()
```

```
# ->
#
 Valores únicos
df_pivot.nunique()

# ->
# Shape
df_pivot.shape

# ->
# Describe
df_pivot.describe()

# ->
# Não podemos ter valores nulos
df_pivot.isnull().sum()

# ->
# Vamos incluir a coluna localidade e para fazer o merge precisamos de uma coluna em comum, nesse caso, id_transacao
df_pivot2 = df_pivot.merge(df_food_delivery[['id_transacao', 'localidade']])

# ->
# Visualiza os dados
df_pivot2.head()

# ->
# Shape
df_pivot2.nunique()
```

Extraindo Granularidade de Tempo

A coluna de horário do pedido tem detalhes como mês, dia e ano. Em algum momento pode ser interessante fazer a segmentação por mês, por exemplo. Vamos então extrair o mês e colocar em uma coluna separada.

```
# ->
# Visualiza os dados
df_item_pedidos.head(3)

# ->
# Extraímos o mês da coluna horario_pedido e gravamos em uma nova coluna
df_item_pedidos['mes'] = df_item_pedidos['horario_pedido'].apply(lambda x: time.strftime("%m", time.strptime(x,"%Y-%m-%d %H:%M:%S")))

# ->
# Visualiza o resultado
df_item_pedidos.head(10)

# ->
# Vamos incluir a coluna localidade e para fazer o merge precisamos de uma coluna em comum, nesse caso, id_transacao
df_pivot = df_pivot.merge(df_item_pedidos[['id_transacao', 'mes']])

# ->
# Visualiza o resultado
df_pivot.head(10)

# ->
# Visualiza valores únicos
df_pivot.nunique()
```

Ajuste de Índices

Para segmentar os pedidos dos clientes, precisamos de uma coluna de identificação de cada registro. Não podemos usar id_transacao, pois essa coluna representa um dado válido e além disso não é um valor único, logo não pode ser usado como índice.
Vamos então criar uma coluna usando o índice atual, o que acha? Vamos checar o índice:

```
# ->
# Dataset
df_item_pedidos

# ->
# Índice
df_item_pedidos.index

# ->
# Fazemos o reset no índice e gravamos o resultado em outro dataframe
df_item_pedidos_idx = df_item_pedidos.reset_index()

# ->
# Pronto, agora temos uma coluna de ID com valor único para cada registro
df_item_pedidos_idx.head()

# ->
# Dataset
```

```
df_item_pedidos

### Análise Descritiva
### Distplot dos Atributos Usados Para Segmentação
# ->
# Plot
# Tamanho da figura
plt.figure(1 , figsize = (15 , 6))
# Inicializa o contador
n = 0
# Loop pelas colunas
for x in ['pizza' , 'sobremesa' , 'salada', 'bebida', 'localidade']:
    n += 1
    plt.subplot(1 , 5 , n)
    plt.subplots_adjust(hspace = 0.5 , wspace = 0.5)
    sns.distplot(df_item_pedidos[x] , bins = 20)
    plt.title('Distplot de {}'.format(x))
plt.show()

### Gráfico de Total de Pedidos Por Localidade
# ->
# Plot
plt.figure(1 , figsize = (15 , 5))
sns.countplot(y = 'localidade' , data = df_item_pedidos)
plt.show()

### Regplot dos Atributos Usados Para Segmentação
# ->
# Relação Entre os Atributos
# Tamanho da figura
plt.figure(1 , figsize = (15 , 7))
# Inicializa o contador
n = 0
# Loop pelos atributos
for x in ['pizza' , 'sobremesa' , 'salada', 'bebida']:
    for y in ['pizza' , 'sobremesa' , 'salada', 'bebida']:
        n += 1
        plt.subplot(4 , 4 , n)
        plt.subplots_adjust(hspace = 0.5 , wspace = 0.5)
        sns.regplot(x = x , y = y , data = df_item_pedidos)
        plt.ylabel(y)
plt.show()

### Definindo as Variáveis Para Segmentação
Vamos remover id_transacao, horario_pedido, localidade e mes para nossas primeiras atividades de clusterização.
# ->
# Filtrando o dataframe por colunas
df_item_pedidos_idx[['index', 'bebida', 'pizza', 'sobremesa', 'salada']]

# ->
# Vamos gerar um novo dataframe com o slice anterior
df = df_item_pedidos_idx[['index', 'bebida', 'pizza', 'sobremesa', 'salada']]

# ->
# Dataset
df.head()

Perfeito. Podemos avançar.
### Análise de Cluster
Clusterização é um processo de aprendizagem não supervisionada, quando entregamos a um algoritmo de Machine Learning somente os dados de entrada e durante o treinamento, o algoritmo cria um modelo capaz de gerar saídas, nesse caso grupos, ou clusters.
![title](imagens/cluster.png)
### Algoritmo de Clusterização - K-means
https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
O K-Means Clustering é um algoritmo de aprendizado de máquina não supervisionado. Em contraste com os algoritmos tradicionais de aprendizado de máquina supervisionado, o K-Means tenta classificar dados sem antes ter sido treinado com dados rotulados. Depois que o algoritmo é executado e os grupos são definidos, qualquer novo dado pode ser facilmente atribuído ao grupo mais relevante.
![title](imagens/kmeans.png)
K-Means é provavelmente o algoritmo de agrupamento mais conhecido. É fácil entender e implementar! Confira o gráfico abaixo para obter uma ilustração.
![title](imagens/k-means.gif)
- Para começar, primeiro selecionamos um número de classes / grupos que desejamos e inicializamos aleatoriamente seus respectivos pontos centrais (centróides). Para descobrir o número de classes a serem usadas, é bom dar uma olhada rápida nos dados e tentar identificar grupos distintos.
- Cada ponto de dados é classificado calculando a distância entre esse ponto e cada centro de grupo e, em seguida, classificando o ponto no grupo cujo centro está mais próximo.
- Com base nesses pontos classificados, recalculamos o centro do grupo, calculando a média de todos os vetores do grupo.
- Repitimos essas etapas para um número definido de iterações ou até que os centros dos grupos não alterem muito entre as iterações. Você também
```

pode optar por inicializar aleatoriamente os centros do grupo algumas vezes e selecionar a execução que parece ter fornecido os melhores resultados.
O K-Means tem a vantagem de ser muito rápido, pois estamos realmente calculando as distâncias entre pontos e centros de grupos; são poucos cálculos! Portanto, possui uma complexidade linear O(n).
Por outro lado, o K-Means tem algumas desvantagens. Primeiro, você deve selecionar quantos grupos / clusters. Isso nem sempre é trivial e, idealmente, com um algoritmo de agrupamento, queremos que ele os descubra, porque o objetivo é obter algumas informações dos dados.
O K-means também começa com uma escolha aleatória de centros de cluster e, portanto, pode produzir resultados de cluster diferentes em execuções diferentes do algoritmo. Assim, os resultados podem não ser repetíveis e não têm consistência. Outros métodos de cluster são mais consistentes.
K-Medians é outro algoritmo de agrupamento relacionado ao K-Means, exceto que, em vez de recalcular os pontos centrais do grupo usando a média, usamos o vetor de medianas do grupo. Esse método é menos sensível a outliers (por causa do uso da Mediana), mas é muito mais lento para conjuntos de dados maiores, pois a classificação é necessária em cada iteração ao calcular o vetor Mediana.
Outros Algoritmos de Clusterização:
- Mean-Shift Clustering
- Density-Based Spatial Clustering of Applications with Noise (DBSCAN)
- Expectation–Maximization (EM) Clustering using Gaussian Mixture Models (GMM)
- Agglomerative Hierarchical Clustering
Métricas de Clusterização - Definição e Interpretação
Está disponível no manual em pdf no próximo item de aprendizagem.
Segmentação 1
Vamos realizar nossa primeira segmentação usando 2 variáveis.
Segmentação 1 - Definindo o Número de Atributos
Usando 2 Variáveis (Pizza e Sobremesa).

```
# ->
# Usaremos duas variáveis
X1 = df[['pizza' , 'sobremesa']].iloc[: , :].values

# ->
# Lista do WCSS
wcss_X1 = []
```

Muitas vezes, os dados com os quais você trabalha têm várias dimensões, dificultando a visualização. Como consequência, o número ideal de clusters não é muito óbvio. Felizmente, temos uma maneira de determinar isso matematicamente.
Representamos graficamente a relação entre o número de clusters e a soma dos quadrados dentro do cluster (Within Cluster Sum of Squares - WCSS) e, em seguida, selecionamos o número de clusters nos quais a mudança no WCSS começa a se estabilizar (Método Elbow).
Segmentação 1 - Encontrando o Valor Ideal de Clusters
Vamos testar diferentes valores de K (valores de cluster) entre 2 e 10.
Para a inicialização dos clusters, usamos o algoritmo k-means++ que oferece convergência mais rápida para o resultado final.
https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

```
# ->
# Loop para testar os valores de K
for n in range(2, 11):
    modelo = (KMeans(n_clusters = n,
                     init = 'k-means++',
                     n_init = 10,
                     max_iter = 300,
                     tol = 0.0001,
                     random_state = 111,
                     algorithm = 'elkan'))
    modelo.fit(X1)
    wcss_X1.append(modelo.inertia_)

# ->
# Plot
plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(2 , 11) , wcss_X1 , 'o')
plt.plot(np.arange(2 , 11) , wcss_X1 , '-' , alpha = 0.5)
plt.xlabel('Número de Clusters') , plt.ylabel('WCSS')
plt.show()
```

Escolhemos o valor ideal de clusters e criamos o modelo final para a Segmentação 1. Observe no gráfico acima que não há certo ou errado. Poderíamos trabalhar com qualquer valor entre 2 e 10 (não faz sentido criar apenas 1 cluster).
O gráfico acima é chamado de Curva de Elbow e normalmente usamos o valor com o menor WCSS. Mas isso deve ser alinhado com as necessidade de negócio. Para esse exemplo, não faria sentido usar 10 clusters. Vamos começar com 2 clusters e avaliar e interpretar os resultados.
Segmentação 1 - Construindo e Treinando o Modelo

```
# ->
# Criação do modelo
modelo_seg1 = KMeans(n_clusters = 2,
                     init = 'k-means++',
                     n_init = 10,
                     max_iter = 300,
                     tol = 0.0001,
                     random_state = 111,
                     algorithm = 'elkan')

# ->
# Treinamento do modelo
modelo_seg1.fit(X1)
```

```
# ->
# Extração dos labels
labels1 = modelo_seg1.labels_
labels1

# ->
# Extração dos centróides
centroids1 = modelo_seg1.cluster_centers_
centroids1
```

Caso queira alterar a combinação de cores dos gráficos, basta alterar a paleta usada. Aqui estão as opções:
https://matplotlib.org/3.2.0/tutorials/colors/colormaps.html
Para o Segmento 1 estamos usando plt.cm.Set2.

Segmentação 1 - Visualização e Interpretação dos Segmentos

```
# ->
# Plot
# Parâmetros do Meshgrid
h = 0.02
x_min, x_max = X1[:, 0].min() - 1, X1[:, 0].max() + 1
y_min, y_max = X1[:, 1].min() - 1, X1[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = modelo_seg1.predict(np.c_[xx.ravel(), yy.ravel()])
plt.figure(1 , figsize = (15, 7) )
plt.clf()
Z = Z.reshape(xx.shape)
# Plot da imagem
plt.imshow(Z,
           interpolation = 'nearest',
           extent = (xx.min(), xx.max(), yy.min(), yy.max()),
           cmap = plt.cm.Set2,
           aspect = 'auto',
           origin = 'lower')
# Plot dos pontos de dados
plt.scatter( x = 'pizza', y = 'sobremesa', data = df, c = labels1, s = 200 )
plt.scatter(x = centroids1[: , 0], y =  centroids1[: , 1], s = 300, c = 'red', alpha = 0.5)
plt.xlabel('Pizza')
plt.ylabel('Sobremesa')
plt.show()
```

Interpretação:
- O ponto vermelho é o centróide de cada cluster (segmento).
- No cluster 1 (área em verde) temos os clientes que pediram 0, 1 ou 2 Pizzas. Em todos os casos houve pedido de Sobremesa.
- No cluster 2 (área em cinza) estão clientes que pediram 2, 3, 4 ou 5 Pizzas. Perceba que à medida que o pedido tem maior número de Pizzas, também aumenta o número de Sobremesas.

Análise:
- Cluster 1 - Clientes que pedem menos Pizzas. Todos pedem sobremesa.
- Cluster 2 - Clientes que pedem mais Pizzas. Todos pedem sobremesa em volume maior.

Como estratégia de Marketing, poderíamos oferecer ao cliente uma sobremesa grátis no caso de comprar mais uma Pizza de maior valor. Com base na Segmentação provavelmente essa estratégia teria sucesso.

Vamos criar mais 4 Segmentações:
- Segmentação 2 - Variáveis Pizza e Salada
- Segmentação 3 - Variáveis Pizza e Localidade
- Segmentação 4 - Variáveis Pizza, Salada e Localidade
- Segmentação 5 - Variáveis Pizza, Salada e Sobremesa

Segmentação 2

Segmentação 2 - Variáveis Pizza e Salada

```
# ->
# Usaremos duas variáveis
X1 = df[['pizza' , 'salada']].iloc[: , :].values
# Lista de valores de Inertia (Inertia e WCSS são a mesma coisa)
inertia = []
# Loop para testar os valores de K
for n in range(2 , 11):
    modelo = (KMeans(n_clusters = n,
                     init = 'k-means++',
                     n_init = 10,
                     max_iter = 300,
                     tol = 0.0001,
                     random_state = 111,
                     algorithm = 'elkan'))
    modelo.fit(X1)
    inertia.append(modelo.inertia_)
# Plot
plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(2 , 11) , inertia , 'o')
```

```
plt.plot(np.arange(2 , 11) , inertia , '-' , alpha = 0.5)
plt.xlabel('Número de Clusters') , plt.ylabel('Inertia')
plt.show()

Vamos criar o modelo com 3 clusters.
# ->
# Criação do modelo com 3 clusters
modelo_seg2 = (KMeans(n_clusters = 3,
                      init = 'k-means++',
                      n_init = 10 ,
                      max_iter = 300,
                      tol = 0.0001,
                      random_state = 111  ,
                      algorithm = 'elkan') )
# Treinamento do modelo
modelo_seg2.fit(X1)
# Labels
labels2 = modelo_seg2.labels_
# Centróides
centroids2 = modelo_seg2.cluster_centers_

# ->
# Plot
# Parâmetros do Meshgrid
h = 0.02
x_min, x_max = X1[:, 0].min() - 1, X1[:, 0].max() + 1
y_min, y_max = X1[:, 1].min() - 1, X1[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = modelo_seg2.predict(np.c_[xx.ravel(), yy.ravel()])
plt.figure(1 , figsize = (15, 7) )
plt.clf()
Z = Z.reshape(xx.shape)
# Plot da imagem
plt.imshow(Z,
           interpolation = 'nearest',
           extent = (xx.min(), xx.max(), yy.min(), yy.max()),
           cmap = plt.cm.Dark2,
           aspect = 'auto',
           origin = 'lower')
# Plot dos pontos de dados
plt.scatter( x = 'pizza', y = 'salada', data = df, c = labels2, s = 200 )
plt.scatter(x = centroids2[: , 0], y =  centroids2[: , 1], s = 300, c = 'red', alpha = 0.5)
plt.xlabel('Pizza')
plt.ylabel('Salada')
plt.show()

**Interpretação**:
- O ponto vermelho é o centróide de cada cluster (segmento).
- No cluster 1 (área em cinza) temos os clientes que pediram menos Pizzas e mais Saladas.
- No cluster 2 (área em verde escuro) temos os clientes que pediram poucas Pizzas e poucas Saladas.
- No cluster 3 (área em verde claro) estão clientes que pediram mais Pizzas e menos Saladas.
**Análise**:
Os clusters 1 e 3 são de clientes com comportamentos opostos. A equipe de Marketing poderia concentrar os esforços nos clientes do cluster 2, pois são clientes que compram Pizzas e Saladas e, portanto, tendem a consumir mais itens variados evitando manter os estoques cheios de um único item.
Ou então, concentrar os esforços nos clientes que consomem produtos que geram mais lucro. Teríamos que verificar qual item, Pizza ou Salada, é mais rentável.
### Segmentação 3
Segmentação 3 - Variáveis Pizza e Localidade
# ->
# Filtrando o dataframe por colunas
df_item_pedidos_idx[['index', 'bebida', 'pizza', 'sobremesa', 'salada', 'localidade']]

# ->
# Criando um novo dataframe
df2 = df_item_pedidos_idx[['index', 'bebida', 'pizza', 'sobremesa', 'salada', 'localidade']]

# ->
# Resumo do dataset
df2.describe()

# ->
# Usaremos duas variáveis
X1 = df2[['pizza' , 'localidade']].iloc[: , :].values
# Lista de valores de Inertia (Inertia e WCSS são a mesma coisa)
inertia = []
# Loop para testar os valores de K
for n in range(2 , 11):
```

```
    modelo = (KMeans(n_clusters = n,
                     init = 'k-means++',
                     n_init = 10,
                     max_iter = 300,
                     tol = 0.0001,
                     random_state = 111,
                     algorithm = 'elkan'))
    modelo.fit(X1)
    inertia.append(modelo.inertia_)
# Plot
plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(2 , 11) , inertia , 'o')
plt.plot(np.arange(2 , 11) , inertia , '-' , alpha = 0.5)
plt.xlabel('Número de Clusters') , plt.ylabel('Inertia')
plt.show()
```

Vamos criar o modelo com 4 clusters.

```
# ->
# Criação do modelo com 4 clusters
modelo_seg3 = (KMeans(n_clusters = 4,
                      init = 'k-means++',
                      n_init = 10 ,
                      max_iter = 300,
                      tol = 0.0001,
                      random_state = 111  ,
                      algorithm = 'elkan') )
# Treinamento do modelo
modelo_seg3.fit(X1)
# Labels
labels3 = modelo_seg3.labels_
# Centróides
centroids3 = modelo_seg3.cluster_centers_

# ->
# Plot
# Parâmetros do Meshgrid
h = 0.02
x_min, x_max = X1[:, 0].min() - 1, X1[:, 0].max() + 1
y_min, y_max = X1[:, 1].min() - 1, X1[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = modelo_seg3.predict(np.c_[xx.ravel(), yy.ravel()])
plt.figure(1 , figsize = (15, 7) )
plt.clf()
Z = Z.reshape(xx.shape)
# Plot da imagem
plt.imshow(Z,
           interpolation = 'nearest',
           extent = (xx.min(), xx.max(), yy.min(), yy.max()),
           cmap = plt.cm.Pastel1,
           aspect = 'auto',
           origin = 'lower')
# Plot dos pontos de dados
plt.scatter( x = 'pizza', y = 'localidade', data = df2, c = labels3, s = 200 )
plt.scatter(x = centroids3[: , 0], y =  centroids3[: , 1], s = 300, c = 'red', alpha = 0.5)
plt.xlabel('Pizza')
plt.ylabel('Localidade')
plt.show()
```

Interpretação:
- O ponto vermelho é o centróide de cada cluster (segmento).
- Observe que os clusters da esquerda no gráfico contém os pedidos de todas as Localidades, mas com menor número de Pizzas. Já os clusters da direita no gráfico contém pedidos de todas as Localidades com com maior número de Pizzas.

Análise:
Queremos aumentar as vendas, certo? Então teríamos que investigar mais a fundo os pedidos dos clusters à esquerda do gráfico e compreender em mais detalhes as características desses pedidos e que tipo de oferta podemos fazer.

Segmentação 4

Segmentação 4 - Variáveis Pizza, Salada e Localidade

```
# ->
# Usaremos três variáveis
X1 = df2[['pizza' , 'salada' , 'localidade']].iloc[: , :].values
# Lista de valores de Inertia (Inertia e WCSS são a mesma coisa)
inertia = []
# Loop para testar os valores de K
for n in range(2 , 11):
    modelo = (KMeans(n_clusters = n,
                     init = 'k-means++',
                     n_init = 10,
                     max_iter = 300,
```

```
                tol = 0.0001,
                random_state = 111,
                algorithm = 'elkan'))
    modelo.fit(X1)
    inertia.append(modelo.inertia_)
# Plot
plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(2 , 11) , inertia , 'o')
plt.plot(np.arange(2 , 11) , inertia , '-' , alpha = 0.5)
plt.xlabel('Número de Clusters') , plt.ylabel('Inertia')
plt.show()
```

Vamos criar o modelo com 4 clusters.

```
# ->
# Criação do modelo com 4 clusters
modelo_seg4 = (KMeans(n_clusters = 4,
                init = 'k-means++',
                n_init = 10 ,
                max_iter = 300,
                tol = 0.0001,
                random_state = 111  ,
                algorithm = 'elkan') )
# Treinamento do modelo
modelo_seg4.fit(X1)
# Labels
labels4 = modelo_seg4.labels_
# Centróides
centroids4 = modelo_seg4.cluster_centers_
```

O Meshgrid que criamos até aqui é útil para duas dimensões, mas com 3 dimensões precisamos de um gráfico mais apropriado. Um Scatter3d.

```
# ->
# Instala o Plotly
!pip install -q plotly

# ->
# Pacotes para o gráfico 3D
import plotly as py
import plotly.graph_objs as go
py.offline.init_notebook_mode(connected = True)

# ->
# Versões dos pacotes usados neste jupyter notebook
%reload_ext watermark
%watermark -a "Data Science Academy" --iversions

# ->
# Plot
# Gráfico 3D
grafico = go.Scatter3d(x = df2['pizza'],
                       y = df2['salada'],

 z = df2['localidade'],
 mode = 'markers',
 marker = dict(color = labels4,
 size = 4,
 line = dict(color = labels4, width = 15),
 opacity = 0.7))
# Layout do gráfico
layout = go.Layout(title = 'Clusters',
 scene = dict(xaxis = dict(title  = 'Pizza'),
 yaxis = dict(title  = 'Salada'),
 zaxis = dict(title  = 'Localidade')))
# Plot da figura (gráfico + layout)
fig = go.Figure(data = grafico, layout = layout)
py.offline.iplot(fig)
```

Interpretação:
- Observamos 2 clusters inferiores e 2 superiores.
- Cada ponto de dado representa uma coordenada de 3 dimensões.

Análise:
Aqui o ideal é avaliar o gráfico de forma interativa aproveitando essa propriedade do Plotly.

Segmentação 5

Segmentação 5 - Variáveis Pizza, Salada e Sobremesa

```
# ->
# Usaremos três variáveis
X1 = df2[['pizza' , 'salada' , 'sobremesa']].iloc[: , :].values
# Lista de valores de Inertia (Inertia e WCSS são a mesma coisa)
inertia = []
```

```
# Loop para testar os valores de K
for n in range(2 , 11):
    modelo = (KMeans(n_clusters = n,
                     init = 'k-means++',
                     n_init = 10,
                     max_iter = 300,
                     tol = 0.0001,
                     random_state = 111,
                     algorithm = 'elkan'))
    modelo.fit(X1)
    inertia.append(modelo.inertia_)
# Plot
plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(2 , 11) , inertia , 'o')
plt.plot(np.arange(2 , 11) , inertia , '-' , alpha = 0.5)
plt.xlabel('Número de Clusters') , plt.ylabel('Inertia')
plt.show()
```

Vamos criar o modelo com 2 clusters.

```
# ->
# Criação do modelo com 2 clusters
modelo_seg5 = (KMeans(n_clusters = 2,
                      init = 'k-means++',
                      n_init = 10 ,
                      max_iter = 300,
                      tol = 0.0001,
                      random_state = 111  ,
                      algorithm = 'elkan') )
# Treinamento do modelo
modelo_seg5.fit(X1)
# Labels
labels5 = modelo_seg5.labels_
# Centróides
centroids5 = modelo_seg5.cluster_centers_

# ->
# Plot
# Gráfico 3D
grafico = go.Scatter3d(x = df2['pizza'],
                       y = df2['salada'],
                       z = df2['sobremesa'],
                       mode = 'markers',
                       marker = dict(color = labels5,
                                     size = 4,
                                     line = dict(color = labels5, width = 15),
                                     opacity = 0.7))
# Layout do gráfico
layout = go.Layout(title = 'Clusters',
                   scene = dict(xaxis = dict(title  = 'Pizza'),
                                yaxis = dict(title  = 'Salada'),
                                zaxis = dict(title  = 'Sobremesa')))
# Plot da figura (gráfico + layout)
fig = go.Figure(data = grafico, layout = layout)
py.offline.iplot(fig)
```

Interpretação:
- Observamos a clara separação entre os dados dois 2 clusters.
- Cada ponto de dado representa uma coordenada de 3 dimensões.

Análise:
Aqui o ideal é avaliar o gráfico de forma interativa aproveitando essa propriedade do Plotly.
Exemplo de Relatório final (Considerando a Segmentação 5)

```
# ->
# Shape dos labels
labels5.shape

# ->
# Tipo
type(labels5)

# ->
# Converte o array para dataframe
df_labels = pd.DataFrame(labels5)

# ->
# Visualiza
df_labels.head(5)

# ->
```

```
# Tipo
type(df_labels)

# ->
# Vamos fazer o merge de df2 e os labels (clusters) encontrados pelo modelo
# Lembre-se que usamos somente 3 variáveis para criar a segmentação
df_final = df2.merge(df_labels, left_index = True, right_index = True)

# ->
# Visualiza
df_final

# ->
# Ajusta o nome da coluna
df_final.rename(columns = {0:"cluster"}, inplace = True)

# ->
# Visualiza
df_final

---------
Agora é com você! Faça modificações nos hiperparâmetros do modelo, use diferentes números de clusters, compare os resultados e gere um
relatório final para a área de negócio.
# Fim
```

8.5. Marketing Analytics – Projeto 2 – Teste A/B

O Que São Testes A/B?

Marketing pode ser difícil – você precisa aproveitar toda vantagem que puder para estar na frente do seu concorrente. E é por isto que você precisa fazer Teste A/B – isso permite que você teste diferentes abordagens e veja qual das alternativas trará o melhor resultado.

O Teste A/B, ou teste de divisão, é um método eficaz em Marketing Digital, como técnica de otimização, para tentar gradualmente aumentar as conversões do funil de vendas (contatos ou vendas). Ou ainda melhorar outras fases iniciais como, por exemplo, taxa de cliques ou taxa de rejeição.

Um Teste A/B consiste em testar duas diferentes estratégias em um grupo específico.

Para ilustrar, digamos que você tenha uma página de venda de livros online com um botão para o usuário se cadastrar e receber seus e-mails – esse elemento é chamado de controle (o elemento existente).

Você quer saber se algo pode ser melhorado na página, para atrair mais inscritos. De todas as estratégias de Marketing, a comunicação via e-mail ainda é a mais eficiente.

Então, você cria outra página com um botão de inscrição diferente – você pode mudar o lugar do botão, a cor, o texto call-to-action, o formato, etc. Este novo elemento é chamado variável.

Agora, você exibe as duas versões da página para divisão 50/50 de toda sua audiência. Ou seja, todos os seus usuários visualizarão o botão de cadastro, mas alguns visualizarão a página onde o botão é azul e outros visualizarão a página onde o botão é verde, por exemplo.

Desta forma, você reunirá os dados analíticos das duas versões, a controle e a variável, e poderá usar como fator decisivo. Qual deles pode trazer mais inscritos?

O que obteve mais resultados deve ser o escolhido.

Isso é apenas a ponta do iceberg! Você pode usar Teste A/B em diversas plataformas de marketing, como social media, marketing visual e muito mais.

Nosso objetivo aqui não é criar o Teste A/B (isso é feito no curso de Marketing Digital da Formação Desenvolvedor Web Para Data Science). Nosso objetivo aqui é analisar o resultado de um Teste A/B.

Funcionamento de Testes A/B

Um exemplo clássico de Teste A/B é mudar a cor de um botão. Digamos que um botão seja azul, mas um Gerente de Marketing vem com uma ótima ideia: o que aconteceria se o tornássemos verde?

O botão azul é a versão A, a versão atual, o controle. O botão verde é a versão B, a nova versão, o teste. Queremos saber: o botão verde é tão incrível quanto pensamos? É uma experiência melhor para nossos usuários? Isso leva a melhores resultados para o nosso negócio? Para descobrir, executamos um Teste A/B. Atribuímos aleatoriamente alguns usuários para ver a versão A e alguns para ver a versão B. Em seguida, medimos algumas métricas de resultados principais para os

usuários em cada grupo. Finalmente, usamos a análise estatística para comparar essas métricas entre os dois grupos e determinar se os resultados são significativos.

A significância estatística é uma maneira formal de medir se um resultado é interessante. Sabemos que existe uma variabilidade natural em nossos usuários. Nem todo mundo se comporta exatamente da mesma maneira. Portanto, queremos verificar se a diferença entre A e B pode ser apenas devido ao acaso. Finja que, em vez disso, executamos um Teste A/A (isso mesmo, A/A). Dividimos aleatoriamente os usuários em dois grupos, mas todos recebem o botão azul. Há uma gama de diferenças (perto de zero) que poderíamos esperar ver. Quando os resultados do Teste A/B são estatisticamente significativos, isso significa que seria muito incomum ver em um teste A/A. Nesse caso, concluiríamos que o botão verde fez a diferença.

Para fazer compras em um portal online, os usuários precisam criar uma conta. Exigir que nossos usuários façam login é ótimo para Testes A/B e para análises em geral. Isso significa que podemos unir todas as ações de um usuário por meio de seu ID de conta, mesmo se ele trocar de navegador ou dispositivo. Isso torna mais fácil medir comportamentos de longo prazo, muito além de uma única sessão. E, como podemos medi-los, podemos fazer um teste A/B para eles.

Um dos resultados comuns que medimos em qualquer negócio é o processo de compra. Um resultado de curto prazo seria: quanto esse usuário gastou na sessão ao ver o botão azul ou verde? Um resultado de longo prazo seria: quanto o usuário gastou durante o Teste A/B? Sempre que um usuário vê o controle ou a experiência de teste, dizemos que eles foram expostos. Um usuário pode ser exposto repetidamente no decorrer de um teste. Acumulamos métricas de resultado desde a primeira exposição até o final do teste. Ao medir os resultados cumulativos, podemos entender melhor o impacto de longo prazo e não nos distrair com os efeitos da novidade.

Análise de Testes A/B

Elevação (Lift)

Normalmente, a análise de Teste A/B mede a diferença entre a versão B e a versão A. Para uma métrica de resultado x, a diferença entre o teste e o controle é xB - xA. Essa diferença, especialmente para resultados cumulativos, pode aumentar com o tempo. Considere o exemplo de gasto por usuário exposto. Conforme o Teste A/B continua, os dois grupos continuam comprando e acumulando mais gastos. A versão B é um sucesso se os gastos do grupo de teste aumentarem mais rápido do que os de controle.

Em vez da diferença, medimos a elevação (Lift) de B sobre A. A elevação dimensiona a diferença pelo valor da linha de base. Para uma métrica de resultado x, a elevação do teste sobre o controle é:

$$(xB - xA) / xA * 100\%$$

Descobrimos que o aumento das métricas cumulativas tende a ser estável ao longo do tempo. Isso é o que chamamos de Lift.

Análise de Potência

Antes de iniciar um Teste A/B, é bom fazer duas perguntas:

1- Que porcentagem de usuários deve fazer o teste versus o controle?
2- Quanto tempo o teste precisa ser executado?

A maneira estatística formal de responder a essas perguntas é uma análise de potência. Primeiro, precisamos saber qual é a menor diferença (ou aumento) que seria significativo para o negócio. Isso é chamado de tamanho do efeito. Em segundo lugar, precisamos saber o quanto a métrica de resultado normalmente flutua. A análise de potência calcula o tamanho da amostra, o número de usuários necessários para detectar esse tamanho de efeito com significância estatística.

Existem dois componentes para usar o tamanho da amostra. A divisão é a fração de usuários em teste versus controle e isso afeta o tamanho da amostra necessário. O tempo para o teste é o tempo total que levará para expor tantos usuários. Como os usuários podem voltar e serem expostos novamente, o número cumulativo exposto aumentará mais lentamente com o passar do tempo. Matematicamente, quanto mais desequilibrada a divisão (quanto mais distante de 50-50 em qualquer direção), mais longo será o teste. Da mesma forma, quanto menor o tamanho do efeito, mais longo será o teste.

Muitas vezes, a análise de potência não conta toda a história. Por exemplo, se em um portal online temos um forte ciclo semanal - as pessoas compram de maneira diferente nos fins de semana dos dias de semana.

Sempre recomendamos a execução de Testes A/B por pelo menos uma semana e, de preferência, em múltiplos de sete dias. Obviamente, se os resultados parecerem drasticamente negativos após o primeiro ou dois dias, é bom desligar o teste mais cedo.

O equilíbrio da divisão afeta a duração da execução do teste, mas também consideramos o nível de risco. Se tivermos uma grande quantidade de produtos à venda, podemos começar com 90% de controle, 10% de teste. Por outro lado, se quisermos ter certeza de que um recurso importante no site continua fornecendo sustentação, podemos manter um controle de 5% e teste de 95%. Mas, se tivermos um teste de baixo risco, como uma pequena mudança na interface do site, uma divisão em 50% do controle, o teste de 50% significará um tempo de teste mais curto.

Distribuição de Bernoulli e o Teorema Central do Limite

Na aula anterior, podemos ver que o grupo de teste converteu mais usuários do que o grupo de controle. Também podemos ver que o pico dos resultados do grupo de teste é inferior ao do grupo de controle.

Mas como interpretamos a diferença no pico da probabilidade?

Devemos nos concentrar, em vez disso, na taxa de conversão para que tenhamos uma comparação de termos equivalentes. Para calcular isso, precisamos padronizar os dados e comparar a probabilidade de sucesso, p, para cada grupo.

Primeiro, considere a distribuição de Bernoulli para o grupo de controle.

$$X \sim Bernoulli(p)$$

onde p é a probabilidade de conversão do grupo de controle. De acordo com as propriedades da distribuição de Bernoulli, a média e a variância são as seguintes:

$$E(X) = p$$

$$Var(X) = p(1-p)$$

De acordo com o Teorema Central do Limite, ao calcular muitas médias amostrais podemos aproximar a média verdadeira da população, μ, da qual os dados para o grupo de controle foram obtidos. A distribuição das médias da amostra, p, será normalmente distribuída em torno da média verdadeira com um desvio padrão igual ao erro padrão da média.

A equação para isso é dada como:

$$\sigma_{\bar{x}} = \frac{s}{\sqrt{n}} = \frac{\sqrt{p(1-p)}}{\sqrt{n}}$$

Portanto, podemos representar ambos os grupos como uma distribuição normal com as seguintes propriedades:

$$\hat{p} \sim Normal\left(\mu = p, \sigma = \frac{\sqrt{p(1-p)}}{\sqrt{n}}\right)$$

O mesmo pode ser feito para o grupo de teste. Portanto, teremos duas distribuições normais para p_A e p_B.

E com as distribuições normais, nosso trabalho de comparação ficará mais fácil.

Variância da Soma

Uma propriedade básica da variância é que a variância da soma de duas variáveis independentes aleatórias é a soma das variâncias.

$$Var(X+Y) = Var(X) + Var(Y)$$

$$Var(X-Y) = Var(X) + Var(Y)$$

Isso significa que a hipótese nula e alternativa terão a mesma variância que será a soma das variâncias para o grupo de controle e o grupo de teste.

$$Var(\hat{d}) = Var(\hat{p}_B - \hat{p}_A) = Var(\hat{p}_A) + Var(\hat{p}_B) = \frac{p_A(1-p_A)}{n_A} + \frac{p_B(1-p_B)}{n_B}$$

O desvio padrão pode então ser calculado como:

$$\sigma = \sqrt{Var(\hat{d})} = \sqrt{\frac{p_A(1-p_A)}{n_A} + \frac{p_B(1-p_B)}{n_B}}$$

Se colocarmos esta equação em termos de desvio padrão para a distribuição de Bernoulli, temos:

$$\sigma = \sqrt{Var(\hat{d})} = \sqrt{\frac{s_A^2}{n_A} + \frac{s_B^2}{n_B}}$$

e obtemos a aproximação de Satterthwaite para o erro padrão agrupado. Se calcularmos a probabilidade combinada e usarmos a probabilidade combinada para calcular o desvio padrão para ambos os grupos, obtemos:

$$\sigma = \sqrt{Var(\hat{d})} = \sqrt{\frac{s_p^2}{n_A} + \frac{s_p^2}{n_B}} = \sqrt{s_p^2\left(\frac{1}{n_A} + \frac{1}{n_B}\right)} = \sqrt{\hat{p}_p(1-\hat{p}_p)\left(\frac{1}{n_A} + \frac{1}{n_B}\right)}$$

Onde:

$$\hat{p}_p = \frac{p_A N_A + p_B N_B}{N_A + N_B}$$

Ambas as equações para o erro padrão combinado fornecerão resultados muito semelhantes.

Com isso, agora temos informações suficientes para construir as distribuições para a hipótese nula e a hipótese alternativa.

Projeto 2 – Teste A/B

```
# <font color='blue'>Data Science Academy</font>
# <font color='blue'>Business Analytics</font>
# <font color='blue'>Capítulo 5 - Marketing Analytics</font>
# <font color='blue'>Projeto 2 - Teste A/B</font>
## <font color='blue'>Páginas com Avaliações de Usuários Aumentam as Vendas de Produtos Online?</font>
# ->
# Versão da Linguagem Python
from platform import python_version
print('Versão da Linguagem Python Usada Neste Jupyter Notebook:', python_version())

![title](imagens/projeto2.png)
## Marketing Analytics
Marketing Analytics compreende os processos e tecnologias que permitem aos profissionais de Marketing avaliar o sucesso de suas iniciativas.
Isso é feito medindo o desempenho das campanhas de Marketing, coletando os dados e analisando os resultados. Marketing Analytics utiliza
métricas importantes de negócios, como ROI (Retorno Sobre o Investimento), Atribuição de Marketing e Eficácia Geral do Marketing. Em outras
palavras, o Marketing Analytics mostra se os programas de Marketing estão sendo efetivos ou não.
Marketing Analytics reúne dados de todos os canais de marketing e os consolida em uma visão de marketing comum. A partir dessa visão comum,
você pode extrair resultados analíticos que podem fornecer assistência inestimável para impulsionar os esforços de marketing.
## O Que São Testes A/B?
Leia o manual em pdf no Capítulo 5 do curso, com a definição completa.
As análises de Testes A/B são comumente realizadas por Analistas e Cientistas de dados.
Para este projeto vamos compreender os resultados de um Teste A/B executado por um site de e-commerce. Sua meta é ajudar a empresa a entender
se as avaliações de usuários (reviews) fazem ou não diferença para o usuário efetuar uma compra no portal.
Os dados são fictícios, mas representam valores válidos para esse tipo de trabalho.
## Como Analisar Testes A/B?
Em geral, realizamos esses 5 passos para analisar um Teste A/B:
1. Configuramos o experimento.
2. Executamos o teste de hipóteses e registramos a taxa de sucesso de cada grupo.
3. Criamos o Plot da distribuição da diferença entre as duas amostras.
4. Calculamos o poder estatístico.
5. Avaliamos como o tamanho das amostras afeta os Testes A/B.
Vamos ao trabalho.
## Carregando o Conjunto de Dados
```

```
# ->
# Para atualizar um pacote, execute o comando abaixo no terminal ou prompt de comando:
# pip install -U nome_pacote
# Para instalar a versão exata de um pacote, execute o comando abaixo no terminal ou prompt de comando:
# pip install nome_pacote==versão_desejada
# Depois de instalar ou atualizar o pacote, reinicie o jupyter notebook.
# Instala o pacote watermark.
# Esse pacote é usado para gravar as versões de outros pacotes usados neste jupyter notebook.
!pip install -q -U watermark

# ->
# Imports
import datetime
import matplotlib
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import scipy.stats as scs
# Formatação dos gráficos
plt.style.use('fivethirtyeight')
plt.figure(1 , figsize = (15 , 6))
%matplotlib inline

# ->
# Versões dos pacotes usados neste jupyter notebook
%reload_ext watermark
%watermark -a "Data Science Academy" --iversions

## Carregando e Compreendendo os Dados
Variante A: Mostra o número atual de comentários e avaliações de usuários.
Variante B: Não mostra os comentários de usuários no site.
# ->
# Carrega o dataset
# Usaremos dados fictícios, que representam valores que seriam possíves em um teste real
df_vendas = pd.read_csv("dados/dataset.csv")

# ->
# Visualiza
df_vendas.head()

# ->
# Visualiza
df_vendas.tail()

## Análise Exploratória e Cálculo de Probabilidade
# ->
# Shape dos dados
df_vendas.shape

# ->
# Tipos de dados
df_vendas.dtypes

# ->
# Data máxima
df_vendas['data'].max()

# ->
# Data mínima
df_vendas['data'].min()

# ->
# Checamos por valores nulos
df_vendas.isnull().sum()

# ->
# Checamos se temos IDs duplicados
df_vendas.id.value_counts().count()

# ->
# Proporção do resultado de conversão
df_vendas.compra.value_counts()

# ->
# Proporção das variantes mostradas aos usuários
df_vendas.variante.value_counts()

Calculando probabilidades básicas.
```

Variante A é o grupo de controle. Variante B é o grupo de teste ou tratamento.

```
# ->
# Probabilidade de um usuário visualizar a variante A
df_vendas[df_vendas.variante == 'A'].shape[0] / df_vendas.shape[0] * 100

# ->
# Probabilidade de um usuário visualizar qualquer variante
df_vendas.shape[0] / df_vendas.shape[0] * 100

# ->
# Probabilidade de um usuário visualizar a variante B
df_vendas[df_vendas.variante == 'B'].shape[0] / df_vendas.shape[0] * 100

# ->
# Total de compras realizadas (conversões)
df_vendas.compra.sum()

# ->
# Total de compras realizadas quando a variante era A
df_vendas[df_vendas.variante == 'A'].compra.sum()

# ->
# Total de compras realizadas quando a variante era B
df_vendas[df_vendas.variante == 'B'].compra.sum()

# ->
# Probabilidade de conversão independente da variante recebida
df_vendas.compra.mean()

# ->
# Dado que um indivíduo estava no grupo de controle, qual é a probabilidade de conversão?
df_vendas[df_vendas.variante == 'A'].compra.mean()

# ->
# Dado que um indivíduo estava no grupo de tratamento, qual é a probabilidade de conversão?
df_vendas[df_vendas.variante == 'B'].compra.mean()
```

Como vemos, a probabilidade de receber a nova página é de aproximadamente 10% e a probabilidade total de conversão é de 19%. Precisamos checar se temos evidências suficientes para dizer que o grupo de tratamento leva a um aumento das conversões.

Tarefa 1 - Configurando o Experimento

Páginas com Avaliações de Usuários Aumentam as Vendas de Produtos Online?

Variante A: Mostra o número atual de comentários e avaliações de usuários

Variante B: Não mostra os comentários de usuários no site

Observe que, devido ao registro de data e hora associado a cada evento, você pode tecnicamente executar um teste de hipótese continuamente à medida que cada evento é observado.

No entanto, a questão difícil é saber quando parar assim que uma variante for considerada significativamente melhor do que outra ou isso precisa acontecer de forma consistente por um determinado período de tempo? Quanto tempo até você decidir que nenhuma variante é melhor que a outra?

Converse com a área de negócio para definir a melhor abordagem para o teste e apresentaremos algumas dicas durante este trabalho.

Essas questões são as partes mais difíceis associadas aos Testes A/B e a análise em geral.

Por enquanto, considere que você precisa tomar uma decisão apenas com base nos dados fornecidos. Se você quiser assumir que a variante A é melhor, a menos que a nova variante prove ser definitivamente melhor em uma taxa de erro Tipo I de 5%, quais deveriam ser suas hipóteses nula e alternativa?

Você pode definir suas hipóteses em termos de palavras ou em notação como p_{A} e p_{B}, que são as probabilidades de conversão para as variantes nova e antiga.

- H0: PB - PA = 0
- H1: PB - PA > 0

H0 nos diz que a diferença de probabilidade dos dois grupos é igual a zero.

H1 nos diz que a diferença de probabilidade dos dois grupos é maior do que zero.

Pré-Processamento dos Dados

Faremos alguns cálculos pesados e para simplificar o processo, iremos filtrar os dados e usar apenas um dos meses. Fique à vontade para usar dados de períodos maiores.

```
# ->
# Função para extrair ano e mês da data
def extrai_data(x):
    return x[:7]

# ->
# Extrai ano e mês da coluna de data
df_vendas['ano_mes'] = df_vendas['data'].apply(extrai_data)

# ->
# Visualiza
df_vendas.head()

# ->
# Vamos trabalhar apenas com os dados de Janeiro/2020 para simplificar o processo didaticamente
df_vendas_2020 = df_vendas[df_vendas['ano_mes'] == '2020-01']
```

```
# ->
# Visualiza
df_vendas_2020.head()

# ->
# shape
df_vendas_2020.shape

### Criação do Baseline
Vamos criar um baseline (linha base) da taxa de conversão antes de executar o teste de hipótese. Assim, saberemos a taxa de conversão base e o aumento desejado em compras que gostaríamos de testar.
* A será o grupo de controle
* B será o grupo de teste
# ->
# Geramos um dataframe
df_ab_data = df_vendas_2020[['variante', 'compra']]
df_ab_data.head()

# ->
# Shape
df_ab_data.shape

# ->
# Altera o nome das colunas
df_ab_data.columns = ['grupo', 'conversao']

# ->
# Visualiza
df_ab_data.head()

# ->
# Tabela pivot para o sumário dos dados
df_ab_sumario = df_ab_data.pivot_table(values = 'conversao', index = 'grupo', aggfunc = np.sum)

# ->
# Visualiza os dados
df_ab_sumario.head()

# ->
# Sumário com total
df_ab_sumario['total'] = df_ab_data.pivot_table(values = 'conversao', index = 'grupo', aggfunc = lambda x: len(x))

# ->
# Sumário com taxa
df_ab_sumario['taxa'] = df_ab_data.pivot_table(values = 'conversao', index = 'grupo')

# ->
# Visualiza os dados
df_ab_sumario.head()

# ->
# Obtemos os valores da variante A
conversao_A = df_ab_sumario['conversao'][0]
total_A = df_ab_sumario['total'][0]
taxa_A = df_ab_sumario['taxa'][0]

# ->
# Imprime os valores de A
print(conversao_A)
print(total_A)
print(taxa_A)

# ->
# Obtemos os valores da variante B
conversao_B = df_ab_sumario['conversao'][1]
total_B = df_ab_sumario['total'][1]
taxa_B = df_ab_sumario['taxa'][1]

# ->
# Imprime os valores de B
print(conversao_B)
print(total_B)
print(taxa_B)

Taxa de conversão da linha de base (Baseline conversion rate).
Igual a $p$ no contexto de uma distribuição binomial e $p$ é a probabilidade de sucesso.
# ->
# Taxa de conversão da linha de base.
```

```
conversao_base = taxa_A
conversao_base

Efeito mínimo detectável (Minimum Detectable Effect).
Às vezes referido como nível de significância prática.
# ->
# Efeito mínimo detectável
efeito_minimo = taxa_B - taxa_A
efeito_minimo

## Tarefa 2 - Execução do Teste de Hipóteses
Executamos o teste de hipóteses e registramos a taxa de sucesso de cada grupo.
Poder estatístico ou sensibilidade.
Igual a 1 - $\beta$.
Normalmente 80% é usado para a maioria das
 análises. É a probabilidade de rejeitar a hipótese nula quando a hipótese nula é de fato falsa.
Parâmetros que usaremos para executar o teste:
1- Alfa (Nível de significância) $\alpha$: normalmente 5%; probabilidade de rejeitar a hipótese nula quando a hipótese nula for verdadeira
2- Beta $\beta$: probabilidade de aceitar a hipótese nula quando a hipótese nula é realmente falsa.
# ->
# Parâmetros que usaremos para executar o teste
alfa = 0.05
beta = 0.2

# ->
# Tamanho da amostra
n = 50000

Podemos supor que a distribuição de nosso grupo de controle é binomial porque os dados são uma série de tentativas de Bernoulli, em que cada tentativa tem apenas dois resultados possíveis (semelhante a um cara ou coroa).
 Para o teste usaremos a função binom() do SciPy:
https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.binom.html
# ->
# Teste Binomial (usando padrão de 5% para o nível de significância)
teste_binom = scs.binom(n, p = conversao_base)

# ->
help(teste_binom)

# ->
# Teste Binomial com o efeito mínimo (no nosso exemplo 0.04 para o nível de significância)
teste_binom_mde = scs.binom(n, p = conversao_base + efeito_minimo)

Agora visualizamos a probability mass function (pmf).
# ->
# Plot
# Área de plotagem
fig, ax = plt.subplots(figsize = (10, 10))
# Definimos diversos valores para x
x = np.linspace(0,int(n), int(n) + 1)
# Plotamos os resultados com a pmf e alfa de 0.5
ax.bar(x, teste_binom.pmf(x), alpha = 0.5)
ax.bar(x, teste_binom_mde.pmf(x), alpha = 0.5)

## Tarefa 3 - Plot da Distribuição
Criamos o Plot da distribuição da diferença entre as duas amostras e comparamos os resultados.
Podemos comparar os dois grupos traçando a distribuição do grupo de controle e calculando a probabilidade de obter o resultado de nosso grupo de teste.
# ->
# Plot da distribuição do grupo A (controle)
# Área de plotagem
fig, ax = plt.subplots(figsize = (14,6))
# Teste de A
x = np.linspace(conversao_A - 49, conversao_A + 50, 100)
y = scs.binom(total_A, taxa_A).pmf(x)
# Cria a barra vertical
ax.bar(x, y, alpha = 0.5)
ax.axvline(x = taxa_B * total_A, c = 'magenta', alpha = 0.75, linestyle = '--')
# Labels
plt.xlabel('Conversão')
plt.ylabel('Probabilidade')

# ->
# Plot da distribuição dos 2 grupos
# Área de plotagem
fig, ax = plt.subplots(figsize = (14,6))
# Gráfico de A
xA = np.linspace(conversao_A - 49, conversao_A + 50, 100)
```

```
yA = scs.binom(total_A, taxa_A).pmf(xA)
ax.bar(xA, yA, alpha = 0.5)
# Gráfico de B
xB = np.linspace(conversao_B - 49, conversao_B + 50, 100)
yB = scs.binom(total_B, taxa_B).pmf(xB)
ax.bar(xB, yB, alpha = 0.5)
# Labels
plt.xlabel('Conversão')
plt.ylabel('Probabilidade')
```

Podemos ver que o grupo de teste converteu mais usuários do que o grupo de controle. Também podemos ver que o pico dos resultados do grupo de teste é inferior ao do grupo de controle.
Mas como interpretamos a diferença no pico da probabilidade?
Devemos nos concentrar, em vez disso, na taxa de conversão para que tenhamos uma comparação de termos equivalentes. Para calcular isso, precisamos padronizar os dados e comparar a probabilidade de sucesso, p, para cada grupo.
Antes de continuar, leia o manual em pdf **Distribuição de Bernoulli e o Teorema Central do Limite** no Capítulo 5.

```
# ->
# Ajusta o nome das variáveis
# Probabilidades (taxas de conversão)
p_A = taxa_A
p_B = taxa_B
# Número de conversões
N_A = 3821
N_B = 5000

# ->
# Erro padrão (standard error) para a média de ambos os grupos
SE_A = np.sqrt(p_A * (1 - p_A)) / np.sqrt(total_A)
SE_B = np.sqrt(p_B * (1 - p_B)) / np.sqrt(total_B)

# ->
# Print
print(SE_A)
print(SE_B)

# ->
# Plot das distribuições das hipóteses nula e alternativa
# Área de plotagem
fig, ax = plt.subplots(figsize = (14,6))
# Dados para a variável aleatória
x = np.linspace(0, p_B - p_A, 100)
# Distribuição de A
yA = scs.norm(p_A, SE_A).pdf(x)
ax.plot(xA, yA)
ax.axvline(x = p_A, c = 'blue', alpha = 0.5, linestyle = '--')
# Distribuição de B
yB = scs.norm(p_B, SE_B).pdf(x)
ax.plot(xB, yB)
ax.axvline(x = p_B, c = 'red', alpha = 0.5, linestyle = '--')
# Labels
plt.xlabel('Proporção de Conversão')
plt.ylabel('PDF - Probability Density Function')
```

As linhas contínuas representam a taxa de conversão média para cada grupo. A distância entre a linha azul e a linha vermelha é igual à diferença média entre o grupo de controle e teste.
Variância da Soma
Lembre-se de que a hipótese nula afirma que a diferença de probabilidade entre os dois grupos é zero. Portanto, a média para essa distribuição normal será zero. A outra propriedade de que precisaremos para a distribuição normal é o desvio padrão ou a variância.
Observação: a variância é o desvio padrão ao quadrado. A variância da diferença dependerá das variâncias da probabilidade para ambos os grupos.
Leia o manual em pdf **Variância da Soma** para compreender uma importante propriedade da variância.
Verificando a Hipótese Nula e a Hipótese Alternativa
Vamos começar relembrando a definição da hipótese nula e da hipótese alternativa.
A hipótese nula é a posição de que a mudança no design feito para o grupo de teste resultaria em nenhuma mudança na taxa de conversão.
A hipótese alternativa é a posição oposta de que a mudança no design do grupo de teste resultaria em uma melhoria (ou redução) na taxa de conversão.
A hipótese nula será uma distribuição normal com uma média de zero e um desvio padrão igual ao erro padrão agrupado.
A hipótese alternativa tem o mesmo desvio padrão que a hipótese nula, mas a média estará localizada na diferença na taxa de conversão, d_hat. Isso faz sentido porque podemos calcular a diferença nas taxas de conversão diretamente dos dados, mas a distribuição normal representa possíveis valores que nosso experimento poderia ter nos dado.
Fórmula para o cálculo de z:
$$ z = \frac{(\bar{x_1}-\bar{x_2})-D_0}{\sqrt{\sigma_1^{2}/n_1+\sigma_{2}^{2}/n_2}} $$
$$ z = \frac{(\hat{p_1}-\hat{p_2})-0}{\sqrt{\hat{p}\hat{q}(\frac{1}{n_1}+\frac{1}{n_2})}}$$

```
# ->
# Calculando a probabilidade agrupada
prob_agrupada = (p_A * N_A + p_B * N_B) / (N_A + N_B)

# ->
# Calculando z
```

```
z = (p_B - p_A) / (prob_agrupada * (1 - prob_agrupada) * (1 / N_A + 1 / N_B))**0.5
z

# ->
# Verificamos se z é maior que 1.64 (nível de significância de 0.05)
z > 1.64

### Plot da Distribuição de Probabilidade
Criaremos uma série de funções auxiliares para o plot das distribuições de probabilidade. Leia atentamente todos os comentários.
# ->
# Função que retorna a probabilidade agrupada para 2 amostras
def prob_agrupada_func(N_A, N_B, X_A, X_B):
    return (X_A + X_B) / (N_A + N_B)

# ->
# Função que retorna o erro padrão agrupado para 2 amostras
def erro_padrao_agrupado_func(N_A, N_B, X_A, X_B):
    p_hat = prob_agrupada_func(N_A, N_B, X_A, X_B)
    SE = np.sqrt(p_hat * (1 - p_hat) * (1 / N_A + 1 / N_B))
    return SE

# ->
# Retorna o valor z para um determinado nível de significância
def z_val(sig_level = 0.05, two_tailed = True):
    # Gera distribuição para o valor z
    z_dist = scs.norm()
    # Verifica se devemos checar as duas caudas
    if two_tailed:
        sig_level = sig_level/2
        area = 1 - sig_level
    else:
        area = 1 - sig_level
    # Valor de z
    z = z_dist.ppf(area)
    return z

# ->
# Calcula o intervalo de confiança
def confidence_interval(sample_mean = 0, sample_std = 1, sample_size = 1, sig_level = 0.05):
    # Calcula o valor de z
    z = z_val(sig_level)
    # Limites à esquerda e direita
    left = sample_mean - z * sample_std / np.sqrt(sample_size)
    right = sample_mean + z * sample_std / np.sqrt(sample_size)
    return (left, right)

# ->
# Função que calcula o intervalo de confiança de duas caudas
def plot_CI(ax,
            mu,
            s,
            sig_level = 0.05,
            color = 'grey'):
    # Calcula o intervalo de confiança
    left, right = confidence_interval(sample_mean = mu, sample_std = s, sig_level = sig_level)
    # Inclui o intervalo no gráfico
    ax.axvline(left, c = color, linestyle = '--', alpha = 0.5)
    ax.axvline(right, c = color, linestyle = '--', alpha = 0.5)

# ->
# Função para o plot de uma distribuição normal
def plot_norm_dist(ax,
                   mu,
                   std,
                   with_CI = False,
                   sig_level = 0.05,
                   label = None):
    # Gera valores para a variável aleatória x
    x = np.linspace(mu - 12 * std, mu + 12 * std, 1000)
    # Cria a distribuição normal
    y = scs.norm(mu, std).pdf(x)
    # Plot
    ax.plot(x, y, label = label)
    # Se tivermos intervalo de confiança, incluímos no plot
    if with_CI:
        plot_CI(ax, mu, std, sig_level = sig_level)
```

Função para o plot da distribuição de hipótese nula onde, se não houver mudança real, a distribuição das diferenças entre os grupos de teste e

```
controle será normalmente distribuída.
# ->
# Função para o plot da distribuição da H0
def plot_H0(ax, stderr):
    plot_norm_dist(ax, 0, stderr, label = "H0 - Hipótese Nula")
    plot_CI(ax, mu = 0, s = stderr, sig_level = 0.05)

Função para o plot da distribuição de hipótese alternativa onde, se houver uma mudança real, a distribuição das diferenças entre o teste e os grupos
de controle será normalmente distribuída e centralizada em torno de d_hat
# ->
# Função para o plot da distribuição da H1
def plot_H1(ax, stderr, d_hat):
    plot_norm_dist(ax, d_hat, stderr, label = "H1 - Hipótese Alternativa")

# ->
# Função que preenche entre o limite de significância superior e a distribuição para hipótese alternativa
def show_area(ax, d_hat, stderr, sig_level):
    # Intervalo de confiança
    left, right = confidence_interval(sample_mean = 0, sample_std = stderr, sig_level = sig_level)
    # Valores para x
    x = np.linspace(-12 * stderr, 12 * stderr, 1000)
    # H0
    null = ab_dist(stderr, 'controle')
    # H1
    alternative = ab_dist(stderr, d_hat, 'teste')
    # Se o tipo da área for igual a power

  # Preenchemos entre o limite de significância superior e a distribuição para hipótese alternativa
    ax.fill_between(x, 0, alternative.pdf(x), color = 'green', alpha = 0.25, where = (x > right))
    ax.text(-3 * stderr, null.pdf(0), 'power = {0:.3f}'.format(1 - alternative.cdf(right)),
                fontsize = 12, ha = 'right', color = 'k')

# ->
# Função que retorna um objeto de distribuição dependendo do tipo de grupo
def ab_dist(stderr, d_hat = 0, group_type = 'controle'):
    # Verifica o tipo de grupo
    if group_type == 'controle':
        sample_mean = 0
    elif group_type == 'teste':
        sample_mean = d_hat
    # Cria uma distribuição normal que depende da média e do desvio padrão
    dist = scs.norm(sample_mean, stderr)
    return dist

# ->
# Função que retorna o valor p
def p_val(N_A, N_B, p_A, p_B):
    return scs.binom(N_A, p_A).pmf(p_B * N_B)

# ->
# Função para o plot da análise do Teste A/B
def abplot_func(N_A,
                N_B,
                bcr,
                d_hat,
                sig_level = 0.05,
                show_p_value = False,
                show_legend = True):
    # Define a área de plotagem
    fig, ax = plt.subplots(figsize = (14, 8))
    # Define parâmetros para encontrar o erro padrão agrupado
    X_A = bcr * N_A
    X_B = (bcr + d_hat) * N_B
    stderr = erro_padrao_agrupado_func(N_A, N_B, X_A, X_B)
    # Plot da distribuição da hipótese nula e alternativa
    plot_H0(ax, stderr)
    plot_H1(ax, stderr, d_hat)
    # Definir a extensão da área do plot
    ax.set_xlim(-8 * stderr, 8 * stderr)
    # Ajustamos o gráfico e preenchemos a área interior
    show_area(ax, d_hat, stderr, sig_level)
    # Mostramos valores-p com base nas distribuições para os dois grupos
    if show_p_value:
        null = ab_dist(stderr, 'controle')
        p_value = p_val(N_A, N_B, bcr, bcr + d_hat)
        ax.text(3 * stderr, null.pdf(0), 'Valor-p = {0:.4f}'.format(p_value), fontsize = 14, ha = 'left')
    # Mostra a legenda
    if show_legend:
```

```
        plt.legend()
    plt.xlabel('d')
    plt.ylabel('PDF')
    plt.show()
```

Agora que entendemos a derivação do erro padrão combinado, podemos apenas plotar diretamente as hipóteses nula e alternativa para experimentos futuros. Tudo que precisamos é executar a célula abaixo.

```
# ->
# Definimos os parâmetros e executamos a função
n = N_A + N_B
conversao_base = p_A
d_hat = p_B - p_A
abplot_func(N_A, N_B, conversao_base, d_hat, show_p_value = True)
```

Visualmente, o gráfico para as hipóteses nula e alternativa é muito semelhante aos outros gráficos acima. Felizmente, as duas curvas têm formato idêntico, portanto, podemos apenas comparar a distância entre as médias das duas distribuições. Podemos ver que a curva de hipótese alternativa sugere que o grupo de teste tem uma taxa de conversão maior do que o grupo de controle. Este gráfico também pode ser usado para determinar diretamente o poder estatístico.

Tarefa 4 - Calculando o Poder Estatístico

Poder Estatístico e Nível de Significância

É mais fácil definir o poder estatístico e o nível de significância mostrando primeiro como eles são representados no gráfico da hipótese nula e alternativa. Podemos retornar uma visualização do poder estatístico adicionando o parâmetro show_power = True

```
# ->
# Executa a função
abplot_func(N_A, N_B, conversao_base, d_hat, show_p_value = True)
```

A área sombreada em verde representa o poder estatístico e o valor calculado para o poder também é exibido no gráfico. As linhas tracejadas em cinza no gráfico acima representam o intervalo de confiança (95% para o gráfico acima) para a hipótese nula. O poder estatístico é calculado encontrando a área sob a distribuição de hipótese alternativa e fora do intervalo de confiança da hipótese nula.

Depois de executar nosso experimento, obtemos uma taxa de conversão resultante para ambos os grupos. Se calcularmos a diferença entre as taxas de conversão, acabamos com um resultado, a diferença ou o efeito da mudança de design da página web, não mostrando as avaliações de usuários. Nossa tarefa é determinar de qual população esse resultado veio, a hipótese nula ou a hipótese alternativa.

A área sob a curva da hipótese alternativa é igual a 1. Se o design alternativo (sem avaliações) for realmente melhor, o poder é a probabilidade de aceitarmos a hipótese alternativa e rejeitarmos a hipótese nula e é igual à área sombreada em verde (verdadeiro positivo). A área oposta sob a curva alternativa é a probabilidade de não rejeitarmos a hipótese nula e rejeitarmos a hipótese alternativa (falso negativo). Isso é conhecido como beta no teste A/B ou teste de hipótese e é mostrado abaixo.

Se a hipótese nula for verdadeira e realmente não houver diferença entre os grupos de controle e teste, o nível de significância é a probabilidade de rejeitarmos a hipótese nula e aceitarmos a hipótese alternativa (falso positivo). Um falso positivo é quando concluímos erroneamente que o novo design é melhor. Este valor é baixo porque queremos limitar essa probabilidade.

Muitas vezes, um problema será fornecido com um nível de confiança desejado em vez do nível de significância. Um nível de confiança típico de 95% para um teste A / B corresponde a um nível de significância de 0,05.

Os experimentos são normalmente configurados para uma potência mínima desejada de 80%. Se nosso novo design for realmente melhor, queremos que nosso experimento mostre que há pelo menos 80% de probabilidade de que esse seja o caso. Sabemos que, se aumentarmos o tamanho da amostra para cada grupo, diminuiremos a variância combinada para nossa hipótese nula e alternativa. Isso tornará nossas distribuições muito mais estreitas e pode aumentar o poder estatístico. Vamos dar uma olhada em como o tamanho da amostra afetará diretamente nossos resultados.

Tarefa 5 - Influência do Tamanho da Amostra no Teste A/B

Nossas curvas para a hipótese nula e alternativa tornaram-se mais estreitas e mais da área sob a curva alternativa está localizada à direita da linha tracejada cinza. O resultado para potência é maior que 0,80 e atende a nossa referência de potência estatística. Agora podemos dizer que nossos resultados são estatisticamente significativos.

O próximo problema que devemos encontrar é determinar o tamanho mínimo da amostra de que precisaremos para o experimento. E isso é útil saber porque está diretamente relacionado à rapidez com que podemos concluir os experimentos e fornecer resultados estatisticamente significativos para a área de negócio.

```
# ->
# Executa a função
abplot_func(2000, 2000, conversao_base, d_hat)
```

Temos a taxa de conversão de linha de base e o efeito mínimo detectável, que é a diferença mínima entre o controle e o grupo de teste que a equipe de negócio determinará para valer a pena o investimento de fazer a mudança de design em primeiro lugar.

```
# ->
# Função para incluir o valor z no plot
def zplot(area = 0.95, two_tailed = True, align_right = False):
    # Cria a área de plotagem
    fig = plt.figure(figsize = (12, 6))
    ax = fig.subplots()
    # Cria a distribuição normal
    norm = scs.norm()
    # Cria os pontos de dados para o plot
    x = np.linspace(-5, 5, 1000)
    y = norm.pdf(x)
    ax.plot(x, y)
    # Código para preencher áreas para testes bidirecionais
    if two_tailed:
        left = norm.ppf(0.5 - area / 2)
        right = norm.ppf(0.5 + area / 2)
        ax.vlines(right, 0, norm.pdf(right), color = 'grey', linestyle = '--')
        ax.vlines(left, 0, norm.pdf(left), color = 'grey', linestyle = '--')
        ax.fill_between(x, 0, y, color = 'grey', alpha = 0.25, where = (x > left) & (x < right))
        plt.xlabel('z')
```

```
        plt.ylabel('PDF')
        plt.text(left, norm.pdf(left), "z = {0:.3f}".format(left),
                 fontsize = 12,
                 rotation = 90,
                 va = "bottom",
                 ha = "right")
        plt.text(right, norm.pdf(right), "z = {0:.3f}".format(right),
                 fontsize = 12,
                 rotation = 90,
                 va = "bottom",
                 ha = "left")
    # Para testes de uma cauda
    else:
        # Alinhamento à direita
        if align_right:
            left = norm.ppf(1-area)
            ax.vlines(left, 0, norm.pdf(left), color = 'grey', linestyle = '--')
            ax.fill_between(x, 0, y, color = 'grey', alpha = 0.25, where = x > left)
            plt.text(left, norm.pdf(left), "z = {0:.3f}".format(left),
                     fontsize = 12,
                     rotation = 90,
                     va = "bottom",
                     ha = "right")
        # Alinhamento à esquerda
        else:
            right = norm.ppf(area)
            ax.vlines(right, 0, norm.pdf(right), color = 'grey', linestyle = '--')
            ax.fill_between(x, 0, y, color = 'grey', alpha = 0.25, where = x < right)
            plt.text(right, norm.pdf(right), "z = {0:.3f}".format(right),
                     fontsize = 12,
                     rotation = 90,
                     va = "bottom",
                     ha = "left")
    # Incluímos texto no plot
    plt.text(0, 0.1, "Área Sombreada = {0:.3f}".format(area), fontsize = 12, ha = 'center')
    # Labels
    plt.xlabel('z')
    plt.ylabel('PDF')
    plt.show()

# ->
# Print do valor z
print(z)
print(z_val(sig_level = 0.05, two_tailed = False))
print(z > z_val(sig_level = 0.05, two_tailed = False))

# ->
# Plot de z
zplot(area = 0.95, two_tailed = False, align_right = False)

Aqui está o código Python que executa o mesmo cálculo para o tamanho mínimo da amostra:
$$ n_A = k*n_B $$
$$ n_B = (\frac{p_A(1-p_A)}{k}+p_B(1-p_B)) (\frac{Z_{1-\alpha} + Z_{1-\beta}}{p_A-p_B})^{2}$$
$$ n = \frac{2(\bar{p})(1-\bar{p})(Z_{1-\beta}+Z_{1-\alpha})^2}{(p_B-p_A)^2}$$
# ->
# Calculamos os valores de z alfa e beta
sig_level = 0.05
beta = 0.2
k = N_A/N_B
standard_norm = scs.norm(0, 1)
Z_beta = standard_norm.ppf(1-beta)
Z_alpha
 = standard_norm.ppf(1-sig_level)
print(Z_beta)
print(Z_alpha)

Vamos calcular o tamanho mínimo necessário para a amostra.
# ->
# Função para encontrar o tamanho mínimo da amostra
def calcula_tamanho_min_amostra(N_A,
                                N_B,
                                p_A,
                                p_B,
                                power = 0.8,
                                sig_level = 0.05,
                                two_sided = False):
    k = N_A/N_B
    # Distribuição normal para determinar os valores z
```

```
    standard_norm = scs.norm(0, 1)
    # Encontramos o valor de z para o poder estatístico
    Z_beta = standard_norm.ppf(power)
    # Encontramos z alfa
    if two_sided == True:
        Z_alpha = standard_norm.ppf(1-sig_level/2)
    else:
        Z_alpha = standard_norm.ppf(1-sig_level)
    # Probabilidade agrupada
    pooled_prob = (p_A + p_B) / 2
    # Tamanho mínimo da amostra
    min_N = (2 * pooled_prob * (1 - pooled_prob) * (Z_beta + Z_alpha)**2 / efeito_minimo**2)
    return min_N

# ->
# Calculamos o tamanho mínimo da amostra com two_sided = True
calcula_tamanho_min_amostra(N_A, N_B, p_A, p_B, power = 0.8, sig_level = 0.05, two_sided = True)

# ->
# Calculamos o tamanho mínimo da amostra com two_sided = False
calcula_tamanho_min_amostra(N_A, N_B, p_A, p_B, power = 0.8, sig_level = 0.05, two_sided = False)
```

Agora vamos calcular o tamanho mínimo da amostra considerando nosso baseline.

```
# ->
conversao_base + efeito_minimo

# ->
# Calcula a probabilidade agrupada
probabilidade_agrupada = (conversao_base + conversao_base + efeito_minimo) / 2
probabilidade_agrupada

# ->
# Soma de z alfa e beta
Z_beta + Z_alpha

# ->
# Tamanho mínimo da amostra para o baseline
min_N = (2 * probabilidade_agrupada * (1 - probabilidade_agrupada) * (Z_beta + Z_alpha)**2 / efeito_minimo**2)
min_N
```

Poder estatístico para o baseline.

```
# ->
# Executa a função para 984 amostras
abplot_func(N_A = 984,
            N_B = 984,
            bcr = p_A,
            d_hat = p_B - p_A,
            sig_level = 0.05,
            show_p_value = False,
            show_legend = True)
```

Poder estatístico para o tamanho de amostra calculado.

```
# ->
# Executa a função para 1249 amostras
abplot_func(N_A = 1249,
            N_B = 1249,
            bcr = p_A,
            d_hat = p_B - p_A,
            sig_level = 0.05,
            show_p_value = False,
            show_legend = True)
```

Conclusão e Considerações Finais

O poder calculado para este tamanho de amostra foi de aproximadamente 0,80. Portanto, para afirmar que a mudança na página removendo as avaliações de usuários realmente aumentou a taxa de conversão precisamos de pelos menos 1249 amostras.

Este mini-projeto foi um passo a passo muito longo, mas básico, dos Testes A/B. Depois de desenvolver uma compreensão e familiaridade com o procedimento, você provavelmente será capaz de executar um experimento e ir diretamente para os gráficos das hipóteses nula e alternativa para determinar se seus resultados alcançaram poder suficiente. Ao calcular o tamanho mínimo da amostra de que você precisa antes do experimento, você pode determinar quanto tempo levará para obter os resultados para que a área de negócio possa tomar uma decisão final.

O segredo do Teste A/B não está na execução do teste de hipótese em si, mas em conseguir obter o poder estatístico necessário para afirmar que as mudanças tiveram ou não efeito na taxa de conversão.

Para o nosso exemplo, com 1249 amostras podemos afirmar que sim, remover as avaliações de usuários da página aumenta a taxa de conversão.

```
# Fim
```

8.6. Marketing Analytics – Projeto 3 – Análise de Indicadores de Performance em Redes de Varejo

O Que São KPIs (Indicadores Chave de Performance)?

KPI é uma sigla que vem do inglês para Key Performance Indicator, ou Indicadores-Chave de Performance. Trata-se de uma ferramenta de gestão empregada para analisar os indicadores mais importantes de um negócio ou empresa.

É utilizado para avaliar se determinadas iniciativas, atitudes ou ações estão atendendo ou superando as expectativas do público quanto à promessa trabalhada na campanha de Marketing.

Os indicadores-chave para fazer tais avaliações são muitos e o que mais vemos hoje são técnicas novas surgindo todos os dias, acompanhando o ritmo com que as informações chegam a cada pessoa no planeta.

Cada empresa deverá ter conhecimento de quais métricas deverão passar por análises para fazer a diferença em suas campanhas.

Saber exatamente o que a empresa precisa vai fazer com que não aconteça nenhum tipo de perda de tempo e desperdício em investimentos.

Cada empresa pode definir seus próprios indicadores.

Como Definir os KPIs?

Cada empresa pode definir seus próprios indicadores e a definição depende do que a empresa gostaria de medir e analisar ao longo do tempo. O segmento de negócio é importante, pois alguns indicadores são relevantes para algumas áreas, mas podem não ter relevância para outras.

Por exemplo: uma empresa no segmento de varejo online muito provavelmente vai querer avaliar o indicador de usuários ativos por mês (total de usuários que fazem compras em cada mês). Já uma empresa no segmento de logística pode estar interessa no tempo médio de descarga e entrega de produtos. A definição do indicador é uma definição do negócio em si.

O link abaixo oferece uma biblioteca de KPIs com exemplos nos mais variados setores:

http://kpilibrary.com

Definindo o Problema

Trabalhamos como analistas em uma empresa que comercializa produtos importados dos mais variados tipos para diversos países ao redor do mundo.

Nosso trabalho é calcular, analisar e interpretar 8 indicadores chave de performance com base nos dados fornecidos. Os dados são fictícios, mas representam valores que podem ser considerados reais.

Os indicadores foram definidos pela área de planejamento estratégico da empresa que precisa acompanhar a evolução das vendas e a efetividade das campanhas de Marketing ao longo do tempo!

Aqui estão os 8 indicadores que farão parte da nossa análise:

• Indicador 1 - Faturamento Mensal
• Indicador 2 - Taxa Percentual de Crescimento Mensal
• Indicador 3 - Clientes Ativos Por Mês em um País (Brasil)
• Indicador 4 - Total de Itens Comprados Por Mês em um País (Brasil)
• Indicador 5 - Faturamento Médio Mensal em um País (Brasil)
• Indicador 6 - Diferença de Faturamento ao Longo do Tempo Entre Clientes Novos e Antigos
• Indicador 7 - Taxa de Novos Clientes
• Indicador 8 - Taxa Mensal de Retenção de Clientes

Projeto 3 – Análise de Indicadores de Performance em Redes de Varejo

```
# <font color='blue'>Data Science Academy</font>
# <font color='blue'>Business Analytics</font>
# <font color='blue'>Capítulo 6 - Marketing Analytics</font>
# <font color='blue'>Projeto 3</font>
## <font color='blue'>Análise de Indicadores de Performance em Redes de Varejo</font>
# ->
# Versão da Linguagem Python
from platform import python_version
print('Versão da Linguagem Python Usada Neste Jupyter Notebook:', python_version())

![title](imagens/projeto3.png)
## Marketing Analytics
Marketing Analytics compreende os processos e tecnologias que permitem aos profissionais de Marketing avaliar o sucesso de suas iniciativas.
Isso é feito medindo o desempenho das campanhas de Marketing, coletando os dados e analisando os resultados. Marketing Analytics utiliza
métricas importantes de negócios, como ROI (Retorno Sobre o Investimento), Atribuição de Marketing e Eficácia Geral do Marketing. Em outras
palavras, o Marketing Analytics mostra se os programas de Marketing estão sendo efetivos ou não.
Marketing Analytics reúne dados de todos os canais de marketing e os consolida em uma visão de marketing comum. A partir dessa visão comum,
você pode extrair resultados analíticos que podem fornecer assistência inestimável para impulsionar os esforços de marketing.
## O Que São KPIs (Indicadores Chave de Performance)?
Leia o manual em pdf no Capítulo 6.
## Como Definir os KPIs?
Leia o manual em pdf no Capítulo 6.
## Definindo o Problema
Leia o manual em pdf no Capítulo 6.
## Instalando e Carregando os Pacotes
# ->
# Para atualizar um pacote, execute o comando abaixo no terminal ou prompt de comando:
# pip install -U nome_pacote
# Para instalar a versão exata de um pacote, execute o comando abaixo no terminal ou prompt de comando:
# !pip install nome_pacote==versão_desejada
# Depois de instalar ou atualizar o pacote, reinicie o jupyter notebook.
# Instala o pacote watermark.
# Esse pacote é usado para gravar as versões de outros pacotes usados neste jupyter notebook.
!pip install -q -U watermark

# ->
# Instala o Plotly
!pip install -q plotly

# ->
# Imports
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib
import plotly
import matplotlib.pyplot as plt
import plotly.offline as pyoff
import plotly.graph_objs as go
from datetime import datetime, timedelta
%matplotlib inline
pyoff.init_notebook_mode()

# ->
# Versões dos pacotes usados neste jupyter notebook
%reload_ext watermark
%watermark -a "Data Science Academy" --iversions

## Carregando os Dados
# ->
```

```
# Carrega os dados
dados = pd.read_csv("dados/dataset.csv", header = 0, encoding = 'unicode_escape')

# ->
# Visualiza os dados
dados.head()
```

Nota: Cada linha (registro) representa um item de um pedido. Observe que a coluna NumeroFatura se repete indicando que é um mesmo pedido com itens diferentes. Para cada item temos o produto, a quantidade, o valor unitário, o cliente e o país.

```
# ->
# Shape
dados.shape

# ->
# Tipos de dados
dados.dtypes

# ->
# Describe
dados.describe()

# ->
# Verificando valores nulos
dados.isna().sum()

# ->
# Range de datas do período que ocorreram as vendas
# (observe que devido ao tipo de dado da coluna o resultado será incorreto)
print('Data Mínima:', dados['DataVenda'].min())
print('Data Máxima:', dados['DataVenda'].max())

# ->
# Converte a coluna de data para o tipo data
dados.DataVenda = pd.to_datetime(dados.DataVenda)

# ->
# Tipos de dados
dados.dtypes

# ->
# Range de datas do período que ocorreram as vendas, agora sim com tipo de dado correto
print('Data Mínima:', dados['DataVenda'].min())
print('Data Máxima:', dados['DataVenda'].max())

# ->
# Países para os quais ocorreram vendas
dados['Pais'].unique()

### Indicador 1 - Faturamento Mensal
Faturamento = Quantidade * Valor_Unitario
# ->
# Extrai o mês da fatura
dados['AnoMes'] = dados['DataVenda'].map(lambda date: 100 * date.year + date.month)

# ->
# Visualiza os dados
dados.head()

# ->
# Calcula o faturamento
dados["Faturamento"] = dados["Quantidade"] * dados["ValorUnitario"]

# ->
# Visualiza os dados
dados.head()

# ->
# Agrupa o faturamento por mês/ano
df_faturamento = dados.groupby(['AnoMes']).agg({'Faturamento': sum}).reset_index()

# ->
# Tabela de dados
df_faturamento

### Visualização do Indicador 1
# ->
# Plot
# Definição dos dados no plot
```

```
plot_data = [go.Scatter(x = df_faturamento['AnoMes'],
                        y = df_faturamento['Faturamento'],)]
# Layout
plot_layout = go.Layout(xaxis = {"type": "category"},
                        title = 'Faturamento Mensal')
# Plot da figura
fig = go.Figure(data = plot_data, layout = plot_layout)
pyoff.iplot(fig)

### Indicador 2 - Taxa Percentual de Crescimento Mensal
Taxa Percentual de Crescimento Mensal = Faturamento Mensal / Faturamento Mensal Anterior * 100
# ->
# Usamos a função pct_change() para calcular a variação percentual mensal
df_faturamento['CrescimentoMensal'] = df_faturamento['Faturamento'].pct_change()

# ->
# Tabela de dados
df_faturamento

### Visualização do Indicador 2
# ->
# Plot
# Definição dos dados no plot (filtramos o mês 12 de 2011 pois não temos dados suficientes)
plot_data = [go.Scatter(x = df_faturamento.query("AnoMes < 201112")['AnoMes'],
                        y = df_faturamento.query("AnoMes < 201112")['CrescimentoMensal'],)]
# Layout
plot_layout = go.Layout(xaxis = {"type": "category"},
                        title = 'Taxa Percentual de Crescimento Mensal')
# Plot da figura
fig = go.Figure(data = plot_data, layout = plot_layout)
pyoff.iplot(fig)

### Indicador 3 - Clientes Ativos Por Mês em um País (Brasil)
Clientes ativos são aqueles que fizeram pelo menos uma compra em cada mês.
# ->
# Cria um dataframe somente com dados do Brasil
dados_brasil = dados.query("Pais=='Brasil'").reset_index(drop = True)

# ->
# Usuários ativos são aqueles que fizeram pelo menos uma compra
df_ativos_mes = dados_brasil.groupby('AnoMes')['IdCliente'].nunique().reset_index()

# ->
# Dados
df_ativos_mes

### Visualização do Indicador 3
# ->
# Plot
# Definição dos dados no plot
plot_data = [go.Bar(x = df_ativos_mes['AnoMes'],
                    y = df_ativos_mes['IdCliente'],)]
# Layout
plot_layout = go.Layout(xaxis = {"type": "category"},
                        title = 'Clientes Ativos Por Mês em um País (Brasil)')
# Plot da figura
fig = go.Figure(data = plot_data, layout = plot_layout)
pyoff.iplot(fig)

### Indicador 4 - Total de Itens Comprados Por Mês em um País (Brasil)
Total de itens comprados por mês.
# ->
# Agrupa os dados para calcular o total de itens comprados por mês no Brasil
df_itens_mes = dados_brasil.groupby('AnoMes')['Quantidade'].sum().reset_index()

# ->
# Dados
df_itens_mes

### Visualização do Indicador 4
# ->
# Plot
# Definição dos dados no plot
plot_data = [go.Bar(x = df_itens_mes['AnoMes'],
                    y = df_itens_mes['Quantidade'],)]
# Layout
plot_layout = go.Layout(xaxis = {"type": "category"},
                        title = 'Total de Itens Comprados Por Mês em um País (Brasil)')
```

```
# Plot da figura
fig = go.Figure(data = plot_data, layout = plot_layout)
pyoff.iplot(fig)

### Indicador 5 - Faturamento Médio Mensal em um País (Brasil)
Faturamento médio por mês em um país.
# ->
# Calcula o faturamento médio
df_faturamento_medio = dados_brasil.groupby('AnoMes')['Faturamento'].mean().reset_index()

# ->
# Dados
df_faturamento_medio

### Visualização do Indicador 5
# ->
# Plot
# Definição dos dados no plot
plot_data = [go.Bar(x = df_faturamento_medio['AnoMes'],
                    y = df_faturamento_medio['Faturamento'],)]
# Layout
plot_layout = go.Layout(xaxis = {"type": "category"},
                        title = 'Faturamento Médio Mensal em um País (Brasil)')
# Plot da figura
fig = go.Figure(data = plot_data, layout = plot_layout)
pyoff.iplot(fig)

# ->
# Calcula o faturamento total por mês
df_faturamento_total = dados_brasil.groupby('AnoMes')['Faturamento'].sum().reset_index()

# ->
# Dados
df_faturamento_total

# ->
# Plot
# Definição dos dados no plot
plot_data = [go.Bar(x = df_faturamento_total['AnoMes'],
                    y = df_faturamento_total['Faturamento'],)]
# Layout
plot_layout = go.Layout(xaxis = {"type": "category"},
                        title = 'Faturamento Total Mensal em um País (Brasil)')
# Plot da figura
fig = go.Figure(data = plot_data, layout = plot_layout)
pyoff.iplot(fig)

### Indicador 6 - Diferença de Faturamento ao Longo do Tempo Entre Clientes Novos e Antigos
Para calcular esse indicador precisaremos de um pouco mais de criatividade. O que é um cliente novo ou antigo?
Vamos considerar cliente novo aquele com baixo volume de compras e cliente antigo aquele com alto volume de compras.
# ->
# Vamos encontrar a data de menor volume de compras de cada cliente
df_compra_minima = dados.groupby('IdCliente')["DataVenda"].min().reset_index()

# ->
# Ajustamos os nomes das colunas
df_compra_minima.columns = ['IdCliente', 'Data_Menor_Compra']

# ->
# Vamos extrair o mês em que ocorreu o menor volume de compras de cada cliente
df_compra_minima['Mes_Menor_Compra_Mensal'] = df_compra_minima['Data_Menor_Compra'].map(lambda date: 100 * date.year + date.month)

# ->
# Dados
df_compra_minima.head()

# ->
# Vamos fazer um merge entre o dataset original e o dataset de volume de compras
dados_compras = pd.merge(dados, df_compra_minima, on = "IdCliente")
dados_compras.head()

# ->
# Vamos criar uma nova coluna de tipo de usuário e preencher como Novo
dados_compras['TipoUsuario'] = 'Novo'

# ->
# Dados
dados_compras['TipoUsuario'].value_counts()
```

```
# ->
# Dados
dados_compras.head()

# ->
# Um cliente antigo é aquele cujo volume de compras no mês é maior que o volume mínimo
# Se for verdadeiro, mudamos a coluna TipoUsuario para "Antigo" e se não, mantemos como "Novo"
dados_compras.loc[dados_compras['AnoMes'] > dados_compras['Mes_Menor_Compra_Mensal'], 'TipoUsuario'] = 'Antigo'

# ->
# Dados
dados_compras['TipoUsuario'].value_counts()


# ->
# Agora calculamos o faturamento por tipo de usuário por mês
df_faturamento_user_mes = dados_compras.groupby(['AnoMes', 'TipoUsuario'])['Faturamento'].sum().reset_index()

# ->
# Removemos o mês 12 de 2011 pois não temos dados suficientes
df_faturamento_user_mes = df_faturamento_user_mes.query("AnoMes != 201012 and AnoMes != 201112")

# ->
# Dados
df_faturamento_user_mes

### Visualização do Indicador 6
# ->
# Plot
# Definição dos dados no plot
plot_data = [go.Scatter(x = df_faturamento_user_mes.query('"TipoUsuario == 'Antigo'"')['AnoMes'],
                        y = df_faturamento_user_mes.query('"TipoUsuario == 'Antigo'"')['Faturamento'],
                        name = 'Cliente Antigo'),
             go.Scatter(x = df_faturamento_user_mes.query('"TipoUsuario == 'Novo'"')['AnoMes'],
                        y = df_faturamento_user_mes.query('"TipoUsuario == 'Novo'"')['Faturamento'],
                        name = 'Cliente Novo')]
# Layout
plot_layout = go.Layout(xaxis = {"type": "category"},
                        title = 'Diferença de Faturamento ao Longo do Tempo Entre Clientes Novos e Antigos')
# Plot da figura
fig = go.Figure(data = plot_data, layout = plot_layout)
pyoff.iplot(fig)

### Indicador 7 - Taxa de Novos Clientes
Como definimos clientes novos e antigos no indicador 6, agora podemos usar os dados e calcular a proporção de novos clientes ao longo do tempo.
# ->
# Calcula a taxa de novos clientes
df_taxa_novos_clientes = dados_compras.query('"TipoUsuario == 'Novo'").groupby(['AnoMes'])['IdCliente'].nunique() /
dados_compras.query('"TipoUsuario == 'Antigo'").groupby(['AnoMes'])['IdCliente'].nunique()

# ->
# Ajustamos índice e removemos valores ausentes
df_taxa_novos_clientes = df_taxa_novos_clientes.reset_index()
df_taxa_novos_clientes = df_taxa_novos_clientes.dropna()

# ->
# Dados
df_taxa_novos_clientes

### Visualização do Indicador 7
# ->
# Plot
# Definição dos dados no plot
plot_data = [go.Bar(x = df_taxa_novos_clientes.query("AnoMes > 201101 and AnoMes < 201112")['AnoMes'],
                    y = df_taxa_novos_clientes.query("AnoMes > 201101 and AnoMes < 201112")['IdCliente'],)]
# Layout
plot_layout = go.Layout(xaxis = {"type": "category"},
                        title = 'Taxa de Novos Clientes')
# Plot da figura
fig = go.Figure(data = plot_data, layout = plot_layout)
pyoff.iplot(fig)

### Indicador 8 - Taxa Mensal de Retenção de Clientes
Taxa Mensal de Retenção de Clientes  = Clientes do Mês Anterior / Total de Clientes Ativos
# ->
# Agrupamos os dados por cliente e mês e somamos o faturamento
dados_compras_clientes = dados_compras.groupby(['IdCliente', 'AnoMes'])['Faturamento'].sum().reset_index()
```

```
# ->
# Dados
dados_compras_clientes.head()

# ->
# Agora definimos a retenção com uma tabela cruzada
df_ret = pd.crosstab(dados_compras_clientes['IdCliente'], dados_compras_clientes['AnoMes']).reset_index()

# ->
# Dados
df_ret.head()

# ->
# Extraímos os meses
meses = df_ret.columns[2:]
meses

# ->
# O loop abaixo vai calcular a retenção ao longo dos meses
# Lista para gravar o resulatdo
lista_ret = []
# Loop
for i in range(len(meses)-1):
    dados_retencao = {}
    mes_corrente = meses[i+1]
    mes_anterior = meses[i]
    dados_retencao['AnoMes'] = int(mes_corrente)
    dados_retencao['TotalUser'] = df_ret[mes_corrente].sum()
    dados_retencao['TotalRetido'] = df_ret[(df_ret[mes_corrente] > 0) & (df_ret[mes_anterior] > 0)][mes_corrente].sum()
    lista_ret.append(dados_retencao)
lista_ret

# ->
# Dados
df_ret_final = pd.DataFrame(lista_ret)
df_ret_final.head()

Agora calculamos a proporção para encontrar o indicador.
# ->
# Calculamos o indicador
df_ret_final['TaxaRetencao'] = df_ret_final['TotalRetido'] / df_ret_final['TotalUser']
df_ret_final

### Visualização do Indicador 8
# ->
# Plot
# Definição dos dados no plot
plot_data = [go.Scatter(x = df_ret_final.query("AnoMes < 201112")['AnoMes'],
                        y = df_ret_final.query("AnoMes < 201112")['TaxaRetencao'],
                        name = "taxa")]
# Layout
plot_layout = go.Layout(xaxis = {"type": "category"},
                        title = 'Taxa Mensal de Retenção de Clientes')
# Plot da figura
fig = go.Figure(data = plot_data, layout = plot_layout)
pyoff.iplot(fig)

# Fim
```

8.7. Marketing Analytics – Projeto 4 – Otimização de Preços e Mix de Produtos

Definindo o Problema

A empresa Lua Smart Tech monta e testa dois modelos de smartphones, Lua1 e Lua2. Para o próximo mês, a empresa quer decidir quantas unidades de cada modelo vai montar e depois testar.

Nenhum smartphone está em estoque desde o mês anterior e, como esses modelos serão trocados depois deste mês, a empresa não quer manter nenhum estoque para o mês seguinte.

Ela acredita que o máximo que pode vender neste mês são 600 unidades do modelo Lua1 e 1200 unidades do modelo Lua2.

Cada modelo Lua1 é vendido por R$300 e cada modelo Lua2 por R$450. O custo dos componentes de um Lua1 é de R$150 e para um Lua2 é R$225.

A mão de obra é necessária para a montagem e teste. Existem no máximo 10.000 horas de montagem e 3.000 horas de teste disponíveis. Cada hora de trabalho para montagem custa R$11 e cada hora de trabalho para teste custa R$15. Cada Lua1 requer cinco horas para montagem e uma hora para teste. Cada Lua2 requer seis horas para montagem e duas horas para teste.

A Lua Smart Tech deseja saber quantas unidades de cada modelo deve produzir (montar e testar) para maximizar seu lucro líquido, mas não pode usar mais horas de trabalho do que as disponíveis e não deseja produzir mais do que pode vender.

Seu trabalho é realizar a otimização de preços e mix dos produtos da Lua Smart Tech.

Projeto 4 – Otimização de Preços e Mix de Produtos

```
# <font color='blue'>Data Science Academy</font>
# <font color='blue'>Business Analytics</font>
# <font color='blue'>Capítulo 7 - Marketing Analytics</font>
# <font color='blue'>Projeto 4</font>
## <font color='blue'>Otimização de Preços e Mix de Produtos</font>
# ->
# Versão da Linguagem Python
from platform import python_version
print('Versão da Linguagem Python Usada Neste Jupyter Notebook:', python_version())

![title](imagens/projeto4.png)
## Marketing Analytics
Marketing Analytics compreende os processos e tecnologias que permitem aos profissionais de Marketing avaliar o sucesso de suas iniciativas. Isso é feito medindo o desempenho das campanhas de Marketing, coletando os dados e analisando os resultados. Marketing Analytics utiliza métricas importantes de negócios, como ROI (Retorno Sobre o Investimento), Atribuição de Marketing e Eficácia Geral do Marketing. Em outras palavras, o Marketing Analytics mostra se os programas de Marketing estão sendo efetivos ou não.
Marketing Analytics reúne dados de todos os canais de marketing e os consolida em uma visão de marketing comum. A partir dessa visão comum, você pode extrair resultados analíticos que podem fornecer assistência inestimável para impulsionar os esforços de marketing.
## Definindo o Problema
Leia o manual em pdf no Capítulo 7.
## Instalando e Carregando os Pacotes
# ->
# Para atualizar um pacote, execute o comando abaixo no terminal ou prompt de comando:
# pip install -U nome_pacote
# Para instalar a versão exata de um pacote, execute o comando abaixo no terminal ou prompt de comando:
# !pip install nome_pacote==versão_desejada
# Depois de instalar ou atualizar o pacote, reinicie o jupyter notebook.
# Instala o pacote watermark.
# Esse pacote é usado para gravar as versões de outros pacotes usados neste jupyter notebook.
!pip install -q -U watermark

# ->
# Instala o PuLP
```

```
# https://pypi.org/project/PuLP/
# https://coin-or.github.io/pulp/
!pip install -q pulp

# ->
# Imports
from pulp import *

# ->
# Versões dos pacotes usados neste jupyter notebook
%reload_ext watermark
%watermark -a "Data Science Academy" --iversions
```

Criando o Modelo Matemático Para a Otimização

Parâmetros

- Ai = Número máximo de smartphones modelo tipo i para vender este mês, onde i pertence ao conjunto {Lua1, Lua2}
- Bi = Preço de venda de smartphones modelo tipo i, onde i pertence ao conjunto {Lua1, Lua2}
- Ci = Preço de custo das peças componentes para smartphones modelo tipo i, onde i pertence ao conjunto {Lua1, Lua2}
- Di = Custo de mão de obra de montagem por hora de smartphones modelo tipo i, onde i pertence ao conjunto {Lua1, Lua2}
- Ei = Custo de mão de obra de teste por hora de smartphones modelo tipo i, onde i pertence ao conjunto {Lua1, Lua2}
- F = Número máximo de horas de trabalho de montagem
- G = Número máximo de horas de trabalho de teste
- Hf,i = Horas de montagem necessárias para construir cada modelo de smartphone tipo i, onde i pertence ao conjunto {Lua1, Lua2}
- Hg,i = Horas de teste necessárias para testar cada modelo de smartphone tipo i, onde i pertence ao conjunto {Lua1, Lua2}

Variável de Decisão

- Xi = Número de smartphones modelo tipo i a produzir este mês, onde i pertence ao conjunto {Lua1, Lua2}

Função Objetivo

Restrições

- O número de smartphones modelo tipo i a serem produzidos não pode ser negativo, ou seja, Xi >= 0, onde i pertence ao conjunto {Lua1, Lua2}.
- O número total de horas de montagem não pode ser maior que o número máximo de horas de mão de obra de montagem disponíveis.
- O número total de horas de teste não pode ser maior do que o máximo de horas de mão de obra de teste disponíveis.
- O número de smartphones modelo tipo i a serem produzidos não pode ser maior do que o número máximo de smartphones modelo tipo i a serem vendidos neste mês, onde i pertence ao conjunto {Lua1, Lua2}.

Implementando o Modelo Matemático

Organizando os Parâmetros

```
# ->
# Número máximo de smartphones para vender este mês
A_Lua1 = 600
A_Lua2 = 1200

# ->
# Preço de venda de smartphones
B_Lua1 = 300
B_Lua2 = 450

# ->
# Preço de custo das peças componentes para smartphones
C_Lua1 = 150
C_Lua2 = 225

# ->
# Custo de mão de obra de montagem por hora de smartphones
D_Lua1 = 11
D_Lua2 = 11

# ->
# Custo de mão de obra de teste por hora de smartphones
E_Lua1 = 15
E_Lua2 = 15

# ->
# Número máximo de horas de trabalho de montagem
F = 10000

# ->
# Número máximo de horas de trabalho de teste
G = 3000

# ->
# Horas de montagem necessárias para construir cada modelo de smartphone
Hfi_Lua1 = 5
Hfi_Lua2 = 6

# ->
# Horas de teste necessárias para testar cada modelo de smartphone
Hgi_Lua1 = 1
Hgi_Lua2 = 2
```

```
### Criação da Variável Para o Problema de Otimização
# ->
help(LpProblem)

# ->
# Variável para o problema
problema = LpProblem("MixProdutos", LpMaximize)

# ->
# Visualiza
problema

### Definindo a Variável de Decisão de Cada Modelo de Smartphone
# ->
help(LpVariable)

# ->
# Define as variáveis
x_Lua1 = LpVariable("Unidades Lua1", 0, None, LpInteger)
x_Lua2 = LpVariable("Unidades Lua2", 0, None, LpInteger)

# ->
# Print
print(x_Lua1)
print(x_Lua2)

### Implementando a Função Objetivo
![title](imagens/formula.png)
# ->
# Faturamento
faturamento = (x_Lua1 * B_Lua1) + (x_Lua2 * B_Lua2)
faturamento

# ->
# Custo de Montagem
custo_montagem = (x_Lua1 * Hfi_Lua1 * D_Lua1) + (x_Lua2 * Hfi_Lua2 * D_Lua2)
custo_montagem

# ->
# Custo de Teste
custo_teste = (x_Lua1 * Hgi_Lua1 * E_Lua1) + (x_Lua2 * Hgi_Lua2 * E_Lua2)
custo_teste

# ->
# Custo de Componentes
custo_componentes = (x_Lua1 * C_Lua1) + (x_Lua2 * C_Lua2)
custo_componentes

# ->
# Lucro = Faturamento - Custo de Montagem - Custo de Teste - Custo de Componentes
problema += faturamento - custo_montagem - custo_teste - custo_componentes
problema

### Restrições
# ->
# Número máximo de horas de montagem
problema += (x_Lua1 * Hfi_Lua1) + (x_Lua2 * Hfi_Lua2) <= F,"Número Máximo de Horas de Montagem"

# ->
# Número máximo de horas de teste
problema += (x_Lua1 * Hgi_Lua1) + (x_Lua2 * Hgi_Lua2) <= G,"Número Máximo de Horas de Teste"

# ->
# Produção menor ou igual a demanda pelo modelo Lua1
problema += x_Lua1 <= A_Lua1,"Produção menor ou igual a demanda pelo modelo Lua1"

# ->
# Produção menor ou igual a demanda pelo modelo Lua2
problema += x_Lua2 <= A_Lua2,"Produção menor ou igual a demanda pelo modelo Lua2"

# ->
# Problema final
problema

### Resolvendo o Problema de Otimização
# ->
# Otimização
```

```
problema.solve()

# ->
# Lucro Maximizado
print("Lucro Maximizado:", value(problema.objective))

# ->
# Número de Unidades do Modelo Lua 1 a Produzir
print("Número de Unidades do Modelo Lua 1 a Produzir:", problema.variables()[0].varValue)

# ->
# Número de Unidades do Modelo Lua 1 a Produzir
print("Número de Unidades do Modelo Lua 2 a Produzir:", problema.variables()[1].varValue)

##  Conclusão
A empresa Lua Smart Tech deve produzir 560 unidades do tipo de smartphone Lua1 e 1200 do tipo Lua2 para atingir o lucro máximo de R$199.600.
# Fim
```

8.8. RH Analytics

Como o People Analytics Está Sendo Usado?

O uso adequado das informações é algo poderoso para a tomada de decisões nas empresas. Normalmente, quanto melhor é a análise prévia feita pelo gestor, maiores as chances de sucesso nas suas decisões.

Em relação à gestão de pessoas, isso não é diferente e é por causa de tal questão que as empresas vêm buscando formas de mapear comportamentos e condições que influenciem positivamente o desempenho de seus colaboradores.

Um dos melhores recursos para aperfeiçoar a performance da equipe é o denominado People Analytics. E um dos cases de maior sucesso é do Google. Vamos definir People Analytics, enquanto discutimos o projeto inovador e pioneiro do Google, que levou ao uso cada vez maior de People Analytics.

Foi o Google que desenvolveu pioneiramente o People Analytics. No início o foco principal do uso do People Analytics era aumentar a eficiência da equipe de recrutamento, sem ter de contratar mais gente para essa área. O Google começou a desenvolver tecnologia de análise automatizada para poder fazer uma previsão de quais seriam os candidatos mais viáveis no processo seletivo utilizado.

Depois de desenvolver uma primeira tecnologia para análise de recrutamento, foi criada uma que media quanto o colaborador estava engajado na empresa. O Google usou o People Analytics para se proteger. Verificou os fatores que aumentavam a chance de um funcionário ficar no Google, já que sempre há organizações tentando contratar seus colaboradores, e investiu nisso. A experiência do Google mostra que o People Analytics possibilita definir as características das melhores lideranças e o melhor papel dos gestores.

Por exemplo, a análise da massa de dados dos funcionários, provou que, para ser um líder bem-sucedido no Google, o indivíduo deve, mais do que demonstrar conhecimento técnico superior, oferecer treinamentos e coaching periódicos e individuais a seus colaboradores, o que significa expressar um real interesse por eles e dar frequentes feedbacks personalizados.

O desenvolvimento de um algoritmo para retenção dos profissionais é outro uso do People Analytics no Google. Por meio dele, a empresa tem conseguido prever com sucesso e de maneira pró-ativa quais colaboradores estão mais propensos a deixar a organização e evitar que isso aconteça.

O algoritmo tem permitido ao Google agir antes que seja tarde demais e também criar soluções e propostas de retenção personalizadas. Pesquisas com diversas empresas nos EUA mostraram que programas de retenção de talentos, fazem a empresa economizar até 7 milhões de dólares com processos de recrutamento. Os bons resultados obtidos pela equipe de People Analytics do Google vêm principalmente da força científica dos dados e das ações que propõem. “Em geral, os executivos são racionais e isso é o tipo de coisa que os influencia.”

As vantagens do People Analytics. O conceito de avaliar dados e informações sobre os funcionários em busca de otimizações traz vantagens diversas, sendo a mais evidente a retenção de talentos na empresa. Outro ponto é que, como a escolha é feita de forma adequada, existem menos chances de que o candidato errado seja escolhido para uma determinada vaga — o que, de modo desfavorável,

implicaria em um processo mais burocrático e na necessidade de reiteradas seleções de recursos humanos.

Entre os já contratados, o favorecimento da retenção acontece porque os funcionários ficam mais satisfeitos, motivados e possuem mais objetivos dentro da própria empresa.

Além disso, é possível gerar impactos positivos no desempenho, tanto por identificar as condições que favorecem a produtividade, quanto pela alocação correta de talentos para as funções e projetos específicos. Essas informações ainda fornecem insights sobre a criatividade de cada colaborador, e como isso pode impactar direta e positivamente nos resultados – um vendedor mais criativo, por exemplo, tem mais chances de fazer negócios do que aquele que é estritamente metódico.

O uso do People Analytics está cada vez mais fazendo parte da realidade das empresas porque é uma forma de melhorar a assertividade da gestão de pessoas, através de dados e modelos preditivos. Big Data e Machine Learning em ação.

Referências:

People analytics at Google: using data to make Google a great place to Work
https://digital.hbs.edu/platform-digit/submission/people-analytics-at-google-using-data-tomake-google-a-great-place-to-work/

Case study: how Google uses People analytics
https://www.sagepeople.com/about-us/news-hub/case-study-how-google-uses-peopleanalytics

Projeto People Analytics – Quais Fatores Mais Causam Atritos no Ambiente de Trabalho?

A partir de agora vamos trabalhar no projeto deste capítulo. Durante o projeto estudaremos não apenas conceitos técnicos, mas definiremos questões e características de projetos na área de RH.

Neste projeto nosso objetivo é analisar uma série de atributos de colaboradores de uma empresa e identificar quais atributos são mais relevantes para gerar atritos entre os colaboradores e então apresentaremos nossas conclusões aos tomadores de decisão.

Trabalharemos com Linguagem R.

```
# Projeto People Analytics - Quais Fatores Mais Causam Atritos no Ambiente de Trabalho?

# Leia a definição do projeto no Capítulo 8 do curso.

# Configurando o diretório de trabalho
# Coloque entre aspas o diretório de trabalho que você está usando no seu computador
# Não use diretórios com espaço no nome
setwd("~/Dropbox/DSA/Business-Analytics2.0/Cap08")
getwd()

# Imports
library(caret)
library(ggplot2)
library(gridExtra)
library(data.table)
library(car)
library(caTools)
library(corrplot)
library(rpart)
library(rpart.plot)

# Carregando o dataset
dados_rh <- fread('dados/dataset.csv')
dim(dados_rh)
```

```
View(dados_rh)
str(dados_rh)
summary(dados_rh)

##### Limpeza e Transformação #####

# Transformando variáveis categóricas para o tipo fator
View(dados_rh)
dados_rh$Attrition               <- as.factor(dados_rh$Attrition)
dados_rh$BusinessTravel          <- as.factor(dados_rh$BusinessTravel)
dados_rh$Department              <- as.factor(dados_rh$Department)
dados_rh$Education               <- as.factor(dados_rh$Education)
dados_rh$EducationField          <- as.factor(dados_rh$EducationField)
dados_rh$'Employee Source'       <- as.factor(dados_rh$'Employee Source')
dados_rh$EnvironmentSatisfaction <- as.factor(dados_rh$EnvironmentSatisfaction)
dados_rh$Gender                  <- as.factor(dados_rh$Gender)
dados_rh$JobInvolvement          <- as.factor(dados_rh$JobInvolvement)
dados_rh$JobLevel                <- as.factor(dados_rh$JobLevel)
dados_rh$JobRole                 <- as.factor(dados_rh$JobRole)
dados_rh$JobSatisfaction         <- as.factor(dados_rh$JobSatisfaction)
dados_rh$MaritalStatus           <- as.factor(dados_rh$MaritalStatus)
dados_rh$OverTime                <- as.factor(dados_rh$OverTime)
dados_rh$PerformanceRating       <- as.factor(dados_rh$PerformanceRating)
dados_rh$RelationshipSatisfaction <- as.factor(dados_rh$RelationshipSatisfaction)
dados_rh$StockOptionLevel        <- as.factor(dados_rh$StockOptionLevel)
dados_rh$WorkLifeBalance         <- as.factor(dados_rh$WorkLifeBalance)
str(dados_rh)

# Transformando variáveis numéricas para o tipo inteiro
View(dados_rh)
dados_rh$DistanceFromHome  <- as.integer(dados_rh$DistanceFromHome)
dados_rh$MonthlyIncome     <- as.integer(dados_rh$MonthlyIncome)
dados_rh$PercentSalaryHike <- as.integer(dados_rh$PercentSalaryHike)

# Drop dos níveis de fatores com 0 count
dados <- droplevels(dados_rh)
str(dados_rh)
summary(dados_rh)
View(dados_rh)

##### Engenharia de Atributos #####

# Criamos uma coluna de anos anteriores de experiência para visualizar melhor o
# perfil de experiência do funcionário.
dados_rh$PriorYearsOfExperience <- dados_rh$TotalWorkingYears - dados_rh$YearsAtCompany
View(dados_rh)

# A estabilidade no emprego (job tenure) é a medida do tempo que um funcionário está empregado
# por seu empregador atual. A estabilidade no emprego de um funcionário é muito importante e
# muitas vezes os empregadores consideram a estabilidade no emprego um critério para a contratação
# de novos funcionários. A permanência no emprego pode ser longa ou curta.

# Criamos um novo recurso de estabilidade média para traçar o perfil de permanência média
# dos funcionários em empresas anteriores.
dados_rh$AverageTenure <- dados_rh$PriorYearsOfExperience / dados_rh$NumCompaniesWorked
View(dados_rh)

# A estabilidade média produz valores como Inf devido à natureza de sua derivação
# Substituímos para zero.
summary(dados_rh$AverageTenure)
dados_rh$AverageTenure[!is.finite(dados_rh$AverageTenure)] <- 0
summary(dados_rh$AverageTenure)
View(dados_rh)

# Analisamos e dividimos os dados como base na coluna Termination, que indica se
# o funcionário foi desligado da empresa.
dados_rh_1 <- dados_rh[dados_rh$Attrition != 'Termination']
dados_rh_1 <- droplevels(dados_rh_1)
dim(dados_rh_1)
summary(dados_rh_1)

# Mesmo filtro anterior, mas agora por demissão voluntária
dados_rh_2 <- dados_rh[dados_rh$Attrition != 'Voluntary Resignation']
dados_rh_2 <-droplevels(dados_rh_2)
dim(dados_rh_2)
summary(dados_rh_2)

##### Análise Exploratória #####
```

```
# Plots de análise univariada
ggplot(dados_rh) + geom_bar(aes(x = Gender))
ggplot(dados_rh) + geom_density(aes(x = Age))
ggplot(dados_rh) + geom_bar(aes(x = Attrition))
ggplot(dados_rh) + geom_bar(aes(x = Department))
ggplot(dados_rh) + geom_bar(aes(x = JobRole))
ggplot(dados_rh) + geom_bar(aes(x = Education)) + facet_grid(~EducationField)

# Multiplot Grid
p.TotalWorkingYears       <- ggplot(dados_rh) + geom_density(aes(TotalWorkingYears))
p.YearsAtCompany          <- ggplot(dados_rh) + geom_density(aes(YearsAtCompany))
p.YearsSinceLastPromotion <- ggplot(dados_rh) + geom_density(aes(YearsSinceLastPromotion))
p.YearsWithCurrManager    <- ggplot(dados_rh) + geom_density(aes(YearsWithCurrManager))
p.YearsInCurrentRole      <- ggplot(dados_rh) + geom_density(aes(YearsInCurrentRole))
p.PriorYearsOfExperience  <- ggplot(dados_rh) + geom_density(aes(PriorYearsOfExperience))

# Organiza no grid
grid.arrange(p.TotalWorkingYears,
             p.YearsAtCompany,
             p.YearsSinceLastPromotion,
             p.YearsWithCurrManager,
             p.YearsInCurrentRole,
             p.PriorYearsOfExperience,
             nrow = 2,
             ncol = 3)

# Tempo de experiência anterior
# Vamos descobrir a proporção de funcionários com menos de alguns anos de experiência
# (valores escolhidos: 1, 3, 5, 7, 10 anos)
length(which(dados_rh$PriorYearsOfExperience < 1)) / length(dados_rh$PriorYearsOfExperience)
length(which(dados_rh$PriorYearsOfExperience < 3)) / length(dados_rh$PriorYearsOfExperience)
length(which(dados_rh$PriorYearsOfExperience < 5)) / length(dados_rh$PriorYearsOfExperience)
length(which(dados_rh$PriorYearsOfExperience < 7)) / length(dados_rh$PriorYearsOfExperience)
length(which(dados_rh$PriorYearsOfExperience < 10)) / length(dados_rh$PriorYearsOfExperience)

# Exemplo de insight:
# 58% dos funcionários têm menos de 3 anos de experiência de trabalho antes de entrar na IBM
# Possíveis problemas: conjuntos de habilidades subdesenvolvidos, base de jovens funcionários,
# mentalidade de "trabalho" imatura.

# Idade
length(which(dados_rh$Age < 30)) / length(dados_rh$Age)

# Exemplo de insight:
# Apenas 22% dos funcionários têm menos de 30 anos, a base de funcionários não é exatamente
# tão jovem como o esperado.

# # Educação
summary(dados_rh$Education)
length(which(dados_rh$Education == 3)) / length(dados_rh$Education)
length(which(dados_rh$Education == 4)) / length(dados_rh$Education)

# Exemplo de insight:
# Cerca de 39% dos funcionários são graduados e 27% realizaram o mestrado.
# A busca pelo ensino superior pode ter levado a uma diminuição da experiência de trabalho.

# Boxplot mostrando a distribuição do salário mensal para todos os 4 níveis
# de satisfação no trabalho de 1-4
ggplot(data = subset(dados_rh, !is.na(JobSatisfaction)), aes(JobSatisfaction, MonthlyIncome)) +
  geom_boxplot()

# Exemplo de Insight
# Não há sinais óbvios de que um salário mais alto leva a uma maior satisfação no trabalho

# Correlação
cor(dados_rh$TotalWorkingYears, dados_rh$YearsAtCompany,          use = "complete.obs")
cor(dados_rh$YearsAtCompany,    dados_rh$YearsInCurrentRole,      use = "complete.obs")
cor(dados_rh$YearsAtCompany,    dados_rh$YearsSinceLastPromotion, use = "complete.obs")
cor(dados_rh$YearsAtCompany,    dados_rh$YearsWithCurrManager,    use = "complete.obs")
cor(dados_rh$TotalWorkingYears, dados_rh$MonthlyIncome,           use = "complete.obs")
cor(dados_rh$YearsAtCompany,    dados_rh$MonthlyIncome,           use = "complete.obs")

# Scatterplots
ggplot(dados_rh) + geom_point(aes(TotalWorkingYears, MonthlyIncome))
ggplot(dados_rh) + geom_point(aes(YearsAtCompany, MonthlyIncome))

# Vamos investigar a relação do equilíbrio entre vida pessoal e profissional e renda mensal
```

```
ggplot(data = subset(dados_rh, !is.na(WorkLifeBalance)), aes(WorkLifeBalance, MonthlyIncome)) +
  geom_boxplot()

# Exemplo de insight
# Os funcionários que avaliaram o equilíbrio entre vida profissional e pessoal igual a 1 também têm renda média mensal
# significativamente mais baixa.
# Baixo equilíbrio entre vida profissional e baixo salário? Um problema que o departamento de RH precisa examinar.

# Verificando a diferença salarial entre homens e mulheres.
ggplot(data = subset(dados_rh, !is.na(Gender)), aes(Gender, MonthlyIncome, fill = Gender)) +
  geom_boxplot() +
  theme(legend.position = "none", plot.title = element_text(hjust = 0.5, size = 10)) +
  labs(x = "Gender", y = "Monthly Income", title = "Salário Mensal Entre Gêneros") +
  coord_flip()

# Exemplo de insight
# Não há sinais de discriminação de gênero; na verdade, as mulheres ganham um pouco mais, em média,
# desconsiderando todos os outros fatores.

# Função
ggplot(data = subset(dados_rh, !is.na(JobRole))) + geom_boxplot(aes(JobRole, MonthlyIncome)) +
  ggtitle("Salário Mensal Por Função")

ggplot(data = subset(dados_rh, !is.na(JobRole))) + geom_boxplot(aes(JobRole, AgeStartedWorking)) +
  ggtitle("Idade Que Iniciou na Função")

ggplot(data = subset(dados_rh, !is.na(JobRole))) + geom_boxplot(aes(JobRole, Age)) +
  ggtitle("Idade Por Função")

ggplot(data = subset(dados_rh, !is.na(JobRole))) + geom_boxplot(aes(JobRole, YearsAtCompany)) +
  ggtitle("Tempo de Empresa (em anos)")

ggplot(data = na.omit(dados_rh)) + geom_bar(aes(JobRole, fill = Education), position = "fill") +
  ggtitle("Nível de Educação Por Função") +
  ylab("Proportion")

# Plots de análise multivariada para variáveis normalmente usadas durante o processo de contratação
ggplot(data = dados_rh_1) +
  geom_bar(aes(x = Education , fill = Attrition), position = 'fill') +
  facet_grid(.~Department)

ggplot(data = dados_rh_1) +
  geom_bar(aes(x = Education , fill = Attrition), position = 'fill') +
  facet_grid(.~JobRole)

ggplot(data = dados_rh_1) +
  geom_bar(aes(x = EducationField , fill
 = Attrition), position = 'fill') +
  facet_grid(.~JobRole) +
  theme(axis.text.x = element_text(angle = -90, hjust = 0))

# Plots de análise multivariada para variáveis normalmente usadas após o processo de contratação
ggplot(dados_rh_1) + geom_bar(aes(x = Age, fill = Attrition), position = 'fill')
ggplot(dados_rh_1) + geom_bar(aes(x = Department, fill = Attrition), position = 'fill')
ggplot(dados_rh_1) + geom_bar(aes(x = DistanceFromHome, fill = Attrition), position = 'fill')
ggplot(dados_rh_1) + geom_bar(aes(x = `Employee Source`, fill = Attrition), position = 'fill')
ggplot(dados_rh_1) + geom_bar(aes(x = JobRole, fill = Attrition), position = 'fill')
ggplot(dados_rh_1) + geom_bar(aes(x = MaritalStatus, fill = Attrition), position = 'fill')
ggplot(dados_rh_1) + geom_bar(aes(x = AverageTenure, fill = Attrition), position = 'fill')
ggplot(dados_rh_1) + geom_bar(aes(x = Education, fill = Attrition), position = 'fill')
ggplot(dados_rh_1) + geom_bar(aes(x = EducationField, fill = Attrition),position ='fill')
ggplot(dados_rh_1) + geom_bar(aes(x = Gender, fill = Attrition), position = 'fill')

# Plots de análise multivariada entre algumas variáveis e o status do funcionário
ggplot(dados_rh_1) + geom_boxplot(aes(Attrition, MonthlyIncome))
ggplot(dados_rh_1) + geom_boxplot(aes(Attrition, PercentSalaryHike))
ggplot(dados_rh_1) + geom_bar(aes(TrainingTimesLastYear, fill = Attrition), position = 'fill')
ggplot(dados_rh_1) + geom_bar(aes(BusinessTravel, fill = Attrition), position = 'fill')
ggplot(dados_rh_1) + geom_bar(aes(OverTime, fill = Attrition), position = 'fill')
ggplot(dados_rh_1) + geom_bar(aes(StockOptionLevel, fill = Attrition), position = 'fill')
ggplot(dados_rh_1) + geom_bar(aes(EnvironmentSatisfaction, fill = Attrition), position = 'fill')
ggplot(dados_rh_1) + geom_bar(aes(JobSatisfaction, fill = Attrition), position = 'fill')
ggplot(dados_rh_1) + geom_bar(aes(JobInvolvement, fill = Attrition), position = 'fill')
ggplot(dados_rh_1) + geom_bar(aes(RelationshipSatisfaction, fill = Attrition), position = 'fill')
ggplot(dados_rh_1) + geom_bar(aes(WorkLifeBalance, fill = Attrition), position = 'fill')

##### Modelagem Preditiva #####
```

```
# Vamos concentrar nosso trabalho em tentar ajudar o RH a recrutar melhor visando evitar atritos
# e, consequentemente, demissões.

# Criaremos 5 versões do modelo e para cada um vamos explorar as opções e interpretar o resultado.

# Primeira versão do modelo com algumas variáveis
?glm
modelo_v1 <- glm(Attrition ~ Age + Department + DistanceFromHome + `Employee Source` +
                   JobRole + MaritalStatus + AverageTenure + PriorYearsOfExperience + Gender +
                   Education + EducationField,
                 family = binomial,
                 data = dados_rh)

summary(modelo_v1)
?vif
vif(modelo_v1)

# Vamos dividir os dados em treino e teste. Vamos trabalhar com os dados sem registros de demitidos.
set.seed(2004)
index_treino <- sample.split(Y = dados_rh_1$Attrition, SplitRatio = 0.7)
dados_rh_1_treino <- subset(dados_rh_1, train = T)
dados_rh_1_teste <- subset(dados_rh_1, train = F)

# Segunda versão do modelo com dados de treino
modelo_v2 <- glm(Attrition ~ Age + Department + DistanceFromHome + `Employee Source` +
                   JobRole + MaritalStatus + AverageTenure + PriorYearsOfExperience + Gender +
                   Education + EducationField,
                 family = binomial,
                 data = dados_rh_1_treino)

summary(modelo_v2)
vif(modelo_v2)

# Previsões
threshold <- 0.5
previsoes_v2 <- predict(modelo_v2, type = 'response', newdata = dados_rh_1_teste)
previsoes_finais_v2 <- ifelse(previsoes_v2 > threshold, 'Voluntary Resignation', 'Current employee')
table(dados_rh_1_teste$Attrition, previsoes_finais_v2)

# Terceira versão do modelo com dados de treino e sem variáveis de educação
modelo_v3 <- glm(Attrition ~ Age + Department + DistanceFromHome + `Employee Source` +
                   JobRole + MaritalStatus + AverageTenure + PriorYearsOfExperience + Gender,
                 family = binomial,
                 data = dados_rh_1_treino)

summary(modelo_v3)
vif(modelo_v3)

# Previsões
threshold <- 0.5
previsoes_v3 <- predict(modelo_v3, type = 'response', newdata = dados_rh_1_teste)
previsoes_finais_v3 <- ifelse(previsoes_v3 > threshold, 'Voluntary Resignation', 'Current employee')
table(dados_rh_1_teste$Attrition, previsoes_finais_v3)

# Quarta versão do modelo com dados de treino e sem variáveis de educação e genero
modelo_v4 <- glm(Attrition ~ Age + Department + DistanceFromHome + `Employee Source` +
                   JobRole + MaritalStatus + AverageTenure + PriorYearsOfExperience,
                 family = binomial,
                 data = dados_rh_1_treino)

summary(modelo_v4)
vif(modelo_v4)

# Previsões
threshold <- 0.5
previsoes_v4 <- predict(modelo_v4, type = 'response', newdata = dados_rh_1_teste)
previsoes_finais_v4 <- ifelse(previsoes_v4 > threshold, 'Voluntary Resignation', 'Current employee')
table(dados_rh_1_teste$Attrition, previsoes_finais_v4)

# Quinta versão do modelo com dados de treino e sem variáveis de educação, genero e outro algoritmo
?rpart
modelo_v5 <- rpart(Attrition ~ Age + Department + DistanceFromHome + JobRole + MaritalStatus +
                     AverageTenure + PriorYearsOfExperience,
                   method = "class",
                   control = rpart.control(minsplit = 500, cp = 0),
                   data = dados_rh_1_treino)

summary(modelo_v5)
```

```
rpart.plot(modelo_v5)


# Fim
```

8.9. Health Analytics

Definição do Problema e Fonte de Dados

O objetivo deste projeto é tentar prever o tempo de sobrevivência de um paciente um ano após o transplante de fígado. Usaremos dados reais disponibilizados publicamente.

Há normalmente uma grande fila de pacientes esperando em um determinado momento para receber um transplante de fígado, pois não há outra cura para o estágio final de doença hepática. Nosso estudo visa ajudar os pacientes a compreender melhor suas chances de sobrevivência após um ano do transplante. As informações do nosso estudo mostram que somos capazes de prever o tempo de sobrevivência com razoável precisão para diferentes intervalos de tempo. Nosso modelo pode auxiliar os pacientes no processo de tomada de decisão (como financeiro por exemplo), enquanto eles esperam na fila para fazer um transplante.

Usando dados do Registro Científico de Recipientes de Transplante (Scientic Registry of Transplant Recipients - SRTR), trabalharemos com variáveis quantitativas e qualitativas criando modelos preditivos usando o tempo de sobrevivência do paciente como nossa variável de resposta (ou seja, teremos um problema de regressão, cujo objetivo é prever um valor numérico).

Usaremos duas técnicas de regressão: regressão linear e rede neural. Ambos os modelos previram o tempo de sobrevida do paciente por 1 a 3 anos após o transplante, usando um conjunto de variáveis mais significativas. As variáveis foram escolhidas após uma pesquisa sobre quais são os fatores mais importantes no processo de transplante de fígado.

Nossos resultados nos dão a certeza de que podemos prever com razoável precisão o tempo de sobrevivência um ano após os pacientes receberem um transplante de fígado. Acreditamos que isso dará aos pacientes uma ideia melhor de suas mudanças de sobrevida por um período prolongado de tempo após o transplante. Usando as informações desses modelos, os pacientes podem tomar uma decisão mais informada quando se trata de transplante de órgão com base em sua condição médica.

Os dados foram extraídos do SRTR Database e modificados para que os alunos pudessem executar o script em suas máquinas. Script e dataset podem ser encontrados ao final do capítulo. Site oficial dos dados:

https://www.srtr.org/about-the-data/the-srtr-database/

Os dados originais devem ser solicitados no link abaixo e a aprovação pode levar até 30 dias, sendo o objetivo estudo e pesquisa:

https://www.srtr.org/requesting-srtr-data/data-requests/

MELD Score

A pontuação MELD (MELD Score - Mayo End-Stage Liver Disease) é uma métrica de quão doente um paciente está. Pontuação MELD

Se você é um adulto com doença hepática que pode exigir um transplante, sua pontuação MELD ajuda a dizer com que rapidez você pode precisar dele.

MELD significa “modelo para doença hepática em estágio terminal”. Os médicos usam um sistema semelhante, chamado PELD (doença hepática em estágio terminal pediátrico), para crianças menores de 12 anos.

O MELD score é um número que varia de 6 a 40, com base em testes de laboratório. Ele classifica o grau de doença, que mostra o quanto você precisa de um transplante de fígado. Quanto maior o número, mais urgente é o seu caso.

Referências:

https://www.mdcalc.com/meld-score-model-end-stage-liver-disease-12-older
https://www.webmd.com/hepatitis/meld-score-for-liver-disease
https://optn.transplant.hrsa.gov/resources/allocation-calculators/meld-calculator/

Projeto – Podemos Prever o Tempo de Sobrevivência dos Pacientes 1 Ano Após Receberem um Transplante?

```
# Health Analytics

# Projeto - Podemos Prever o Tempo de Sobrevivência dos Pacientes 1 Ano Após Receberem um Transplante?

# Leia a definição do projeto e detalhes sobre a fonte de dados no manual em pdf no Capítulo 9.

# Diretório de trabalho
setwd("~/Dropbox/DSA/Business-Analytics2.0/Cap09/R")
getwd()

# Pacotes
library(dplyr)
library(ggcorrplot)
library(forecast)
library(nnet)
library(neuralnet)

# Carregando os dados
dados <- read.csv("dados/dataset.csv", header = TRUE, na.strings = c(""))
dim(dados)

# Análise Exploratória, Limpeza, Transformação e Manipulação de Dados (Data Wrangling)

# Visualizando os dados
View(dados)

# Tipos dos dados
str(dados)

# Explorando os dados das variáveis numéricas
hist(dados$AGE)
hist(dados$AGE_DON)
hist(dados$PTIME)
hist(dados$DAYSWAIT_CHRON)
hist(dados$FINAL_MELD_SCORE)

# Explorando os dados das variáveis categóricas
dados$DIAB <- as.factor(dados$DIAB)
table(dados$DIAB)

dados$PSTATUS <- as.factor(dados$PSTATUS)
table(dados$PSTATUS)

dados$GENDER <- as.factor(dados$GENDER)
dados$GENDER_DON <- as.factor(dados$GENDER_DON)
table(dados$GENDER)
table(dados$GENDER_DON)

dados$REGION <- as.factor(dados$REGION)
table(dados$REGION)

dados$TX_Year <- as.factor(dados$TX_Year)
table(dados$TX_Year)
```

```
dados$MALIG <- as.factor(dados$MALIG)
table(dados$MALIG)

dados$HIST_CANCER_DON <- as.factor(dados$HIST_CANCER_DON)
table(dados$HIST_CANCER_DON)

# Considerando apenas os pacientes que sobreviveram ao primeiro ano de cirurgia
dados1 <- dados %>%
  filter(PTIME > 365) %>%
  mutate(PTIME = (PTIME - 365))

dim(dados1)

# Dos pacientes que sobreviveram ao primeiro ano da cirurgia,
# filtramos os que permaneceram vivos por até três anos depois da cirurgia.
dados2 <- dados1 %>%
  filter(PTIME <= 1095)

dim(dados2)
View(dados2)

# Vamos separar variáveis numéricas e categóricas
dados_num <- dados2[,!unlist(lapply(dados2, is.factor))]
dim(dados_num)

dados_fator <- dados2[,unlist(lapply(dados2, is.factor))]
dim(dados_fator)

# Correlação entre as variáveis numéricas
# Para variáveis categóricas usamos associação
df_corr <- round(cor(dados_num, use = "complete.obs"), 2)
?ggcorrplot
ggcorrplot(df_corr)

# Padronização das variáveis numéricas e combinação em um novo dataframe
# com as variáveis categóricas

# Padronização
dados_num_norm <- scale(dados_num)
dados_final <- cbind(dados_num_norm, dados_fator)
dim(dados_final)
View(dados_final)

# Divisão dos dados em treino e teste
set.seed(1)
index <- sample(1:nrow(dados_final), dim(dados_final)[1]*.7)
dados_treino <- dados_final[index,]
dados_teste <- dados_final[-index,]

# Remove os registros dos anos de 2001 e 2002 (pois foram os primeiros anos da coleta de dados)
dados_treino <- dados_treino %>%
  filter(TX_Year != 2001) %>%
  filter(TX_Year != 2002)

dados_teste <- dados_teste %>%
  filter(TX_Year != 2001) %>%
  filter(TX_Year != 2002)

# Modelagem Preditiva com Modelo de Regressão

# Vamos trabalhar apenas com algumas variáveis mais significativas para o problema.
# Isso também reduz o tempo total de treinamento.
?lm
modelo_v1 <- lm(PTIME ~ FINAL_MELD_SCORE +
                  REGION +
                  LiverSize +
                  LiverSizeDon +
                  ALCOHOL_HEAVY_DON +
                  MALIG +
                  TX_Year,
                data = dados_treino)

summary(modelo_v1)

# Avaliação do modelo

# Com dados de treino
```

```
modelo_v1_pred_1 = predict(modelo_v1, newdata = dados_treino)
?accuracy
accuracy(modelo_v1_pred_1, dados_treino$PTIME)

# Com dados de teste
modelo_v1_pred_2 = predict(modelo_v1, newdata = dados_teste)
accuracy(modelo_v1_pred_2, dados_teste$PTIME)

# Distribuição do erro de validação
par(mfrow = c(1,1))
residuos <- dados_teste$PTIME - modelo_v1_pred_2
hist(residuos, xlab = "Resíduos", main = "Sobreviventes de 1 a 3 Anos")

# Agora vamos padronizar os dados de treino e teste de forma separada.
# Executamos o procedimento anterior, mas de forma separada em cada subset.
set.seed(1)
index <- sample(1:nrow(dados2), dim(dados2)[1]*.7)
dados_treino <- dados2[index,]
dados_teste <- dados2[-index,]

# Vamos separar variáveis numéricas e categóricas (treino)
dados_treino_num <- dados_treino[,!unlist(lapply(dados_treino, is.factor))]
dim(dados_treino_num)

dados_treino_fator <- dados_treino[,unlist(lapply(dados_treino, is.factor))]
dim(dados_treino_fator)

# Vamos separar variáveis numéricas e categóricas (teste)
dados_teste_num <- dados_teste[,!unlist(lapply(dados_teste, is.factor))]
dim(dados_teste_num)

dados_teste_fator <- dados_teste[,unlist(lapply(dados_teste, is.factor))]
dim(dados_teste_fator)

# Padronização
dados_treino_num_norm <- scale(dados_treino_num)
dados_treino_final <- cbind(dados_treino_num_norm, dados_treino_fator)
dim(dados_treino_final)

# Padronização
dados_teste_num_norm <- scale(dados_teste_num)
dados_teste_final <- cbind(dados_teste_num_norm, dados_teste_fator)
dim(dados_teste_final)

# Filtra os anos de 2001 e 2002
dados_treino_final <- dados_treino_final %>%
  filter(TX_Year != 2001) %>%
  filter(TX_Year != 2002)

dados_teste_final <- dados_teste_final %>%
  filter(TX_Year != 2001) %>%
  filter(TX_Year != 2002)

# Cria novamente o modelo agora com o outro dataset de treino
modelo_v1 <- lm(PTIME ~ FINAL_MELD_SCORE +
                  REGION +
                  LiverSize +
                  LiverSizeDon +
                  ALCOHOL_HEAVY_DON +
                  MALIG +
                  TX_Year,
                data = dados_treino_final)

summary(modelo_v1)

# Avaliação do modelo

# Com dados de treino
modelo_v1_pred_1 = predict(modelo_v1, newdata = dados_treino_final)
accuracy(modelo_v1_pred_1, dados_treino_final$PTIME)

# Com dados de teste
modelo_v1_pred_2 = predict(modelo_v1, newdata = dados_teste_final)
accuracy(modelo_v1_pred_2, dados_teste_final$PTIME)

# Distribuição do erro de validação
par(mfrow = c(1,1))
residuos <- dados_teste_final$PTIME - modelo_v1_pred_2
```

```
hist(residuos, xlab = "Resíduos", main = "Sobreviventes de 1 a 3 Anos")

# Vamos desfazer a escala dos dados
variaveis_amostra <- c("PTIME",
                       "FINAL_MELD_SCORE",
                       "REGION",
                       "LiverSize",
                       "LiverSizeDon",
                       "ALCOHOL_HEAVY_DON",
                       "MALIG",
                       "TX_Year")

# Removemos valores NA das variáveis que usaremos para aplicar o unscale
dados_unscale <- na.omit(dados2[,variaveis_amostra])

# Retorna os dados unscale
dados_final_unscale <- dados_unscale[-index,] %>%
  filter(TX_Year!= 2001) %>%
  filter(TX_Year!= 2002)

# Histograma dos dados sem escala (formato original)
previsoes = predict(modelo_v1, newdata = dados_final_unscale)
hist(previsoes)
accuracy(previsoes, dados_final_unscale$PTIME)

# Modelagem Preditiva com Modelo de Rede Neural

# Preparação dos dados
dados_final2 <- na.omit(dados_final[,variaveis_amostra])
dim(dados_final2)
str(dados_final2)

# Retorna somente as variáveis que não são do tipo fator
variaveis_numericas <- !unlist(lapply(dados_final2, is.factor))
View(variaveis_numericas)

# Retorna o nome das variáveis numéricas
variaveis_numericas_nomes <- names(dados_final2[,!unlist(lapply(dados_final2, is.factor))])
View(variaveis_numericas_nomes)

# Gera o dataframe final com variáveis dummy
?class.ind
df_final = cbind(dados_final2[,variaveis_numericas],
                 class.ind(dados_final2$REGION),
                 class.ind(dados_final2$ALCOHOL_HEAVY_DON),
                 class.ind(dados_final2$MALIG),
                 class.ind(dados_final2$TX_Year))

dim(df_final)
View(df_final)

# Nomes das variáveis
names(df_final) = c(variaveis_numericas_nomes,
                    paste("REGION", c(1:11),sep = ""),
                    paste("ALCOHOL_HEAVY_DON", c(1:3),sep = ""),
                    paste("MALIG", c(1:3), sep = ""),
                    paste("LISTYR", c(01:18), sep = ""))

dim(df_final)
View(df_final)

# Divisão em dados de treino e teste
index2 <- sample(1:nrow(df_final), dim(df_final)[1]*.70)
dados_treino2 <- df_final[index2,]
dados_teste2 <- df_final[-index2,]
print(dados_teste2[1,])

# Modelo
# www.deeplearningbook.com.br
?neuralnet
modelo_v2 <- neuralnet::neuralnet(PTIME ~ .,
                                  data = dados_treino2,
                                  linear.output = TRUE,
                                  hidden = 2,
                                  stepmax = 1e7)

# Plot
plot(modelo_v2,
```

```
    col.entry.synapse = "red",
    col.entry = "brown",
    col.hidden = "green",
    col.hidden.synapse = "black",
    col.out = "yellow",
    col.out.synapse = "purple",
    col.intercept = "green",
    fontsize = 10,
    show.weights = TRUE ,
    rep = "best")

# Avaliação do modelo

# Com dados de treino
modelo_v2_pred_1 <- compute(modelo_v2, dados_treino2)
accuracy(unlist(modelo_v2_pred_1), dados_treino2$PTIME)

# Com dados de teste
modelo_v2_pred_2 <- compute(modelo_v2, dados_teste2)
accuracy(unlist(modelo_v2_pred_2), dados_teste2$PTIME)

# Conclusão: O modelo de regressão linear apresentou uma taxa de erro menor e, portanto,
# deve ser usado como versão final.

# Sim, conseguimos prever o tempo de sobrevivência dos pacientes 1 ano após receberem um transplante.

# Fim
```

8.10. Financial Analytics

O Que é Financial Analytics?

Financial Analytics envolve a aplicação de modelos estatísticos clássicos e algoritmos de Machine Learning em dados do mercado financeiro e carteiras de investimento.

Financial Analytics é uma área multidisciplinar em sua combinação de estatística, finanças e ciência da computação.

Exemplos de Questões de Simulações Analíticas Financeiras

Estes são alguns exemplos de tipos de questões a serem abordadas por simulações analíticas financeiras:

• Qual é o retorno otimizado e qual é o nível de risco assumido?
• Que tipos de métricas financeiras podem se tornar boas variáveis aleatórias e como elas são distribuídas?
• Que conjuntos de dados estão disponíveis para amostrar estas variáveis aleatórias, analiticamente?
• Quais métricas financeiras estão altamente correlacionadas?
• Quais são relativamente independentes?
• O resultado de um algoritmo pode ser uma vantagem sobre uma estratégia simples de retenção, ao gerar transações?

Essas perguntas devem ser feitas no início de qualquer projeto de Financial Analytics e empregamos Data Science para respondê-las.

As Finanças no Século XXI

No ambiente de negócios em constante mudança de hoje, os executivos estão explorando maneiras pelas quais a função financeira pode trazer maior valor para suas organizações. Para este fim, eles estão transformando suas organizações e se concentrando principalmente em relatórios mais eficazes, fornecendo as informações que a gestão interna precisa para "gerir" de forma mais eficiente o negócio.

Os executivos agora devem pensar além das informações financeiras tradicionais contidas nos sistemas de contabilidade geral e considerar a melhor forma de fornecer os insights gerados a partir de métodos analíticos necessários para conduzir decisões em empresas em um ambiente de negócios cada vez mais complexo e dinâmico.

Para atingir esses objetivos, áreas contábeis, financeiras, fiscais e outras áreas financeiras estão sendo diretamente beneficiadas, por ambientes de Big Data combinados com análises avançadas para atender às necessidades de toda a empresa. Nos referimos a esta capacidade avançada de suporte à decisão para finanças, como Financial Analytics.

A evolução da análise financeira tem sido impulsionada pelo surgimento de novos modelos de negócios, a mudança do papel do departamento financeiro tradicional, modificações nos processos de negócios e avanços tecnológicos. Este ambiente dinâmico apresenta a função de finanças enormes oportunidades e desafios.

B2B x B2C x B2E e Financial Analytics

Com a introdução da Internet duas décadas atrás, surgiram três novos modelos de ebusiness: business-to-business (B2B), business-to-consumer (B2C) e business-to-employee (B2E).

Estes novos modelos estão moldando o futuro da análise financeira, pois exigem uma conexão translúcida e fluidez de informações entre os departamentos dentro de qualquer empresa. Além desses novos modelos de negócios, vem ocorrendo uma mudança fundamental nos valores, desde ativos físicos até ativos intangíveis como patentes, marcas registradas, franquias, programas de computador, pesquisa e desenvolvimento, inteligência de negócios e relacionamentos. Consequentemente, o valor da informação está aumentando.

No modelo B2C por exemplo, muitas organizações que são puramente B2C não desenvolvem os produtos ou serviços que vendem. Ao invés disso, essas empresas terceirizam sua fabricação e outras operações não essenciais. Elas concentram seus recursos no desenvolvimento de uma marca, gerenciando uma rede de clientes e outros aspectos diferenciados da gestão do negócio. Dito isto, estes objetivos só podem ser alcançados através da utilização mais eficaz da informação. Ou seja, o novo ambiente de negócios do século XXI.

O Financial Analytics tem tradicionalmente focado em como uma empresa utiliza ativos tangíveis, como dinheiro, imóveis, máquinas, etc. No entanto, muitas empresas concorrentes na nova economia eletrônica são valorizadas com base em seus ativos intangíveis. Estes ativos cada vez mais importantes são muitas vezes difíceis de medir e gerir. Como resultado, empresas de negócios online estão confiando em Financial Analytics para ajudá-las a:

• Compreender o desempenho geral da organização.
• Identificar maneiras de medir e maximizar o valor dos ativos intangíveis.
• Reduzir os custos operacionais e gerenciar com eficiência os investimentos em toda a empresa.
• Antecipar variações no mercado.
• Otimizar as capacidades dos sistemas de informação.
• Melhorar os processos de negócios.

Assim como a economia continua a evoluir, o mesmo acontece com o papel da função de finanças dentro de uma organização.

Guiado por investimentos em planejamento de recursos empresariais, serviços compartilhados e mudanças em sua função de relatório, a maioria das funções financeiras estão se tornando mais eficientes - exigindo menos recursos para gerenciá-los e alinhando-se estreitamente com a estrutura de negócios da empresa. Isto é especialmente verdadeiro na área de processamento de transações onde a automação das transações financeiras permitiu que o pessoal de finanças expandisse seu papel e passasse mais tempo apoiando processos de tomada de decisão, em vez de apenas processar e reconciliar transações.

Cada vez mais organizações globais estão integrando e padronizando seus processos e sistemas de negócios, permitindo que os usuários finais com funções financeiras ou não, atualizem e obtenham informações financeiras de qualquer localização geográfica.

Isso melhorou significativamente o suporte à decisão dentro da organização. Mais uma vez, gestão do século XXI.

Processos de Negócios e Customer Value Management

À medida que os processos de negócios evoluem e as questões de negócios se tornam mais complexas, as análises necessárias para responder e agir sobre essas questões exigem um maior nível de integração de dados e colaboração organizacional. Historicamente, os departamentos de finanças foram muitas vezes os únicos departamentos com acesso a informações precisas sobre os resultados financeiros de uma empresa. No entanto, esta informação foi geralmente a um nível agregado e não estava disponível até vários dias, às vezes semanas, após o final do mês.

Durante a última década, no entanto, as empresas têm buscado integrar os processos de back-office e os fluxos de informação em toda a empresa, substituindo sistemas legados baseados em funções por um único sistema ERP, reengenharia de processos de negócio e simplificação de transações comerciais. Isso permitiu que executivos e gerentes acessassem informações financeiras detalhadas, bem como não-financeiras, mais precisas e consistentes, sobre a organização ao longo do mês.

Em meados dos anos noventa, novos produtos de software capazes de maximizar o valor da Internet foram introduzidos no mercado. As empresas começaram a implementar gerenciamento de cadeia de suprimentos (Supply Chain Management), gerenciamento de relacionamento com clientes (Customer Relationship Management) e outras soluções de sistemas sofisticados para otimizar suas operações de ponta a ponta. Ao mesmo tempo, as organizações - cada uma com seus próprios sistemas legados - começaram a fortalecer suas relações com clientes e fornecedores. Tudo isso contribuiu para o novo desafio que as organizações enfrentam hoje: um ambiente de informação complexo que obriga as organizações a adotar um novo nível de integração em toda a cadeia de valor.

E a evolução do Financial Analytics, permitiu unir dados dessas diferentes fontes, a fim de permitir uma gestão mais eficiente. Por exemplo, ao combinar as medidas financeiras tradicionais (receita e custo) com informações de CRM (histórico do cliente) e aplicar ferramentas e técnicas de modelagem preditiva, as empresas agora podem projetar a lucratividade futura associada a um cliente individual ou conjunto de clientes. Referimo-nos a isso como gestão de valor do cliente (ou CVM – Customer Value Management). A CVM permite às organizações monitorar continuamente o valor de cada cliente para o negócio e agir em conformidade.

O Que é Uma Criptomoeda?

Uma criptomoeda é um meio de troca, podendo ser centralizado ou descentralizado que se utiliza da tecnologia Blockchain e da criptografia para assegurar a validade das transações e a criação de novas unidades da moeda.

O Bitcoin, a primeira criptomoeda descentralizada, foi criado em 2009 por um usuário que usou o pseudônimo Satoshi Nakamoto. Desde então, muitas outras criptomoedas foram criadas. Mais recentemente, tem-se assistido a um fenômeno de explosão de inúmeros tokens que têm sido criados com base no protocolo Ethereum, principalmente após a onda massiva de Ofertas Iniciais de Moedas (usualmente referida como ICO, do inglês Initial Coin Offering) que ocorreu em 2017.

Ao contrário de sistemas bancários centralizados, grande parte das criptomoedas usam um sistema de controle descentralizado com base na tecnologia Blockchain, que é um tipo de livroregistro distribuído operado em uma rede ponto-a-ponto (peer-to-peer) de milhares de computadores, onde todos possuem uma cópia igual de todo o histórico de transações, impedindo que uma entidade central promova alterações no registro ou no software unilateralmente sem ser excluída da rede.

O Que é Uma Cryptocurrency Exchange?

Uma Cryptocurrency Exchange, ou uma Digital Currency Exchange (DCE), é uma empresa que permite aos clientes negociar criptomoedas ou moedas digitais por outros ativos, como moeda fiduciária convencional ou outras moedas digitais. Em português costuma ser chamado de “Bolsa de Criptomoedas”.

As Exchanges podem aceitar pagamentos com cartão de crédito, transferências eletrônicas ou outras formas de pagamento em troca de moedas digitais ou criptomoedas. Uma Cryptocurrency Exchange pode receber os spreads de compra e venda com uma comissão de transação por seu serviço ou como uma plataforma de correspondência, simplesmente cobrando taxas.

Algumas corretoras que também se concentram em outros ativos, como ações, permitem que os usuários comprem, mas não retirem criptomoedas das carteiras de criptomoeda (exemplo como Robinhood e eToro). No entanto, as bolsas de criptomoedas dedicadas, como Binance e Coinbase, permitem retiradas de criptomoedas.

As Exchanges podem enviar criptomoedas para a carteira pessoal de criptomoedas de um usuário. Algumas podem converter saldos de moeda digital em cartões pré-pagos anônimos que podem ser usados para retirar fundos de caixas eletrônicos em todo o mundo, enquanto outras moedas digitais são apoiadas por commodities do mundo real, como ouro.

Mini-Projeto 1 – AI Bot Trader – Robô Investidor Para Recomendação de Compra e Venda de Criptomoedas

Nosso Bot vai analisar dados financeiros históricos e recomendar a quantidade de criptomoedas que devemos comprar e vender para otimizar nosso portfólio e recomendar quando isso deve ser feito.

Este é um projeto completo e funcional, mas uma versão mais avançada deste projeto está disponível no curso de Inteligência Artificial Aplicada a Finanças da Formação Engenheiro Blockchain.

Este mini-projeto é em Linguagem Python. Os scripts estão ao final do capítulo.

```
# <font color='blue'>Data Science Academy</font>
# <font color='blue'>Business Analytics</font>
# <font color='blue'>Capítulo 10 - Financial Analytics</font>
## <font color='blue'>Mini-Projeto 1</font>
### <font color='blue'>AI Bot Trader - Robô Investidor Para Recomendação de Compra e Venda de Criptomoedas</font>
# ->
# Versão da Linguagem Python
from platform import python_version
print('Versão da Linguagem Python Usada Neste Jupyter Notebook:', python_version())

![title](imagens/mini-projeto1.png)
## Avisos Antes de Começar
- 1- Nosso objetivo aqui é didático! Não use essa aplicação para fins comerciais sem antes realizar muitos testes.
- 2- A execução do processo de treinamento pode ser bastante demorada dependendo do hardware do seu computador.
- 3- Experimente a aplicação com outras técnicas.
- 4- Este projeto é completo e funcional, mas uma versão mais avançada pode ser encontrada no curso de IA Para Finanças da Formação Engenheiro Blockchain.
- 5- Se executar muitas vezes seguidas este projeto seu acesso à API pode ser bloqueado. Estamos fornecendo o arquivo de dados extraídos no dia da gravação das aulas.
## Definição do Problema e Fonte de Dados
Leia o manual em pdf no Capítulo 10.
## Etapas do Projeto
- Parte 1 - Extração dos Dados em Tempo Real
- Parte 2 - Análise de Dados
- Parte 3 - Construção do Modelo e Otimização Bayesiana
- Parte 4 - Execução do AI Bot Trader
- Parte 5 - Conclusão e Próximos Passos
## Instalando e Carregando os Pacotes
```

```
# ->
# Para atualizar um pacote, execute o comando abaixo no terminal ou prompt de comando:
# pip install -U nome_pacote
# Para instalar a versão exata de um pacote, execute o comando abaixo no terminal ou prompt de comando:
# !pip install nome_pacote==versão_desejada
# Depois de instalar ou atualizar o pacote, reinicie o jupyter notebook.
# Instala o pacote watermark.
# Esse pacote é usado para gravar as versões de outros pacotes usados neste jupyter notebook.
!pip install -q -U watermark

# ->
# CryptoCurrency eXchange Trading Library
# https://pypi.org/project/ccxt/
!pip install -q ccxt

# ->
# https://pypi.org/project/bayesian-optimization/
!pip install -q bayesian-optimization==1.2

# ->
# Imports
import csv
import ccxt
import time
import random
import types
import pkg_resources
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from bayes_opt import BayesianOptimization
from pprint import pprint
from datetime import datetime
sns.set()

# ->
# Versões dos pacotes usados neste jupyter notebook
%reload_ext watermark
%watermark -a "Data Science Academy" --iversions

## Parte 1 - Extração dos Dados em Tempo Real
### Função de Gravação dos Dados
# ->
# Função para salvar dados em formato csv
def grava_csv(arquivo, dados):
    # Abre o arquivo para escrita
    with open(arquivo, mode = 'w') as arquivo_saida:
        # Gera o cabeçalho
        arquivo_saida.write("Date,Open,High,Low,Close,Adj Close,Volume\n")
        # Grava os dados
        csv_writer = csv.writer(arquivo_saida, delimiter = ',', quotechar = "", quoting = csv.QUOTE_MINIMAL)
        csv_writer.writerows(dados)

### Função de Conexão à Operadora de Criptomoeda
# ->
# Função para fazer conexão à exchange para extração dos dados
# https://www.bitmex.com/
# https://www.bitmex.com/app/apiOverview
def conecta_exchange(exchange, max_retries, symbol, timeframe, since, limit):
    # Zera o número de tentativas
    num_retries = 0
    # Tenta fazer a conexão
    try:
        num_retries += 1
        ohlcv = exchange.fetch_ohlcv(symbol, timeframe, since)
        return ohlcv
    except Exception:
        if num_retries > max_retries:
            raise

### Funções Para Extração dos Dados
# ->
# Função para extração dos dados
def extrai_dados(exchange, max_retries, symbol, timeframe, since, limit):
    # Timestamp
    earliest_timestamp = exchange.milliseconds()
    # Duração da janela em segundos
```

```
    timeframe_duration_in_seconds = exchange.parse_timeframe(timeframe)
    # Duração da janela em milisegundos
    timeframe_duration_in_ms = timeframe_duration_in_seconds * 1000
    # Diferença de tempo
    timedelta = limit * timeframe_duration_in_ms
    # Lista para os dados
    all_ohlcv = []
    # Loop
    while True:
        # Data de início para extração dos dados
        fetch_since = earliest_timestamp - timedelta
        # Conecta na exchange e extrai os dados
        ohlcv = conecta_exchange(exchange, max_retries, symbol, timeframe, fetch_since, limit)
        # Se alcançamos o limite, finaliza o loop
        if ohlcv[0][0] >= earliest_timestamp:
            break
        # Atualiza o tempo mais cedo
        earliest_timestamp = ohlcv[0][0]
        # Atualiza os dados
        all_ohlcv = ohlcv + all_ohlcv
        # Print do andamento
        print(len(all_ohlcv), 'registros extraídos de', exchange.iso8601(all_ohlcv[0][0]), 'a', exchange.iso8601(all_ohlcv[-1][0]))
        if fetch_since < since:
            break
    return all_ohlcv

# ->
# Função para extrair os dados e salvar em formato csv
def extrai_dados_para_csv(filename, exchange_id, max_retries, symbol, timeframe, since, limit):
    # Obtém o id da exchange com o pacote ccxt
    exchange = getattr(ccxt, exchange_id)({'enableRateLimit': True,})
    # Checa a consistência
    if isinstance(since, str):
        since = exchange.parse8601(since)
    # Extrai o que está sendo comercializado
    exchange.load_markets()
    # Extrai os dados
    ohlcv = extrai_dados(exchange, max_retries, symbol, timeframe, since, limit)
    # Contador
    key = 0
    # Loop
    for item in ohlcv:
        epoch = int(item[0]) / 1000
        ohlcv[key][0] = datetime.utcfromtimestamp(epoch).strftime('%Y-%m-%d')
        ohlcv[key][5] = int(item[5])
        ohlcv[key].append(ohlcv[key][5])
        ohlcv[key][5] = ohlcv[key][4]
        key += 1
    # Comprimento de dados extraídos
    ohlen = len(ohlcv)
    # Print do andamento
    pprint("Número de Registros: " + str(ohlen))
    # Vamos manter um limite para os dados
    if ohlen > 399:
        ohrem = ohlen - 399
        pprint("Removendo: " + str(ohrem))
        ohlcv = ohlcv[ohrem:]
    # Grava os dados em csv
    grava_csv(filename, ohlcv)
    # Print
    print('Salvos', len(ohlcv), 'registros no arquivo', filename)

### Parâmetros Para Extração dos Dados
# ->
# Define os parâmetros de extração dos dados
# Exchange: https://www.bitmex.com/app/apiOverview
exchange = "bitmex"
# Símbolo da criptomoeda
simbolo = "BTC/USD"
# Janela
janela = "1d"
# Data de início
data_inicio = "2018-01-01T00:00:00Z"
# Arquivo de saída
outfile = "dados/dataset.csv"

### Extração dos Dados
# ->
```

```
# Executa a extração dos dados
extrai_dados_para_csv(outfile, exchange, 3, simbolo, janela, data_inicio, 100)

## Parte 2 - Análise de Dados
### Carregando e Explorando o Arquivo de Dados
# ->
# Carregando o arquivo do disco
df = pd.read_csv(outfile)

# ->
# Dados
df.head()

# ->
# Shape
df.shape

# ->
# Dados de fechamento
close = df.Close.values.tolist()

# ->
# Outros parâmetros para a versão base do modelo
window_size = 30
skip = 5
l = len(close) - 1

## Parte 3 - Construção do Modelo e Otimização Bayesiana
A parte 3 considera que você tem algum conhecimento em Aprendizado Por Reforço. Esse conhecimento pode ser obtido no curso de Deep Learning II em nível avançado ou em nível básico no curso gratuito de Python Fundamentos aqui na DSA.
### Estratégia de Treinamento
https://arxiv.org/abs/1712.06560
https://openai.com/blog/evolution-strategies/
https://gist.github.com/karpathy/77fbb6a8dac5395f1b73e7a89300318d
http://www.deeplearningbook.com.br/
# ->
# Classe para a estratégia de treinamento
# Usamos Deep Evolution Strategy do OpenAI
class PoliticaTrader:
    # Inputs
    inputs = None
    # Construtor
    def __init__(self, weights, reward_function, population_size, sigma, learning_rate):
        # Inicializa os atributos da classe
        self.weights = weights
        self.reward_function = reward_function
        self.population_size = population_size
        self.sigma = sigma
        self.learning_rate = learning_rate
    # Obtém o peso a partir da população
    def get_weights_population(self, weights, population):
        # Lista para os pesos
        weights_population = []
        # Loop pela população
        for index, i in enumerate(population):
            jittered = self.sigma * i
            weights_population.append(weights[index] + jittered)
        return weights_population
    # Obtém os pesos
    def get_weights(self):
        return self.weights
    # Treinamento
    def treinamento(self, epoch = 100, print_every = 1):
        # Time
        lasttime = time.time()
        # Loop pelas épocas
        for i in range(epoch):
            # Lista para a população
            population = []
            # Recompensas
            rewards = np.zeros(self.population_size)
            # Loop pelo population_size
            for k in range(self.population_size):
                x = []
                # Loop
                for w in self.weights:
                    x.append(np.random.randn(*w.shape))
                population.append(x)
```

```
            # Loop
            for k in range(self.population_size):
                weights_population = self.get_weights_population(self.weights, population[k])
                rewards[k] = self.reward_function(weights_population)
            # Recompensas
            rewards = (rewards - np.mean(rewards)) / np.std(rewards)
            # Loop
            for index, w in enumerate(self.weights):
                A = np.array([p[index] for p in population])
                # Pesos da rede neural
                self.weights[index] = (w + self.learning_rate / (self.population_size * self.sigma) * np.dot(A.T, rewards).T)
            if (i + 1) % print_every == 0:
                print('Iteração %d. Recompensa: %f' % (i + 1, self.reward_function(self.weights)))

        print('Tempo Total de Treinamento:', time.time() - lasttime, 'segundos')

### Arquitetura do Modelo de Rede Neural
# ->
# Classe do Modelo
class Modelo:
    # Método construtor
    def __init__(self, input_size, layer_size, output_size):
        self.weights = [np.random.randn(input_size, layer_size),
                        np.random.randn(layer_size, output_size),
                        np.random.randn(layer_size, 1),
                        np.random.randn(1, layer_size),]
    # Função para previsão
    def predict(self, inputs):
        # Feed forward
        feed = np.dot(inputs, self.weights[0]) + self.weights[-1]
        # Decisão de compra (previsão)
        decision = np.dot(feed, self.weights[1])
        # Compra (decisão)
        buy = np.dot(feed, self.weights[2])
        return decision, buy
    def get_weights(self):
        return self.weights
    def set_weights(self, weights):
        self.weights = weights

### Configuração do AI Bot Trader
# ->
# Função para obter o estado dos dados
def get_state(data, t, n):
    d = t - n + 1
    block = data[d : t + 1] if d >= 0 else -d * [data[0]] + data[0 : t + 1]
    res = []
    for i in range(n - 1):
        res.append(block[i + 1] - block[i])
    return np.array([res])

# ->
# Classe para o agente inteligente (Trader)
class Trader:
    # Método construtor
    def __init__(self, population_size, sigma, learning_rate, model, money, max_buy, max_sell, skip, window_size,):
        # Inicializa os atributos
        self.window_size = window_size
        self.skip = skip
        self.POPULATION_SIZE = population_size
        self.SIGMA = sigma
        self.LEARNING_RATE = learning_rate
        self.model = model
        self.initial_money = money
        self.max_buy = max_buy
        self.max_sell = max_sell
        self.es = PoliticaTrader(self.model.get_weights(),
                                 self.get_reward,
                                 self.POPULATION_SIZE,
                                 self.SIGMA,
                                 self.LEARNING_RATE,)
    # Método de ação
    def agir(self, sequence):
        decision, buy = self.model.predict(np.array(sequence))
        return np.argmax(decision[0]), int(buy[0])
    # Método para obter recompensa
    def get_reward(self, weights):
        # Valor inicial investido
```

```
    initial_money = self.initial_money
    starting_money = initial_money
    # Pesos
    self.model.weights = weights
    # Estado
    state = get_state(close, 0, self.window_size + 1)
    # Objetos de controle
    inventory = []
    quantity = 0
    # Loop
    for t in range(0, l, self.skip):
        # Ação e compra/venda
        action, buy = self.agir(state)
        # Próximo estado
        next_state = get_state(close, t + 1, self.window_size + 1)
        # Verifica ação e valor inicial investido
        if action == 1 and initial_money >= close[t]:
            if buy < 0:
                buy = 1
            if buy > self.max_buy:
                buy_units = self.max_buy
            else:
                buy_units = buy
            total_buy = buy_units * close[t]
            initial_money -= total_buy
            inventory.append(total_buy)
            quantity += buy_units
        elif action == 2 and len(inventory) > 0:
            if quantity > self.max_sell:
                sell_units = self.max_sell
            else:
                sell_units = quantity
            quantity -= sell_units
            total_sell = sell_units * close[t]
            initial_money += total_sell
        # Próximo estado
        state = next_state
    return ((initial_money - starting_money) / starting_money) * 100
# Treinamento do Trader
def fit(self, iterations, checkpoint):
    self.es.treinamento(iterations, print_every = checkpoint)
# Método para recomendação
def investir(self):
    # Valor inicial
    initial_money = self.initial_money
    starting_money = initial_money
    # Estado
    state = get_state(close, 0, self.window_size + 1)
    # Listas de controle
    states_sell = []
    states_buy = []
    inventory = []
    quantity = 0
    # Loop
    for t in range(0, l, self.skip):
        # Ação e compra
        action, buy = self.agir(state)
        # Próximo estado
        next_state = get_state(close, t + 1, self.window_size + 1)
        # Verifica ação e valor inicial investido
        if action == 1 and initial_money >= close[t]:
            if buy < 0:
                buy = 1
            if buy > self.max_buy:
                buy_units = self.max_buy
            else:
                buy_units = buy
            total_buy = buy_units * close[t]
            initial_money -= total_buy
            inventory.append(total_buy)
            quantity += buy_units
            states_buy.append(t)
            print('Dia %d: comprar %d unidades ao preço de %f, saldo total %f' % (t, buy_units, total_buy, initial_money))
        elif action == 2 and len(inventory) > 0:
            bought_price = inventory.pop(0)
            if quantity > self.max_sell:
                sell_units = self.max_sell
            else:
```

```
                sell_units = quantity
            if sell_units < 1:
                continue
            quantity -= sell_units
            total_sell = sell_units * close[t]
            initial_money += total_sell
            states_sell.append(t)
            try:
                invest = ((total_sell - bought_price) / bought_price) * 100
            except:
                invest = 0
            print('Dia %d, vender %d unidades ao preço de %f, investimento %f %%, saldo total %f,' % (t, sell_units, total_sell, invest,
initial_money))
        # Próximo estado
        state = next_state
    # Investimento
    invest = ((initial_money - starting_money) / starting_money) * 100
    print('\nGanho Total %f, Valor Total Investido %f' % (initial_money - starting_money, invest))
    plt.figure(figsize = (20, 10))
    plt.plot(close, label = 'Valor Real de Fechamento', c = 'g')
    plt.plot(close, 'X', label = 'Previsão de Compra', markevery = states_buy, c = 'b')
    plt.plot(close, 'o', label = 'Previsão de Venda', markevery = states_sell, c = 'r')
    plt.legend()
    plt.show()

### Funções Para Buscar o Melhor Trader
# ->
# Função para encontrar o melhor trader
def melhor_trader(window_size, skip, population_size, sigma, learning_rate, size_network):
    # Cria o modelo
    model = Modelo(window_size, size_network, 3)
    # Cria o trader
    trader = Trader(population_size, sigma, learning_rate, model, 10000, 5, 5, skip, window_size,)
    # Treinamento
    try:
        trader.fit(100, 1000)
        return trader.es.reward_function(trader.es.weights)
    except:
        return 0

# ->
# Função para encontrar o melhor trader de acordo com os hiperparâmetros
def busca_melhor_trader(window_size, skip, population_size, sigma, learning_rate, size_network):
    # Variável global
    global accbest
    # Hiperparâmetros
    param = {'window_size': int(np.around(window_size)),
             'skip': int(np.around(skip)),
             'population_size': int(np.around(population_size)),
             'sigma': max(min(sigma, 1), 0.0001),
             'learning_rate': max(min(learning_rate, 0.5), 0.000001),
             'size_network': int(np.around(size_network)),}
    print('\nBuscando Parâmetros %s' % (param))
    # Investimento feito pelo melhor trader
    investment = melhor_trader(**param)
    print('Após 100 iterações o investimento foi de %f' % (investment))
    return investment

### Otimização Bayesiana Para os Hiperparâmetros do Modelo
# ->
# Modelo para otimização bayesiana de hiperparâmetros
otimizacao_bayesiana = BayesianOptimization(busca_melhor_trader, {'window_size': (2, 50),
                                                   'skip': (1, 15),
                                                   'population_size': (1, 50),
                                                   'sigma': (0.01, 0.99),
                                                   'learning_rate': (0.000001, 0.49),
                                                   'size_network': (10, 1000),},)

# ->
%%time
otimizacao_bayesiana.maximize(init_points = 30, n_iter = 50, acq = 'ei', xi = 0.0)

# ->
type(otimizacao_bayesiana)

# ->
# Visualiza o resultado
otimizacao_bayesiana.res
```

```
Vamos obter o maior valor para cada hiperparâmetro.
# ->
max([dic['target'] for dic in otimizacao_bayesiana.res])

# ->
[dic['params'] for dic in otimizacao_bayesiana.res]

# ->
max([d['learning_rate'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]])

# ->
max([d['population_size'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]])

# ->
max([d['sigma'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]])

# ->
max([d['size_network'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]])

# ->
max([d['skip'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]])

# ->
max([d['window_size'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]])

## Parte 4 - Execução do AI Bot Trader
### Execução do AI Bot Trader - Modelo Base
# ->
%%time
melhor_trader(window_size = 30,
              skip = 1,
              population_size = 15,
              sigma = 0.1,
              learning_rate = 0.03,
              size_network = 500)

# ->
# Cria o modelo
modelo_base = Modelo(input_size = 30, layer_size = 500, output_size = 3)

# ->
# Cria o trader
 base
trader_base = Trader(population_size = 15,
                     sigma = 0.1,
                     learning_rate = 0.03,
                     model = modelo_base,
                     money = 10000,
                     max_buy = 5,
                     max_sell = 5,
                     skip = 1,
                     window_size = 30)

Aqui treinamos o trader!
# ->
%%time
trader_base.fit(500, 100)

# ->
# Recomendações
trader_base.investir()

### Execução do AI Bot Trader - Modelo Otimizado
# ->
%%time
melhor_trader(window_size = int(np.around(max([d['window_size'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]]))),
              skip = int(np.around(max([d['skip'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]]))),
              population_size = int(max([d['population_size'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]])),
              sigma = max([d['sigma'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]]),
              learning_rate = max([d['learning_rate'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]]),
              size_network = int(np.around(max([d['size_network'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]]))))

# ->
%%time
modelo_otim = Modelo(input_size = int(np.around(max([d['window_size'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]]))),
                     layer_size = int(np.around(max([d['size_network'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]]))),
                     output_size = 3)
```

```
# ->
# Cria o trader com otimização
trader_otim = Trader(population_size = int(np.around(max([d['population_size'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]]))),
                     sigma = max([d['sigma'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]]),
                     learning_rate = max([d['learning_rate'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]]),
                     model = modelo_otim,
                     money = 10000,
                     max_buy = 5,
                     max_sell = 5,
                     skip = int(np.around(max([d['skip'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]]))),
                     window_size = int(np.around(max([d['window_size'] for d in [dic['params'] for dic in otimizacao_bayesiana.res]]))))

Aqui treinamos o trader!
# ->
%%time
trader_otim.fit(500, 100)

# ->
%%time
trader_otim.investir()

## Parte 5 - Conclusão, Lições Aprendidas e Próximos Passos
- Um projeto como esse é complexo e requer conhecimento avançado.
- Nosso Trader demonstrou bons resultados com baixo volume de dados. Teríamos que validar isso com volumes de dados maiores.
- Diferentes políticas podem ser usadas para treinar o Trader e essa é uma decisão humana.
- Diferentes arquiteturas podem ser usadas no modelo e essa é uma decisão humana.
- Todas as técnicas usadas neste projeto podem ser refinadas com procedimentos mais avançados, tornando o AI Bot Trader ainda mais preciso.
# Fim
```

Otimização Bayesiana

Para compreender como funciona a Otimização Bayesiana, acesses esses dois manuais:

https://distill.pub/2020/bayesian-optimization/
https://arxiv.org/pdf/1807.02811.pdf

Aplicações de Machine Learning em Finanças

A Aprendizagem de Máquinas tem sido amplamente utilizada em finanças bem antes do advento de aplicativos de smartphones, bots de bate-papo (chatbots) ou motores de busca. Dado o volume elevado, registros históricos precisos e natureza quantitativa do mundo das finanças, poucas indústrias são mais adequadas para a inteligência artificial. Há mais casos de uso de aprendizagem de máquinas em finanças do que nunca, uma tendência perpetuada por maior capacidade de computação e aprendizagem de máquina mais acessível do que nunca.

Atualmente, Machine Learning passou a desempenhar um papel fundamental em muitas fases do ecossistema financeiro, desde a aprovação de empréstimos até a gestão de ativos e a avaliação de riscos. No entanto, poucos profissionais têm uma visão exata de quantas maneiras a aprendizagem de máquina é ou pode ser usada, em suas tarefas diárias.

Machine Learning em Finanças – Aplicações Atuais

Abaixo estão exemplos de Machine Learning em Financial Analytics. Tenha em mente que algumas dessas aplicações alavancam múltiplas abordagens de Inteligência Artificial - não exclusivamente aprendizagem de máquina.

Gerenciamento de Portfólio

O termo "robo-conselheiro" era essencialmente inédito há apenas cinco anos, mas agora é comum no panorama financeiro. O termo é enganoso e não envolve robôs em tudo. Em vez disso, robo-

consultores (empresas como Betterment, Wealthfront e outros) são algoritmos construídos para calibrar um portfólio financeiro para os objetivos e tolerância ao risco do usuário. Os usuários registram seus objetivos (por exemplo, aposentar aos 65 anos com economia de R$ 250.000,00), idade, renda e ativos financeiros atuais. O consultor (que seria mais precisamente referido como um "alocador"), em seguida, espalha investimentos em classes de ativos e instrumentos financeiros, a fim de atingir as metas do usuário. O sistema então calibra as mudanças nos objetivos do usuário e em mudanças em tempo real no mercado, visando sempre encontrar o melhor ajuste para os objetivos originais do usuário. Robô-consultores ganharam significativa tração com os consumidores que não precisam de um conselheiro físico para se sentir confortáveis investindo e que são menos capazes de validar as taxas pagas a consultores humanos.

Trading Algorítmico

Com origens que remontam aos anos 70, o Trading Algorítmico ou negociação algorítmica (às vezes chamada de “Sistemas de Negociação Automatizada”, que é, sem dúvida, uma descrição mais precisa) envolve o uso de sistemas complexos de Inteligência Artificial para tomar decisões comerciais extremamente rápidas. Sistemas algorítmicos muitas vezes fazem milhares ou milhões de negócios em um dia, daí o termo "negociação de alta freqüência" (HFT), que é considerado um subconjunto de negociação algorítmica. A maioria dos fundos de hedge e instituições financeiras não revelam abertamente suas abordagens de Inteligência Artificial para negociação (por uma boa razão), mas acredita-se que Machine Learning e Deep Learning estão desempenhando um papel cada vez mais importante na calibração de decisões comerciais em tempo real.

Detecção de Fraudes

Combine o poder de computação mais acessível, a internet tornando-se mais comumente usada e uma quantidade crescente de valiosos dados da empresa sendo armazenados on-line, e você tem uma "tempestade perfeita" para o risco de segurança de dados. Enquanto os sistemas de detecção de fraudes financeiras anteriores dependiam fortemente de conjuntos complexos e robustos de regras, a detecção de fraude moderna vai além de seguir uma lista de fatores de risco - ela aprende e calibra ativamente novas ameaças de segurança potenciais (ou reais). Este é o lugar da aprendizagem de máquinas em finanças para a detecção de fraudes - mas os mesmos princípios são válidos para outros problemas de segurança de dados. Usando Machine Learning, os sistemas podem detectar atividades ou comportamentos únicos ("anomalias") e sinalizá-los para equipes de segurança. O desafio para esses sistemas é evitar falsos positivos - situações em que os "riscos" são sinalizados, mas que não eram riscos realmente. Dada a quantidade incalculavelmente elevada de formas que a segurança pode ser violada, sistemas genuinamente "de aprendizagem" serão uma necessidade básica nos próximos cinco a dez anos.

Empréstimo / Seguros

Especialmente em grandes empresas (grandes bancos e companhias de seguros), os algoritmos de aprendizado de máquina podem ser treinados em milhões de exemplos de dados de consumidor (idade, trabalho, estado civil, etc ...), a fim de prever a probabilidade de pagar o empréstimo a tempo, entrar em um acidente de carro, etc .... As tendências podem ser avaliadas com algoritmos e continuamente analisadas para detectar outras tendências que podem influenciar os empréstimos a fim de aumentar o grau de previsibilidade de pagamento ou inadimplência. Estes resultados têm um enorme rendimento tangível para as empresas - mas, no momento, são reservados principalmente

para grandes empresas com os recursos para contratar Cientistas de Dados e armazenar os enormes volumes de dados passados e presentes para treinar seus algoritmos.

Machine Learning e o Futuro em Finanças

Machine Learning em Finanças – O Futuro

As aplicações abaixo são aquelas que consideramos promissoras. Algumas têm aplicações relativamente ativas hoje (embora não tão ativo quanto os casos de uso mais estabelecidos listados na parte 1) e outros são ainda relativamente novos.

Serviço ao Cliente

Os bots de bate-papo (chatbots) e as interfaces de conversação são uma área em rápida expansão de investimento de risco e atendimento ao cliente (chatbots são apontados como o mais promissor aplicativo de serviço ao consumidor no curto prazo). Empresas como a Kasisto já estão construindo chatbots específicos para financiamentos a fim de ajudar os clientes a fazer perguntas via bate-papo, como "Quanto eu gastei em compras no mês passado?" E "Qual foi o saldo de minha conta poupança há 60 dias? Esses chatbots tiveram de ser construídos com motores de processamento de linguagem natural, bem como dados de interações de clientes específicos de finanças. Bancos e instituições financeiras que permitem tais consultas e interações rápidas poderão rapidamente conquistar clientes de bancos maiores e burocráticos que exigem que as pessoas façam login em um portal online tradicional e procurem as informações por si mesmos. Este tipo de experiência de conversa (ou no futuro - voz) não é a norma hoje em bancos ou instituições financeiras, mas pode ser uma opção viável para milhões nos próximos cinco anos. Este aplicativo vai além da Machine Learning em finanças e é provável que se manifeste em uma variedade de campos e indústrias, como aliás já está ocorrendo.

Segurança 2.0

Nomes de usuários, senhas e questões de segurança podem não ser mais a regra para a segurança do usuário em cinco anos. A segurança do usuário é um jogo de apostas particularmente alto. Além de aplicações de detecção de anomalias como as que estão sendo desenvolvidas e usadas em detecção de fraudes, as futuras medidas de segurança podem exigir reconhecimento facial, reconhecimento de voz ou outros dados biométricos.

Análise de Sentimento / Notícias

Os fundos de hedge possuem sistemas avançados de Inteligência Artificial, que realizam análise de sentimentos e de notícias, como forma de gerar novos parâmetros de análise na toma de decisão. Mas pouco se sabe sobre esses sistemas, já que eles são a essência de negócios bilionários. No entanto, supõe-se que grande parte das futuras aplicações da aprendizagem de máquinas será baseada na compreensão das mídias sociais, tendências de notícias e outras fontes de dados - e não apenas os preços das ações. O mercado de ações move-se em resposta a uma miríade de fatores humanos relacionados e a esperança é que Machine Learning seja capaz de replicar e melhorar a intuição humana na atividade financeira, descobrindo novas tendências e percebendo sinais. O aprendizado de máquina em finanças estará longe de ser limitado a dados de ações e commodities - e que os fundos de hedge terão que aperfeiçoar seus sistemas de Inteligência Artificial se quiserem permanecer à frente da concorrência.

Vendas / Recomendações de Produtos Financeiros

Atualmente existem aplicações de vendas automatizadas de produtos financeiros. Um robô-conselheiro pode sugerir mudanças de carteira e há uma abundância de sites de recomendação de seguro que usam Inteligência Artificial para sugerir um carro particular ou plano de seguro de casa. No futuro, aplicações cada vez mais personalizadas e calibradas e assistentes pessoais virtuais podem ser percebidos como mais confiáveis e objetivos do que os assessores humanos. Assim como a Amazon e a Netflix podem recomendar livros e filmes melhor do que qualquer "especialista" humano, as conversas em andamento com assistentes virtuais podem fazer o mesmo com produtos financeiros, como vemos começando a acontecer na indústria de seguros

Análise Financeira, Liquidez e Análise Dupont

Análise Financeira

Uma análise consiste em distinguir e separar as partes de um todo para conhecer os seus elementos e princípios. Trata-se do exame que se realiza de uma realidade suscetível de estudo intelectual. Através da análise, é possível estudar os limites, as características e as possíveis soluções de um problema.

Financeiro, por outro lado, é um adjetivo que diz respeito às finanças, que é um conceito relacionado com o Tesouro Público, os bens e os fluxos. A noção de finanças é usada para se referir ao estudo da circulação do dinheiro entre indivíduos, empresas ou Estados.

A análise financeira, por conseguinte, é um método que permite analisar as consequências financeiras das decisões de negócios. E para isso, é necessário aplicar técnicas que permitam recolher informações relevantes, avaliar uma série de indicadores e tirar conclusões.

Graças à análise financeira, é possível estimar o rendimento de um investimento, estudar o seu risco e saber se o fluxo de fundos de uma empresa é suficiente para fazer face aos pagamentos, entre outras questões. A análise financeira ajuda a compreender o funcionamento do negócio e a maximizar a rentabilidade a partir da atuação sobre os recursos existentes. Os gerentes podem ter acesso à informação sobre o efeito esperado das decisões estratégicas. Os investidores também recorrem à análise financeira para analisar o grau de risco das suas decisões e para fixar os objetivos a cumprir. Os emprestadores e os credores, por fim, usam a análise financeira para determinar os riscos existentes para cobrar um crédito ou um empréstimo.

Análise Financeira Não é uma Ciência Exata

Nos anos 80, houve uma revolução financeira, marcada pelo aumento dos riscos financeiros a que as empresas estavam expostas, pela crescente importância das emissões de títulos no financiamento das empresas e pela consequente relevância quer da gestão de tesouraria, quer das variáveis financeiras. A implicação desta revolução foi o aumento do número de entidades que emitem pareceres sobre a situação financeira das empresas, não estando já este papel reservado apenas aos bancos, aparecendo aí os auditores e suas empresas de auditoria.

Esta evolução foi acompanhada de alterações na gestão das empresas, no sentido de questionar os tradicionais instrumentos de análise, não para os inviabilizar mas para os enriquecer. A medida que a Análise Financeira evolui, torna-se assim cada vez mais claro que não se trata de uma ciência exata e que não há uma forma única de analisar a saúde financeira de uma empresa.

O diagnóstico da situação financeira de uma empresa pode surgir de uma necessidade interna ou externa. Um diagnóstico interno pode ser necessário para tomar decisões de gestão, utilizando a análise financeira como um instrumento de previsão no âmbito de planos de financiamento e investimento ou para o controlo interno (acompanhamento da atividade e comparação entre as previsões e a performance real). Pode também servir um objetivo informacional, usando a informação financeira como instrumento de comunicação interna ou como elemento de relações sociais na empresa.

Quanto ao diagnóstico externo, surge como instrumento de decisão para entidades externas com influência junto da empresa, tais como bancos, investidores institucionais, governos, fornecedores e clientes. O diagnóstico externo surge também como ferramenta de comunicação com os investidores, o público, as agências de rating e as entidades que elaboram estudos estatísticos financeiros.

Fontes da Informação

Os dados utilizados para calcular os indicadores são na sua grande maioria retirados dos principais documentos financeiros das empresas, já mencionados anteriormente neste capítulo. Vejamos uma breve revisão.

Balanço

Documento contabilístico que expressa a situação patrimonial de uma empresa num determinado momento (geralmente um trimestre, semestre ou ano). Este documento permite comparar o ativo (bens que a empresa possui assim como o dinheiro que tem e as dívidas de terceiros), com o passivo ou capital alheio (o que a empresa deve a terceiros, quer seja empréstimos bancários, responsabilidades para com o Estado, dívidas a fornecedores, etc.). A diferença entre o que tem e o que deve é designada de “situação líquida” (composta pelo Capital que foi usado para criar a empresa, pelo acumular de resultados positivos ou negativos ao longo dos anos de funcionamento da empresa, e por eventuais reavaliações de componentes do ativo).

Num Balanço, o Ativo é igual à soma do Capital de Terceiro com a Situação líquida, ou seja, o Capital Alheio (Passivo) e a Situação líquida (Capitais próprios) financiam o Ativo, logo:

Ativo = Capital de Terceiro + Situação líquida

Devido às empresas se financiarem com Capitais Próprios e Alheios, o Ativo, normalmente, é maior que o Capital Alheio. Caso não se verifique, ou seja, sempre que o Ativo for inferior ao Passivo (valor do Capital Próprio vai ser negativo), existe falência ou insolvência técnica da empresa.

Demonstração do Resultado Líquido

Evidencia a formação de resultados num certo período (entre dois balanços) e serve para avaliar a situação econômica da empresa. Esta formação de resultados evidencia-se pela síntese dos custos e proveitos em grupos homogêneos, sendo que existem duas formas de elaboração da demonstração de resultados:

• Demonstração de resultados por natureza
• Demonstração de resultados por funções

É de salientar que existe uma distinção nítida entre resultados operacionais e resultados financeiros. Nos resultados operacionais integram-se os proveitos e custos relacionados à exploração, enquanto

que nos resultados financeiros consideram-se os proveitos de aplicações de capital e os custos dos financiamentos, assim como ganhos ou perdas resultantes de aplicações financeiras.

Demonstração dos Fluxos de Caixa (Cash-Flows)

A demonstração de fluxos de caixa deve relatar os movimentos de caixa durante o período, classificados por atividades operacionais, de investimento e de financiamento, de forma a proporcionar informação que permita aos interessados das demonstrações financeiras determinar o impacto dessas atividades na posição financeira da empresa e nas quantias de caixa e seus equivalentes.

Pode-se dizer que para cada um destes itens, temos a seguinte definição:

• Balanço → Posição Financeira
• Demonstração de Resultados → Desempenho
• Demonstração de Fluxos de Caixa→ Alterações na Posição Financeira

Devido ao carácter sintético dos documentos contabilísticos referidos, estes não respondem por inteiro às exigências da análise financeira. Resumindo, temos como principais limitações dos documentos contabilísticos, os seguintes aspectos:

• Não refletem valores atuais
• Existem contas para as quais é preciso fazer algumas estimativas
• Enquanto a depreciação de ativos é uma prática corrente, a sua revalorização é normalmente ignorada
• Os diferentes critérios utilizados podem provocar diferenciações nas várias empresas
• As normas contabilísticas são muitas vezes ditadas por imperativos fiscais que reduzem o significado econômico-financeiro da informação contabilística.

Dados externos

São também utilizados dados externos como:

• Dados macroeconômicos
• Cotações
• Taxas de juro
• Valores de referência de empresas concorrentes para comparação

Liquidez

Os índices de liquidez indicam a capacidade de uma empresa para honrar seus compromissos, sejam eles de curto prazo ou de longo prazo. Para chegarmos aos índices desejados, temos que extrair os dados do balanço patrimonial. Principais índices de liquidez:

Liquidez Corrente

É o índice correspondente ao cumprimento das obrigações de curto prazo, utilizamos os dados do ativo circulante (caixa, banco, duplicatas a receber, estoque, etc.) e do passivo circulante (duplicatas a pagar, fornecedores a pagar, empréstimos de curto prazo, etc.). Fórmula:

Liquidez Corrente = Ativo Circulante / Passivo Circulante

Liquidez Seca

O mesmo conceito da liquidez corrente, porém se desconta o valor do estoque no cálculo, o que ocasiona uma análise mais crítica da situação do índice de liquidez, considerando que as obrigações deveriam ser cumpridas sem comprometer o estoque. Fórmula:

Liquidez Seca = (Ativo Circulante – Estoques) / Passivo Circulante

Liquidez Imediata

Índice altamente rígido em relação ao cumprimento das obrigações de curto prazo, pois considera apenas os valores que a empresa tem disponível no exato momento, em contas como caixa, conta bancária, aplicações financeiras e etc...

Liquidez Imediata = Recursos disponíveis imediatos / Passivo Circulante

Liquidez Geral

A liquidez geral apresenta a situação da empresa no longo prazo, obtendo uma visão bem mais ampla e que tem que ser analisada criteriosamente dentro de seus subgrupos. Fórmula:

Liquidez Geral = (Ativo Circulante + Realizável a longo prazo) /
(Passivo Circulante + Passivo não Circulante)

Resumindo:

Quanto menor o valor em relação a 1, menos liquidez no item analisado, se for igual 1 está equilibrado, e maior que 1, existe uma folga para honrar os compromissos.

Os índices de liquidez são apenas mais uma ferramenta que nos auxilia na avaliação de uma empresa. Ela deve ser utilizada em conjunto com outras ferramentas.

Análise Dupont

Um modelo surgido em meados da década de 1920, denominado mais comumente como Análise DuPont, funciona como uma técnica de busca para localizar as áreas responsáveis pelo desempenho da empresa. Este sistema funde a demonstração do resultado e o balanço patrimonial em duas medidas-sínteses da rentabilidade dos ativos: a lucratividade das vendas, representada através da Margem Líquida, evidenciando o ganho no preço; e a produtividade, visualizada a partir do Giro do Ativo, que demonstra o ganho na quantidade e indica qual a eficiência da empresa na utilização dos seus ativos para geração de vendas.

Quando buscamos entender os indicadores de Análise Fundamentalista, é importante começar a fazer relação entre eles. Entre todos existentes, temos um bastante importante, conhecido pela sigla ROE.

ROE é o retorno sobre o investimento, é o Return on Equity, o retorno sobre um patrimônio líquido. Esse indicador orienta ao conhecimento de quanto o investimento alvo (neste caso, a empresa) rende em determinado espaço de tempo (geralmente um ano) em relação ao capital investido. Para a análise em empresas, é o lucro em relação ao dinheiro colocado no negócio por sócios.

A Análise DuPont une os indicadores e a importância do ROE, e permite o entendimento sobre como o retorno de um negócio é construído.

Por que você deve entendê-la?

É a forma mais simples de entender o perfil de lucratividade de uma empresa, de um negócio e assim poder interpretar seus riscos e benefícios. O ROE é formado por outros três indicadores:

• Margem líquida
• Giro do Ativo
• Alavancagem

O ROE pode ser obtido de duas formas:

• ROE = MARGEM LÍQUIDA x GIRO DO ATIVO x ALAVANCAGEM
• ROE = RESULT. LÍQUIDO / PATRIMÔNIO LÍQUIDO

Vamos explorar cada indicador e tirar conclusões. Antes de começar, tenha em mente que pelo fato de multiplicarmos todos entre si, cada vez que um tem o valor maior, isso proporcionalmente aumentará o ROE. Entretanto, aumentar um indicador nem sempre é um fato positivo para o negócio.

Margem Líquida

É o resultado líquido na DRE (lucro ou prejuízo) em relação à receita líquida (faturamento menos impostos sobre vendas, devoluções e outra deduções), também demonstrada na DRE.

É o quanto que sobra de faturamento para os sócios. Após esse cálculo, os valores podem chegar até no máximo 1 e não tem limite para ser negativo, caso a empresa apresente prejuízo. É o percentual das vendas que aumentam ou diminuem o patrimônio da empresa, desta forma, isoladamente, quanto maior melhor.

Ele é o mais importante dos três. Qual o motivo?

Porque não adianta vender mais e aumentar a quantidade de produtos a serem ofertados (aumentar o giro do ativo nestes casos) ou mesmo tomar empréstimos (aumentar a alavancagem), se o negócio não der lucro, não tiver uma margem líquida positiva, pois tais ações podem apenas aumentar o ROE negativo. Por exemplo, um negócio que está iniciando e tem altos valores fixos investidos, precisará de um aumento de vendas para que a margem de contribuição de seus produtos possa cobrir os custos fixos e assim sua margem líquida se torne positiva. Mas ainda sim, esse plano de negócio tem que estar realista na necessidade de se ter estes custos fixos iniciais e em relação à expectativa de venda no mercado, pois, caso não venha a apresentar margem líquida positiva, não alcançará crescimento patrimonial.

Giro do Ativo:

É a receita líquida (faturamento menos impostos sobre vendas, devoluções e outra deduções) em relação ao total de Ativo demonstrado no Balanço Patrimonial.

Teoricamente, esse indicador mostra quantas vezes está se vendendo o ativo da empresa no período, quando adicionada a margem bruta de lucro. Quanto maior, melhor e para isso, irá testar se a aplicação em ativos está se tornando cada vez menor e se paralelamente o volume de vendas é cada vez maior. Observe a fórmula:

Receita Líquida / Ativo

Exigimos que o ativo seja menor e as vendas maiores, justamente pelo fator matemático. Em uma divisão, quando o numerador é maior e o denominador é menor, consequentemente temos um resultado maior, o que também aumentará o ROE. Vale dizer também que constantes em uma das duas posições e aumento das vendas ou redução dos ativos, também resultam em melhores resultados.

A partir disso, a missão do negócio passa a se tornar mais eficiente em suas aplicações e aumentar as vendas.

No caso da empresa crescer em suas vendas, existem efeitos colaterais na Margem Líquida que devem ser comparados em termos de lucratividade. A tendência é que essa margem diminua, pois há um aumento de despesas e custos fixos para sustentar uma estrutura de novas vendas. Tais despesas e custos podem ser também variáveis. O aumento de vendas também aumenta o lucro líquido e consequentemente, o retorno.

Um bom sinal a ser identificado na análise de balanços, é uma empresa crescer seus ativos constantemente, aumentar o giro do ativo constantemente ou ao menos manter um valor constante (ambos apontarão liquidez constante das aplicações) e além disso, mantendo uma Margem Líquida constante ou crescente. É nesse cenário que o investidor terá plena convicção de que seu dinheiro está sendo bem investido e proporciona retornos crescentes. Essa situação pode ser encontrada na análise de balanços com cálculo de indicadores e análise horizontal.

Alavancagem

Mostra o quanto a empresa aplica em relação ao investimento dos sócios. Ex.: Se o resultado for 1,2, ela tem na estrutura de aplicação de 20% a mais que o capital investido. Isso potencializa a base de ativos para conseguir retornos, mas envolve terceiros que nem sempre compartilham do risco do negócio e querem receber independente dos resultados.

Esse valor de 1,2, consequentemente, aumentará o ROE. Nem sempre uma alavancagem tem custos financeiros, então não podemos julgar se ela é boa ou ruim. Isso envolve saber se os juros pagos pelo passivo são menores que os juros recebidos pelo ativo e também saber se o negócio tem capacidade de pagamento dos juros propostos, das amortizações, para saber se a transação aumentará o patrimônio sem problemas. Essa última análise é feita no fluxo de caixa.

Outro fator contábil que impacta na alavancagem, é a provisão de despesas. Esse passivo não tem custos, quando pagos até o vencimento, e compõem o passivo de um mês para o outro. A contabilidade provisiona valores de despesas por competência, a serem desembolsados em um mês posterior. Esse fenômeno sempre formará uma alavancagem ligeiramente maior que 1. E perceba que, esse valor nunca será menor que 1.

Resumindo, o Giro do Ativo e a Alavancagem, são duas formas de potencializar o ROE. Essas formas de potencializar também têm efeitos colaterais e podem piorar a lucratividade de um negócio.

Um exemplo para ficarmos de olho na análise de balanço, são de empresas que pagam mais juros do que recebem de retorno em relação aos seus ativos e continuam a distribuir muito dividendo para o acionista, evitando quitar empréstimos redutores de patrimônio. Nesta situação, é melhor ser credor da empresa, pois o retorno sobre o capital aplicado é maior e a depender da capacidade de geração de caixa do negócio, o retorno será garantido.

Capital Asset Pricing Model

O Modelo de Precificação de Ativos Financeiros (MPAF), mais conhecido mundialmente pela sigla em inglês CAPM (Capital Asset Pricing Model), é utilizado em finanças para determinar a taxa de retorno teórica apropriada de um determinado ativo em relação a uma carteira de mercado perfeitamente diversificada. O modelo leva em consideração a sensibilidade do ativo ao risco não-diversificável (também conhecido como risco sistêmico ou risco de mercado), representado pela variável conhecida como índice beta ou coeficiente beta (β), assim como o retorno esperado do mercado e o retorno esperado de um ativo teoricamente livre de riscos.

O CAPM é um modelo que mostra o retorno que um investidor aceitaria por investir em uma empresa. Trata de uma maneira de encontrar uma taxa de retorno exigido que leva em conta o risco sistemático (não diversificável), por meio do coeficiente Beta. Para entender o que é CAPM acompanhe este texto, pois todos estes conceitos ficarão mais claros!

Uma vez que o custo de capital de uma empresa é composto por uma parcela de capital de terceiros e outra de capital próprio, o CAPM entra no sentido de estimar o custo do capital próprio, ou seja, o retorno que os acionistas esperam obter por terem injetado dinheiro na companhia.

O modelo CAPM é derivado da conhecida Teoria do Portfólio, de Harry Markowitz, e busca trazer respostas mais efetivas quanto ao risco e retorno na avaliação de ativos.

Um dos aspectos mais controvertidos no campo de finanças tem sido a forma como devem ser relacionados dois componentes de extrema importância na avaliação de ativos: o risco e o retorno.

O modelo CAPM, consegue exatamente dimensionar esses dois componentes e seus reflexos sobre a taxa de retorno esperada de um investimento.

Dentro da teoria de finanças, a preocupação com o cálculo do valor de um ativo, conhecido como o valor intrínseco ou valor justo, tem sido constante. E, como no cálculo do valor, um componente vital é a taxa aplicada, o dimensionamento dos componentes risco e retorno, vistos simultaneamente e refletidos no valor intrínseco de um ativo, revestem de grande importância o modelo desenvolvido.

A mensuração desses dois componentes, risco e retorno, é também uma das tarefas primordiais dos analistas de mercado, e o resultado dessa mensuração é ingrediente crucial na construção e formação das carteiras de títulos.

Quais são as relações entre risco e retorno? Como é que risco e retorno devem ser levados em conta na administração de ativos ou de carteira de títulos?

A dificuldade em medir esses componentes risco e retorno pode ser entendida se imaginássemos um analista tentando delinear cada evento possível (preço de uma ação, por exemplo) e estimar a sua probabilidade de ocorrência e o efeito de cada um desses preços sobre suas alternativas de investimento. Isso seria impraticável. Na realidade, isso pode ser evitado se os retornos médios (ou retornos esperados) forem diretamente estimados, e a seu lado a divergência provável de cada retorno, com relação a sua média (ou ao seu valor esperado). Dessa forma, usaremos a média ponderada como medida do retorno esperado e os desvios dessa média (variância ou desvio-padrão) como medida de risco.

A maior vantagem do modelo de avaliação de ativos (CAPM) está em que ele considera a incerteza diretamente, permitindo, portanto, estudar o impacto duplo e simultâneo da lucratividade e do risco

sobre o valor da ação. Além dos pressupostos do mercado eficiente, o modelo pressupõe também que o investidor é avesso ao risco e se utiliza dos conceitos de média e variância na escolha das alternativas.

Para que o modelo possa ser entendido, vamos ter de recorrer a alguns conceitos básicos da teoria da utilidade, segundo a qual cada indivíduo tem um comportamento próprio quando confrontado com as variáveis risco e retorno.

Dessa forma, todo investidor pode ser caracterizado pelo seu maior ou menor grau de aversão ao risco, de tal modo que existem investidores neutros em relação ao risco, investidores avessos ao risco e aqueles investidores que preferem o risco a qualquer alternativa de certeza. Esses últimos são normalmente denominados de amantes do risco.

A teoria da utilidade conclui que, de maneira geral, o investidor pode ser considerado avesso ao risco. Ou seja, entre duas condições, uma com certeza e outra com algum grau de risco, a preferência cairá, seguramente, na alternativa que apresenta o menor componente de risco possível. Ainda assim, entre os investidores avessos ao risco, vai haver maior ou menor grau de aversão, de acordo, mais uma vez, com as características individuais de cada um.

Otimização de Portfólio

A teoria moderna do portfólio, ou simplesmente teoria do portfólio, explica como investidores racionais irão usar o princípio da diversificação para otimizar as suas carteiras de investimentos, e como um ativo arriscado deve ser precificado. O desenvolvimento de modelos de otimização de portfólio tem origem na área econômico-financeira.

O trabalho pioneiro na área de otimização de portfólio foi à proposição do modelo média-variância por Markowitz (1952). A teoria do portfólio estabelece que decisões relacionadas à seleção de investimentos devam ser tomadas com base na relação risco-retorno. Para auxiliar neste processo, modelos de otimização de portfólio têm sido desenvolvidos. De modo a serem efetivos, tais modelos devem ser capazes de quantificar os níveis de risco e retorno dos investimentos e consideram indicadores financeiros e podem ser usados para projetos de Financial Analytics.

Um investidor racional irá otimizar o seu retorno tendo em conta o determinado risco que está disposto a correr, ou irá minimizar o risco tendo em conta um determinado retorno que quer atingir. A medida exata desse trade-off será diferente para cada investidor com base nas suas características individuais a aversão ao risco.

Com o aparecimento da teoria de Markowitz começou-se a olhar para os títulos de investimento como um todo e não simplesmente selecionar os melhores individualmente, passou a olhar-se para como os ativos se relacionavam entre si, de modo a conseguir melhores resultados conjuntos. Quando nos referimos à teoria de Markowitz estamos falando do modelo de Media-Variância, que ficou assim conhecido pelo fato de girar em torno do retorno esperado (média) e no desvio-padrão (variância) das carteiras de ativos.

Pressupostos da teoria de Markowitz:

1. Risco de um portfólio é baseado na variabilidade dos retornos do portfólio
2. Um investidor é avesso ao risco
3. Um investidor prefere aumentar o consumo
4. Função utilidade do investidor, é côncava e crescente, devido à sua aversão ao risco e preferência

de consumo
5. A análise é baseada no modelo de período único de investimento

Todos os dias temos milhares de pessoas tentando investir da melhor maneira possível as suas poupanças. Para muitos os retornos fixos e baixos (retornos sem risco) seriam uma forma lenta de empobrecimento e como solução tendem a procurar carteiras de ativos, ativos reais ou ativos financeiros. Estas carteiras de ativos permitem ganhos extraordinários, no entanto como tudo na vida existe o reverso da moeda, neste caso o risco, poderemos ganhar muito ou então perder tudo.

Como fazer as pessoas ganharem o que desejam com o mínimo risco possível? Para tal, será necessário estudar as várias maneiras de medir o nosso risco (medidas de risco), conseguindo quantificar o risco poderemos controla-lo, e conseguir um melhor Trade-off risco/retorno adaptado a cada especulador. Com este intuito, podemos modelar diversas medidas de risco, sendo assim possível ao longo da mesma, comparar depois os nossos resultados a nível numérico e gráfico e conseguindo resolver problemas que analiticamente seriam impossíveis.

Risco

Existem várias definições de risco, um exemplo de uma definição simples é: “O risco em seu sentido fundamental, pode ser definido como a possibilidade de prejuízos financeiros”. Também podemos ver o risco como a incerteza associada a um ativo ou uma carteira de ativos, podemos ainda ver o risco como a variabilidade/volatilidade em relação a um valor esperado. No entanto, o risco de um investimento estará sempre associado ao fato de o retorno efetivo de um investimento poder estar abaixo do retorno esperado, gerando desse modo uma perda.

Carteiras de investimentos

Carteiras de investimentos são um conjunto de ativos pertencentes a um investidor, pessoa física ou pessoa jurídica, podendo este ser constituído por ações, fundos, títulos públicos, aplicações imobiliárias, entre outros.

Retorno esperado

É o valor que se espera que uma ação ou conjunto de ações possa gerar no próximo período, este valor é meramente especulativo, podendo o valor efetivamente observado ser maior ou menor que o esperado. Quando falamos do retorno esperado de uma carteira de investimentos estamo-nos a referir a média ponderada dos retornos dos ativos que a compõem.

Retorno simples

Retorno simples é dado pela fórmula:

$$\frac{Pt - Pt-1}{Pt-1}$$

Onde o Pt é o nosso retorno no dia t, e o nosso Pt-1, é o nosso no retorno no dia t-1

Em 1964, William Sharpe desenvolveu um modelo imaginando um mundo onde todos os investidores utilizam a teoria da seleção de carteiras de Markowitz através da tomada de decisões usando a avaliação das médias e variâncias dos ativos. Sharpe supõe que os investidores compartilham dos mesmos retornos esperados, variâncias e covariâncias.

Mas ele não assume que os investidores tenham todos o mesmo grau de aversão ao risco. Eles podem reduzir o grau de exposição ao risco selecionando mais ativos de menor risco, ou construindo carteiras combinando muitos ativos de risco. O CAPM descreve a relação entre o risco de mercado e as taxas de retorno exigidas.

Otimização

Quando em matemática se fala em otimização, referimo-nos sobretudo à resolução de problemas de minimização/maximização de uma função, e na otimização de portfólio, o objetivo é sobretudo minimizar as diversas medidas de risco, sujeitas a restrições de igualdade e desigualdade, de modo a obter carteiras ótimas. Existem vários métodos numéricos de otimização para resolução de problemas de otimização com restrições lineares e não lineares. No nosso caso especifico apenas nos interessa os métodos que resolvam problemas numéricos com restrições não lineares, nomeadamente: Métodos de penalização, Programação quadrática sequencial (SQP) e Método dos pontos interiores. A Programação quadrática será usada no Projeto 2, ao final deste capítulo.

Quando se investe em ações ou fundos de ações, é inevitável a existência de um risco sempre associado, como tal, uma das principais necessidades é conseguir quantificar esse risco, para conseguir otimizar os investimentos e minimizar os riscos. Existem várias medidas que foram sendo abordadas ao longo do tempo, podemos, no entanto, subdividir estas medidas de risco em dois grupos de medidas de risco não coerentes e medidas de risco coerentes. Exemplos de medidas de risco não coerentes:

1- Desvio-Padrão (variância);
2- Valor em Risco (VaR);

Desvio Padrão

O desvio-padrão é uma das principais medidas de risco utilizadas devido à sua simplicidade e facilidade de utilização, é uma medida estatística que serve para medir a dispersão de uma variável em torno da sua média, no caso de uma carteira de ações serve para medir a variabilidade dos retornos em relação ao retorno médio. Neste caso, quanto maior for o desvio-padrão de um portfólio maior será o nível de incerteza/risco associado a esse portfólio, ou seja, maior a probabilidade de haver retornos positivos ou negativos extremos. Tem como principais desvantagens o fato de tratar ganhos e perdas da mesma forma e ser insensível a caudas pesadas.

VaR

O VaR surge da necessidade de controlar e quantificar a exposição das carteiras aos riscos de mercado, mais focado nas grandes perdas. Esta medida de risco teve um crescimento na sua utilização depois da falência do banco Barings PLC, em Fevereiro de 1995, na altura um grande banco com 233 anos de história. Esta medida de risco permite-nos quantificar o risco inerente a cada carteira. O VaR de uma carteira por definição pode ser apresentada como uma estimativa da perda máxima em valor monetário para um determinado nível de confiança estatístico, assumindo um determinado horizonte temporal de permanência em posse dessa mesma carteira.

Estudo de Caso – Modelo Analítico Para Otimização de Portfólio

O estudo de caso é uma aplicação do RStudio Shiny para otimização de portfólios com base em um modelo de média-variância simples e um modelo Black-Litterman (módulo não concluído). O aplicativo exibe porcentagens a serem alocadas para cada classe de ativos (assets).

O Modelo Black-Litterman combina a opinião dos investidores sobre classes de ativos no modelo de otimização de portfólio, sendo muito utilizado em Fundos Hedge, com carteiras de bilhões de dólares.

Esse aplicativo foi construído em módulos. Você pode expandi-lo e agregar mais módulos e outros tipos de análises. Para esse aplicativo, consideramos ativos americanos, por ser mais fácil encontrar documentação sobre tais ativos.

Para executar a app, carregue os 3 scripts no RStudio e execute Run App.

```
# Projeto - Otimização de Portfólio

# Esse é aplicativo RStudio Shiny para otimização de portfólios com base em um modelo de média-variância simples
# e um modelo Black-Litterman (módulo não concluído).
# O aplicativo exibe porcentagens a serem alocadas para cada classe de ativos (assets).

# O Modelo BL combina a opinião dos investidores sobre classes de ativos no modelo de otimização de portfólio.

# Esse aplicativo foi construído em módulos. Você pode expandí-lo e agregar mais módulos e outros tipos de análises.
# Para este aplicativo, consideramos ativos americanos, por ser mais fácil encontrar documentação sobre esses ativos.

library(quadprog)

# Função do módulo de construção de portfólio
constructPortfolio <- function(histReturns, covMatrix, numRVals)
{
  # Obtém a quantidade de assets
  numAssets = length(histReturns)

  # Variáveis
  weights = matrix(0, numAssets, numRVals)
  stdDevs = NULL

  # Retorno histórico
  Returns <- histReturns * 0.01

  # 10 taxas de retorno desejadas
  RVals = seq(min(Returns), max(Returns), length.out = numRVals)


  # Define e reolve problema de otimização com solve.QP7
  Dmat <- 2*covMatrix
  dvec <- rep(0,numAssets)

  # Matriz de constraints
  Amat <- matrix(cbind(rep(1,numAssets), rep(-1,numAssets), Returns, diag(numAssets)), nrow=numAssets)

  for(i in 1:length(RVals))
    {
      bvec <- c(1,-1,RVals[i], rep(0,numAssets))

      # Resolvendo programação quadrática
      solution <- solve.QP(Dmat,dvec,Amat,bvec)

      # Obtém desvio padrão e peso do portfólio em cada iteração
      stdDevs[i] <- sqrt(solution$value)
      weights[,i] <- solution$solution
    }

  finalSolution <- matrix(weights, nrow=numAssets, ncol=length(RVals))

  # Retorna os pesos de cada classe de asset e portfólio
  return(list(weights = finalSolution, stdDevs = stdDevs, RVals = RVals))
}
```

```
library(shiny)
source("constructPortfolio.R")

covMatrix <- matrix()
histReturns <- c()
assetClasses <- c("US Bonds",
                  "Int'l Bonds",
                  "US Large Growth",
```

```
                "US Large Value",
                "US Small Growth",
                "US Small Value",
                "Intl Dev Equity",
                "Intl Emerg Equity")
numViews = 1

shinyServer(function(input, output, clientData, session) {

  #---------------------------------- Output da Tab de Descrição ----------------------------------------
  output$cMat <- renderTable({

    # Matriz de covariância padrão
    covMatrix <<- matrix(c(0.001005,0.001328,-0.000579,-0.000675,0.000121,0.000128,-0.000445,-0.000437,
                           0.001328,0.007277,-0.001307,-0.00061,-0.002237,-0.000989,0.001442,-0.001535,
                           -0.000579,-0.001307,0.059852,0.027588,0.063497,0.023036,0.032967,0.048039,
                           -0.000675,-0.00061,0.027588,0.029609,0.026572,0.021465,0.020697,0.029854,
                           0.000121,-0.002237,0.063497,0.026572,0.102488,0.042744,0.039943,0.065994,
                           0.000128,-0.000989,0.023036,0.021465,0.042744,0.032056,0.019881,0.032235,
                           -0.000445,0.001442,0.032967,0.020697,0.039943,0.019881,0.028355,0.035064,
                           -0.000437,-0.001535,0.048039,0.029854,0.065994,0.032235,0.035064,0.079958),
                         byrow = TRUE, nrow = 8, ncol = 8)
    covMatrix

  }, digits = 6
  )

  # Atualiza dados históricos com inputs do usuário
  output$histReturns <- renderTable({
    histReturns <<- c(input$usBonds, input$intlBonds,
                      input$usLargeG, input$usLargeV,
                      input$usSmallG, input$usSmallV,
                      input$intlDevEq, input$intlEmergEq)

    data.frame(
      Asset = c("US Bonds", "Int'l Bonds",
                "US Large Growth", "US Large Value",
                "US Small Growth", "US Small Value",
                "Int'l Dev. Equity", "Intl Emerg Equity"),
      Return = histReturns, stringsAsFactors = FALSE)
  })



#---------------------------------- Tab da Média-Variância Simples ----------------------------------------
observe({
  histReturns <<- c(input$usBonds, input$intlBonds,
                    input$usLargeG, input$usLargeV,
                    input$usSmallG, input$usSmallV,
                    input$intlDevEq, input$intlEmergEq)

  solution <- constructPortfolio(histReturns, covMatrix, input$numRVals)

  list_RVals <- 1:length(solution$RVals)
  names(list_RVals) <- solution$RVals*100
  updateSelectInput(session, "selectRVals", choices = list_RVals)

  # Plot dos pesos
  output$meanVarAlloc <- renderPlot({
    barplot(solution$weights[,as.numeric(input$selectRVals)], ylab = "% do Portfolio", xlab = "Classe de Ativo", main = "Alocação de Portfolio")
  })

  # Plot efficient frontier
  output$meanVarFrontier <- renderPlot({
    plot(solution$stdDevs, solution$RVals, type = 'b', xlab = "Risco (Desvio Padrão)", ylab = "Retorno", main = "Efficient Frontier")
  })
})


#---------------------------------- Tab Modelo Black-Litterman ----------------------------------------
  output$ui <- renderUI({
    switch(input$selectType,
      "relative" = selectInput("newAssetClass", label = "Escolha Outra Classe de Ativos", choices = assetClasses[-
which(assetClasses==input$assetClass)])
    )
  })

  observe({
```

```
  updateSelectInput(session, "newAssetClass", choices = assetClasses[-which(assetClasses==input$assetClass)])
 })

})
```

```
library(shiny)


shinyUI(navbarPage("Otimização de Portfolio",

 tabPanel("Descrição da App",
  sidebarLayout(
   sidebarPanel(
     p("Este aplicativo é para otimização de portfólios contendo 8 classes de ativos com base em dois modelos:
       Um modelo de otimização da média-variância simples e um modelo Black-Litterman (BL) que
       é baseado nas crenças dos investidores sobre os retornos nas classes de ativos. As saídas finais são porcentagens
       de cada ativo que você deve alocar nas carteiras para obter o nível mínimo de risco para um
       retorno desejado. As fronteiras eficientes para ambos os modelos são plotadas para comparação."),
     br(),
     p("Os retornos históricos para cada classe de ativos são especificados na guia pressupostos e
       incluídos também na matriz de covariância usada para calcular os pesos ótimos da carteira."),
     br(),
     helpText("Você pode optar por ajustar os retornos históricos. Os valores estão em porcentagens."),
     numericInput("usBonds", label = strong("US Bonds"), min=0, max=100, value = .08),
     numericInput("intlBonds", label = strong("Intl Bonds"), min=0, max=100, value = .067),
     numericInput("usLargeG", label = strong("US Large Growth"), min=0, max=100, value = 6.41),
     numericInput("usLargeV", label = strong("US Large Value"), min=0, max=100, value = 4.08),
     numericInput("usSmallG", label = strong("US Small Growth"), min=0, max=100, value = 7.43),
     numericInput("usSmallV", label = strong("US Small Value"), min=0, max=100, value = 3.70),
     numericInput("intlDevEq", label = strong("Intl Dev Equity"), min=0, max=100, value = 4.80),
     numericInput("intlEmergEq", label = strong("Intl Emerg Equity"), min=0, max=100, value = 6.60)

   ),
   mainPanel(
    tabsetPanel(type="tabs",
     tabPanel("Suposições de Entrada",
      h4("Matriz de Covariância de Excessos de Retorno"),
       tableOutput("cMat"),

      h4("Retornos históricos de classes de ativos"),
      tableOutput("histReturns")

     ),
     tabPanel("Modelo Black-Litterman"))
   )
)
),

 tabPanel("Modelo Média-Variância",
  sidebarLayout(
   sidebarPanel(

   ),

   mainPanel(
    numericInput("numRVals", label = strong("Quantas Taxas de Retorno Você quer Simular?"), min = 0, max = 500, value = 5),
    selectInput("selectRVals", label = "Especifique a Taxa de Retorno", c("label 1" = "option1")),
    plotOutput("meanVarAlloc"),
    plotOutput("meanVarFrontier")
    )
     )),

 tabPanel("Modelo Black-Litterman",
  sidebarLayout(
   sidebarPanel(

    ),
   mainPanel(
    numericInput("tau", label = "Especifique o índice τ", min = 0, max = 500, value = 2.5),
    selectInput("selectType", label = "A sua opinião é absoluta ou relativa?", choices=list("absolute","relative")),
    selectInput("assetClass", label = "Escolha uma classe de ativo",
                choices = c("US Bonds", "Int'l Bonds", "US Large Growth","US Large Value",
                            "US Small Growth", "US Small Value", "Intl Dev Equity", "Intl Emerg Equity")),
    numericInput("viewInput", label = "A sua avaliação sobre esta classe de ativos", min = 0, max = 100, value = 4.1),
    uiOutput("ui"),

    plotOutput("blAlloc")
    )
```

```
    )
  )
  )
)
```

Mini-Projeto 2 – Análise de Risco em Operações Financeiras

Neste projeto seu trabalho é construir um modelo capaz de prever o risco que um cliente oferece a uma instituição ao realizar operações financeiras. Nossa previsão é uma classificação de risco: 0 (baixo risco) ou 1 (alto risco).

Usando dados históricos repletos de problemas, faremos o processo de limpeza e engenharia de atributos. Vamos explorar as variáveis e compreender seus comportamentos e particularidades. Construiremos 3 versões do modelo e faremos a avaliação. Ao final, vamos comparar as métricas e selecionar o melhor modelo, extraindo os coeficientes que serão usados nas previsões. São 2 scripts:

Script 1: Mini-Projeto2-Modelo.R possui todo o trabalho de modelagem.
Script 2: Mini-Projeto2-Previsoes.R possui o deploy do modelo com dados de novos clientes.

Execute primeiro o Script 1 linha a linha. Na sequência execute o Script 2. Os scripts possuem todas as indicações que você precisa para compreender o trabalho.

```
# Mini-Projeto 2 - Análise de Risco em Operações Financeiras

# Leia os manuais em pdf no Capítulo 10 do curso de Business Analytics com a definição do problema e outros detalhes.
# Você vai precisar do dicionário de dados, também disponível no no Capítulo 10, para compreender os nomes das variáveis.
# Execute o script linha a linha e leia os comentários.

# TARGET_FLAG é a nossa variável alvo.

# Modelagem

# Pacotes
library(ggplot2)
library(readr)
library(plyr)
library(car)
library(fBasics)
library(gmodels)
library(MASS)
library(gridExtra)
library(pROC)

# Carrega os dados
dados <- read_csv("dados/dados_historicos.csv")

# Visaliza no formato de tabela
View(dados)

##### Análise Exploratória e Limpeza dos Dados #####

# Dimensões do dataset
dim(dados)

# Tipos das variáveis
str(dados)

# Sumário estatístico
summary(dados)

# Vamos visualizar as estatísticas das variáveis numéricas
# Analise a tabela em detalhes, especialmente a coluna que indica valores NA
?basicStats
View(t(basicStats(dados[sapply(dados, is.numeric)])))

# Calculando a frequência das variaveis categóricas
# Analise a tabela em detalhes
char_Freq <- lapply(dados[sapply(dados, is.character)], FUN = count)
```

```
char_stats <- ldply(char_Freq, data.frame)
names(char_stats) <- c("Variavel", "Valor", "Frequencia")
View(char_stats)

# Tratamento dos valores ausentes

# Com base na análise das estatísticas anteriores vamos definir as regras do tratamento de valores ausentes.
# Vamos limpar as variáveis usando regras diferentes para cada uma.

# Se tiver valor ausente na variável idade preenche com 45 (mediana da variável idade), senão, mantém a idade atual
dados$IDADE <- ifelse(is.na(dados$IDADE), 45, dados$IDADE)

# Se tiver valor ausente na variável YOJ preenche com 11 (mediana), senão, mantém o valor atual
dados$YOJ <- ifelse(is.na(dados$YOJ), 11, dados$YOJ)

# Vamos tratar os valores ausentes da variável salário de acordo com o tipo de trabalho, usando a estratégia de imputação
dados$SALARIO <- ifelse(is.na(dados$SALARIO) & is.na(dados$JOB), 54000,
                 ifelse(is.na(dados$SALARIO) & (dados$JOB == "Medico"), 128000,
                 ifelse(is.na(dados$SALARIO) & (dados$JOB == "Advogado"), 88000,
                 ifelse(is.na(dados$SALARIO) & (dados$JOB == "Gerente"), 87000,
                 ifelse(is.na(dados$SALARIO) & (dados$JOB == "Profissional"), 76000,
                 ifelse(is.na(dados$SALARIO) & (dados$JOB == "Trabalho Administrativo"), 58000,
                 ifelse(is.na(dados$SALARIO) & (dados$JOB == "Trabalho Manual"), 33000,
                 ifelse(is.na(dados$SALARIO) & (dados$JOB == "Do Lar"), 12000,
                 ifelse(is.na(dados$SALARIO) & (dados$JOB == "Estudante"), 6300,
                 dados$SALARIO)))))))))

# Imputação da mediana para a variável HOME_VAL
dados$HOME_VAL <- ifelse(is.na(dados$HOME_VAL), 162000, dados$HOME_VAL)

# Preenchemos com 8 (mediana da variável TEMPO_VIDA_PROD) se tver valor nulo.
# Se for valor negativo preenchemos com zero (valor negativo não faz sentido nesta variável).
dados$TEMPO_VIDA_PROD <- ifelse(is.na(dados$TEMPO_VIDA_PROD), 8, ifelse(dados$TEMPO_VIDA_PROD < 0, 0,
dados$TEMPO_VIDA_PROD))

# Para tratar os valores ausentes da variável JOB, checamos o salário.
# Por exemplo: Se a variável JOB tiver valor ausente e o salário for maior que 150.000 por ano,
# deve ser Medico (pois esse é o salário dos demais médicos no dataset) e preenchemos com essa categoria o valor ausente.
dados$JOB <- ifelse(is.na(dados$JOB) & dados$SALARIO > 150000, "Medico",
             ifelse(is.na(dados$JOB) & dados$SALARIO > 100000, "Advogado",
             ifelse(is.na(dados$JOB) & dados$SALARIO > 85000, "Gerente",
             ifelse(is.na(dados$JOB) & dados$SALARIO > 75000, "Profissional",
             ifelse(is.na(dados$JOB) & dados$SALARIO > 60000, "Trabalho Administrativo",
             ifelse(is.na(dados$JOB) & dados$SALARIO > 35000, "Trabalho Manual",
             ifelse(is.na(dados$JOB) & dados$SALARIO >= 12000, "Do Lar",
             ifelse(is.na(dados$JOB) & dados$SALARIO < 12000, "Estudante",
             dados$JOB)))))))

# Visualiza o dataset
View(dados)

# Vamos checar as estatística após o tratamento dos valores ausentes.

# Visualizando as estatísticas das variáveis numéricas
View(t(basicStats(dados[sapply(dados,is.numeric)])))

# Calculando a frequência das variaveis categóricas
char_Freq <- lapply(dados[sapply(dados,is.character)], FUN = count)
char_stats <- ldply(char_Freq, data.frame)
names(char_stats) <- c("Variavel", "Valor", "Frequencia")
View(char_stats)

# Valores ausentes tratados com sucesso!

# Vamos seguir com a exploração dos dados e construir alguns gráficos.
# Para diferentes cores digite: colors() no console e escolha a sua preferida.

# Exploração de algumas variáveis numéricas (fique à vontade para explorar outras variáveis numéricas).
# Analise e interprete os gráficos conforme você aprendeu no curso até aqui. Use o dicionário de dados se necessário.

# Análise de Risco Por Atrasos de Pagamentos
ggplot(dados, mapping = aes(x = CLM_FREQ, y = (..density..)  , fill = TARGET_FLAG)) +
  geom_histogram(colour = "tomato3") +
  facet_grid(~TARGET_FLAG, labeller = label_both ) +
  theme_bw() +
  labs(title = "Análise de Risco Por Atrasos de Pagamentos", x = "CLM_FREQ", y = "Density") +
  scale_y_continuous(breaks = seq(0, 10, by = 0.5))
```

```
# Análise de Risco Por Filhos Morando ou Não na Mesma Casa
ggplot(dados, mapping = aes(x = KIDSDRIV, y = (..density..) , fill = TARGET_FLAG)) +
  geom_histogram(colour = "magenta") +
  facet_grid(~TARGET_FLAG, labeller = label_both ) +
  theme_bw() +
  labs(title = "Análise de Risco Por Filhos Morando ou Não na Mesma Casa", x = "KIDSDRIV", y = "Density") +
  scale_y_continuous(breaks = seq(0, 10, by = 0.5))

# Análise de Risco Por Tempo Como Cliente
ggplot(dados, mapping = aes(x = TEMPO_CLIENTE, y = (..density..) , fill = TARGET_FLAG)) +
  geom_histogram(colour = "magenta") +
  facet_grid(~TARGET_FLAG, labeller = label_both ) +
  theme_bw() +
  labs(title = "Análise de Risco Por Tempo Como Cliente", x = "TEMPO_CLIENTE", y = "Density") +
  scale_y_continuous(breaks = seq(0, 10, by = 0.5))

# Análise de Risco Por Salário
ggplot(dados, mapping = aes(x = SALARIO, y = (..density..) , fill = TARGET_FLAG)) +
  geom_histogram(colour = "magenta") +
  facet_grid(~TARGET_FLAG, labeller = label_both ) +
  theme_bw() +
  labs(title = "Análise de Risco Por Salário", x = "SALARIO", y = "Density") +
  scale_y_continuous(breaks = seq(0, 10, by = 0.5))

# Análise de Risco Por Valor da Casa Onde Mora
ggplot(dados, mapping = aes(x = HOME_VAL, y = (..density..) , fill = TARGET_FLAG)) +
  geom_histogram(colour = "magenta") +
  facet_grid(~TARGET_FLAG, labeller = label_both ) +
  theme_bw() +
  labs(title = "Análise de Risco Por Valor da Casa Onde Mora", x = "HOME_VAL", y = "Density") +
  scale_y_continuous(breaks = seq(0, 10, by = 0.5))

# Exploração de algumas variáveis categóricas (fique à vontade para explorar outras variáveis categóricas)

# Análise de Risco Por Nível Educacional
ggplot(dados, mapping = aes(x = TARGET_FLAG, y = (..density..),  group = EDUCATION, fill = EDUCATION)) +
  geom_histogram(colour = "yellowgreen") +
  facet_grid(~EDUCATION ) +
  theme_bw() +
  labs(title = "Análise de Risco Por Nível Educacional", x = "TARGET_FLAG", y = "Density") +
  scale_y_continuous(breaks = seq(0, 100, by = 5)) +
  scale_x_continuous(breaks = seq(0, 1, by = 0.5))

# Análise de Risco Por Tipo de Profissão
ggplot(dados, mapping = aes(x = TARGET_FLAG, y = (..density..),  group = JOB, fill = JOB)) +
  geom_histogram(colour = "violetred4") +
  facet_grid(~JOB ) +
  theme_bw() +
  labs(title = "Análise de Risco Por Tipo de Profissão", x = "TARGET_FLAG", y = "Density") +
  scale_y_continuous(breaks = seq(0, 30, by = 5))  +
  scale_x_continuous(breaks = seq(0, 1, by = 0.5))

# Análise de Risco Por Número de Contas
ggplot(dados, mapping = aes(x = TARGET_FLAG, y = (..density..),  group = CONTA_CLI, fill = CONTA_CLI)) +
  geom_histogram(colour = "slateblue2") +
  facet_grid(~CONTA_CLI, labeller = label_both ) +
  theme_bw() +
  labs(title = "Análise de Risco Por Número de Contas", x = "TARGET_FLAG", y = "Density") +
  scale_y_continuous(breaks = seq(0, 30, by = 5))

# Análise de Risco Por Estado Civil
ggplot(dados, mapping = aes(x = TARGET_FLAG, y = (..density..),  group = CASADO, fill = CASADO)) +
  geom_histogram(colour = "slateblue3") +
  facet_grid(~CASADO, labeller = label_both ) +
  theme_bw() +
  labs(title = "Análise de Risco Por Estado Civil", x = "TARGET_FLAG", y = "Density") +
  scale_y_continuous(breaks = seq(0, 30, by = 5))

# Análise de Risco Por Gênero
ggplot(dados, mapping = aes(x = TARGET_FLAG, y = (..density..),  group = SEX, fill = SEX)) +
  geom_histogram(colour = "slateblue4") +
  facet_grid(~SEX, labeller = label_both ) +
  theme_bw() +
  labs(title = "Análise de Risco Por Gênero", x = "TARGET_FLAG", y = "Density") +
  scale_y_continuous(breaks = seq(0, 30, by = 5))

##### Engenharia de Atributos #####
```

```
# Vamos criar algumas tabelas cruzadas e compreender como os dados estão organizados.
# Com base nessa análise definiremos as decisões na engenharia de atributos.
# O objetivo é compreender a proporção de registros de acordo com o risco envolvido na operação financeira.

#
 Analise cada uma das tabelas conforme você aprendeu no curso até aqui.

# Tabela 1
CrossTable('Total' = dados$JOB, dados$TARGET_FLAG)

# Tabela 2
CrossTable(dados$KIDSDRIV,
           dados$TARGET_FLAG,
           prop.r = TRUE,
           prop.c = FALSE,
           prop.t = FALSE,
           prop.chisq = FALSE,
           row.labels = TRUE,
           dnn = c("KIDSDRIV", "TARGET_FLAG"))

# Tabela 3
CrossTable(dados$HOMEKIDS,
           dados$TARGET_FLAG,
           prop.r = TRUE,
           prop.c = FALSE,
           prop.t = FALSE,
           prop.chisq = FALSE,
           row.labels = TRUE,
           dnn = c("HOMEKIDS", "TARGET_FLAG"))

# Tabela 4
CrossTable(dados$CONTA_CLI,
           dados$TARGET_FLAG,
           prop.r = TRUE,
           prop.c = FALSE,
           prop.t = FALSE,
           prop.chisq = FALSE,
           row.labels = TRUE,
           dnn = c("CONTA_CLI", "TARGET_FLAG"))

# Tabela 5
CrossTable(dados$CASADO,
           dados$TARGET_FLAG,
           prop.r = TRUE,
           prop.c = FALSE,
           prop.t = FALSE,
           prop.chisq = FALSE,
           row.labels = TRUE,
           dnn = c("CASADO", "TARGET_FLAG"))

# Tabela 6
CrossTable(dados$SEX,
           dados$TARGET_FLAG,
           prop.r = TRUE,
           prop.c = FALSE,
           prop.t = FALSE,
           prop.chisq = FALSE,
           row.labels = TRUE,
           dnn = c("SEX", "TARGET_FLAG"))

# Tabela 7
CrossTable(dados$EDUCATION,
           dados$TARGET_FLAG,
           prop.r = TRUE,
           prop.c = FALSE,
           prop.t = FALSE,
           prop.chisq = FALSE,
           row.labels = TRUE,
           dnn = c("EDUCATION", "TARGET_FLAG"))

# Tabela 8
CrossTable(dados$JOB,
           dados$TARGET_FLAG,
           prop.r = TRUE,
           prop.c = FALSE,
           prop.t = FALSE,
           prop.chisq = FALSE,
           row.labels = TRUE,
```

```
        dnn = c("JOB", "TARGET_FLAG"))

# Tabela 9
CrossTable(dados$USO_PROD,
        dados$TARGET_FLAG,
        prop.r = TRUE,
        prop.c = FALSE,
        prop.t = FALSE,
        prop.chisq = FALSE,
        row.labels = TRUE,
        dnn = c("USO_PROD", "TARGET_FLAG"))

# Tabela 10
CrossTable(dados$TIPO_PROD,
        dados$TARGET_FLAG,
        prop.r = TRUE,
        prop.c = FALSE,
        prop.t = FALSE,
        prop.chisq = FALSE,
        row.labels = TRUE,
        dnn = c("TIPO_PROD", "TARGET_FLAG"))

# Tabela 11
CrossTable(dados$PROD_ALTO_VALOR,
        dados$TARGET_FLAG,
        prop.r = TRUE,
        prop.c = FALSE,
        prop.t = FALSE,
        prop.chisq = FALSE,
        row.labels = TRUE,
        dnn = c("PROD_ALTO_VALOR", "TARGET_FLAG"))

# Tabela 12
CrossTable(dados$TRAN_SUSP,
        dados$TARGET_FLAG,
        prop.r = TRUE,
        prop.c = FALSE,
        prop.t = FALSE,
        prop.chisq = FALSE,
        row.labels = TRUE,
        dnn = c("TRAN_SUSP", "TARGET_FLAG"))

# Tabela 13
CrossTable(dados$URBANICITY,
        dados$TARGET_FLAG,
        prop.r = TRUE,
        prop.c = FALSE,
        prop.t = FALSE,
        prop.chisq = FALSE,
        row.labels = TRUE,
        dnn = c("URBANICITY", "TARGET_FLAG"))

# Tabela 14
CrossTable(dados$IDADE,
        dados$TARGET_FLAG,
        prop.r = TRUE,
        prop.c = FALSE,
        prop.t = FALSE,
        prop.chisq = FALSE,
        row.labels = TRUE,
        dnn = c("IDADE", "TARGET_FLAG"))

# Tabela 15
CrossTable(dados$YOJ,
        dados$TARGET_FLAG,
        prop.r = TRUE,
        prop.c = FALSE,
        prop.t = FALSE,
        prop.chisq = FALSE,
        row.labels = TRUE,
        dnn = c("YOJ", "TARGET_FLAG"))

# Tabela 16
CrossTable(dados$TEMPO_VIDA_PROD,
        dados$TARGET_FLAG,
        prop.r = TRUE,
        prop.c = FALSE,
        prop.t = FALSE,
```

```
        prop.chisq = FALSE,
        row.labels = TRUE,
        dnn = c("TEMPO_VIDA_PROD", "TARGET_FLAG"))

# Agora preparamos as variáveis categóricas, criando variáveis dummy
# (classes das variáveis categóricas com uma representação numérica)
# Nosso objetivo é usar variáveis binárias para representar a maioria das informações.
dados$CONTA_CLI_Ind <- ifelse(dados$CONTA_CLI == 'Yes', 1, 0)
dados$CASADO_Ind <- ifelse(dados$CASADO == 'Yes', 0, 1)
dados$SEX_Ind <- ifelse(dados$SEX == 'M', 0, 1)
dados$Ed_Non_Degree_Ind <- ifelse(dados$EDUCATION == 'Ensino Medio' | dados$EDUCATION == 'Ensino Fundamental', 1, 0)
dados$USO_PROD_Ind <- ifelse(dados$USO_PROD == 'Comercial', 1, 0)
dados$Renda_Variavel_Ind <- ifelse(dados$TIPO_PROD == 'Renda Variavel', 1, 0)
dados$Poupanca_Ind <- ifelse(dados$TIPO_PROD == 'Poupanca', 1, 0)
dados$Criptomeda_Ind <- ifelse(dados$TIPO_PROD == 'Criptomeda', 1, 0)
dados$Renda_Fixa_Ind <- ifelse(dados$TIPO_PROD == 'Renda Fixa', 1, 0)
dados$Tesouro_Direto_Ind <- ifelse(dados$TIPO_PROD == 'Tesouro Direto', 1, 0)
dados$TRAN_SUSP_Ind <- ifelse(dados$TRAN_SUSP == 'Yes', 1, 0)
dados$URB_Ind <- ifelse(dados$URBANICITY == 'Urbano', 1, 0)
dados$JOB_White_Collar_Ind <- ifelse(dados$JOB == 'Trabalho Administrativo' | dados$JOB == 'Medico' | dados$JOB == 'Advogado' |
dados$JOB == 'Gerente' | dados$JOB == 'Profissional', 1, 0)
dados$JOB_Blue_Collar_Ind <- ifelse(dados$JOB == 'Trabalho Manual', 1, 0)
dados$JOB_Student_Ind <- ifelse(dados$JOB == 'Estudante', 1, 0)
dados$Home_Owner_else_Renter_Ind <- ifelse(dados$HOME_VAL != 0 , 1, 0)

# Visualiza o dataset
View(dados)

# Visualiza as estatísticas
View(t(basicStats(dados[sapply(dados, is.numeric)])))

# Algumas variáveis podem ser removidas. Vamos preparar a lista com as variáveis.
names(dados)

# Cria a lista com os nomes das variáveis que podem ser removidas
lista_remover <- c('INDEX', 'TARGET_AMT', 'CONTA_CLI', 'CASADO', 'SEX', 'EDUCATION', 'USO_PROD', 'TIPO_PROD', 'TRAN_SUSP',
                   'URBANICITY', 'JOB', 'HOME_VAL', 'VALOR_SOLICITADO', 'PROD_ALTO_VALOR', 'TEMPO_VIDA_PROD', 'TRAVTIME',
'TIF')

# Remove as variáveis do dataset
dados <- dados[,!(names(dados) %in% lista_remover )]

# Confere o resultado
names(dados)
View(dados)

# Vamos criar uma variável aleatória seguindo uma distribuição uniforme que será usada como índice para divisão treino/teste
dados$indice <- runif(n = dim(dados)[1], min = 0, max = 1)

# Se o valor do índice for menor que 0.70 colocaremos no dataset de treino, senão, dataset de teste
dados$treino <- ifelse(dados$indice < 0.70, 1, 0)

# Verificamos o resultado e a proporção 70/30 (treino/teste)
# 0 é para registro de teste
# 1 é para registro de treino
table(dados$treino)
table(dados$treino)/dim(dados)[1]

# Confere o resultado
names(dados)
View(dados)

# Salvamos o dataframe em disco
saveRDS(dados, file= "dados/dados.RData")

##### Modelagem #####

# Carrega os dados do disco
df <- readRDS(file= "dados/dados.RData")
names(df)
View(df)

# Criamos as amostras de treino e teste
df_treino <- subset(df, treino == 1)
dim(df_treino)
df_teste <- subset(df, treino == 0)
dim(df_teste)
```

```
# Modelo Versão 1 com Regressão Logística
# Observe atentamente que estamos criando 3 modelos que depois serão usados no stepAIC.
?glm

# Modelo full com todas as variáveis (exceto as variáveis de controle treino e indice)
modelo_full <- glm(TARGET_FLAG ~ . -treino -indice, family = binomial(link = "logit"), data = df_treino)
summary(modelo_full)
sort(vif(modelo_full), decreasing = TRUE)

# Modelo apenas com o intercepto
modelo_intercepto <- glm(TARGET_FLAG ~ 1, family = binomial(link = "logit"), data = df_treino)
summary(modelo_intercepto)

# Modelo com uma das variáveis
modelo_start <- glm(TARGET_FLAG ~ CLM_FREQ, family = binomial(link = "logit"), data = df_treino)
summary(modelo_start)

# Agora criamos o modelo versão 1 usando stepAIC
# O stepAIC vai eliminando as variáveis e criando diversos modelos. Ao final entrega o modelo com melhor performance.
?stepAIC

# Execute a linha de código abaixo e observe o que ocorre no console
modelo_v1 <- stepAIC(object = modelo_start,
                     scope = list(upper = formula(modelo_full), lower = ~1),
                     direction = c('both'))

summary(modelo_v1)
sort(vif(modelo_v1), decreasing = TRUE)

# Avaliação do Modelo Versão 1

# Definimos o cutoff para a definição de classe.
# Isso é necessário pois o resultado da previsão do modelo é em probabilidade.
# Definiremos o cutoff em 50% de probabilidade e usaremos para classificar em classe 1 ou 0.
cutoff <- 0.50

# Previsões
pred_v1_treino <- predict(modelo_v1, type = "response")
pred_v1_teste <- predict(modelo_v1, newdata = df_teste, type = "response")

# Aplica o cutoff
class_pred_v1_treino <- ifelse(pred_v1_treino > cutoff, 1, 0)
class_pred_v1_teste <- ifelse(pred_v1_teste > cutoff, 1, 0)

# Matriz de confusão
cm_v1_treino <- table(df_treino$TARGET_FLAG, class_pred_v1_treino)
cm_v1_teste <- table(df_teste$TARGET_FLAG, class_pred_v1_teste)

# Acurácia
acc_v1_treino <- sum(diag(cm_v1_treino)) / nrow(df_treino)
print(acc_v1_treino)
acc_v1_teste <- sum(diag(cm_v1_teste)) / nrow(df_teste)
print(acc_v1_teste)

# ROC
roc_v1_treino <- roc(df_treino$TARGET_FLAG, pred_v1_treino)
roc_v1_teste <- roc(df_teste$TARGET_FLAG, pred_v1_teste )

# Plot da Curva ROC
par(mfrow = c(1,2))
plot.roc(roc_v1_treino, col = "green", main = "Curva ROC com Dados de Treino")
plot.roc(roc_v1_teste, col = "green", main = "Curva ROC com Dados de Teste")

# Score AUC
auc(roc_v1_treino)
auc(roc_v1_teste)

# AIC
AIC(modelo_v1)

# Ao final vamos comparar e interpretar as versões dos modelos.

# Modelo Versão 2 com Regressão Logística
# Para essa versão usaremos as variáveis que se mostraram mais relevantes na versão 1 do modelo.
names(df)
modelo_v2 <- glm(TARGET_FLAG ~ CLM_FREQ + URB_Ind + SALARIO + USO_PROD_Ind + TRAN_SUSP_Ind + CONTA_CLI_Ind +
                   HOMEKIDS + Ed_Non_Degree_Ind
 + CASADO_Ind + KIDSDRIV + Renda_Variavel_Ind + Poupanca_Ind +
```

```
              Criptomeda_Ind + Renda_Fixa_Ind + OLDCLAIM + SEX_Ind + Home_Owner_else_Renter_Ind,
            family = binomial(link = "logit"),
            data = df_treino)

summary(modelo_v2)
sort(vif(modelo_v2), decreasing = TRUE)

# Avaliação do Modelo Versão 2

# Definimos o cutoff
cutoff <- 0.50

# Previsões
pred_v2_treino <- predict(modelo_v2, type = "response")
pred_v2_teste <- predict(modelo_v2, newdata = df_teste, type = "response")

# Aplica o cutoff
class_pred_v2_treino <- ifelse(pred_v2_treino > cutoff, 1, 0)
class_pred_v2_teste <- ifelse(pred_v2_teste > cutoff, 1, 0)

# Matriz de confusão
cm_v2_treino <- table(df_treino$TARGET_FLAG, class_pred_v2_treino)
cm_v2_teste <- table(df_teste$TARGET_FLAG, class_pred_v2_teste)

# Acurácia
acc_v2_treino <- sum(diag(cm_v2_treino)) / nrow(df_treino)
print(acc_v2_treino)
acc_v2_teste <- sum(diag(cm_v2_teste)) / nrow(df_teste)
print(acc_v2_teste)

# ROC
roc_v2_treino <- roc(df_treino$TARGET_FLAG, pred_v2_treino)
roc_v2_teste <- roc(df_teste$TARGET_FLAG, pred_v2_teste )

# Plot da Curva ROC
par(mfrow = c(1,2))
plot.roc(roc_v2_treino, col = "magenta", main = "Curva ROC com Dados de Treino")
plot.roc(roc_v2_teste, col = "magenta", main = "Curva ROC com Dados de Teste")

# Score AUC
auc(roc_v2_treino)
auc(roc_v2_teste)

# AIC
AIC(modelo_v2)


# Versão 3 do Modelo com Regressão Logística
# Usaremos as variáveis mais relevantes da versão 2.
modelo_v3 <- glm(TARGET_FLAG ~ CLM_FREQ + URB_Ind + SALARIO + USO_PROD_Ind + TRAN_SUSP_Ind +
                   Ed_Non_Degree_Ind + CASADO_Ind + KIDSDRIV + Poupanca_Ind,
                 family = binomial(link = "logit"),
                 data = df_treino )

summary(modelo_v3)
sort(vif(modelo_v3), decreasing = TRUE)


# Avaliação do Modelo Versão 3

# Definimos o cutoff
cutoff <- 0.50

# Previsões
pred_v3_treino <- predict(modelo_v3, type = "response")
pred_v3_teste <- predict(modelo_v3, newdata = df_teste, type = "response")

# Aplica o cutoff
class_pred_v3_treino <- ifelse(pred_v3_treino > cutoff, 1, 0)
class_pred_v3_teste <- ifelse(pred_v3_teste > cutoff, 1, 0)

# Matriz de confusão
cm_v3_treino <- table(df_treino$TARGET_FLAG, class_pred_v3_treino)
cm_v3_teste <- table(df_teste$TARGET_FLAG, class_pred_v3_teste)

# Acurácia
acc_v3_treino <- sum(diag(cm_v3_treino)) / nrow(df_treino)
print(acc_v3_treino)
```

```
acc_v3_teste <- sum(diag(cm_v3_teste)) / nrow(df_teste)
print(acc_v3_teste)

# ROC
roc_v3_treino <- roc(df_treino$TARGET_FLAG, pred_v3_treino)
roc_v3_teste <- roc(df_teste$TARGET_FLAG, pred_v3_teste )

# Plot da Curva ROC
par(mfrow = c(1,2))
plot.roc(roc_v3_treino, col = "blue", main = "Curva ROC com Dados de Treino")
plot.roc(roc_v3_teste, col = "blue", main = "Curva ROC com Dados de Teste")

# Score AUC
auc(roc_v3_treino)
auc(roc_v3_teste)

# AIC
AIC(modelo_v3)


# Seleção do modelo
# Vamos comparar as métricas dos modelos e escolher qual será levado para produção.

# Acurácia em teste (quanto MAIOR, melhor)
acc_v1_teste
acc_v2_teste
acc_v3_teste

# AUC em teste (quanto MAIOR, melhor)
auc(roc_v1_teste)
auc(roc_v2_teste)
auc(roc_v3_teste)

# AIC dos modelos (quanto MENOR, melhor)
AIC(modelo_v1)
AIC(modelo_v2)
AIC(modelo_v3)

# Vamos trabalhar com o Modelo Versão 1, pois foi o que apresentou melhor performance global.

# Um modelo de Machine Learning nada mais é do que o conjunto de coeficientes que
# foram aprendidos durante o treinamento. Vamos visualizar os coeficientes da versão 1:

options(scipen = 999)
df_coef <- as.data.frame(modelo_v1$coefficients)
print(df_coef)

# Este foi o nosso resultado:

# (Intercept)                  -2.523369633534
# CLM_FREQ                      0.171030108830
# URB_Ind                       2.182256461605
# SALARIO                      -0.000007053102
# USO_PROD_Ind                  0.823279491688
# CONTA_CLI_Ind                 0.327474507031
# TRAN_SUSP_Ind                 0.879376388237
# Poupanca_Ind                 -1.197398965813
# TEMPO_CLIENTE                 0.128643649068
# Home_Owner_else_Renter_Ind   -0.313317550816
# Ed_Non_Degree_Ind             0.484220797137
# KIDSDRIV                      0.432252022246
# CASADO_Ind                    0.472944124897
# Tesouro_Direto_Ind           -1.049482910648
# OLDCLAIM                     -0.000011406187
# Criptomeda_Ind               -0.633869337039
# Renda_Fixa_Ind               -0.650033714145
# SEX_Ind                      -0.240057122738
# Renda_Variavel_Ind           -0.252483050147
# IDADE                        -0.008585587562

# Levamos os valores dos coeficientes para o script de previsões.
# Perceba que alguns coeficentes são valores negativos e outros positivos.
# Como o processo de criação do modelo segue passos aleatórios os resultados podem ser diferentes a cada execução.
```

```
# Mini-Projeto 2 - Análise de Risco em Operações Financeiras

# Neste script usamos o modelo para fazer previsões e salvar em arquivo csv.
```

```
# Pacotes
library(ggplot2)
library(readr)
library(plyr)
library(car)
library(fBasics)
library(gmodels)
library(MASS)
library(gridExtra)
library(pROC)

# Importamos o arquivo com dados de novos clientes
novos_dados <- read_csv("dados/novos_clientes.csv")

# Visão geral dos dados
dim(novos_dados)
summary(novos_dados)
str(novos_dados)
View(novos_dados)

# Preparação dos Dados

# Toda e qualquer limpeza, transformação ou processamento que foram aplicados aos dados de treino
# devem ser aplicados aos novos dados.

novos_dados$IDADE <- ifelse(is.na(novos_dados$IDADE), 45, novos_dados$IDADE)

novos_dados$YOJ <- ifelse(is.na(novos_dados$YOJ), 11, novos_dados$YOJ)

novos_dados$SALARIO <- ifelse(is.na(novos_dados$SALARIO) & is.na(novos_dados$JOB), 54000,
                              ifelse(is.na(novos_dados$SALARIO) & (novos_dados$JOB == "Medico"), 128000,
                                     ifelse(is.na(novos_dados$SALARIO) & (novos_dados$JOB == "Advogado"), 88000,
                                            ifelse(is.na(novos_dados$SALARIO) & (novos_dados$JOB == "Gerente"), 87000,
                                                   ifelse(is.na(novos_dados$SALARIO) & (novos_dados$JOB == "Profissional"), 76000,
                                                          ifelse(is.na(novos_dados$SALARIO) & (novos_dados$JOB == "Trabalho Administrativo"), 33000,
                                                                 ifelse(is.na(novos_dados$SALARIO) & (novos_dados$JOB == "Trabalho Manual"), 58000,
                                                                        ifelse(is.na(novos_dados$SALARIO) & (novos_dados$JOB == "Do Lar"), 12000,
                                                                               ifelse(is.na(novos_dados$SALARIO) & (novos_dados$JOB == "Estudante"), 6300,
                                                                                      novos_dados$SALARIO)))))))))

novos_dados$HOME_VAL <- ifelse(is.na(novos_dados$HOME_VAL), 162000, novos_dados$HOME_VAL)

novos_dados$TEMPO_VIDA_PROD <- ifelse(is.na(novos_dados$TEMPO_VIDA_PROD), 8,
                                      ifelse(novos_dados$TEMPO_VIDA_PROD < 0, 0,
                                             novos_dados$TEMPO_VIDA_PROD))

novos_dados$JOB <- ifelse(is.na(novos_dados$JOB) & novos_dados$SALARIO > 150000, "Medico",
                          ifelse(is.na(novos_dados$JOB) & novos_dados$SALARIO > 100000, "Advogado",
                                 ifelse(is.na(novos_dados$JOB) & novos_dados$SALARIO > 85000, "Gerente",
                                        ifelse(is.na(novos_dados$JOB) & novos_dados$SALARIO > 75000, "Profissional",
                                               ifelse(is.na(novos_dados$JOB) & novos_dados$SALARIO > 60000, "Trabalho Administrativo",
                                                      ifelse(is.na(novos_dados$JOB) & novos_dados$SALARIO > 35000, "Trabalho Manual",
                                                             ifelse(is.na(novos_dados$JOB) & novos_dados$SALARIO >= 12000, "Do Lar",
                                                                    ifelse(is.na(novos_dados$JOB) & novos_dados$SALARIO < 12000, "Estudante",
                                                                           novos_dados$JOB)))))))))

View(novos_dados)

# Criamos então as variáveis que também foram criadas para treinar o modelo.

novos_dados$CONTA_CLI_Ind <- ifelse(novos_dados$CONTA_CLI == 'Yes', 1, 0)
novos_dados$CASADO_Ind <- ifelse(novos_dados$CASADO == 'Yes', 0, 1)
novos_dados$SEX_Ind <- ifelse(novos_dados$SEX == 'M', 0, 1)
novos_dados$Ed_Non_Degree_Ind <- ifelse(novos_dados$EDUCATION == 'Ensino Medio' | novos_dados$EDUCATION == 'Ensino
Fundamental', 1, 0)
novos_dados$USO_PROD_Ind <- ifelse(novos_dados$USO_PROD == 'Comercial', 1, 0)
novos_dados$Renda_Variavel_Ind <- ifelse(novos_dados$TIPO_PROD == 'Renda Variavel', 1, 0)
novos_dados$Poupanca_Ind <- ifelse(novos_dados$TIPO_PROD == 'Poupanca', 1, 0)
novos_dados$Criptomeda_Ind <- ifelse(novos_dados$TIPO_PROD == 'Criptomeda', 1, 0)
novos_dados$Renda_Fixa_Ind <- ifelse(novos_dados$TIPO_PROD == 'Renda Fixa', 1, 0)
novos_dados$Tesouro_Direto_Ind <- ifelse(novos_dados$TIPO_PROD == 'Tesouro Direto', 1, 0)
novos_dados$TRAN_SUSP_Ind <- ifelse(novos_dados$TRAN_SUSP == 'Yes', 1, 0)
novos_dados$URB_Ind <- ifelse(novos_dados$URBANICITY == 'Urbano', 1, 0)
novos_dados$JOB_White_Collar_Ind <- ifelse(novos_dados$JOB == 'Trabalho Administrativo' | novos_dados$JOB == 'Medico' |
novos_dados$JOB == 'Advogado' | novos_dados$JOB == 'Gerente' | novos_dados$JOB == 'Profissional', 1, 0)
novos_dados$JOB_Blue_Collar_Ind <- ifelse(novos_dados$JOB == 'Trabalho Manual', 1, 0)
novos_dados$JOB_Student_Ind <- ifelse(novos_dados$JOB == 'Estudante', 1, 0)
```

```
novos_dados$Home_Owner_else_Renter_Ind <- ifelse(novos_dados$HOME_VAL != 0 , 1, 0)

View(novos_dados)

# Como a previsão do modelo é uma equação matemática, qualquer coluna que tenha valores NA vai gerar como previsão valor NA.
# Se tivermos valores NA nos novos dados que não apareceram nos dados de treino, o ideal é treinar o modelo com essa regra.
# Não podemos definir uma nova regra de tratamento de valores ausentes em novos dados se essa regra não fez parte do treino.

# Deploy do Modelo

# Vamos montar a equação de previsão do modelo usando novos dados e os
# coeficientes do modelo escolhido após na fase de modelagem.

resultado <- with(novos_dados, -2.523369633534
                  + 0.171030108830 * CLM_FREQ
                  + 2.182256461605 * URB_Ind
                  - 0.000007053102 * SALARIO
                  + 0.823279491688 * USO_PROD_Ind
                  + 0.327474507031 * CONTA_CLI_Ind
                  + 0.879376388237 * TRAN_SUSP_Ind
                  + 0.126402607277 * TEMPO_CLIENTE
                  - 1.197398965813 * Poupanca_Ind
                  + 0.128643649068 * TEMPO_CLIENTE
                  - 0.313317550816 * Home_Owner_else_Renter_Ind
                  + 0.484220797137 * Ed_Non_Degree_Ind
                  + 0.432252022246 * KIDSDRIV
                  + 0.472944124897 * CASADO_Ind
                  - 1.049482910648 * Tesouro_Direto_Ind
                  - 0.000011406187 * OLDCLAIM
                  - 0.633869337039 * Criptomeda_Ind
                  - 0.650033714145 * Renda_Fixa_Ind
                  - 0.240057122738 * SEX_Ind
                  - 0.252483050147 * Renda_Variavel_Ind
                  - 0.008585587562 * IDADE)

# Calcuamos o exponencial do resultado (cálculo matemático da regressão logística)
ODDS <- exp(resultado)

# Fazemos as previsões
Previsoes_TARGET_FLAG <- ODDS / (1 + ODDS)

# Gravamos os resultados das previsões no dataframe
novos_dados$Previsoes_TARGET_FLAG <- Previsoes_TARGET_FLAG
View(novos_dados)

# Definimos o cutoff
cutoff <- 0.50

# Gravamos as previsões conforme regra do cutoff
novos_dados$TARGET_FLAG <- ifelse(Previsoes_TARGET_FLAG > cutoff, 1, 0)

# Visualiza os dados
View(novos_dados)

# Vamos salvar as previsões em um dataframe e depois exportar para CSV.
# Incluiremos algumas colunas (você pode incluir outras se desejar)
names(novos_dados)
resultado_final <- with(novos_dados, cbind.data.frame(INDEX, IDADE, EDUCATION, JOB, SALARIO, TARGET_FLAG))

# Ajustamos os nomes das colunas
colnames(resultado_final) <- c("Id_Cliente", "Idade", "Escolaridade", "Trabalho", "Salario", "Previsao_Risco")
View(resultado_final)

# Vamos deixar mais amigável para o tomador de decisão
resultado_final$Previsao_Risco <- ifelse(resultado_final$Previsao_Risco > 0, "Alto Risco", "Baixo Risco")
View(resultado_final)

# Gravamos o arquivo em disco com as previsões
write.csv(resultado_final, "dados/previsoes.csv")

# Agora leve o dataframe com as previsões para uma ferramenta de criação de gráficos como o Power BI, crie visualizações e
# apresente o resultado ao tomador de decisão.
```

8.11. Fraud Analytics (Análise Para Detecção de Fraudes)

O Que é Fraud Analytics?

Fraud Analytics não é uma atividade trivial, pois os fraudadores normalmente empregam técnicas avançadas que são difíceis de detectar e identificar.

Os fraudadores podem também adotar novas técnicas de fraude, resultando em novos padrões de comportamento, dificultando ainda mais a tarefa de detecção.

Estima-se que uma organização perde cerca de 5% de suas receitas devido à fraude, a cada ano.

Fraud Analytics é a aplicação da Ciência de Dados para detectar, identificar e prever possíveis anomalias que indiquem uma possível fraude.

O maior desafio em Fraud Analytics é o baixo volume de transações/operações fraudulentas. Em cada 1000 transações podemos ter 1, 2 ou mesmo 0 registros que indiquem fraude.

O Impacto das Fraudes na Economia

- Uma organização perde 5% de suas receitas por fraude anualmente
- O custo total da fraude de seguro nos EUA é estimado em mais de U$ 40 bilhões por ano
- A fraude custa ao Reino Unido 73 bilhões de libras por ano
- Empresas de cartões de crédito “perdem aproximadamente sete centavos por cada cem dólares de transações devido a fraude”
- O tamanho médio da economia informal nos países em desenvolvimento é de 41%, nos países em transição de 38% e nos países da OCDE de 18%.

Conceito de Fraude

Com o passar do tempo, os fraudadores se tornaram cada vez mais profissionais para encontrar formas de burlar sistemas ou processos e isso tornou a prevenção à fraude cada vez mais complexa e desafiadora. Mas antes que possamos falar em prevenção e detecção de fraudes, precisamos definir o que é fraude.

Segundo o dicionário, a definição de fraude está relacionada à distorção intencional da verdade ou de um fato, que busca em geral a obtenção de lucro ilícito.

Por um lado, esta definição capta a essência da fraude e abrange as diferentes formas e tipos de fraude que serão discutidos daqui a pouco. Por outro lado, não descreve com muita precisão a natureza e as características da fraude e, como tal, não proporciona muita orientação para discutir os requisitos de um sistema de detecção de fraude. Vamos compreender um pouco melhor o que é e o que motiva atividades fraudulentas.

A fraude não é definitivamente um fenômeno recente, exclusivo da sociedade moderna. A fraude é um crime raro, imperceptivelmente escondido, que evolui no tempo e muitas vezes cuidadosamente organizado, que aparece em muitos tipos de formas.

Independentemente da definição exata ou da aplicação, apenas uma minoria da população está realmente envolvida em casos tipicamente relacionados à fraude, dos quais, além disso, apenas um

número limitado será realmente detectado como fraude. Isso torna ainda mais difícil a detecção de fraude, uma vez que os casos fraudulentos são cobertos pelos não fraudulentos, o que também dificulta a construção de sistemas de detecção de fraudes, uma vez que poucos exemplos de atividades fraudulentas estão disponíveis e, como você sabe, para treinar modelos de Machine Learning, precisamos de muitos exemplos de fatos históricos.

Outra característica dos fraudadores, é a capacidade de se manterem anônimos para que não sejam notados e para permanecer cobertos por não fraudadores. Isso efetivamente torna a fraude imperceptivelmente escondida, já que os fraudadores conseguem se esconder planejando como cometer fraudes com precisão. Seu comportamento não é definitivamente impulsivo ou não planejado, uma vez que se fosse, a detecção seria muito mais fácil.

Análise de Redes Sociais Como Ferramenta de Detecção de Fraudes

Fraude é na maioria das vezes um crime cuidadosamente organizado, o que significa que os fraudadores muitas vezes não operam independentemente, têm aliados e podem induzir colaboradores.

Além disso, vários tipos de fraude, envolvem estruturas complexas que são criadas para cometer fraudes de forma organizada. Isso torna a fraude não um evento isolado e, como tal, a fim de detectar a fraude, o contexto (por exemplo, a rede social de fraudadores) deve ser levado em conta.

Pesquisas mostram que as empresas fraudulentas estão, de fato, mais ligadas a outras empresas fraudulentas do que a empresas não fraudulentas, como mostra um estudo de caso de evasão fiscal de empresas, cujo link você encontra na seção de links úteis.

As análises de redes sociais para a detecção de fraudes, parecem ser uma poderosa ferramenta para desmascarar a fraude, fazendo uso inteligente de informações contextuais que descrevem a rede ou o ambiente de uma entidade. Um último elemento na descrição da fraude indica os muitos tipos diferentes de formas em que a fraude ocorre, e falaremos mais sobre isso nas próximas aulas.

O Triângulo da Fraude

Mas as fraudes não são ações apenas de criminosos. O chamado triângulo da fraude, fornece uma explicação mais elaborada dos motivos para cometer fraude ocupacional.

O triângulo da fraude se origina de uma hipótese formulada por Donald R. Cressey em seu livro de 1953, chamado “O dinheiro de outras pessoas: um estudo da psicologia social do desfalque” (você encontra o link do livro na seção de links úteis). Donald dizia:

“Pessoas confiáveis tornam-se violadores de confiança quando se concebem como tendo um problema financeiro que não é compartilhável. Estão conscientes de que este problema pode ser secretamente resolvido por violação da posição de confiança financeira e são capazes de aplicar a sua própria conduta nessa situação, verbalizações que lhes permitam ajustar suas concepções de si mesmos como pessoas de confiança, com suas concepções de si mesmos como usuários dos fundos ou bens confiados”.

A linguagem é um pouco complexa, já que remonta há muitas décadas atrás, mas vamos simplificar: o que Donald quis dizer é que muitas vezes, a fraude é cometida por alguém sem histórico de crimes, mas que devido a uma situação não prevista, se vale de sua condição para fraudar sistemas e obter ganhos. São conhecidos muitos casos de gerentes que fraudaram sistemas financeiros da

empresa para pagar dívidas pessoais, sob o pretexto que devolveriam o dinheiro assim que pudessem. Isso não torna esse tipo de fraude algo lícito, mas dificulta sua detecção, uma vez que não há históricos anteriores da pessoa que o cometeu.

Este modelo do triângulo da fraude, conceitual e básico, explica os fatores que, juntos, causam ou explicam os fatores que levam um indivíduo a cometer fraude ocupacional, e fornece uma visão útil do fenômeno de fraude também de um ponto de vista mais amplo. O modelo tem três pernas que juntos instituem comportamento fraudulento:

Motivação é a primeira etapa e diz respeito à principal motivação para cometer fraude. Um indivíduo comete a fraude porque está sob pressão devido a um problema de natureza financeira, social, ou de qualquer outra natureza, e não pode ser resolvido ou aliviado em uma maneira autorizada.

Oportunidade é a segunda etapa do modelo, e diz respeito à condição prévia para que um indivíduo possa cometer fraude. Atividades fraudulentas só podem ser cometidas quando existe a oportunidade de o indivíduo resolver ou aliviar a pressão ou o problema experimentado de forma não autorizada, mas de forma dissimulada ou escondida. Por isso, muitas vezes a melhor forma de prevenir fraudes é deixar mais de uma pessoa responsável por determinada atividade.

Racionalização é o mecanismo psicológico que explica porque os fraudadores não se abstêm de cometer a fraude e consideram sua conduta como aceitável.

Se os três elementos estiverem presentes, são grandes as chances de ocorrer uma fraude.

Principais Tipos de Fraudes

- Fraude de Cartão de Crédito
- Fraude de Seguro
- Corrupção
- Fraude em Votação
- Fraude de Garantia do Produto
- Fraude em Planos de Saúde
- Fraude de Cliques e de Transações Online
- Fraude de Imóveis e Financiamentos
- Lavagem de Dinheiro
- Evasão Fiscal
- Pirataria
- Scam

Detecção de Fraude x Prevenção de Fraude

Dois componentes que são partes essenciais de quase qualquer estratégia eficaz para combater a fraude envolvem a detecção de fraude e prevenção de fraude.

Detecção de fraude refere-se à capacidade de reconhecer ou descobrir atividades fraudulentas. A prevenção de fraude refere-se a medidas que podem ser tomadas para evitar ou reduzir a fraude.

A diferença entre ambos é clara: a primeira é uma abordagem reativa, enquanto a segunda uma abordagem proativa. Ambas as estratégias podem e provavelmente devem ser usadas de forma complementar para perseguir o objetivo compartilhado, a redução da fraude. Contudo, as ações preventivas mudarão as estratégias de fraude e, consequentemente, o poder de impacto da detecção. A instalação de um sistema de detecção fará com que os fraudadores se adaptem e alterem o seu comportamento, pelo que o próprio sistema de detecção irá prejudicar eventualmente a sua própria potência de detecção. Portanto, a detecção e a prevenção de fraudes não são independentes e devem ser alinhadas e consideradas como um todo.

Sistema de Regras Para Detecção de Fraudes

A compreensão do mecanismo ou padrão de fraude é muitas vezes implementado com base em regras, ou seja, sob a forma de um conjunto de regras Se-Então, adicionando regras que descrevem o mecanismo de fraude recentemente detectado.

Estas regras, junto a as regras que descrevem padrões de fraude detectados anteriormente, são aplicadas a casos ou transações futuras e ativam um alerta ou sinal quando a fraude é ou pode ser cometida pelo uso deste mecanismo. Um exemplo simples, mas possivelmente muito eficaz, de uma regra de detecção de fraude em um cenário de fraude de reivindicação de seguro seria esse:

SE:

- O montante doo sinistro está acima de um limite OU
- Um acidente grave, mas nenhum relatório policial OU
- Lesão grave, mas nenhum relatório médico OU
- O requerente tem múltiplas versões do acidente OU
- Múltiplos recibos enviados

ENTÃO:

- Reivindicação recebe um sinal de alerta, sendo colocada como suspeita

Confiar apenas na tecnologia para aprovar ou reprovar pedidos já não é suficiente, é preciso também contar com uma análise humana sofisticada. Isso porque enquanto modelos estatísticos são ferramentas valiosas na luta contra fraude, ao mesmo tempo requerem contribuição humana e discernimento para criar uma solução abrangente que produza os melhores resultados. Não há dúvida de que as ferramentas de análise automática são componentes importantíssimos em programas de redução de fraude, mas depender somente delas pode não ser eficiente no longo prazo. O motivo é que a tecnologia sozinha não é eficaz para definir ou identificar todos os aspectos subjetivos que envolvem uma transação fraudulenta.

Mini-Projeto 3 – Data Quality Report e Detecção de Fraudes em Transações Imobiliárias

Uma empresa fornece seguros para imóveis na cidade de Nova Iorque nos EUA. Visando calcular o valor do seguro da melhor forma possível, a empresa contratou você, Cientista de Dados, para uma

análise e detecção de fraude nos dados de transações imobiliárias de toda cidade. Os dados são públicos e fornecidos pelo portal de dados abertos da cidade de Nova Iorque.

Primeiro a empresa precisa de um relatório que demonstre que os dados são confiáveis e coerentes e que a qualidade da informação que eles oferecem pode ser usada para prever possíveis fraudes. Sendo assim, você deve apresentar um DQR – Data Quality Report.

Na segunda etapa a empresa precisa de um score para cada transação a fim de checar aquelas com maior possibilidade de ter alguma fraude. O Score de Fraude (Fraud Score) deve ser o mais preciso possível e deve ser gerado um score para cada transação. Não há informação prévia se uma transação foi ou não fraudulenta.

Seu trabalho é fazer isso acontecer

```
# <font color='blue'>Data Science Academy</font>
# <font color='blue'>Business Analytics</font>
# <font color='blue'>Capítulo 11 - Fraud Analytics</font>
## <font color='blue'>Mini-Projeto 3</font>
### <font color='blue'>Data Quality Report e Detecção de Fraudes em Transações Imobiliárias</font>
## Etapa 1 - Data Quality Report (DQR)
O DQR é um relatório analítico com o objetivo de compreender a organização dos dados, se estão coerentes, se há alguma anomalia amplamente visível e resumir os dados (ou pelo menos as variáveis mais importantes) com base na compreensão do problema de negócio.
A descrição do que representa cada variável está disponível no Dicionário de Dados e no material complementar no link abaixo:
https://www1.nyc.gov/site/finance/taxes/definitions-of-property-assessment-terms.page
![title](imagens/mini-projeto3.png)
# ->
# Versão da Linguagem Python
from platform import python_version
print('Versão da Linguagem Python Usada Neste Jupyter Notebook:', python_version())

## Instalando e Carregando os Pacotes
# ->
# Para atualizar um pacote, execute o comando abaixo no terminal ou prompt de comando:
# pip install -U nome_pacote
# Para instalar a versão exata de um pacote, execute o comando abaixo no terminal ou prompt de comando:
# !pip install nome_pacote==versão_desejada
# Depois de instalar ou atualizar o pacote, reinicie o jupyter notebook.
# Instala o pacote watermark.
# Esse pacote é usado para gravar as versões de outros pacotes usados neste jupyter notebook.
!pip install -q -U watermark

# ->
# Imports
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.pyplot as pyplot
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
pd.set_option('display.float_format', lambda x : '%.2f' % x)
%matplotlib inline

# ->
# Versões dos pacotes usados neste jupyter notebook
%reload_ext watermark
%watermark -a "Data Science Academy" --iversions

## Resumo dos Dados
# ->
# Carrega os dados
dados = pd.read_csv('dados/dataset.csv', index_col = 0)

# ->
# Shape
dados.shape

# ->
# Visualiza
dados.head()

# ->
```

```
# Resumo
print("Linhas: ", dados.shape[0])
print("Colunas: ", dados.shape[1])
print("\nVariáveis: \n", dados.columns.tolist())
print("\nValores Ausentes: \n" , dados.isnull().sum())
print("\nValores Únicos: \n", dados.nunique())

# ->
# Info
dados.info()

# ->
# Colunas numéricas (quantitativas)
num_cols = ['LTFRONT', 'LTDEPTH', 'STORIES', 'FULLVAL', 'AVLAND', 'AVTOT', 'EXLAND', 'EXTOT', 'BLDFRONT', 'BLDDEPTH', \
            'AVLAND2', 'AVTOT2', 'EXLAND2', 'EXTOT2']

# ->
# Colunas categóricas
cat_cols = ['BBLE', 'B', 'BLOCK', 'LOT', 'EASEMENT', 'OWNER', 'BLDGCL', 'TAXCLASS', 'EXT', 'EXCD1', 'STADDR', 'ZIP',\
            'EXMPTCL', 'EXCD2', 'PERIOD', 'YEAR', 'VALTYPE']

# ->
# Verifica se o total de variáveis está coberto nos objetos anteriores
len(num_cols) + len(cat_cols) == 31

# ->
# Dataframes com os tipos diferentes de variáveis
df_num = dados[num_cols]
df_cat = dados[cat_cols]

# ->
# Sumário estatístico das variáveis numéricas
summ_num = pd.DataFrame(index = df_num.columns)
summ_num['Tipo de Dado'] = df_num.dtypes.values
summ_num['# Registros Não Nulos'] = df_num.count().values
summ_num['# Registros Não Zero'] = df_num.astype(bool).sum(axis = 0)
summ_num['% Populado'] = round(summ_num['# Registros Não Nulos'] / df_num.shape[0]*100,2)
summ_num['# Valores Únicos'] = df_num.nunique().values
summ_num['Mean'] = round(df_num.mean(),2)
summ_num['Std'] = round(df_num.std(),2)
summ_num['Min'] = round(df_num.min(),2)
summ_num['Max'] = round(df_num.max(),2)
summ_num

# ->
# Sumário estatístico das variáveis categóricas
summ_cat = pd.DataFrame(index = df_cat.columns)
summ_cat['Tipo de Dado'] = df_cat.dtypes.values
summ_cat['# Registros Não Nulos'] = df_cat.count().values
summ_cat['% Populado'] = round(summ_cat['# Registros Não Nulos'] / df_cat.shape[0]*100,2)
summ_cat['# Valores Únicos'] = df_cat.nunique().values

# ->
# Adiciona mais uma coluna com valores mais comuns
temp = []
for col in cat_cols:
    temp.append(df_cat[col].value_counts().idxmax())
summ_cat['Valores Mais Comuns'] = temp

# ->
summ_cat

## Identificação, Exploração e Visualização das Variáveis
# ->
# Variáveis
dados.columns

# ->
# Visualiza
dados.head()

**Variável 1** \
Nome da Variável: BBLE \
Descrição: Concatenação de código Borough, código de bloco, código LOT; um número exclusivo para cada registro.
**Variável 2** \
Nome da Variável: B \
Descrição: Códigos Borough
# ->
```

```
# Visualização da variável 2
sns.set_theme(style = 'whitegrid')
plt.figure(figsize = (12, 6))
fig1 = sns.countplot(x = 'B', data = dados, order = dados['B'].value_counts().index)
plt.title("Número de Propriedades em Diferentes Bairros")

**Variável 3** \
Nome da Variável: BLOCK \
Descrição: Número de até 5 dígitos que representam códigos de bloco em diferentes bairros
# ->
# Contagem
BLOCK = df_cat['BLOCK'].value_counts().rename_axis('Unique_values_BLOCK').reset_index(name = 'Counts')
BLOCK[:15]

**Variável 4** \
Nome da Variável: LOT \
Descrição: Número de até 4 dígitos que representam códigos de lote em diferentes Borough & Block
# ->
# Contagem
LOT = df_cat['LOT'].value_counts().rename_axis('Unique_values_LOT').reset_index(name = 'Counts')[:15]
LOT[:15]

**Variável 5** \
Nome da Variável: EASEMENT  \
Descrição: Tipos de easement
# ->
# Visualização da variável 5
sns.set_theme(style = 'whitegrid')
plt.figure(figsize = (12, 6))
fig2 = sns.countplot(x = 'EASEMENT', data = dados, order = dados['EASEMENT'].value_counts().index)
fig2.set_yscale("log")
fig2.set_title('Quantidade de Imóveis com Diversos Tipos de Easement')

**Variável 6** \
Nome da Variável: OWNER \
Descrição: Proprietários dos imóveis
# ->
# Contagem
OWNER = df_cat['OWNER'].value_counts().rename_axis('Unique_values_OWNER').reset_index(name = 'Counts')

# ->
OWNER.head()

# ->
OWNER.tail()

**Variável 7** \
Nome da Variável: BLDGCL \
Descrição: Classe do imóvel
# ->
# Contagem
BLDGCL = df_cat['BLDGCL'].value_counts().rename_axis('Unique_values_BLDCGL').reset_index(name = 'Counts')

# ->
BLDGCL.tail()

# ->
BLDGCL.head()

**Variável 8** \
Nome da Variável: TAXCLASS \
Descrição: Código de classe de imposto de propriedade
# ->
# Visualização da variável 8
sns.set_theme(style = 'darkgrid')
plt.figure(figsize = (14, 6))
fig3 = sns.countplot(x = 'TAXCLASS', data = dados, order = dados['TAXCLASS'].value_counts().index)
fig3.set_yscale("log")
fig3.set_title('Número de Propriedades com Vários Tipos de Classe')

**Variável 9** \
Nome da Variável: LTFRONT \
Descrição: Frente do lote em pés (feet)
# ->
# Divide em percentis
dados['LTFRONT'].describe(percentiles = [0.5,0.75,0.995])

# ->
```

```
# Filtra por valores iguais ou menores que 375
tmp = dados[dados['LTFRONT'] <= 375]

# ->
# Visualização da variável 9
sns.set_theme(style = 'whitegrid')
plt.figure(figsize = (10, 5))
fig4 = sns.distplot(tmp.LTFRONT, bins = 50)
fig4.set_title('Distribuição da Variável LTFRONT')
```

Variável 10 \
Nome da Variável: LTDEPTH \
Descrição: Profundidade do lote em pés (feet)

```
# ->
# Divide em percentis
dados['LTDEPTH'].describe(percentiles = [0.18,0.25,0.4,0.83,0.98,0.9995])

# ->
# Filtra por valores iguais ou menores que 425
tmp = dados[dados['LTDEPTH'] <= 425]

# ->
# Visualização da variável 10
sns.set_theme(style = 'whitegrid')
plt.figure(figsize = (12, 6))
fig5 = sns.distplot(tmp.LTDEPTH, bins = 50)
fig5.set_yscale("log")
fig5.set_title('Distribuição da Variável LTDEPTH')
```

Variável 11 \
Nome da Variável: EXT \
Descrição: E- Extension, G- Garage, EG- Extension e Garage

```
# ->
# Visualização da variável 11
sns.set_theme(style = 'whitegrid')
plt.figure(figsize = (12, 6))
fig6 = sns.countplot(x = 'EXT', data = dados, order = dados['EXT'].value_counts().index)
fig6.set_title('Número de Propriedades com Vários Tipos de Extensões / Garagem')
```

Variável 12 \
Nome da Variável: STORIES \
Descrição: Número de andares do edifício

```
# ->
# Divide em percentis
dados['STORIES'].describe(percentiles = [0.5,0.75,0.995])

# ->
# Filtra
tmp = dados[dados['STORIES'] <= 50]

# ->
# Visualização da variável 12
sns.set_theme(style = 'whitegrid')
plt.figure(figsize = (12, 6))
fig7 = sns.distplot(tmp['STORIES'], kde = False, bins = 50)
fig7.set_yscale('log', basey = 2)
fig7.set_title('Distribuição de Andares das Propriedades')
```

Variável 13 \
Nome da Variável: FULLVAL \
Descrição: Valor de Mercado Total

```
# ->
# Divide em percentis
dados['FULLVAL'].describe(percentiles = [0.5,0.75,0.95])

# ->
# Visualiza os dados
dados['FULLVAL'].head()

# ->
# Filtra os dados para simplificar a visualização e evitar valores extremos
tmp = dados[dados['FULLVAL'] <= 3000000]

# ->
# Visualização da variável 13
dimensoes = (12, 8)
fig, ax = pyplot.subplots(figsize = dimensoes)
fig8 = sns.distplot(tmp.FULLVAL, kde = False, bins = 70)
```

```
fig8.set_title('Distribuição do Valor de Mercado Total das Propriedades')

**Variável 14** \
Nome da Variável: AVLAND \
Descrição: Valor de mercado do terreno
# ->
# Divide em percentis
dados['AVLAND'].describe(percentiles = [0.5,0.75,0.95])

# ->
# Filtra os dados
tmp = dados[dados['AVLAND'] <= 50000]

# ->
# Visualização da variável 14
dimensoes = (12, 8)
fig, ax = pyplot.subplots(figsize = dimensoes)
fig9 = sns.distplot(tmp.AVLAND, kde = False, bins = 80)
fig9.set_title('Distribuição do Valor de Mercado do Terreno das Propriedades')

**Variável 15** \
Nome da Variável: EXLAND \
Descrição: Valor provisório do terreno com isenção temporária
# ->
# Divide em percentis
dados['EXLAND'].describe(percentiles = [0.5,0.75,0.95])

# ->
# Filtro
tmp = dados[dados['EXLAND'] <= 20000]

# ->
# Visualização da variável 15
dimensoes = (14, 8)
fig, ax = pyplot.subplots(figsize = dimensoes)
fig11 = sns.distplot(tmp.EXLAND, kde = False, bins = 100)
fig11.set_yscale('log', basey = 2)
fig11.set_title('Valor Provisório do Terreno com Isenção Temporária')

# Fim
```

```
# <font color='blue'>Data Science Academy</font>
# <font color='blue'>Business Analytics</font>
# <font color='blue'>Capítulo 11 - Fraud Analytics</font>
## <font color='blue'>Mini-Projeto 3</font>
### <font color='blue'>Data Quality Report e Detecção de Fraudes em Transações Imobiliárias</font>
## Etapa 2 - Modelagem
- Existem diversas técnicas para análise e detecção de fraude.
- Aplicaremos aqui uma abordagem via aprendizado **não supervisionado** criando scores (pontuações) de fraude para cada transação imobiliária.
- Serão criados 2 scores com técnicas diferentes de Machine Learning e depois vamos unir os scores e apresentar o score final.
- A Engenharia de Atributos será parte fundamental do processo.
- Este trabalho tem nível intermediário de dificuldade.
![title](imagens/mini-projeto3.png)
# ->
# Versão da Linguagem Python
from platform import python_version
print('Versão da Linguagem Python Usada Neste Jupyter Notebook:', python_version())

## Instalando e Carregando os Pacotes
# ->
# Para atualizar um pacote, execute o comando abaixo no terminal ou prompt de comando:
# pip install -U nome_pacote
# Para instalar a versão exata de um pacote, execute o comando abaixo no terminal ou prompt de comando:
# !pip install nome_pacote==versão_desejada
# Depois de instalar ou atualizar o pacote, reinicie o jupyter notebook.
# Instala o pacote watermark.
# Esse pacote é usado para gravar as versões de outros pacotes usados neste jupyter notebook.
!pip install -q -U watermark

# ->
# Imports
import math
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)
```

```
import warnings
warnings.filterwarnings("ignore")

# ->
# Versões dos pacotes usados neste jupyter notebook
%reload_ext watermark
%watermark -a "Data Science Academy" --iversions

## Carregando os Dados
# ->
# Carrega os dados
df = pd.read_csv('dados/dataset.csv')

# ->
# Shape
df.shape

# ->
# Visualiza amostra dos dados
df.head()

# ->
# Colunas
df.columns

# ->
# Resumo
print("Linhas: ", df.shape[0])
print("Colunas: ", df.shape[1])
print("\nVariáveis: \n", df.columns.tolist())
print("\nValores Ausentes: \n" , df.isnull().sum())
print("\nValores Únicos: \n", df.nunique())

# ->
# Vamos trabalhar com uma cópia do dataframe
df_temp = df.copy()

## Limpeza e Transformação dos Dados
### Limpando Valores Ausentes da Variável ZIP
# ->
# Contagem de valores ausentes
df_temp['ZIP'].isna().sum()

# ->
# Vamos calcular o valor da variável zip que aparece com maior frequência (calcular a moda)
# Se não houver nenhum valor de moda, substitua por 1
def calcula_moda(x):
    m = pd.Series.mode(x)
    if m.empty:
        x = np.array(range(1,11))
        return x[0]
    else:
        return m.values[0]

# ->
# Definindo a função de contagem para calcular a frequência do valor da moda de cada grupo (que vamos definir)
def contagem(x):
    return x.value_counts().head(1)

# ->
# Agrupando valores pela variável 'B' e 'BLOCK' e usando as duas funções acima para criar dataframe auxiliar
# Teremos o valor da moda da variável ZIP para cada grupo com registros com as variáveis B e Block
df_zip_group = df_temp.groupby(['B','BLOCK'])['ZIP'].agg(ZIP = calcula_moda, Count = contagem).reset_index()

# ->
# Verificando os 25 primeiros registros
df_zip_group.head(25)

# ->
# Para os grupos sem valor de moda, comparamos a frequência do grupo, um grupo abaixo e um grupo acima
# Inserimos o valor do ZIP do grupo com maior frequência
for i in range(len(df_zip_group)):
    if (df_zip_group.loc[i,'ZIP'] == 1):
        if df_zip_group.loc[i - 1,'Count'] > df_zip_group.loc[i + 1,'Count']:
            val = df_zip_group.loc[i - 1,'ZIP']
        else:
            val = df_zip_group.loc[i + 1,'ZIP']
        df_zip_group.loc[i,'ZIP'] = val
```

```
# ->
# Verificando os 25 primeiros registros
df_zip_group.head(25)

# ->
# Definindo a função que preenche os registros com valor ZIP ausente por meio do dataframe auxiliar
def preenche_moda(x):
    if pd.isnull(x['ZIP']):
        return df_zip_group[(df_zip_group['B'] == x['B']) & (df_zip_group['BLOCK'] == x['BLOCK'])]['ZIP'].values[0]
    else:
        return x['ZIP']

# ->
# Usando a função para preencher os valores ZIP ausentes
df_temp['ZIP'] = df_temp.apply(preenche_moda, axis = 1)

# ->
# Verificando se há algum valor ZIP restante com valor NA
df_temp['ZIP'].isna().sum()

# ->
# Verificando manualmente uma das ocorrências para checar se a fórmula funcionou corretamente
df_temp[(df_temp['B']==1) & (df_temp['BLOCK']==36)]['ZIP']

# ->
# Visualiza
df_temp.head()

# ->
# Vamos salvar o dataframe resultante em disco
df_temp.to_csv('dados/dataset_zip_limpo.csv')

# ->
df_temp['ZIP'] = df_temp['ZIP'].astype(int).astype(str)

# ->
# Cria cópia da estrutura do dataframe
df_temp2 = df_temp.copy()

# ->
# Carrega os dados do disco
df_temp2 = pd.read_csv('dados/dataset_zip_limpo.csv')

### Limpando Valores Ausentes das Variáveis:
FULLVAL, AVLAND, AVTOT, BLDFRONT, BLDDEPTH, LTFRONT e LTDEPTH
# ->
# Substituindo valores vazios por valores NaN para garantir que a função fillna funcione nas etapas posteriores
df_temp2.replace(0, np.nan, inplace = True)

# ->
# Definindo a lista de variáveis a serem preenchidas
list_fill = ['FULLVAL', 'AVLAND', 'AVTOT', 'BLDFRONT', 'BLDDEPTH', 'LTFRONT', 'LTDEPTH']

# ->
# Contando o número de valores NA para cada uma das variáveis
for i in list_fill:
    print(i, "--> Número de valores ausentes:", df_temp2[i].isna().sum())

# ->
# Preenchendo os registros vazios com a mediana do grupo ZIP e BLDGCL se o tamanho do grupo for maior ou igual a 5
for i in list_fill:
    df_temp2[i] = df_temp2.groupby(['ZIP','BLDGCL'])[i].apply(lambda x: x.fillna(x.median()) if len(x) >= 5 else x)

# ->
# Contando o número de valores NA para cada uma das variáveis
for i in list_fill:
    print(i, "--> Número de valores ausentes:", df_temp2[i].isna().sum())

# ->
# Preenchendo os registros vazios com a mediana do grupo ZIP e TAXCLASS se o tamanho do grupo for maior ou igual a 5
for i in list_fill:
    df_temp2[i] = df_temp2.groupby(['ZIP','TAXCLASS'])[i].apply(lambda x: x.fillna(x.median()) if len(x) >= 5 else x)

# ->
# Contando o número de valores NA para cada uma das variáveis
for i in list_fill:
    print(i, "--> Número de valores ausentes:", df_temp2[i].isna().sum())
```

```
# ->
# Preenchendo os registros ausentes com a mediana do grupo B e TAXCLASS se o tamanho do grupo for maior ou igual a 5
for i in list_fill:
    df_temp2[i] = df_temp2.groupby(['B','TAXCLASS'])[i].apply(lambda x: x.fillna(x.median()) if len(x) >= 5 else x)

# ->
# Contando o número de valores NA para cada uma das variáveis
for i in list_fill:
    print(i, "--> Número de valores ausentes:", df_temp2[i].isna().sum())

# ->
# Preenchendo os registros ausentes com a mediana do grupo B se o tamanho do grupo for maior ou igual a 5
for i in list_fill:
    df_temp2[i] = df_temp2.groupby(['B'])[i].apply(lambda x: x.fillna(x.median()) if len(x) >= 5 else x)

# ->
for i in list_fill:
    print(i, "--> Número de valores ausentes:", df_temp2[i].isna().sum())

### Limpando Valores Ausentes da Variável STORIES
# ->
# Checa valores ausentes
print('STORIES', "--> Número de valores ausentes:", df_temp2['STORIES'].isna().sum())

# ->
# Imputação da mediana com base nos grupos de ZIP e BLDGCL
df_temp2['STORIES'] = df_temp2.groupby(['ZIP','BLDGCL'])['STORIES'].apply(lambda x: x.fillna(x.median()) if len(x) >= 5 else x)

# ->
# Checa valores ausentes
print('STORIES', "--> Número de valores ausentes:", df_temp2['STORIES'].isna().sum())

# ->
# Imputação da mediana com base nos grupos de BLDGCL e STORIES
df_temp2['STORIES'] = df_temp2.groupby(['BLDGCL'])['STORIES'].apply(lambda x: x.fillna(x.median()) if len(x) >= 5 else x)

# ->
# Checa valores ausentes
print('STORIES', "--> Número de valores ausentes:", df_temp2['STORIES'].isna().sum())

# ->
# Imputação da mediana com base nos grupos de TAXCLASS e STORIES
df_temp2['STORIES'] = df_temp2.groupby(['TAXCLASS'])['STORIES'].apply(lambda x: x.fillna(x.median()) if len(x) >= 5 else x)

# ->
# Checa valores ausentes
print('STORIES', "--> Número de valores ausentes:", df_temp2['STORIES'].isna().sum())

# ->
# Salvando resultado em disco
df_temp2.to_csv('dados/dataset_variaveis_limpo.csv')

## Engenharia de Atributos
Vamos criar algumas variáveis a partir de operações com as variáveis existentes.
# ->
# Cópia do dataframe
df_proc = df_temp2.copy()

# ->
# LTFRONT * LTDEPTH
df_proc['AREA1'] = df_proc['LTFRONT'] * df_proc['LTDEPTH']

# ->
# Visualiza amostra
df_proc[['AREA1', 'LTFRONT', 'LTDEPTH']].head()

# ->
# BLDFRONT * BLDDEPTH
df_proc['AREA2'] = df_proc['BLDFRONT'] * df_proc['BLDDEPTH']

# ->
# Visualiza amostra
df_proc[['AREA2', 'BLDFRONT', 'BLDDEPTH']].head()

# ->
# AREA2 * STORIES
df_proc['AREA3'] = df_proc['AREA2'] * df_proc['STORIES']
```

```
# ->
# Visualiza amostra
df_proc[['AREA2', 'STORIES', 'AREA3']].head()

# ->
# Dividindo FULLVAL, AVLAND e AVTOT pelas variáveis recém criadas AREA1, AREA2 e AREA3 para gerar um índice
k = 1
for i in ['FULLVAL', 'AVLAND', 'AVTOT']:
    for j in ['AREA1', 'AREA2', 'AREA3']:
        indice_area = 'ind' + str(k)
        df_proc[indice_area] = df_proc[i] / df_proc[j]
        print(indice_area + ' é a combinação feita entre:', i, j)
        k += 1

# ->
# Visualiza amostra
df_proc.head()

# ->
df_proc['ZIP'].dtype

# ->
# Converte o tipo da variável zip para string
df_proc['ZIP'] = df_proc['ZIP'].astype(int).astype(str)

# ->
df_proc['ZIP'].dtype

# ->
# Visualiza
df_proc.head()

# ->
# Lista das variáveis para o agrupamento abaixo
list_groupby = ['ZIP', 'TAXCLASS', 'B']

# ->
# Cria variáveis que representam as médias das áreas dos imóveis por grupo com base no índice criado
# Lista
ind_lista = ['ind' + str(i) for i in range(1,10)]
# Loop
for i in list_groupby:

    for j in ind_lista:
        name_col = 'media_' + str(j) + '_grupo_' + str(i)
        df_proc[name_col] = df_proc.groupby(i)[j].transform('mean')
        name_col_final = str(j) + '_' + name_col
        df_proc[name_col_final] = df_proc[j] / df_proc[name_col]

# ->
# Visualiza
df_proc.head()

# ->
# Shape
df_proc.shape

# ->
# Média para todas as variáveis
for j in ind_lista:
    name_col = 'media_' + str(j) + '_grupo_All'
    df_proc[name_col] = df_proc[j].mean()
    name_col_final = str(j) + '_' + name_col
    df_proc[name_col_final] = df_proc[j] / df_proc[name_col]

# ->
# Shape
df_proc.shape

# ->
# Visualiza
df_proc.head()

# ->
# Lista para as colunas criadas
cols_created = []
```

```
# ->
# Loop pelas variáveis do group by
for i in list_groupby:
    for j in ind_lista:
        cols_created.append(j + '_media_' + j + '_grupo_' + i)

# ->
len(cols_created)

# ->
# Loop pelas colunas criadas
for j in ind_lista:
    cols_created.append(j + '_media_' + j + '_grupo_All')

# ->
len(cols_created)

# ->
# Colunas finais para os dados
cols = list(df.columns) + cols_created

# ->
# Dataframe final
df_final = df_proc[cols]

# ->
# Shape
df_final.shape

As variáveis ind nos ajudam a verificar quão "normal" é cada registro.
# ->
# Colunas
df_final.columns

# ->
# Visualiza
df_final.head()

# ->
# Salva em disco
df_final.to_csv('dados/dataset_final.csv')

# ->
# Carrega do disco
df_final = pd.read_csv('dados/dataset_final.csv')

# ->
# Dataframe final para a modelagem
df = df_final.copy()

## Calculando o Score de Fraude 1 com PCA
Vamos criar o score de fraude usando PCA em escala Z. Essa é uma abordagem de aprendizado não supervisionado.
# ->
# Imports
import sklearn
from sklearn import preprocessing
from sklearn.decomposition import PCA

# ->
# Versões dos pacotes usados neste jupyter notebook
%reload_ext watermark
%watermark -a "Data Science Academy" --iversions

### Padronização das Variáveis
# ->
# Shape
df.shape

# ->
# Colunas
df.columns

# ->
# Tipos de dados
df.dtypes

# ->
# Busca as 36 últimas colunas (com as variáveis que nós criamos na Engenharia de Atributos)
```

```
df.iloc[:,-36:].columns

# ->
# Cria o objeto scaler
scaler = preprocessing.StandardScaler()

# ->
# Aplica o scaler às variáveis
arr_scaled = scaler.fit_transform(df.iloc[:,-36:])

# ->
# Cria o dataframe
df_scaled = pd.DataFrame(arr_scaled, columns = cols_created)

# ->
# Visualiza a amostra
df_scaled.head()

### Aplicando PCA
# ->
# Cria e treina o modelo PCA
# Queremos encontrar o valor ideal de componentes principais
pca = PCA().fit(df_scaled)

# ->
# Visualiza a Soma Cumulativa da Variância Explicada
plt.figure()
plt.plot(np.cumsum(pca.explained_variance_ratio_))
plt.xlabel('Número de Componentes')
plt.ylabel('Percentual de Variância Explicada')
plt.title('Variância Explicada')
plt.show()

# ->
# Pelo gráfico acima 10 componentes principais são suficentes. Usaremos esse valor.
pca = PCA(n_components = 10)

# ->
# Fit e transform do modelo pca
arr_pca = pca.fit_transform(df_scaled)

# ->
# Cria a lista colunas doo PCA
cols_PCA = ['PCA' + str(i) for i in range(1,11)]
cols_PCA

# ->
# Armazena os componentes em um dataframe
df_scaled_pca = pd.DataFrame(arr_pca, columns = cols_PCA)

# ->
# Visualiza amostra
df_scaled_pca.head()

### Aplicação da Escala Z em Componentes PCA Padronizados
# ->
# Fit e transform nos dados
arr_pca_scaled_z = scaler.fit_transform(df_scaled_pca)

# ->
# Cria o dataframe
df_pca_scaled_z = pd.DataFrame(arr_pca_scaled_z, columns = cols_PCA)

# ->
# Visualiza
df_pca_scaled_z.head()

### Calculando o Fraud Score
# ->
# Loop para calcular o score por componente PCA
for i in range(1,11):
    col_name = 'score' + str(i)
    pca_col_name = 'PCA' + str(i)
    df_pca_scaled_z[col_name] = df_pca_scaled_z[pca_col_name] ** 2

# ->
# Visualiza
df_pca_scaled_z.head()
```

```
# ->
# Colunas
col_scores = ['score' + str(i) for i in range(1,11)]

# ->
# Calcula o score de fraude
df_pca_scaled_z['Fraud Score 1'] = df_pca_scaled_z[col_scores].sum(axis = 1) ** (1 / 2)

# ->
# Visualiza
df_pca_scaled_z.head()

# ->
# Dataframe final com o score
df_score1 = pd.merge(df, df_pca_scaled_z.iloc[:,-1], left_index = True, right_index = True)

# ->
# Shape
df_score1.shape

# ->
# Visualiza
df_score1.head()

## Calculando o Score de Fraude 2 com Deep Learning
Usaremos um modelo Autoencoder. Essa é uma abordagem de aprendizado não supervisionado.
https://www.deeplearningbook.com.br/
# ->
# Imports
import tensorflow as tf
import tensorflow.keras as keras
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Sequential, Model, load_model
from tensorflow.keras.layers import Input, Dense
from tensorflow.keras.callbacks import ModelCheckpoint, TensorBoard
from tensorflow.keras import regularizers

# ->
# Versões dos pacotes usados neste jupyter notebook
%reload_ext watermark
%watermark -a "Data Science Academy" --iversions

# ->
# Carrega os dados com os 10 componentes principais
df_pca = df_pca_scaled_z.iloc[:,:10]

# ->
# Divisão em treino e teste
X_treino, X_teste = train_test_split(df_pca, test_size = 0.2, random_state = 42)

# ->
# Extrai os valores
X_treino = X_treino.values
X_teste = X_teste.values

# ->
# Shape
X_treino.shape

### Construção do Modelo
# ->
# Hiperparâmetros do modelo
input_dim = X_treino.shape[1]
encoding_dim = 5
input_layer = Input(shape = (input_dim, ))

input_layer --> encoder --> decoder
# ->
# Modelo
# Encoder
encoder = Dense(encoding_dim, activation = "tanh", activity_regularizer = regularizers.l1(10e-5))(input_layer)
encoder = Dense(int(encoding_dim / 2), activation = "relu")(encoder)
# Decoder
decoder = Dense(int(encoding_dim / 2), activation = 'tanh')(encoder)
decoder = Dense(input_dim, activation='relu')(decoder)
# Modelo final
autoencoder = Model(inputs = input_layer, outputs = decoder)
```

```
autoencoder.summary()

# ->
# Hiperparâmetros de treinamento
num_epoch = 50
batch_size = 256

# ->
# Compila o modelo
autoencoder.compile(optimizer = 'adam', loss = 'mean_squared_error', metrics = ['accuracy'])

# ->
# Checkpoint
checkpointer = ModelCheckpoint(filepath = "modelo/modelo_autoencoder.h5", verbose = 0, save_best_only = True)

# ->
tensorboard = TensorBoard(log_dir = '.\logs',
                          histogram_freq = 0,
                          write_graph = True,
                          write_images = True,
                          profile_batch = 100000000)

# ->
# Treinamento
history = autoencoder.fit(X_treino,
                          X_treino,
                          epochs = num_epoch,
                          batch_size = batch_size,
                          shuffle = True,
                          validation_data = (X_teste, X_teste),
                          verbose = 1,
                          callbacks = [checkpointer, tensorboard]).history

### Avaliação do Modelo
# ->
# Carrega o modelo do disco
autoencoder = load_model('modelo/modelo_autoencoder.h5')

# ->
# Plot
plt.plot(history['loss'])
plt.plot(history['val_loss'])
plt.title('Erro do Modelo')
plt.ylabel('Erro')
plt.xlabel('Número de Épocas')
plt.legend(['Treino', 'Teste'], loc = 'upper right');

# ->
# Previsões
previsoes = autoencoder.predict(df_pca)
previsoes

# ->
# Previsões em dataframe
df_previsoes = pd.DataFrame(previsoes, columns = cols_PCA)

# ->
# Visualiza
df_previsoes.head()

# ->
# Shape
df_previsoes.shape

# ->
# Colunas do PCA
cols_pca = ['PCA' + str(i) for i in range(1,11)]

# ->
# Dataframe
df_autoencod = pd.DataFrame(0, index = np.arange(len(df_previsoes)), columns = cols_pca)

# ->
# Loop
for i in range(0,10):
    df_autoencod.iloc[:,i] = (df_previsoes.iloc[:,i] - df_pca_scaled_z.iloc[:,i]) ** 2

# ->
```

```
# Calcula o score 2
df_autoencod['Fraud Score 2'] = df_autoencod[cols_pca].sum(axis = 1) ** (1 / 2)

# ->
# Visualiza
df_autoencod.head()

# ->
# Dataframe
df_score2 = pd.merge(df_score1.iloc[:,1:], df_autoencod.iloc[:,-1], left_index = True, right_index = True)

# ->
# Visualiza
df_score2.head()

## Calculando o Score Final do Score de Fraude e Apresentando os Resultados
https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.rank.html
https://www.mathsisfun.com/algebra/matrix-rank.html
O posto (português brasileiro) ou característica (português europeu) de uma matriz (em inglês, "matrix rank") é o número de linhas não-nulas da matriz, quando escrita na forma escalonada por linhas. Equivalentemente, corresponde ao número de linhas ou colunas linearmente independentes da matriz. A característica de uma matriz tem várias implicações em relação à independência linear e a dimensão de um espaço vetorial.
# ->
# Cópia do dataframe
df_fraudes = df_score2.copy()

# ->
# Calcula o Rank do Score 1
df_fraudes['Rank_Fraud Score 1'] = df_fraudes['Fraud Score 1'].rank(ascending = True, method = 'first')
df_fraudes.head()

# ->
# Calcula o Rank do Score 2
df_fraudes['Rank_Fraud Score 2'] = df_fraudes['Fraud Score 2'].rank(ascending = True, method = 'first')
df_fraudes.head()

# ->
# Organiza os dados
df_fraudes.sort_values('Rank_Fraud Score 1', ascending = False).head()

# ->
# Score final
df_fraudes['Final Score'] = (df_fraudes['Fraud Score 1'] * df_fraudes['Rank_Fraud Score 1']) + (df_fraudes['Fraud Score 2'] *
df_fraudes['Rank_Fraud Score 2'])

# ->
# Rank do Score Final
df_fraudes['Final Rank'] = df_fraudes['Final Score'].rank(ascending = False, method = 'first')

# ->
# Resultado
df_fraudes.sort_values('Final Rank', ascending = True).head(20)

Prepare o relatório para os tomadores de decisão com o score final de fraude de cada transação imobiliária, apresente o resultado e inicie seu
próximo projeto.
# Fim
```

Big Data e Detecção de Fraudes

Quando as atividades fraudulentas são detectadas e confirmadas, tipicamente são tomadas duas medidas:

1- Medidas corretivas que visam resolver a fraude e corrigir as consequências ilícitas - por exemplo, através da restituição ou compensação pelas perdas incorridas. Essas medidas corretivas podem também incluir ações para detectar retrospectivamente o rastro do evento fraudulento e corrigir o problema que não havia sido detectado antes.
2- Medidas preventivas, que podem incluir ações que visam evitar futuras fraudes por parte do fraudador (por exemplo, rescindindo um contrato, ou tomando ações que visam evitar fraudes do mesmo tipo por outros indivíduos). Quando uma abordagem especializada é adotada, um exemplo de medida preventiva é estender o mecanismo de regras incorporando regras adicionais que

permitam detectar e impedir que o mecanismo de fraude descoberto seja aplicado no futuro. Um caso de fraude deve ser investigado minuciosamente para que o mecanismo possa ser desvendado, ampliando o conhecimento especializado disponível, tornando a organização mais robusta e menos vulnerável a fraudes e ajustando o sistema de detecção e prevenção.

Tipicamente, quanto mais cedo forem tomadas as medidas corretivas e, portanto, quanto mais cedo for detectada uma fraude, mais eficazes serão essas medidas e mais perdas poderão ser evitadas. Por outro lado, a fraude torna-se mais fácil de detectar quanto mais tempo passa, por uma série de razões particulares. Quando existe um mecanismo ou rastro de fraude - o que significa uma lacuna no sistema de detecção e prevenção de uma organização - o número de vezes que esse caminho será seguido cresce no tempo e, portanto, o número de ocorrências deste tipo particular de fraude.

Quanto mais uma fraude é aplicada, mais aparente ela se torna e, estatisticamente, é mais fácil de ser detectada. Espera-se que o número de ocorrências de um determinado tipo de fraude aumente, uma vez que muitos fraudadores parecem ser reincidentes. Além disso, um mecanismo de fraude pode muito bem ser descoberto por vários indivíduos ou o conhecimento compartilhado entre os fraudadores. Certamente alguns tipos de fraude tendem a se espalhar de forma viral e exibir o que são chamados efeitos de rede social, indicando que os fraudadores compartilham seus conhecimentos sobre como cometer fraudes. Este efeito, também, leva a um número crescente de ocorrências e, portanto, um maior risco ou chance, dependendo da sua perspectiva de detecção.

E se temos mais dados (Big Data) à disposição, mais eficaz se torna o processo de detecção de fraudes.

Uma outra razão pela qual a fraude se torna mais fácil de detectar quanto mais tempo passa é porque as melhores técnicas de detecção estão sendo desenvolvidas, estão ficando prontamente disponíveis e estão sendo implementadas e aplicadas por uma quantidade crescente de organizações. Um importante motor para melhorias no que diz respeito às técnicas de detecção é a crescente disponibilidade de dados. A informatização e digitalização de quase todos os aspectos da sociedade e da vida cotidiana leva a uma abundância de dados disponíveis. Estes dados, algo que você já deve conhecer bem a esta altura da Formação Cientista de Dados, são o Big Data, que podem ser explorados para uma série de propósitos, incluindo detecção de fraude, a um custo muito baixo.

A utilização do Big Data para análise de fraudes no Brasil ainda é mínima, o que claro leva a um oceano de oportunidades, para aqueles dispostos a desbravar esse mercado. É necessário que as empresas saibam lidar com o enorme volume de dados gerados, pois é certo que eles irão aumentar cada vez mais. Elas precisam olhar para o futuro e imaginar todos os dispositivos que estarão conectados por seus clientes. Calcula-se que até 2020, 30 bilhões de dispositivos estejam conectados, contra 10 bilhões em 2013. Quanto mais dados gerados, maior a possibilidade de se perder nessas informações e não identificar o que realmente interessa.

O primeiro passo para utilizar o Big Data na detecção e prevenção de fraudes, é identificar o perfil do cliente, utilizar as ferramentas necessárias e estar preparado para receber e analisar essas informações. A combinação ideal de várias fontes de dados e a melhoria da qualidade dos processos de análise dos mesmos são outros desafios para as companhias. É nesse contexto que a participação de um Cientista de Dados torna-se ainda mais essencial. Ele terá o conhecimento necessário para analisar os resultados gerados e trabalhar na prevenção à fraude com análises mais precisas.

Segundo as estimativas, pelo menos 10% dos pagamentos da companhia de seguros são para reivindicações fraudulentas, e a soma global destes pagamentos fraudulentos equivale a bilhões ou

possivelmente trilhões de dólares. Embora a fraude de seguros não seja um problema novo, a gravidade do problema está aumentando e os perpetradores de fraudes de seguros estão se tornando cada vez mais sofisticados. Por exemplo, uma reivindicação de ferimento pessoal poderia potencialmente incluir reivindicações médicas falsificadas ou um acidente encenado. O Big Data Analytics, pode rapidamente procurar padrões em reivindicações históricas e identificar semelhanças ou levantar questões em uma nova reivindicação antes que o processo seja concluído.

Considere o seguinte exemplo. Uma companhia de seguros quer melhorar sua capacidade de tomar decisões em tempo real ao decidir como processar uma nova reivindicação. A despesa de custo da companhia incluindo pagamentos do litígio relacionados às reivindicações fraudulentas, tem aumentado firmemente. A empresa tem políticas extensas para ajudar os seguradores a avaliarem a legitimidade dos sinistros, mas os analistas muitas vezes não tinham os dados no momento certo para tomar uma decisão informada.

A empresa implementou uma grande plataforma de Big Data Analytics para fornecer a integração e análise de dados de várias fontes. A plataforma incorpora o uso extensivo de dados de mídia social e streaming de dados para ajudar a fornecer uma visão em tempo real. Agentes de call center são capazes de ter uma visão muito mais profunda em possíveis padrões de comportamento e relações entre outros requerentes e prestadores de serviços quando uma chamada ocorre.

Um agente pode receber um alerta sobre uma nova reivindicação que indica que o requerente foi testemunha anterior sobre uma reclamação semelhante há seis meses. Depois de descobrir outros padrões incomuns de comportamento e apresentar essas informações ao reclamante, o processo de reivindicação pode ser interrompido antes que ele realmente comece a andar. Em outras situações, dados de mídia social podem indicar que as condições descritas em uma reivindicação não ocorreram no dia em questão. Por exemplo, um requerente indicou que seu carro foi atingido em uma enchente, mas a documentação das mídias sociais mostrou que o carro tinha sido realmente em outra cidade no dia em que ocorreu a inundação.

Fraude de seguros causam enormes prejuízos para as empresas e os executivos estão se movendo com o objetivo de incorporar Big Data Analytics e outras tecnologias avançadas para resolver o problema da fraude. As companhias de seguros não só sentem o impacto desses prejuízos, como também repassam esses custos em forma de taxas para os clientes honestos, que pagam mais pelo serviços. Soluções de Big Data Analytics para detecção de fraudes podem ajudar a evitar perdas de todos os lados!

Mini-Projeto 4 – Detecção de Fraudes em Transações de Vendas Online

Neste Mini-Projeto vamos construir um modelo capaz de prever a probabilidade de que uma transação online seja fraudulenta, conforme indicado pela variável target isFraud. Conforme você já sabe criaremos o modelo usando dados históricos e depois faremos previsões em novos dados.

Os dados históricos estão divididos em dois arquivos: de identidade (identifica o cliente) e de transação (identifica a transação), que são unidos pela coluna TransactionID. Nem todas as transações possuem informações de identidade correspondentes.

Como este é um problema de classificação (queremos prever sim/não para cada transação) usaremos um algoritmo de classificação em aprendizagem supervisionada.

As previsões devem ser salvas em arquivo csv e entregues ao cliente/área de negócio.

```
# Mini-Projeto 4 - Detecção de Fraudes em Transações de Vendas Online

# Leia os manuais em pdf no Capítulo 11 do curso de Business Analytics com a descrição do projeto,
# definição do problema e fonte de dados.

# Defina sua pasta de trabalho
getwd()
setwd("~/Dropbox/DSA/Business-Analytics2.0/Cap11/R")

# Pacotes
library(ggplot2)
library(caret)
library(dplyr)
library(tibble)
library(tidyverse)
library(data.table)

# https://cran.r-project.org/web/packages/arsenal/vignettes/comparedf.html
library(arsenal)

# Carregando os Dados
treino_id = read.csv(file = "dados/train_identity.csv", na.strings = "")
treino_transacoes = read.csv(file = "dados/train_transaction.csv", na.strings = "")
teste_id = read.csv(file = "dados/test_identity.csv", na.strings = "")
teste_transacoes = read.csv("dados/test_transaction.csv", na.strings = "")

# Exploração Inicial dos Dados Históricos de Transações Online

# Dados de treino
dim(treino_id)
dim(treino_transacoes)

str(treino_id)
str(treino_transacoes)

View(treino_id)
View(treino_transacoes)

# Dados de teste
dim(teste_id)
dim(teste_transacoes)

str(teste_id)
str(teste_transacoes)

View(teste_id)
View(teste_transacoes)

# Engenharia de Atributos

# Nomes das colunas
names(treino_transacoes)
names(teste_transacoes)

# Ajustando a variável target em treino e teste
treino_transacoes[,"isFraud"] = factor(treino_transacoes[,"isFraud"])
teste_transacoes$isFraud <- NA

# Conferimos as dimensões
dim(treino_transacoes)
dim(teste_transacoes)

# Merge dos dataframnes para facilitar o trabalho de limpeza
?merge
dados_treino = merge(x = treino_id, y = treino_transacoes, by = "TransactionID")
dados_teste = merge(x = teste_id, y = teste_transacoes, by = "TransactionID")

# Dimensões
dim(dados_treino)
dim(dados_teste)

# Criamos uma coluna para identificar mais tarde se o registro é de treino ou teste
dados_treino$label = "treino"
dados_teste$label = "teste"

# Dimensões
dim(dados_treino)
dim(dados_teste)
```

```
# Vamos gerar um dataset completo com as duas amostras de dados (treino e teste)
dados_full <- rbind(dados_treino, dados_teste)

# TAREFA DE TROUBLESHOOTING

# Nomes das colunas
names(dados_treino)
names(dados_teste)

# Vamos organizar os nomes das colunas
dados_treino <- dados_treino %>%
  select(-label, -isFraud, everything())

dados_teste <- dados_teste %>%
  select(-label, -isFraud, everything())

# Nomes das colunas
names(dados_treino)
names(dados_teste)

# Dimensões
dim(dados_treino)
dim(dados_teste)

# Vamos gerar o dataset como as duas amostras
dados_full <- rbind(dados_treino, dados_teste)

# Vamos comparar os dataframes
?comparedf
comparedf(dados_treino, dados_teste)

# Nomes das colunas
names(dados_treino)
names(dados_teste)

# Lista de dataframes
dfs <- c("dados_treino", "dados_teste")

# Loop por todas as colunas dos dataframes para ajustar os nomes das colunas
for (eachdf in dfs) {
  df.tmp <- get(eachdf)
  for (eachcol in 1:length(df.tmp)){
    colnames(df.tmp)[eachcol] <- str_trim(str_to_lower(str_replace_all(colnames(df.tmp)[eachcol], "_", ".")))
  }
  assign(eachdf, df.tmp)
}

# Nomes das colunas
names(dados_treino)
names(dados_teste)

# Vamos gerar o dataset como as duas amostras
dados_full <- rbind(dados_treino, dados_teste)
dim(dados_full)
str(dados_full)

# Tratamento de Valores Ausentes

# Estratégia 1 - Remover variáveis cujo percentual de valor ausente for superior a 50%
# Estratégia 2 - Para as variáveis remanescentes, atribuir o valor "Desconhecido" se for variável categórica
# Estratégia 3 - Para as variáveis remanescentes, atribuir a média se for variável quantitativa

# Aplicando a Estratégia 1

# Calculando o percentual de valores ausentes por coluna
percentual_valores_ausentes = (colSums(is.na(dados_full)) / nrow(dados_full)) * 100

# Dataframe com o resultado anterior para criar o plot
df_percent_NA = data.frame(colnames(dados_full), percentual_valores_ausentes)
colnames(df_percent_NA) <- c("Variavel", "Percentual_Ausente")
df_percent_NA = df_percent_NA[order(df_percent_NA$Percentual_Ausente, decreasing = TRUE), ]

# Visualiza
View(df_percent_NA)

# Plot
plot(df_percent_NA$Percentual_Ausente,
```

```
    ylab = "% de Valores Ausentes",
    main = "Percentual de Valores Ausentes")

# Vamos remover as colunas com mais de 50% de valores ausentes
dim(dados_full)
dados_full <- dados_full[percentual_valores_ausentes < 50]
dim(dados_full)

# Aplicando as Estratégias 2 e 3

# Colunas ainda com valores ausentes
outrasNAcol <- (dados_full)[!colSums(is.na(dados_full))==0]
outrasNAcol <- colnames(outrasNAcol)

# Vamos colocar o valor "Desconhecido" onde estiver NA se for variável qualitativa
# Para variáveis quantitativas substituímos NA pela méda
for(f in outrasNAcol){

  if(any(is.na(dados_full[[f]]))){

    if(is.factor(dados_full[,f])){

      dados_full[,f] <- as.character(dados_full[,f])

      # Estratégia 2
      dados_full[,f][is.na(dados_full[,f])] <- "Desconhecido"
      dados_full[,f] <- factor(dados_full[,f])

    }
    else{

      # Estratégia 3
      dados_full[is.na(dados_full[,f]),f] <- mean(dados_full[,f], na.rm = TRUE)
    }
  }
}

# Verifica o dataframe
str(dados_full)
names(dados_full)
dim(dados_full)
sum(is.na(dados_full))

# Pré-Processamento das Variáveis Categóricas

# Convertemos as variáveis categóoricas para o tipo fator
str(dados_full)
dados_full[,"card1"] = factor(dados_full[,"card1"])
dados_full[,"card2"] = factor(dados_full[,"card2"])
dados_full[,"card3"] = factor(dados_full[,"card3"])
dados_full[,"card4"] = factor(dados_full[,"card4"])
dados_full[,"card5"] = factor(dados_full[,"card5"])
dados_full[,"card6"] = factor(dados_full[,"card6"])
dados_full[,"addr1"] = factor(dados_full[,"addr1"])
dados_full[,"addr2"] = factor(dados_full[,"addr2"])
dados_full[,"p.emaildomain"] = factor(dados_full[,"p.emaildomain"])
dados_full[,"r.emaildomain"] = factor(dados_full[,"r.emaildomain"])
dados_full[,"devicetype"] = factor(dados_full[,"devicetype"])
dados_full[,"deviceinfo"] = factor(dados_full[,"deviceinfo"])
dados_full[,"id.12"] = factor(dados_full[,"id.12"])
dados_full[,"id.13"] = factor(dados_full[,"id.13"])
dados_full[,"id.14"] = factor(dados_full[,"id.14"])
dados_full[,"id.15"] = factor(dados_full[,"id.15"])
dados_full[,"id.16"] = factor(dados_full[,"id.16"])
dados_full[,"id.17"] = factor(dados_full[,"id.17"])
dados_full[,"id.19"] = factor(dados_full[,"id.19"])
dados_full[,"id.20"] = factor(dados_full[,"id.20"])
dados_full[,"id.28"] = factor(dados_full[,"id.28"])
dados_full[,"id.29"] = factor(dados_full[,"id.29"])
dados_full[,"id.30"] = factor(dados_full[,"id.30"])
dados_full[,"id.31"] = factor(dados_full[,"id.31"])
dados_full[,"id.32"] = factor(dados_full[,"id.32"])
dados_full[,"id.33"] = factor(dados_full[,"id.33"])
dados_full[,"id.34"] = factor(dados_full[,"id.34"])
dados_full[,"id.35"] = factor(dados_full[,"id.35"])
dados_full[,"id.36"] = factor(dados_full[,"id.36"])
dados_full[,"id.37"] = factor(dados_full[,"id.37"])
dados_full[,"id.38"] = factor(dados_full[,"id.38"])
```

```
# Em variáveis do tipo texto vamos aplicar limpeza ao texto para poder separar as categorias

# Variável deviceinfo
View(table(dados_full$deviceinfo))
names_deviceinfo <- dados_full$deviceinfo
dados_full$deviceinfo <- factor(gsub("([A-Za-z]+).*",  "\\1", names_deviceinfo, ignore.case = FALSE))
View(table(dados_full$deviceinfo))

# Variável id.30
View(table(dados_full$id.30))
names_id.30 <- dados_full$id.30
dados_full$id.30 <- factor(gsub("([A-Za-z]+).*",  "\\1", names_id.30, ignore.case = FALSE))
View(table(dados_full$id.30))

# Variável card4
View(table(dados_full$card4))
dados_full$card4 = recode_factor(dados_full$card4,
                                 'american express' = "OTHER",
                                 'discover' = "OTHER",
                                 'visa' = "visa",
                                 'mastercard' = "mastercard",
                                 .default = "OTHER")
View(table(dados_full$card4))

# Variável card6
View(table(dados_full$card6))
dados_full$card6 = recode_factor(dados_full$card6,
                                 'credit' = "credit",
                                 'debit' = "debit",
                                 .default = "OTHER")
View(table(dados_full$card6))

# Ajusta a variável alvo removendo o nível onde a categoria for "Desconhecido"
View(table(dados_full$isfraud))
dados_full$isfraud = factor(x = dados_full$isfraud, exclude = "Desconhecido")
View(table(dados_full$isfraud))

# Dimensões
dim(dados_full)
colnames(dados_full)

# Divisão dos Dados em Treino e Teste
dados_treino_final = subset(dados_full, label == "treino")
dados_teste_final = subset(dados_full, label == "teste")

# Dimensões
dim(dados_treino_final)
dim(dados_teste_final)

# Colunas
names(dados_treino_final)
names(dados_teste_final)

# Análise Exploratória
ggplot(dados_treino_final,
       aes(x = factor(devicetype), fill = isfraud)) +
  geom_bar() +
  scale_fill_brewer(palette = "Dark2")

ggplot(dados_treino_final,
       aes(x = factor(productcd), fill = isfraud)) +
  geom_bar(position = "fill") +
  scale_fill_brewer(palette = "Set2")

ggplot(dados_treino_final,
       aes(x = factor(p.emaildomain), fill = isfraud)) +
  geom_bar() +
  coord_flip() +
  scale_fill_brewer(palette = "Set1")

# Checando outliers na quantidade das transações
ggplot(dados_treino_final,
       aes(factor(isfraud), transactionamt)) +
  geom_boxplot()

# Proporção por classe
table(dados_treino_final$isfraud)
```

```
# Preparação dos Dados Para Modelagem

# Divisão em dados de treino e teste
set.seed(100)
?createDataPartition
indice <- createDataPartition(dados_treino_final$isfraud, p = .7, list = F)
df_treino <- dados_treino_final[indice, ]
df_valid <- dados_treino_final[-indice, ]

# Dimensões
dim(df_treino)
dim(df_valid)

# Remove a coluna de label e amostra os dados
set.seed(100)
names(df_treino)
df_treino_final = select(df_treino, -395)
dim(df_treino_final)
names(df_treino_final)
df_treino_final_sample
 <- sample(df_treino_final, replace = FALSE, prob = NULL)
dim(df_treino_final_sample)
names(df_treino_final_sample)

# Modelagem

# Modelo de Regressão Logística
?glm
modelo_v1 <- glm(formula = isfraud ~ productcd + card4 + card6 + devicetype + id.30,
                 data = df_treino_final_sample,
                 family = "binomial")

summary(modelo_v1)

# Avaliação do Modelo
previsoes <- predict(modelo_v1, newdata = df_valid, type = "response")
View(previsoes)

# Cutoff
y_pred_num <- ifelse(previsoes > 0.5, 1, 0)

# Previsões de classe
y_pred <- factor(y_pred_num, levels = c(0, 1))

# Valor real de y
y_act <- df_valid$isfraud
y_act <- factor(y_act, levels=c(0, 1))

# Matriz de Confusão e Acurácia
confusionMatrix(y_act, y_pred)

# Previsões com Novos Dados
previsoes_novos_dados = predict(modelo_v1, newdata = dados_teste_final, type = "response")
previsoes <- data.frame(TransactionID = dados_teste_final$transactionid , fraud = previsoes_novos_dados)
View(previsoes)
write.csv(previsoes , file = 'dados/previsoes.csv' , row.names = FALSE )

# Fim
```

Cientistas de Dados Especializados em Detecção de Fraude

Discutimos ao longo deste capítulo, as características de um bom modelo de detecção de fraude, e vamos agora listar as características-chave de um bom Cientista de Dados Especializado em Análise de Fraude, do ponto de vista do gerente de contratação. Baseia-se em nossa experiência de consultoria e pesquisa, tendo colaborado com muitas empresas sobre o Big Data, análise e detecção de fraudes.

Um Cientista de Dados Especializado em Análise de Fraude deve ter sólidas habilidades analíticas

Obviamente, um Cientista de Dados Especializado em Análise de Fraude deve ter uma base sólida em estatísticas, aprendizagem de máquinas e mineração de dados. A distinção entre estas várias disciplinas está ficando cada vez mais desfocada e não é realmente relevante. Todos elas fornecem um conjunto de técnicas quantitativas para analisar dados e encontrar padrões de negócios relevantes dentro de um contexto particular, como detecção de fraude. Um Cientista de Dados deve estar ciente de qual técnica pode ser aplicada quando e como. Ele/ela não deve se concentrar muito nos detalhes matemáticos (por exemplo, otimização), mas sim ter uma boa compreensão do problema analítico que uma técnica resolve e como seus resultados devem ser interpretados. Nesse contexto, a formação de engenheiros em informática e/ou engenharia empresarial/industrial deve ter como objetivo uma visão integrada e multidisciplinar, com graduados formados tanto no uso das técnicas como com a perspicácia empresarial necessária para que novos empreendimentos se concretizem. Também é importante gastar tempo suficiente para validar os resultados analíticos obtidos para evitar situações muitas vezes referidas como massagem de dados e/ou tortura de dados, por meio dos quais os dados são falsificados intencionalmente e/ou foco excessivo é gasto discutindo correlações espúrias. Ao selecionar a técnica quantitativa ótima, o Cientista de Dados especializado em Análise de Fraude deve levar em conta as especificidades do contexto e do problema ou aplicação de detecção de fraude. O Cientista de Dados deve ser capaz de selecionar a melhor técnica analítica para resolver o problema de negócio específico.

Um Cientista de Dados Especializado em Análise de Fraude deve ser um bom programador

Por definição, os Cientistas de Dados trabalham com dados. Isso envolve muitas atividades como amostragem e pré-processamento de dados, estimativa de modelo e pós-processamento (por exemplo, análise de sensibilidade, implantação de modelo, backtesting, validação de modelo). Embora muitas ferramentas de software amigáveis ao usuário estejam no mercado hoje em dia para automatizar e suportar essas tarefas, todo exercício analítico requer etapas personalizadas para abordar as especificidades de um determinado problema e configuração de negócios. Para realizar com êxito essas etapas, a programação precisa ser feita. Assim, um bom Cientista de Dados deve possuir habilidades de programação (por exemplo, SAS, R, Python, etc.). A linguagem de programação em si não é tão importante como tal, desde que esteja familiarizado com os conceitos básicos de programação e saiba como usá-los para automatizar tarefas repetitivas ou executar rotinas específicas.

Um Cientista de Dados Especializado em Análise de Fraude deve ser um bom comunicador

Goste ou não, a análise é um exercício técnico. Neste momento, há uma enorme lacuna entre os modelos analíticos e os usuários de negócios. Para preencher essa lacuna, comunicação e visualização são fundamentais. Assim, os Cientistas de Dados devem saber como representar modelos analíticos, suas estatísticas e relatórios, de maneiras amigáveis usando abordagens de semáforo, regras de negócios Se-Então e assim por diante. Eles devem ser capazes de comunicar a quantidade certa de informação sem se perder em detalhes complexos (por exemplo, estatísticos), o que inibirá a implantação bem-sucedida de um modelo. Ao fazer isso, os usuários de negócios entenderão melhor as características e comportamento em seus dados, o que melhorará sua atitude e aceitação dos modelos analíticos resultantes.

Um Cientista de Dados Especializado em Análise de Fraude deve ter uma sólida compreensão de negócios

Um Cientista de Dados Especializado em Análise de Fraude deve ter uma sólida compreensão de negócios. Embora isso possa ser óbvio, temos testemunhado (também) muitos projetos de ciência de dados que falharam porque o analista respectivo não entendeu o problema do negócio em questão. Por "negócio" referimo-nos à respectiva área de aplicação. Cada um desses campos tem suas próprias particularidades que são importantes para um Cientista de Dados Especializado em Análise de Fraude conhecer e entender, a fim de ser capaz de projetar e implementar um sistema personalizado de detecção de fraude. Quanto mais alinhado o sistema de detecção com o ambiente, melhor será seu desempenho, avaliado em cada uma das dimensões já discutidas.

Um Cientista de Dados precisa de criatividade em pelo menos dois níveis

Primeiro, em um nível técnico, é importante ser criativo no que diz respeito à seleção de recursos, transformação de dados e limpeza. Essas etapas do processo de análise padrão devem ser adaptadas a cada aplicativo em particular, e muitas vezes o "palpite correto" pode fazer uma grande diferença. Segundo, Big Data Analytics é um campo em rápida evolução. Novos problemas, tecnologias e desafios correspondentes surgem de forma contínua. Além disso, também os fraudadores são muito criativos e adaptar suas táticas e métodos em uma base contínua. Portanto, é crucial que um Cientista de Dados Especializado em Análise de Fraude acompanhe essas novas evoluções e tecnologias e tenha criatividade suficiente para ver como elas podem criar novas oportunidades.

8.12. Text Analytics

O Que é Text Analytics?

Text Analytics é a aplicação de Ciência de Dados para análise de texto.

Mini-Projeto 5 – Text Analytics Para Analisar a Reação do Mercado Sobre as Notícias de Uma Empresa

Fundada em 1984 em Dallas nos EUA, a Gamestop Corp é uma empresa de eletrônicos, especificamente no varejo de jogos de computador, os famosos videogames.

Recentemente a Gamestop ganhou atenção da mídia e do mercado em geral quando o grupo de usuário chamado WallStreetBets fez com que o preço das ações da empresa disparasse, com um aumento de 1.500%, alcançando o pico de USD 347.51 por ação na bolsa de valores.

O WallStreetBets é uma comunidade de usuários na rede social Reddit com mais de 9 milhões de participantes e que foi fundada em 2012. O grupo discute investimentos de risco e oportunidades de compra e venda de ações. Usuários do grupo definem estratégias agressivas de investimentos e em uma dessas estratégias a Gamestop foi supervalorizada em poucos dias, fazendo investidores menores ganharem muito dinheiro. Isso abalou o mercado de investimentos especialmente nos EUA.

Nosso objetivo neste projeto é tentar compreender a reação do mercado a esse acontecimento, analisando os textos de artigos de alguns importantes veículos de comunicação nos EUA.

```
# Mini-Projeto 5 - Text Analytics Para Analisar a Reação do Mercado Sobre as Notícias de Uma Empresa

# Leia o manual em pdf no Capítulo 12 com a descrição do projeto

# Diretório de Trabalho
setwd("~/Dropbox/DSA/Business-Analytics2.0/Cap12/R")
getwd()

# Pacotes
library(tm)
library(topicmodels)
library(textdata)
library(rvest)
library(tidyverse)
library(tidytext)
library(dplyr)
library(tidyr)
library(reshape2)
library(forcats)
library(scales)
library(stringr)
library(ggplot2)
library(wordcloud)
library(igraph)
library(ggraph)

########## Extração dos Dados de Texto via Web Scraping ##########

# Web Scraping do site do New York Times
?read_html
artigo_nytimes <- read_html("https://www.nytimes.com/2021/02/01/business/gamestop-how-much-worth.html")
class(artigo_nytimes)
View(artigo_nytimes)

# Obtendo o título do artigo
titulo_artigo_nytimes <- artigo_nytimes %>%
  html_nodes("title") %>%
  html_text()
```

```
# Visualizando o título
print(titulo_artigo_nytimes)

# Extraindo o texto do artigo
texto_artigo_nytimes <- artigo_nytimes %>%
  html_nodes("p") %>%
  html_text()

# Visualizando o texto
class(texto_artigo_nytimes)
View(texto_artigo_nytimes)

# Web Scraping do site do Yahoo Finance
artigo_yahoo <- read_html("https://finance.yahoo.com/news/gamestop-amc-reddit-investing-213609595.html")

# Obtendo o título do artigo
titulo_artigo_yahoo <- artigo_yahoo %>%
  html_nodes("title") %>%
  html_text()

# Visualizando o título
print(titulo_artigo_yahoo)

# Extraindo o texto do artigo
texto_artigo_yahoo <- artigo_yahoo %>%
  html_nodes("p")%>%
  html_text()

# Visualizando o texto
View(texto_artigo_yahoo)

# Web Scraping do site da Time Magazine
artigo_timemag <- read_html("https://time.com/5933242/gamestop-stock-gme/")

# Obtendo o título do artigo
titulo_artigo_timemag <- artigo_timemag %>%
  html_nodes("title") %>%
  html_text()

# Visualizando o título
print(titulo_artigo_timemag)

# Extraindo o texto do artigo
texto_artigo_timemag <- artigo_timemag %>%
  html_nodes("p") %>%
  html_text()

# Visualizando o texto
View(texto_artigo_timemag)

########## Preparação dos Dados ##########

# Vamos criar um dataframe para o texto de cada artigo

df_nytimes <- data.frame(line = 1, text = texto_artigo_nytimes, stringsAsFactors = FALSE)
class(df_nytimes)
View(df_nytimes)

df_yahoo <- data.frame(line = 1, text = texto_artigo_yahoo, stringsAsFactors = FALSE)
View(df_yahoo)

df_timemag <- data.frame(line = 1, text = texto_artigo_timemag, stringsAsFactors = FALSE)
View(df_timemag)

########## Processamento de Linguagem Natural ##########

# Tokenização

?unnest_tokens
?dplyr::anti_join

tokens_nytimes <- df_nytimes %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  count(word, sort = TRUE)

View(tokens_nytimes)
```

```
tokens_yahoo <- df_yahoo %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  count(word, sort = TRUE)

View(tokens_yahoo)

tokens_timemag <- df_timemag %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  count(word, sort = TRUE)

View(tokens_timemag)

# Histogramas de Frequência

# Vamos verificar quais palavras aparecem com mais frequência

hist_nytimes <- df_nytimes %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  count(word, sort = TRUE) %>%
  filter(n > 6) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_col() +
  xlab(NULL) +
  coord_flip()

print(hist_nytimes + ggtitle("Histograma de Frequência do Artigo do New York Times"))

hist_yahoo <- df_yahoo %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  count(word, sort = TRUE) %>%
  filter(n > 13) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_col() +
  xlab(NULL) +
  coord_flip()

print(hist_yahoo + ggtitle("Histograma de Frequência do Artigo do Yahoo Finance"))

hist_timemag <- df_timemag %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  count(word, sort = TRUE) %>%
  filter(n > 3) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_col() +
  xlab(NULL) +
  coord_flip()

print(hist_timemag + ggtitle("Histograma de Frequência do Artigo da Time Magazine"))

# Removendo as Stop Words

nytimes_clean <- df_nytimes %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words)

yahoo_clean <- df_yahoo %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words)

timemag_clean <- df_timemag %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words)

# Combinando os Dataframes

# Calculamos a frequência
?bind_rows
?str_extract
frequencia <- bind_rows(mutate(nytimes_clean, author = "NYTIMES"),
```

```
                    mutate(yahoo_clean, author = "YAHOO"),
                    mutate(timemag_clean, author = "TIME")) %>%
  mutate(word = str_extract(word, "[a-z']+")) %>%
  count(author, word) %>%
  group_by(author) %>%
  mutate(proportion = n/sum(n))%>%
  select(-n) %>%
  spread(author, proportion) %>%
  gather(author, proportion, `TIME`, `NYTIMES`)

print(frequencia)

# Plot de um Correlograma para comparar os artigos
ggplot(frequencia, aes(x = proportion, y = `YAHOO`, color = abs(`YAHOO`- proportion))) +
  geom_abline(color = "grey40", lty = 2) +
  geom_jitter(alpha = .1, size = 2.5, width = 0.3, height = 0.3) +
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = percent_format()) +
  scale_y_log10(labels= percent_format()) +
  scale_color_gradient(limits = c(0,0.001), low = "darkslategray4", high = "gray75") +
  facet_wrap(~author, ncol=2) +
  theme(legend.position = "none") +
  labs(y = "Yahoo", x = NULL)

# Correlação entre os artigos do Yahoo e NYTimes
cor.test(data = frequencia[frequencia$author == "NYTIMES",], ~proportion + `YAHOO`)

# Correlação entre os artigos do Yahoo e Time
cor.test(data = frequencia[frequencia$author == "TIME",], ~proportion + `YAHOO`)

########## Análise de Sentimentos ##########

# Léxicos de sentimentos
?get_sentiments
afinn <- get_sentiments("afinn")
nrc <- get_sentiments("nrc")
bing <- get_sentiments("bing")

# Análise de sentimentos do artigo do New York Times

# Método AFINN
nytimes_clean %>%
  inner_join(get_sentiments("afinn")) %>%
  summarise(sentiment = sum(value)) %>%
  mutate(method = "AFINN")

# Método Bing
nytimes_clean %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = T)

# Método NRC
sentimentos_nytimes <- nytimes_clean %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort = T) %>%
  ungroup()

# Plot de todos os sentimentos
sentimentos_nytimes %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Sentimentos do Artigo do New York Times", x = NULL) +
  coord_flip()

# Palavras mais associadas a sentimentos positivos
sentimentos_nytimes %>%
  group_by(sentiment) %>%
  filter(sentiment == "positive") %>%
  top_n(5) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) + ggtitle("Palavras Mais Associadas a Sentimentos Positivos") +
  geom_col(show.legend = FALSE, fill = "palegreen2") +
```

```
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Frequência", x = NULL) +
  coord_flip()

# Palavras mais associadas a sentimentos negativos
sentimentos_nytimes %>%
  group_by(sentiment) %>%
  filter(sentiment == "negative") %>%
  top_n(5) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) + ggtitle("Palavras Mais Associadas a Sentimentos Negativos") +
  geom_col(show.legend = FALSE, fill = "red3") +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Frequência", x = NULL) +
  coord_flip()

# Análise de sentimentos do artigo do Yahoo Finance

# Método AFINN
yahoo_clean %>%
  inner_join(get_sentiments("afinn")) %>%
  summarise(sentiment = sum(value)) %>%
  mutate(method = "AFINN")

# Método Bing
yahoo_clean %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = T)

# Método NRC
sentimentos_yahoo <- yahoo_clean %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort = T) %>%
  ungroup()

# Plot de todos os sentimentos
sentimentos_yahoo %>%
  group_by(sentiment) %>%
  top_n(8) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Sentimentos do Artigo do Yahoo Finance", x = NULL) +
  coord_flip()

# Palavras mais associadas a sentimentos positivos
sentimentos_yahoo %>%
  group_by(sentiment) %>%
  filter(sentiment == "positive") %>%
  top_n(5) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) + ggtitle("Palavras Mais Associadas a Sentimentos Positivos")+
  geom_col(show.legend = FALSE, fill = "palegreen2") +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Frequência", x = NULL) +
  coord_flip()

# Palavras mais associadas a sentimentos negativos
sentimentos_yahoo %>%
  group_by(sentiment) %>%
  filter(sentiment == "negative") %>%
  top_n(5) %>%
  ungroup() %>%
  mutate(word=reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) + ggtitle("Palavras Mais Associadas a Sentimentos Negativos") +
  geom_col(show.legend = FALSE, fill = "red3") +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Frequência", x = NULL) +
  coord_flip()

# Análise de sentimentos do artigo da Time Magazine

# Método AFINN
timemag_clean %>%
```

```
  inner_join(get_sentiments("afinn")) %>%
  summarise(sentiment = sum(value)) %>%
  mutate(method = "AFINN")

# Método Bing
timemag_clean %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = T)

# Método NRC
sentimentos_timemag <- timemag_clean %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort = T) %>%
  ungroup()

# Plot
 de todos os sentimentos
sentimentos_timemag %>%
  group_by(sentiment) %>%
  top_n(5) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Sentimentos do Artigo da Time Magazine", x = NULL) +
  coord_flip()

# Palavras mais associadas a sentimentos positivos
sentimentos_timemag %>%
  group_by(sentiment) %>%
  filter(sentiment == "positive") %>%
  top_n(5) %>%
  ungroup() %>%
  mutate(word=reorder(word, n)) %>%
  ggplot(aes(word, n, fill=sentiment)) + ggtitle("Palavras Mais Associadas a Sentimentos Positivos")+
  geom_col(show.legend = FALSE, fill = "palegreen2") +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Frequência", x = NULL) +
  coord_flip()

# Palavras mais associadas a sentimentos negativos
sentimentos_timemag %>%
  group_by(sentiment) %>%
  filter(sentiment == "negative")%>%
  top_n(5) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) + ggtitle("Palavras Mais Associadas a Sentimentos Negativos") +
  geom_col(show.legend = FALSE, fill = "red3") +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Frequência", x = NULL) +
  coord_flip()

# Encontrando as Palavras Mais Comuns nos Textos Usando TF-IDF

# Combinando os artigoos
df_combinado <- bind_rows(mutate(df_nytimes, author = "Yahoo"),
                          mutate(df_yahoo, author = "TIME"),
                          mutate(df_timemag, author = "NYtimes"))

# Remoção de stop words e tokenização
df_combinado_limpo <- df_combinado %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  count(author, word, sort = TRUE) %>%
  ungroup()

# Total de palavras
total_palavras <- df_combinado_limpo %>%
  group_by(author) %>%
  summarize(total = sum(n))

# Left join
df_combinado_limpo <- left_join(df_combinado_limpo, total_palavras)

# TF-IDF
?bind_tf_idf
```

```
df_tf_idf <- df_combinado_limpo %>%
  bind_tf_idf(word, author, n)

View(df_tf_idf)

# Plot das palavras mais frequentes usando TF-IDF
df_tf_idf %>%
  arrange(desc(tf_idf)) %>%
  mutate(word = factor(word, levels = rev(unique(word)))) %>%
  group_by(author) %>%
  top_n(10) %>%
  ungroup() %>%
  ggplot(aes(tf_idf, fct_reorder(word, tf_idf), fill = author)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~author, ncol = 2, scales = "free") +
  labs(x = "Palavras Mais Frequentes", y = NULL)


# Word Clouds

# Remoção de stop words e tokenização
df_combinado2 <- df_combinado %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  count(author, word, sort = TRUE)

# Plot com NRC
df_combinado2 %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"), max.words = 200)

# Plot com Bing
df_combinado2 %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"), max.words = 200)

########### Modelagem de Tópicos ###########

# Criando a Document Term Matrix para todos os artigos
doc_term_matrix <- df_combinado %>%
  unnest_tokens(word, text) %>%
  count(author, word) %>%
  cast_dtm(author, word, n)

print(doc_term_matrix)

dtm_artigos <- df_combinado_limpo %>%
  cast_dtm(author, word, n)

# LDA (Latent Dirichlet Allocation)
?LDA
modelo_lda <- LDA(dtm_artigos, k = 3, control = list(seed = 123))
modelo_lda

# Dataframe de tópicos
df_topics <- tidy(modelo_lda, matrix = "beta")
print(df_topics)

# Top termos
top_termos <- df_topics %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

# Tópicos - Top Termos
top_termos %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip()

# Conclusão: Não há evidência que demonstre reação positiva ou negativa do mercado com relação aos artigos
```

```
# analisados. O objetivo principal dos artigos parece estar em prover informação mantendo um tom neutro,
# levemente positivo.
```

O Que é Modelagem de Tópicos?

A maneira padrão de procurar documentos na internet é através de palavras-chave ou frases-chave. Isso é o que o Google e outros motores de busca fazem rotineiramente, e eles fazem isso muito bem. No entanto, por mais útil que seja, tem suas limitações. Considere, por exemplo, uma situação na qual você é confrontado com uma grande coleção de documentos, mas não tem ideia sobre o assunto que eles abordam. Uma das primeiras coisas que você pode querer fazer é classificar esses documentos em tópicos ou temas. Entre outras coisas isso iria ajudá-lo a descobrir o que há de interessante, enquanto também poderia orientá-lo sobre o subconjunto relevante dos dados. Para coleções pequenas, pode-se fazer isso simplesmente passando por cada documento, mas isso é claramente inviável para corpus contendo milhares de documentos.

A modelagem de tópicos – Topic Modeling – trata do problema de classificar automaticamente conjuntos de documentos em temas.

Topic Modeling é uma forma de mineração de texto, uma forma de identificar padrões em um corpus. Você agrupa palavras de todo o corpus em tópicos. A modelagem de tópicos é “um método para encontrar e traçar clusters de palavras (chamados de ‘tópicos’) em grandes corpus de textos”.

O que, então, é um tópico? Uma definição oferecida no Twitter durante uma conferência sobre modelagem de tópicos descreveu um tópico como “um padrão recorrente de palavras co-ocorrentes”. Uma ferramenta de modelagem de tópicos examina um corpus para esses grupos de palavras e os agrupa em conjunto por um processo de similaridade.

(ATENÇÃO: LDA também é a abreviatura de Linear Discriminant Analysis, uma técnica de classificação. Aqui falaremos de outro LDA, o Latent Dirichlet Allocation).

Latent Dirichlet Allocation

Em essência, LDA é uma técnica que facilita a descoberta automática de temas em uma coleção de documentos (corpus).

A suposição básica por trás de LDA é que cada um dos documentos em uma coleção consiste em uma mistura de tópicos de toda a coleção. No entanto, na realidade, observamos apenas documentos e palavras, não tópicos - estes últimos fazem parte da estrutura oculta (ou latente) dos documentos. O objetivo é inferir a estrutura tópica latente dada as palavras e o documento. A LDA faz isso recriando os documentos no corpus e ajustando a importância relativa de tópicos em documentos e palavras em tópicos, iterativamente. O processo iterativo é implementado usando uma técnica chamada amostragem de Gibbs.

O termo "Dirichlet" em LDA refere-se ao fato de que tópicos e palavras são assumidos para seguir as distribuições de Dirichlet. Não há nenhuma "boa" razão para isso, além da conveniência - distribuições Dirichlet fornecem boas aproximações para distribuições de palavras em documentos e, talvez mais importante, são computacionalmente convenientes.

No link abaixo há uma descrição completa deste algoritmo, incluindo seu fundamento matemático:

http://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf

Modelagem de Tópicos usando LDA

Agora estamos prontos para fazer alguma modelagem de tópicos. Usaremos o pacote topicmodels em R. Especificamente, usaremos a função LDA com a opção de amostragem Gibbs mencionada anteriormente. A função LDA tem um número bastante grande de parâmetros e usaremos valores padrões, embora você possa fazer um tunning dos parâmetros e customizar seus resultados.

A amostragem de Gibbs funciona executando uma caminhada aleatória de tal forma que reflete as características de uma distribuição desejada. Como o ponto de partida da caminhada é escolhido ao acaso, é necessário descartar os primeiros passos da caminhada (pois estes não refletem corretamente as propriedades da distribuição). Isto é referido como o período de burn-in.

Deve-se enfatizar que as configurações não garantem a convergência do algoritmo para uma solução globalmente ótima. Na verdade, a amostragem de Gibbs, na melhor das hipóteses, encontrará apenas uma solução localmente ótima, e mesmo isso é difícil de provar matematicamente em problemas práticos específicos, como o que estamos lidando aqui. O resultado disso é que é melhor fazer lotes de execuções com diferentes configurações de parâmetros para verificar a estabilidade de seus resultados. A questão é que nosso interesse é puramente prático por isso é bom o suficiente se os resultados fazem sentido.

Há um parâmetro importante que deve ser especificado inicialmente: k, o número de tópicos que o algoritmo deve usar para classificar documentos. Existem abordagens matemáticas para isso, mas muitas vezes não produzem escolhas semanticamente significativas de k. Do ponto de vista prático, pode-se simplesmente executar o algoritmo para diferentes valores de k e fazer uma escolha baseada na inspeção dos resultados.

8.13. Social Network Analytics

Mídias Sociais e Sua Importância

A expectativa é que os investimentos em propaganda na internet, especialmente através de mídias sociais, mantenha um nível e presença cada vez maiores. Nenhuma empresa pode se dar ao luxo de ignorar isso!

Estes números demonstram que campanhas de marketing digital e links patrocinados em sites de redes e mídias sociais, como Facebook, LinkedIn e YouTube, por exemplo, e sites buscadores, como Google e Bing, entre outros, deixaram de ser uma aposta, para ser fundamentais em estratégias de publicidade e marketing.

Através de pequenos aplicativos, as marcas têm acesso às suas preferências e ficam aptas a conhecer melhor seus hábitos e costumes e dessa forma, podem desenvolver produtos e serviços com maior chance de agradar a você!

Alguns números:

- São 103 milhões de contas ativas nas redes sociais
- 88 milhões de aplicativos de social media em dispositivos móveis
- 267,1 milhões de pessoas com aparelhos celulares

O que faz das mídias sociais um lugar tão interessante para a sua marca ou o seu produto: seu público está lá.

As pessoas passam um bom tempo do dia nas mídias sociais e a empresa quer estar presente na vida delas nesses momentos.

Na teoria de mídia social, as pessoas são consideradas os blocos de construção básicos de um mundo criado nas bases fornecidas pela mídia social.

Desafios em Social Network Analytics

Veja o desafio quando falamos em análise de mídias sociais:

- Big Data: Devemos usar o gosto de um amigo de um amigo da pessoa de interesse, que tem estudado em um colégio particular e cuja cidade natal era uma cidade particular, para recomendar algo para a pessoa do interesse?
- Suficiência: Devemos limitar as pessoas a ver apenas as pessoas de sua cidade natal a fim de recomendar alguma coisa e não usar os gostos de seus amigos?
- Remoção de Ruído: O ruído por sua definição é uma quantidade subjetiva e pode sempre ser confundido e portanto, esta etapa pode acabar introduzindo mais erros no reconhecimento de padrões.
- Dilema de avaliação: Devido ao grande tamanho dos dados de mídias sociais, não é possível obter um conjunto de dados adequadamente rotulado para treinar um algoritmo supervisionado de aprendizado de máquina.

Técnicas de Social Media Analytics

São 2 abordagens principais:

- Graph Mining:
  - Gráficos de rede compõem a estrutura de dados dominante e aparecem essencialmente em todas as formas de mídia social. Normalmente, as comunidades de usuários constituem um grupo de nós em tais gráficos, onde os nós dentro da mesma comunidade ou cluster, tendem a compartilhar recursos comuns.
  - A mineração de gráficos pode ser descrita como o processo de extração de conhecimento útil, ou seja, padrões, outliers, etc., de uma relação social, entre os membros da comunidade, que pode ser representada por um gráfico.
  - Ferramentas utilizadas: Gephi, NodeXL, Netlytic, Flocker, Digimind
- Text Mining:
  - Extração de significado de dados de texto não estruturados, presentes em mídias sociais, é descrito como mineração de texto.
  - Os alvos principais são blogs, microblogs e outras mídias sociais que utilizam texto de forma massiva. Porém, é aplicável a outras redes sociais, como facebook por exemplo, que contêm links para postagens, blogs e artigos de notícias.
  - Ferramentas utilizadas: Linguagem R e Python

O Processo de Social Media Analytics

Etapas Genéricas para Obter Informações a Partir dos Dados:

1. Obter autenticação na mídia social

   OAuth: Um protocolo aberto para permitir a autorização segura em um método simples e padrão, de aplicações web, móveis e de desktop. Tornou-se um padrão para acessar APIs baseadas na web e que exigem uma autorização antes que as funções da API possam ser chamadas.

   Autenticação é quando se valida a identidade de um usuário, pedindo login e senha, por exemplo. Enquanto que Autorização é a verificação das permissões que um usuário existente já possui.

   Alguns servidores de API emitem tokens OAuth que expiram depois de um tempo. Atualizar tokens expirados é um processo complicado, mas temos funções em R e Python que cuidam disso de forma transparente.

   Outras APIs, como por exemplo a do Facebook, utilizam tokens de acesso que expiram após um tempo, mas não podem ser renovados automaticamente.

2. Visualização de Dados e Análise Exploratória

   1. Text:

   Vários pacotes de visualização de dados de texto estão disponíveis em R e Python.

Uma das bibliotecas de visualização mais simples e mais utilizadas é a nuvem de palavras (ou wordcloud). A wordcloud ajuda o usuário a compreender os pesos de uma palavra ou termo, em relação a matriz *PFDF (atenção: termo pode estar errado)*.

Existem pacotes que podem gerar uma nuvem de palavras, junto aos sentimentos que cada palavra representa.

Algumas das técnicas de análise de sentimento são:

Classify_emotion
Classify_polarity (polaridade geral das emoções: positiva ou negativa)

2. Graph:

A biblioteca de visualização mais utilizada para graph mining de dados do Facebook é a Gephi.

3. Limpeza e Pré-processamento

Principais processos utilizados durante a faze de limpeza e pré-processamento:

1. Entendimento dos Dados
2. Limpeza dos Dados
3. Eliminação de Dados Incorretos
4. Eliminação de Dados Duplicados
5. Tratamento de Dados Ausentes (Dados Missing)
6. Seleção dos Dados
7. Seleção dos Atributos
8. Seleção das Instâncias
9. Transformação dos Dados
10. Padronização dos Dados

É no pré-processamento que os documentos são transformados em forma numérica, onde o conteúdo de cada documento é decomposto em termos e a frequência de cada um. Os termos menos significativos são descartados e os que estão presentes em um grande número de documentos da coleção são valorizados (recebem um peso).

O resultado do pré-processamento é a geração da “bag of words” (saco de palavras), que é uma representação numérica da coleção de documentos.

Term-Frequency Metric	Term(1)	Term(2)	Term(3)	Term(4)	Term(5)	Term(6)	Term(n)
Document(1)	225	300	0	0	0	25	0
Document(2)	78	87	0	92	0	175	0
Document(3)	58	137	0	0	237	0	21
Document(4)	0	12	101	0	0	0	0
Document(5)	3	15	0	24	0	48	87
Document(6)	0	0	71	0	0	0	0
Document(n)	109	0	901	221	331	441	551

Para a geração de uma bag of words são necessárias 4 etapas:

1. Leitura
2. Extração e Limpeza dos Termos
    1. Tokenização (delimitação dos termos – espaço em branco, quebra de linhas, tabulações e alguns outros caracteres especiais)
    2. Limpeza (remoção das stop words e verificação de sinônimos)
    3. Stemming (redução de um termo ao seu radical)
3. Contagem dos Termos
4. Cálculo da Frequência

4. Modelagem de dados

    Opinion Mining (Sentiment Analysis) – método onde tentamos avaliar a opinião ou sentimento presente em uma frase.

    Análise de sentimentos é a tarefa de identificar se a opinião que foi expressada em um determinado texto, é positiva ou negativa.

    Para começar, é importante compreender que opiniões e sentimentos, bem como seus conceitos relacionados como avaliação, atitude, emoção e humor são influenciadores do comportamento humano.

    A análise de sentimento pode ser realizada em 3 níveis de granularidade:

    1. no nível de documento, observando o sentimento global expresso no texto.
    2. no nível da sentença, classificando a polaridade de cada sentença no texto.
    3. no nível de característica, analisando a polaridade das opiniões sobre características/atributos do objeto.

5. Visualização do resultado

    Conseguimos uma melhor compreensão dos dados representando-os em uma plataforma gráfica.

    Alguns métodos utilizados para visualização:

    • Boxplots
    • Scatter plots
    • Nuvens de palavras
    • Árvores de decisão
    • Várias ferramentas de análise de redes sociais como Igraph, MuxViz, NetworkX

Grafo G(V, A), onde V é um conjunto não vazio de objetos denominados vértices e A é um conjunto de pares não ordenados de V, chamado arestas.

Análise de Redes Sociais – Twitter

```
# Social Network Analytics - Twitter
# Comparando dados de diferentes temas

# Obs: Caso tenha problemas com a acentuação, consulte este link:
```

```
# https://support.rstudio.com/hc/en-us/articles/200532197-Character-Encoding

# Configurando o diretório de trabalho
# Coloque entre aspas o diretório de trabalho que você está usando no seu computador
# Não use diretórios com espaço no nome
setwd("Z:/Dropbox/DSA/Business-Analytics/R/Cap08/Twitter")
getwd()

# Carrega biblioteca com functions auxiliares e autenticação
source('utils.R')
source('autentication.R')

# Instalando os pacotes
#install.packages(c("devtools", "httr"))
#install.packages(c("tm", "wordcloud", "RColorBrewer"))
#install.packages(c("twitteR", "ROAuth"))

# Carregando os pacotes
library(devtools)
library(twitteR)
library(ROAuth)
library(tm)
library(wordcloud)
library(RColorBrewer)
library(stringr)

# Coletando os tweets
?searchTwitter
bigdata_tweets = searchTwitter("BigData", n = 100, lang = "pt")
datascience_tweets = searchTwitter("DataScience", n = 100, lang = "pt")
#dsa_Tweets = userTimeline(getUser('dsacademybr'), n = 100)

# Print
head(bigdata_tweets)
class(bigdata_tweets)
head(datascience_tweets)
class(datascience_tweets)

# Convertendo os tweets para texto
textos_bigdata = sapply(bigdata_tweets, function(x) x$getText())
head(bigdata_tweets)
textos_datascience = sapply(datascience_tweets, function(x) x$getText())
textos_bigdata[1:10]
class(textos_bigdata)

# Limpeza dos tweets
textos_bigdata_limpo = textos_bigdata
textos_bigdata_limpo <- limpaTweets(textos_bigdata_limpo)
head(textos_bigdata_limpo)
names(textos_bigdata_limpo) = NULL
textos_bigdata_limpo = textos_bigdata_limpo[textos_bigdata_limpo != ""]
textos_bigdata_limpo[1:10]
class(textos_bigdata_limpo)

textos_datascience_limpo = textos_datascience
textos_datascience_limpo <- limpaTweets(textos_datascience_limpo)
names(textos_datascience_limpo) = NULL
textos_datascience_limpo = textos_datascience_limpo[textos_datascience_limpo != ""]
textos_datascience_limpo[1:10]

# Converte para Corpus
tweetcorpus_bigdata <- Corpus(VectorSource(textos_bigdata_limpo))
tweetcorpus_bigdata <- limpaCorpus(tweetcorpus_bigdata)
tweetcorpus_datascience <- Corpus(VectorSource(textos_datascience_limpo))
tweetcorpus_datascience <- limpaCorpus(tweetcorpus_datascience)

# Converte o texto para a matriz de termos
termo_por_documento_bigdata     = as.matrix(TermDocumentMatrix(tweetcorpus_bigdata), control = list(stopwords = c(stopwords("portuguese"))))
termo_por_documento_datascience = as.matrix(TermDocumentMatrix(tweetcorpus_datascience), control = list(stopwords =
c(stopwords("portuguese"))))

# Verifica os primeiros 10 termos (linhas) com os primeiros 10 documentos (colunas)
termo_por_documento_bigdata[1:10,1:10]
termo_por_documento_datascience[1:10,1:10]

# Calcula a frequência de cada termo ao somar cada linha e coloca em ordem decrescente
frequencia_dos_termos_bigdata = sort(rowSums(termo_por_documento_bigdata), decreasing = TRUE)
head(frequencia_dos_termos_bigdata)
```

```
frequencia_dos_termos_datascience = sort(rowSums(termo_por_documento_datascience), decreasing = TRUE)
head(frequencia_dos_termos_datascience)

# Cria um dataframe com o termo (palavra) e sua respectiva frequência
df_bigdata = data.frame(termo = names(frequencia_dos_termos_bigdata), frequencia = frequencia_dos_termos_bigdata)
df_datascience = data.frame(termo = names(frequencia_dos_termos_datascience), frequencia = frequencia_dos_termos_datascience)

# Remove o termo mais frequente
df_bigdata = df_bigdata[-1,]
class(df_bigdata)
df_datascience = df_datascience[-1,]
class(df_datascience)

# Desenha a nuvem de palavras
wordcloud(df_bigdata$termo,
        df_bigdata$frequencia,
        max.words = 100,
        min.freq = 2,
        scale = c(3,.5),
        random.order = FALSE,
        colors = brewer.pal(8, "Dark2"))

wordcloud(df_datascience$termo,
        df_datascience$frequencia,
        max.words = 100,
        min.freq = 3,
        scale = c(3,.5),
        random.order = FALSE,
        colors = brewer.pal(8, "Dark2"))

# Merge dos dataframes
df_merge <- merge(df_bigdata, df_datascience, by = "termo")
head(df_merge)
df_merge$freq_total <- df_merge$frequencia.x + df_merge$frequencia.y
head(df_merge)

# Wordcloud do novo dataframe
wordcloud(df_merge$termo,
        df_merge$freq_total,
        max.words = 100,
        min.freq = 2,
        scale = c(3,.5),
        random.order = FALSE,
        colors = brewer.pal(8, "Dark2"))
```

```
# Autenticação

library(twitteR)

# Definindo as chaves de acesso
api_key <- "xxx"
api_secret <- "xxx"
access_token <- "xxx"
access_token_secret <- "xxx"

# Autenticando no Twitter
setup_twitter_oauth(api_key, api_secret, access_token, access_token_secret)
```

```
# Funções auxiliares

# Function para limpeza dos tweets
limpaTweets <- function(tweet){
  # Remove http links
  tweet = gsub("(f|ht)(tp)(s?)(://)(.*)[.|/](.*)", " ", tweet)
  tweet = gsub("http\\w+", "", tweet)
  # Remove retweets
  tweet = gsub("(RT|via)((?:\\b\\W*@\\w+)+)", " ", tweet)
  # Remove “#Hashtag”
  tweet = gsub("#\\w+", " ", tweet)
  # Remove nomes de usuarios “@people”
  tweet = gsub("@\\w+", " ", tweet)
  # Remove pontuacão
  tweet = gsub("[[:punct:]]", " ", tweet)
  # Remove os números
  tweet = gsub("[[:digit:]]", " ", tweet)
  # Remove espacos desnecessários
  tweet = gsub("[ \t]{2,}", " ", tweet)
```

```
  tweet = gsub("^\\s+|\\s+$", "", tweet)
  # Convertendo encoding de caracteres e convertendo para letra minúscula
  tweet <- stringi::stri_trans_general(tweet, "latin-ascii")
  tweet <- tryTolower(tweet)
  tweet <- iconv(tweet, from = "UTF-8", to = "ASCII")
}

# Function para limpeza de Corpus
limpaCorpus <- function(myCorpus){
  library(tm)
  myCorpus <- tm_map(myCorpus, tolower)
  # Remove pontuação
  myCorpus <- tm_map(myCorpus, removePunctuation)
  # Remove números
  myCorpus <- tm_map(myCorpus, removeNumbers)
}

# Converte para minúsculo
tryTolower = function(x)
{
  # Cria um dado missing (NA)
  y = NA
  # faz o tratamento do erro
  try_error = tryCatch(tolower(x), error=function(e) e)
  # se não der erro, transforma em minúsculo
  if (!inherits(try_error, "error"))
    y = tolower(x)
  # Retorna o resultado
  return(y)
}
```

Análise de Redes Sociais – Twitter – TweetBot

```
# Social Network Analytics - Twitter Bot
# Coletando dados da web e twittando em tempo real

# Obs: Caso tenha problemas com a acentuação, consulte este link:
# https://support.rstudio.com/hc/en-us/articles/200532197-Character-Encoding

# Configurando o diretório de trabalho
# Coloque entre aspas o diretório de trabalho que você está usando no seu computador
# Não use diretórios com espaço no nome
setwd("Z:/Dropbox/DSA/Business-Analytics/R/Cap08/Twitter")
getwd()


# Instalando pacotes
# install.packages("rvest")
# install.packages("stringr")
# install.packages("dplyr")
# install.packages("twitteR")

# Carregando os pacotes
library(rvest)
library(stringr)
library(dplyr)
library(twitteR)

# Web Scraping
url <- "https://cran.r-project.org/web/packages"

# Lendo a url
page <- read_html(url)

# Obtendo o num de pacotes
n_packages <- page %>%
  html_text() %>%
  str_extract("[[:digit:]]* available packages") %>%
  str_extract("[[:digit:]]*") %>%
  as.numeric()

print(n_packages)

# Num de pacotes lidos na ultima leitura
```

```
n_packages_last_time <- read.table(file = "n_packages.csv",stringsAsFactors = F, sep = ";")
n_packages_last_time <- n_packages_last_time$V2[nrow(n_packages_last_time)]

# Envia tweet somente se o num de pacotes mudou desde a ultima leitura
if(n_packages > n_packages_last_time) {

  # Definindo as chaves de acesso
  api_key <- "xxx"
  api_secret <- "xxx"
  access_token <- "xxx"
  access_token_secret <- "xxx"

  # Autenticando no Twitter
  setup_twitter_oauth(api_key, api_secret, access_token, access_token_secret)

  # Time
  time <- Sys.time()

  # Cria o tweet
  tweet_text <- paste0("Olá a todos, são ", time, " e neste momento existem ", n_packages, " pacotes R no CRAN. Aqui é o TweetBot DSA em
ação!")

  # Envia o tweet
  tweet(tweet_text)

  # Gravando no arquivo
  n_packages_df <- data.frame(time = Sys.time(), n = n_packages)
  write.table(n_packages_df, file = "n_packages.csv", row.names = FALSE,col.names = FALSE, append = TRUE, sep = ";")

}
```

Análise de Redes Sociais – Twitter Analytics

```
# Social Network Analytics - Twitter Analytics
# Analisando dados do Twitter

# Procedimento para solicitar seus tweets
# https://support.twitter.com/articles/20170330

# Obs: Caso tenha problemas com a acentuação, consulte este link:
# https://support.rstudio.com/hc/en-us/articles/200532197-Character-Encoding

# Configurando o diretório de trabalho
# Coloque entre aspas o diretório de trabalho que você está usando no seu computador
# Não use diretórios com espaço no nome
setwd("Z:/Dropbox/DSA/Business-Analytics/R/Cap08/Twitter")
getwd()

# Instalando os pacotes
install.packages("ggplot2")
install.packages("lubridate")
install.packages("scales")

# Carregando os pacotes
library(ggplot2)
library(lubridate)
library(scales)

# Carregando o dataset
tweets <- read.csv("tweets.csv", stringsAsFactors = FALSE)
head(tweets)
class(tweets)

# Ajustando a data
tweets$timestamp <- ymd_hms(tweets$timestamp)
tweets$timestamp <- with_tz(tweets$timestamp, "America/Sao_Paulo")
Sys.timezone(location = TRUE)

# Histograma com distribuição de tweets ao longo do tempo
ggplot(data = tweets, aes(x = timestamp)) +
  geom_histogram(aes(fill = ..count..)) +
  theme(legend.position = "none") +
  xlab("Time") + ylab("Número de tweets") +
  scale_fill_gradient(low = "midnightblue", high = "aquamarine4")
```

```
# Tweets por ano
ggplot(data = tweets, aes(x = year(timestamp))) +
  geom_bar(stat = "count", aes(fill = ..count..)) +
  theme(legend.position = "none") +
  xlab("Time") + ylab("Número de tweets") +
  scale_fill_gradient(low = "midnightblue", high = "aquamarine4")

# Tweets por dia da semana
ggplot(data = tweets, aes(x = wday(timestamp, label = TRUE))) +
  geom_bar(stat = "count", aes(fill = ..count..)) +
  theme(legend.position = "none") +
  xlab("Dia da Semana") + ylab("Número de tweets") +
  scale_fill_gradient(low = "midnightblue", high = "aquamarine4")


# Testes Estatísticos

# Um teste apropriado nesse caso é um teste de qui-quadrado de amostra única.
# Com esse teste poderemos ver se a distribuição que temos é consistente com uma determinada hipótese,
# dentro de erro de amostragem aleatória.
# Por exemplo, poderíamos ter obtido esta distribuição de tweets apenas por acaso, ou realmente temos menos tweets
# nos fins de semana? Este tipo de teste pode ser feito com qualquer tipo de frequência esperada,
# mas primeiro vamos comparar contra a hipótese de frequências esperadas iguais, ou seja, a hipótese de que a DSA "tweeta"
# na mesma taxa em todos os dias e que essa distribuição de tweets seja apenas por acaso e amostragem aleatória.
chisq.test(table(wday(tweets$timestamp, label = TRUE)))

# O teste do qui-quadrado indica que a distribuição de meus tweets é altamente improvável de ser uniforme.
# Podemos rejeitar a hipótese nula (a hipótese de que a DSA "tweeta" na mesma taxa em todos os dias),
# com um alto grau de confiança.

# A distribuição nos tweets pode ser explicada apenas como uma diferença entre o comportamento
# dos dias úteis e dos fins de semana?
# Parece que de segunda a quinta-feira são mais altos do que sexta-feira a domingo.
myTable <- table(wday(tweets$timestamp, label = TRUE))
mean(myTable[c(2:5)])/mean(myTable[c(1,6,7)])

# Os valores para segunda-feira a quinta-feira são 1.370274 mais elevados do que os outros dias, em média, perto de 5/4.
# Vamos ver se o teste chi-quadrado diz que o meu padrão de tweets é consistente com "tweetar" 1.37 vezes mais vezes
# entre segunda-feira e quinta-feira do que entre sexta-feira até domingo.
chisq.test(table(wday(tweets$timestamp, label = TRUE)), p = c(4, 5, 5, 5, 5, 4, 4)/32)

# O valor-p aqui é ainda muito baixo, assim nós podemos rejeitar esta hipótese simples.


# Visualizando os Resultados

# Padrão mensal de tweets
ggplot(data = tweets, aes(x = month(timestamp, label = TRUE))) +
  geom_bar(aes(fill = ..count..)) +
  theme(legend.position = "none") +
  xlab("Mês") + ylab("Número de tweets") +
  scale_fill_gradient(low = "midnightblue", high = "aquamarine4")

# Este é um padrão anual bastante interessante. A diminuição em janeiro e fevereiro faz sentido porque esta é geralmente
# uma época de férias.

# Distribuição de tweets por hora
tweets$timeonly <- as.numeric(tweets$timestamp - trunc(tweets$timestamp, "days"))
tweets[(minute(tweets$timestamp) == 0 & second(tweets$timestamp) == 0),11] <- NA
mean(is.na(tweets$timeonly))

class(tweets$timeonly) <- "POSIXct"
ggplot(data = tweets, aes(x = timeonly)) +
  geom_histogram(aes(fill = ..count..)) +
  theme(legend.position = "none") +
  xlab("Time") + ylab("Número de tweets") +
  scale_x_datetime(breaks = date_breaks("3 hours"),
                   labels = date_format("%H:00")) +
  scale_fill_gradient(low = "midnightblue", high = "aquamarine4")

# Tweets no fim do dia
latenighttweets <- tweets[(hour(tweets$timestamp) < 6),]
ggplot(data = latenighttweets, aes(x = timestamp)) +
  geom_histogram(aes(fill = ..count..)) +
  theme(legend.position = "none") +
  xlab("Time") + ylab("Número de tweets") + ggtitle("Tweets Noturnos") +
  scale_fill_gradient(low = "midnightblue", high = "aquamarine4")
```

```
# Tweets com ou sem hashtags
ggplot(tweets, aes(factor(grepl("#", tweets$text)))) +
  geom_bar(fill = "midnightblue") +
  theme(legend.position="none", axis.title.x = element_blank()) +
  ylab("Número de tweets") +
  ggtitle("Tweets com Hashtags") +
  scale_x_discrete(labels=c("Sem hashtags", "Tweets com hashtags"))

# Retweets
ggplot(tweets, aes(factor(!is.na(retweeted_status_id)))) +
  geom_bar(fill = "midnightblue") +
  theme(legend.position="none", axis.title.x = element_blank()) +
  ylab("Número de tweets") +
  ggtitle("Retweeted Tweets") +
  scale_x_discrete(labels=c("Sem retweet", "Com Retweeted"))

# Tweets respondidos
ggplot(tweets, aes(factor(!is.na(in_reply_to_status_id)))) +
  geom_bar(fill = "midnightblue") +
  theme(legend.position="none", axis.title.x = element_blank()) +
  ylab("Número de tweets") +
  ggtitle("Tweets Respondidos") +
  scale_x_discrete(labels=c("Sem resposta", "Com resposta"))

# Unindo todas as informaçções em um único gráfico
tweets$type <- "tweet"
tweets[(!is.na(tweets$retweeted_status_id)),12] <- "RT"
tweets[(!is.na(tweets$in_reply_to_status_id)),12] <- "reply"
tweets$type <- as.factor(tweets$type)
tweets$type = factor(tweets$type,levels(tweets$type)[c(3,1,2)])

ggplot(data = tweets, aes(x = timestamp, fill = type)) +
  geom_histogram() +
  xlab("Time") + ylab("Número de tweets") +
  scale_fill_manual(values = c("midnightblue", "deepskyblue4", "aquamarine3"))
```

Análise de Redes Sociais – Facebook

```
# Facebook Analytics
# https://github.com/pablobarbera/Rfacebook

# Obs: Caso tenha problemas com a acentuação, consulte este link:
# https://support.rstudio.com/hc/en-us/articles/200532197-Character-Encoding

# Configurando o diretório de trabalho
# Coloque entre aspas o diretório de trabalho que você está usando no seu computador
# Não use diretórios com espaço no nome
setwd("Z:/Dropbox/DSA/Business-Analytics/R/Cap08/Facebook")
getwd()

# Instalando pacote RFacebook a partir do Github
library(devtools)
install_github("pablobarbera/Rfacebook", subdir = "Rfacebook", force = TRUE)

# Carregando o pacote
library(Rfacebook)

# Usando token com duração de 2 horas
token <- "xxxxx"

# Obtendo dados do usuário
user <- getUsers("radiocbn", token, private_info = TRUE)
user$name

# Configurando autenticação
# https://developers.facebook.com/apps/
require("Rfacebook")

# Autenticando por 2 meses
fb_oauth <- fbOAuth(app_id = "xxxxxx", app_secret = "xxxxxx", extended_permissions = TRUE)

# Salvando a configuração
save(fb_oauth, file = "fb_oauth")
```

```
load("fb_oauth")

# Obtendo a lista de usuários
friends <- getFriends(token, simplify = TRUE)
head(friends)

# Obtendo dados sobre os amigos
friends_data <- getUsers(friends$id, token, private_info = TRUE)

# Sexo
table(friends_data$gender)

# Idioma
table(substr(friends_data$locale, 1, 2))

# País
table(substr(friends_data$locale, 4, 5))

# Status
table(friends_data$relationship_status)


## Network analysis
?getNetwork
network <- getNetwork(token, format = "adj.matrix")
head(network)

# Usando igraph
install.packages("igraph")
require(igraph)

# Social graph
?graph.adjacency
social_graph <- graph.adjacency(network)
layout <- layout.drl(social_graph, options = list(simmer.attraction = 0))

# Plot
plot(social_graph,
    vertex.size = 10,
    vertex.color = "green",
    vertex.label = NA,
    vertex.label.cex = 0.5,
    edge.arrow.size = 0,
    edge.curved = TRUE,
    layout = layout.fruchterman.reingold)

# Salvando em png
dev.copy(png,filename = "network.png", width = 600, height = 600)
dev.off()

# Degree - número de conexões diretas que um node tem dentro da rede
# Um degree alto significa que o node tem muitas conexões diretas dentro da rede
degree(social_graph,
      v = V(social_graph),
      mode = c("all", "out", "in", "total"),
      loops = TRUE,
      normalized = FALSE)

degree.distribution(social_graph, cumulative = FALSE)

# Betweenness - conceito de centralização
# É calculado com base em quantos pares de indivíduos (outros nós na rede) teriam que passar por
# você (nó para o qual ele é calculado), a fim de alcançar um outro no número mínimo de saltos.
# O nó com maior alcance terá uma maior influência no fluxo da informação.
betweenness(social_graph,
          v = V(social_graph),
          directed = TRUE,
          weights = NULL,
          nobigint = TRUE,
          normalized = FALSE)

# Closeness
# O quão central você está (nó para o qual é calculado) depende do comprimento do caminho mais curto
# médio entre o nó de medição e todos os outros nós na rede. Os nós com alta proximidade são muito
# importantes porque estão em uma excelente posição para monitorar o que está acontecendo na rede,
# ou seja, nós com maior visibilidade. Esta medida pode não ser muito útil quando nossa rede tem
# muitos componentes desconectados.
closeness(social_graph,
```

```
        vids = V(social_graph),
        mode = c("out", "in", "all", "total"),
        weights = NULL,
        normalized = FALSE)

# Cluster in network
# Cluster é uma medida em que os nós da rede tendem a se agrupar uns com os outros.
# Podemos ver quantos clusters existem na nossa rede usando a seguinte função:
is.connected(social_graph, mode = c("weak", "strong"))
clusters(social_graph, mode = c("weak", "strong"))

# Comunidade
# Depois de verificar o número de clusters na rede, vamos verificar como esses clusters são espalhados na rede.
# A função modularity() é utilizada para detectar as comunidades na rede. Ela mede como modular uma
# divisão dada de um gráfico de rede em subgrafos é, isto é, quão forte é uma divisão dentro de uma rede.
# As redes com alto grau de modularidade têm conexões fortes entre os nós dentro de seu cluster
# (grupo / comunidade).
network_Community <- walktrap.community(social_graph)
modularity(network_Community)

# Plot
plot(network_Community,
    social_graph,
    vertex.size = 10,
    vertex.label.cex = 0.5,
    vertex.label = NA,
    edge.arrow.size = 0,
    edge.curved = TRUE,
    layout=layout.fruchterman.reingold)

# Salva a imagem
dev.copy(png,filename = "comunidade.png", width = 600, height = 600)
dev.off()
```

Mini-Projeto 4 – Analisando Dados do Facebook

O Facebook tem atualmente 1.86 bilhão de usuários, com cerca de 1.23 bilhão de usuários conectando na rede todos os dias. Como poderia alguma empresa ignorar todos esses potenciais consumidores e clientes? Não poderia.

Qualquer estratégia de Marketing de qualquer empresa atenta aos movimentos das mídias sociais, deve levar em consideração ações no Facebook, como forma de divulgar produtos e serviços e interagir em tempo real com seus clientes e parceiros.

Neste projeto, vamos conectar no Facebook e buscar dados que podem ser usados na tomada de decisões e como forma de avaliar a efetividade de campanhas de Marketing.

Primeiro você vai coletar posts, likes, comentários e compartilhamentos, vai analisar esses dados, buscar posts com mais comentários e/ou likes e ordenar os usuários com maior número de interações com os posts.

Na sequência você vai coletar comentários em diversos posts e minerar esses dados usando linguagem SQL e extraindo informações úteis que podem ser usadas mais os mais variados fins, inclusive processamento de linguagem natural e análise de sentimentos.

Finalmente, você vai avaliar a performance de uma página no Facebook e calcular a evolução dos posts desta página ao longo do tempo, gerando um gráfico como este abaixo.

Utilize esse projeto para realizar as análises na página da sua empresa, avaliar a efetividade de campanhas de Marketing e ajudar os tomadores de decisão a definir novas estratégias.

```
# Facebook Analytics
```

```
# https://github.com/pablobarbera/Rfacebook

# Obs: Caso tenha problemas com a acentuação, consulte este link:
# https://support.rstudio.com/hc/en-us/articles/200532197-Character-Encoding

# Configurando o diretório de trabalho
# Coloque entre aspas o diretório de trabalho que você está usando no seu computador
# Não use diretórios com espaço no nome
setwd("Z:/Dropbox/DSA/Business-Analytics/R/Cap08/Facebook")
getwd()

# Instalando pacote RFacebook a partir do Github
library(devtools)
install_github("pablobarbera/Rfacebook", subdir = "Rfacebook", force = TRUE)

# Carregando o pacote
library(Rfacebook)

# Usando token com duração de 2 horas
token <- "xxx"

# Configurando autenticação
# https://developers.facebook.com/apps/
require("Rfacebook")

# Autenticando por 2 meses
fb_oauth <- fbOAuth(app_id = "xxxx", app_secret = "xxxx", extended_permissions = TRUE)

# Salvando a configuração
save(fb_oauth, file = "fb_oauth")
load("fb_oauth")


## Dados de Páginas no Facebook

### TROCANDO NOME DA PAGINA NVIDIA
# Obtém dados de uma página
?getPage
page <- getPage("NVIDIAGeForce.BR", token, n = 400)
View(page)
class(page)

# Busca por páginas
?searchPages
pages <- searchPages(string = "Data Science", token = token, n = 200)
View(pages)

# Post com maior número de likes
page[which.max(page$likes_count), ]

# Filtrando pela data mais recente
pageRecent <- page[which(page$created_time > "2017-04-02"), ]

# Ordenando pelo número de likes
top <- pageRecent[order(- pageRecent$likes),]
head(top, n = 2)

# Verificando trend
post1 <- getPost("130196237030986_1290400904343841", token, n = 1000, likes = FALSE, comments = FALSE)
post2 <- getPost("130196237030986_1317789011605030", token, n = 1000, likes = FALSE, comments = FALSE)
View(post2)


# Usando Linguagem SQL e Minerando dados do Facebook
post_id <- head('130196237030986_1290400904343841', n = 1)

# Busca likes e comentários
post <- getPost(post_id, token, n = 1000, likes = TRUE, comments = TRUE)
head(post$comments, n = 2)

# Gravando os comentários
comments <- post$comments
class(comments)
View(comments)

# Usando SQL
install.packages("sqldf")
library(sqldf)
```

```
# Usuários mais influenciadores
infusers <- sqldf("select from_name, sum(likes_count) as totlikes from comments group by from_name")
head(infusers)

# Buscando os "top" influenciadores
infusers$totlikes <- as.numeric(infusers$totlikes)
top <- infusers[order(- infusers$totlikes),]
head(top, n = 10)

# Buscando comentários de vários posts simultaneamente
post_id <- head(page$id, n = 100)
head(post_id, n = 10)
post_id <- as.matrix(post_id)
allcomments <- ""

# Percorrendo todos os posts e coletando comentários de todos eles
for (i in 1:nrow(post_id))
{
  post <- getPost(post_id[i,], token, n = 1000, likes = TRUE, comments = TRUE)
  comments <- post$comments
  allcomments <- rbind(allcomments, comments)
}

# Usuários mais influentes
infusers <- sqldf("select from_name, sum(likes_count) as totlikes from allcomments group by from_name")
infusers$totlikes <- as.numeric(infusers$totlikes)
top <- infusers[order(- infusers$totlikes),]
head(top, n = 20)
View(allcomments)


# Performance da página

# Converte o formato de data do Facebook para o formato do R
format.facebook.date <- function(datestring) {
  date <- as.POSIXct(datestring, format = "%Y-%m-%dT%H:%M:%S+0000", tz = "GMT")
}

# Agregando a métrica de counts por mês
aggregate.metric <- function(metric) {
  m <- aggregate(page[[paste0(metric, "_count")]], list(month = page$month), mean)
  m$month <- as.Date(paste0(m$month, "-15"))
  m$metric <- metric
  return(m)
}

# Obtendo os posts da página da Nvidia
page <- getPage("NVIDIAGeForce.BR", token, n = 500)

# Aplicando a agregação aos dados coletados
page$datetime <- format.facebook.date(page$created_time)
page$month <- format(page$datetime, "%Y-%m")
df.list <- lapply(c("likes", "comments", "shares"), aggregate.metric)
df <- do.call(rbind, df.list)

# Plot
library(ggplot2)
library(scales)

ggplot(df, aes(x = month, y = x, group = metric)) +
  geom_line(aes(color = metric)) +
  scale_y_log10("Média de Counts por Mês", breaks = c(10, 100, 1000, 10000, 50000)) +
  theme_bw() +
  theme(axis.title.x = element_blank()) +
  ggtitle("Performance da Página no Facebook")

# Salvando a imagem
ggsave(file="chart.png", dpi = 500)
```

Análise de Redes Sociais – Instagram

```
# Instagram Analytics
```

```
# Obs: Caso tenha problemas com a acentuação, consulte este link:
# https://support.rstudio.com/hc/en-us/articles/200532197-Character-Encoding

# Configurando o diretório de trabalho
# Coloque entre aspas o diretório de trabalho que você está usando no seu computador
# Não use diretórios com espaço no nome
setwd("Z:/Dropbox/DSA/Business-Analytics/R/Cap08/Instagram")
getwd()

# Pacotes
library(devtools)
library(RCurl)
library(httr)
library(rjson)

# Pacote instaR
# https://cran.r-project.org/web/packages/instaR/instaR.pdf
install_github("pablobarbera/instaR/instaR")
library(instaR)

# Dados para autenticação
app_name <- "DSA-app"
client_id <- "xxx"
client_secret <- "xxx"
scope = "public_content"

# Gerando token
token <- instaOAuth(client_id, client_secret, scope = "public_content")
print(token)
my_token <- token$credentials$access_token

# Salvando o token em disco
save(token, file = "instagram-token")
load(token)

# Acessando end-points
# https://www.instagram.com/developer/endpoints/users/

# Coletando info do usuário
user_info <- paste('https://api.instagram.com/v1/users/self/?access_token=', my_token, sep = "")
user_info_data <- getURL(user_info)
json_data_user <- fromJSON(user_info_data)
View(json_data_user)

# Coletando mídias mais recentes
recent_media <- paste('https://api.instagram.com/v1/users/self/media/recent/?access_token=', my_token, sep = "")
media_data <- getURL(recent_media)
json_data_media <- fromJSON(media_data)
print(json_data_media)

# Coletando likes
likes <- paste('https://api.instagram.com/v1/users/self/media/liked?access_token=', my_token, sep = "")
likes_data <- getURL(likes)
json_data_likes <- fromJSON(likes_data)
print(json_data_likes)


### PERMISSÃO SENDO AVALIADA
###################### Somente funciona em Apps com Permissões ######################
# Você precisa submeter sua app para aprovação pelo Instagram. Todos os detalhes na página da api.

# Coletando dados
?searchInstagram
data <- searchInstagram("DataScience", token, n = 5, folder = "DataScience")
head(data,2)
names(data)
head(data$comments_count)

# Obtendo fotos de uma localidade
data <- searchInstagram("DataScience", token, n = 10, lat = 13.1633, lng = 72.5456, folder = "DataScience")

# Download das 15 fotos mais atuais
instag <- getUserMedia("dsacademybr", token, n = 15, folder = "instagram")
names(instag)
head(instag)

# User profile
usr <- getUser("dsacademybr", token)
```

```
head(usr)
names(usr)

# Obtendo a lista de usuários seguidores
instaf <- getFollowers("instagram", token)
head(instaf,2)
names(instaf)
nrow(instaf)

# Lista de usuários que são seguidos
instaff <- getFollows("instagram", token)
head(instaff,3)
nrow(instaff)

# Obtendo comentários
comm <- getComments("1026058420312409485_25025320", token)
comm$text
tail(comm)
names(comm)
?getComments
class(comm)

# Obtendo a quantidade de vezes que uma hashtag é usada
tag1 <- getTagCount("trump", token)
tag1
tag2 <- getTagCount("bigdata", token)
tag2


###################### Exemplos usando arquivos CSV ######################
# Veremos como realizar análises com dados do Instagram sem a necessidade de conectar com a api.

## Preparando os dados
userAndTags <- read.csv("dataset1-hashtags.csv")
names(userAndTags)
head(userAndTags)

# Extraindo os usuários
users <- userAndTags$Users
users <- as.matrix(users)

# User media
usermedia <- read.csv("dataset2-usermedia.csv")
head(usermedia)
tags <- userAndTags$Hashtags
tags <- as.matrix(tags)
hashmedia <- read.csv("dataset3-hashmedia.csv")
head(hashmedia)

# Juntando os datasets
alldata <- rbind(usermedia, hashmedia)
head(alldata)

## Follows data
users <- userAndTags$Users
users <- as.matrix(users)
userfollows <- read.csv("dataset4-userfollows.csv")
View(userfollows)

## Consolidando user profile

users <- userAndTags$Users
users <- as.matrix(users)
userprofiles <- read.csv("dataset5-userprofiles.csv")
View(userprofiles)

## Quem possui mais seguidores
mostfollowed <- userprofiles[with(userprofiles, order(-followed_by)), ]
head(mostfollowed$full_name, 15)

## Quem segue mais?
mostfollows <- userprofiles[with(userprofiles, order(-follows)), ]
head(mostfollows$full_name, 15)

# Quem publica mais?
mostmedias <- userprofiles[with(userprofiles, order(-media_count)), ]
head(mostmedias$full_name, 15)
```

```
# Top users
userprofiles$overallmetric <- ((userprofiles$media_count/max(userprofiles$media_count)) +
(userprofiles$followed_by/max(userprofiles$followed_by)) +(userprofiles$followed_by/max(userprofiles$followed_by)))*100
overallmet <- userprofiles[with(userprofiles, order(-overallmetric)), ]
head(overallmet$full_name, 15)
View(overallmet)

# Mais comentados
head(alldata)
mostcomm <- alldata[with(alldata, order(-comments_count)), ]
View(mostcomm)

# Com mais likes
head(alldata)
mostlikes <- alldata[with(alldata, order(-likes_count)), ]
View(mostlikes)

## Todas as localidades
library(sqldf)
names(alldata)
allloc <- sqldf("select distinct location_name from alldata")
allloc <- na.omit(allloc)
head(allloc, 20)
nrow(allloc)

# localidades com mais likes
loclikes <- sqldf("select location_name, sum(likes_count) as totlikes from alldata group by location_name")
loc <- loclikes[with(loclikes, order(-totlikes)), ]
loc <- na.omit(loc)
head(loc, 25)

# Localidade mais falada
loccomments <- sqldf("select location_name, sum(comments_count) as totcomm from alldata group by location_name")
loccomm <- loccomments[with(loccomments, order(-totcomm)), ]
loccomm <- na.omit(loccomm)
head(loccomm, 15)

# Localidade que ocorre com mais frequência
locations <- sqldf("select location_name, count(location_id) as locid from alldata group by location_name")
location <- locations[with(locations, order(-locid)), ]
location <- na.omit(location)
head(location,5)

# Com mais likes, por usuário
names(alldata)
userlikes <- sqldf("select username, sum(likes_count) as totlikes from alldata group by username")
user <- userlikes[with(userlikes, order(-totlikes)), ]
user <- na.omit(user)
head(user, 25)

# Wordcloud
library(wordcloud)
library(tm)

# Limpeza nos dados
options(warn=-1)
words <- strsplit(as.character(alldata$caption), " ")
words <- lapply(words, function(x) x[grep("^[A-Za-z0-9]+$", x)])
words <- unlist(words)
words <- tolower(words)
words <- words[-grep("^[rm]t$", words)]

# Removendo stop words
stopWords <- stopwords("en")
"%!in%" <- function(x,table) match(x,table, nomatch = 0) == 0
words <- words[words %!in% stopWords]

# Criando um objeto com a wordcloud
allwords <- as.data.frame(table(words))
wc <- wordcloud(allwords$words, allwords$Freq,
                random.order = FALSE,
                min.freq = 5,
                colors = brewer.pal(2, "Dark2"))

# Gravando a wordcloud
dev.copy(png,filename = "wordcloud.png", width = 600, height = 875);
dev.off ();
```

```
## Clustering
??fpc
library(fpc)

head(alldata)
data <- alldata

# Extraindo algumas colunas
cdata <- data.frame(data$type, data$comments_count, data$likes_count, data$user_id)

# Nomeando as colunas
colnames(cdata) <- c("type","comments","likes","user_id")

# Convertendo para integer (alguns algoritmos de clusters somente aceitam números como entrada)
cdata$user_id <- as.integer(cdata$user_id)
cdata$type <- as.integer(cdata$type)

# Visualizando os dados
View(data)
View(cdata)

# Estimando o número de clusters (automático)
?pamk
clusters <- pamk(cdata)
n <- clusters$nc
n

# Estimando o número de clusters (manual)
# Calculando a soma dos erros quadrados
wss <- (nrow(cdata)-1)*sum(apply(cdata,2,var))

# Buscando a soma dos erros quadrados dentro dos grupos
for (i in 2:25) wss[i] <- sum(kmeans(cdata, centers=i)$withinss)

# Plot dos clusters
# Logicamente, à medida que o número de clusters aumenta, a soma dos erros quadrados reduz.
# Se houver n objetos em um conjunto de dados, então n clusters resultaria em erro 0, mas idealmente
# precisamos parar em algum ponto. De acordo com as teorias, a taxa de diminuição na soma dos erros irá
# cair de repente em um ponto e que deve ser considerado como o número ideal de clusters.
# De acordo com o gráfico a seguir, o número ideal de cluster é 4.
plot(1:25, wss, type = "b", xlab = "Número de clusters", ylab = "Soma dos Quadrados Dentro dos grupos")

# Salvando o plot
dev.copy(png,filename = "clustering.png", width = 600, height = 875)
dev.off ();

## K-Means

# K-Means Cluster Analysis
?kmeans
fit <- kmeans(cdata, 4)
fit$cluster

# Número de elementos em cada cluster
table(fit$cluster)

# Médias dos clusters
aggregate(cdata, by = list(fit$cluster), FUN = mean)

# Observações em cada cluster
resultado <- cbind(cdata, clusterNum = fit$cluster)
View(resultado)

# Plot do cluster
library(fpc)
plotcluster(cdata, fit$cluster)
dev.copy(png, filename = "clusterPlot.png", width = 600, height = 875);
dev.off ();



## Sistema de Recomendação Baseado em Usuário
## Projeto 5 - Construindo um Sistema de Recomendação
library(data.table)

# Carregando o dataset com seguidores
```

```
userfollows <- read.csv("dataset4-userfollows.csv")
head(userfollows)
names(userfollows)

# Temos a variável precedente no conjunto de dados. Para construir o mecanismo de recomendação e fornecer
# recomendações aos usuários, precisamos apenas de duas colunas. Portanto, selecionamos essas duas colunas
# usando a função data.frame:
fdata <- data.frame(userfollows$users.i..1., userfollows$username)
colnames(fdata) <- c("user","follows")
head(fdata)

# Pivot dos dados
# Agora, temos de manipular o conjunto de dados de tal forma que os usuários se tornem a coluna e
# os usuários que seguem, se tornem as linhas. Assim, torna-se fácil calcular a correlação entre os usuários.
# Para girar os dados, precisamos usar a função dcast.data.table, que requer o pacote data.table.
pivoting <- data.table(fdata)
pivotdata <- dcast.data.table(pivoting, follows ~ user, fun.aggregate = length, value.var = "user")
write.csv(pivotdata, "dataset5-pivot-follows-temp.csv")

# Após deletar a coluna
 de índice e o usuário nulo

# Lendo os dados
data <- read.csv("dataset5-pivot-follows.csv")
head(data)
colnames(data)

# Removendo a coluna user
data.ubs <- (data[,!(names(data) %in% c("users"))])

# Função que calcula a distância entre 2 vetores
# Podemos calcular a similaridade dos usuários usando diferentes métodos.
# Em nosso caso, usaremos a técnica de similaridade de coseno para obter a pontuação de similaridade
# para todos os pares de usuários. Em nosso conjunto de dados, zero significa que o usuário não está seguindo.
# Se considerarmos essas linhas com zero, enquanto calculamos a similaridade usando correlação ou qualquer
# outra técnica, vamos acabar com uma saída tendenciosa que está longe da realidade.
# Assim, ao calcular a pontuação de similaridade, consideraremos somente as linhas não nulas.
# A seguinte função calcula a similaridade entre os usuários usando o método de similaridade de coseno.
getCosine <- function(x,y)
{
  dat <- cbind(x,y)
  f <- as.data.frame(dat)
  # Remove as linhas com zeros
  datn<- f[-which(rowSums(f==0)>0),]
  if(nrow(datn) > 2)
  {
    this.cosine <- sum(x*y) / (sqrt(sum(x*x)) * sqrt(sum(y*y)))
  }
  else
  {
    this.cosine <- 0
  }
  return(this.cosine)
}

# Agora, precisamos construir uma matriz de similaridade que nos dirá como os usuários são
# semelhantes entre si. Antes de computar a similaridade, vamos construir uma matriz vazia que pode
# ser usada para armazenar a similaridade:
data.ubs.similarity <- matrix(NA,
                              nrow = ncol(data.ubs),
                              ncol = ncol(data.ubs),
                              dimnames = list(colnames(data.ubs),
                                              colnames(data.ubs)))

# Aplicando o cálculo de similaridade a todas as colunas
# Agora podemos começar a substituir as células vazias na matriz de similaridade com a pontuação de
# similaridade real. No caso da similaridade de coseno, o intervalo será de -1 a + 1.
# O seguinte loop ajudará a calcular a similaridade entre todos os usuários.
# Se não houver dados suficientes para calcular a similaridade de acordo com nossa função,
# ela retornará zero. A instrução print no seguinte loop nos ajudará a entender o progresso do loop.
# Dependendo do conjunto de dados, o tempo necessário varia.
for(i in 1:ncol(data.ubs)) {
  # Loop em todas as colunas
  for(j in 1:ncol(data.ubs)) {
    # Calcula a similaridade e alimenta o dataframe
    data.ubs.similarity[i,j] <- getCosine(as.matrix(data.ubs[i]),as.matrix(data.ubs[j]))
  }
  print(i)
```

```
}

# Converte a matriz de similaridade em dataframe
data.ubs.similarity <- as.data.frame(data.ubs.similarity)

# Replace NA com 0
data.ubs.similarity[is.na(data.ubs.similarity)] <- 0
head(data.ubs.similarity)

# Obtendo os 10 vizinhos de cada usuário
data.neighbours <- matrix(NA,
                          nrow = ncol(data.ubs.similarity),
                          ncol = 11,
                          dimnames = list(colnames(data.ubs.similarity)))

# Gerando as recomendações
for(i in 1:ncol(data.ubs))
{
  # Evitando valores zero
  n <- length(data.ubs.similarity[,i])
  thres <- sort(data.ubs.similarity[,i],partial=n-10)[n-10]
  if(thres > 0.020)
  {
    # Selecionando as recomendações Top 10
    data.neighbours[i,] <- (t(head(n=11,rownames(data.ubs.similarity[order(data.ubs.similarity[,i],decreasing=TRUE),][i]))))
  }
  else
  {
    data.neighbours[i,] <- ""
  }
}

# Visualizando as recomendações
# No código anterior, pegamos um usuário de cada vez e, em seguida, classificamos a pontuação de similaridade
# do usuário com todos os outros usuários, de modo que o par com a maior similaridade venha primeiro.
# Então, paramos apenas filtrando os 10 primeiros para cada um dos usuários. Isso é recomendado para nós.
# Podemos ver as recomendações dadas para os usuários::
View(data.neighbours)

# Gravando as recomendações
write.csv(data.neighbours, "Recomendacoes.csv")
```

Mini-Projeto 5 – Construindo Um Sistema de Recomendação

Recomendações tornaram-se muito comuns hoje em dia. Muitas empresas online como Amazon, Facebook, LinkedIn e Instagram fornecem recomendações. Estas recomendações são produzidas pelo sistema de recomendação que não é nada mais do que um algoritmo que usa alguns dos dados históricos para prever o que o usuário gostaria. A recomendação pode ser implementada usando o algoritmo de filtragem colaborativa, que pode ser implementado usando uma metodologia baseada em usuário (user based) ou item (item based). Neste projeto, veremos em detalhes a implementação do algoritmo usando a filtragem baseada no usuário.

Utilizaremos as informações dos usuários cujos dados foram baixados. Embora esses dados sejam do Instagram, o projeto pode ser aplicado a qualquer conjunto de dados com características semelhantes.

Nosso objetivo é criar um sistema que recomende aos usuários, perfis no Instagram, de acordo com os perfis que são seguidos atualmente pelo mesmo usuário. De forma automática.

```
# Instagram Analytics

# Obs: Caso tenha problemas com a acentuação, consulte este link:
# https://support.rstudio.com/hc/en-us/articles/200532197-Character-Encoding

# Configurando o diretório de trabalho
```

```
# Coloque entre aspas o diretório de trabalho que você está usando no seu computador
# Não use diretórios com espaço no nome
setwd("Z:/Dropbox/DSA/Business-Analytics/R/Cap08/Instagram")
getwd()

# Pacotes
library(devtools)
library(RCurl)
library(httr)
library(rjson)

# Pacote instaR
# https://cran.r-project.org/web/packages/instaR/instaR.pdf
install_github("pablobarbera/instaR/instaR")
library(instaR)

# Dados para autenticação
app_name <- "DSA-app"
client_id <- "xxx"
client_secret <- "xxx"
scope = "public_content"

# Gerando token
token <- instaOAuth(client_id, client_secret, scope = "public_content")
print(token)
my_token <- token$credentials$access_token

# Salvando o token em disco
save(token, file = "instagram-token")
load(token)

# Acessando end-points
# https://www.instagram.com/developer/endpoints/users/

# Coletando info do usuário
user_info <- paste('https://api.instagram.com/v1/users/self/?access_token=', my_token, sep = "")
user_info_data <- getURL(user_info)
json_data_user <- fromJSON(user_info_data)
View(json_data_user)

# Coletando mídias mais recentes
recent_media <- paste('https://api.instagram.com/v1/users/self/media/recent/?access_token=', my_token, sep = "")
media_data <- getURL(recent_media)
json_data_media <- fromJSON(media_data)
print(json_data_media)

# Coletando likes
likes <- paste('https://api.instagram.com/v1/users/self/media/liked?access_token=', my_token, sep = "")
likes_data <- getURL(likes)
json_data_likes <- fromJSON(likes_data)
print(json_data_likes)


### PERMISSÃO SENDO AVALIADA
###################### Somente funciona em Apps com Permissões ######################
# Você precisa submeter sua app para aprovação pelo Instagram. Todos os detalhes na página da api.

# Coletando dados
?searchInstagram
data <- searchInstagram("DataScience", token, n = 5, folder = "DataScience")
head(data,2)
names(data)
head(data$comments_count)

# Obtendo fotos de uma localidade
data <- searchInstagram("DataScience", token, n = 10, lat = 13.1633, lng = 72.5456, folder = "DataScience")

# Download das 15 fotos mais atuais
instag <- getUserMedia("dsacademybr", token, n = 15, folder = "instagram")
names(instag)
head(instag)

# User profile
usr <- getUser("dsacademybr", token)
head(usr)
names(usr)

# Obtendo a lista de usuários seguidores
```

```
instaf <- getFollowers("instagram", token)
head(instaf,2)
names(instaf)
nrow(instaf)

# Lista de usuários que são seguidos
instaff <- getFollows("instagram", token)
head(instaff,3)
nrow(instaff)

# Obtendo comentários
comm <- getComments("1026058420312409485_25025320", token)
comm$text
tail(comm)
names(comm)
?getComments
class(comm)

# Obtendo a quantidade de vezes que uma hashtag é usada
tag1 <- getTagCount("trump", token)
tag1
tag2 <- getTagCount("bigdata", token)
tag2


###################### Exemplos usando arquivos CSV ######################
# Veremos como realizar análises com dados do Instagram sem a necessidade de conectar com a api.

## Preparando os dados
userAndTags <- read.csv("dataset1-hashtags.csv")
names(userAndTags)
head(userAndTags)

# Extraindo os usuários
users <- userAndTags$Users
users <- as.matrix(users)

# User media
usermedia <- read.csv("dataset2-usermedia.csv")
head(usermedia)
tags <- userAndTags$Hashtags
tags <- as.matrix(tags)
hashmedia <- read.csv("dataset3-hashmedia.csv")
head(hashmedia)

# Juntando os datasets
alldata <- rbind(usermedia, hashmedia)
head(alldata)

## Follows data
users <- userAndTags$Users
users <- as.matrix(users)
userfollows <- read.csv("dataset4-userfollows.csv")
View(userfollows)

## Consolidando user profile

users <- userAndTags$Users
users <- as.matrix(users)
userprofiles <- read.csv("dataset5-userprofiles.csv")
View(userprofiles)

## Quem possui mais seguidores
mostfollowed <- userprofiles[with(userprofiles, order(-followed_by)), ]
head(mostfollowed$full_name, 15)

## Quem segue mais?
mostfollows <- userprofiles[with(userprofiles, order(-follows)), ]
head(mostfollows$full_name, 15)

# Quem publica mais?
mostmedias <- userprofiles[with(userprofiles, order(-media_count)), ]
head(mostmedias$full_name, 15)

# Top users
userprofiles$overallmetric <- ((userprofiles$media_count/max(userprofiles$media_count)) +
(userprofiles$followed_by/max(userprofiles$followed_by)) +(userprofiles$followed_by/max(userprofiles$followed_by)))*100
overallmet <- userprofiles[with(userprofiles, order(-overallmetric)), ]
```

```
head(overallmet$full_name, 15)
View(overallmet)

# Mais comentados
head(alldata)
mostcomm <- alldata[with(alldata, order(-comments_count)), ]
View(mostcomm)

# Com mais likes
head(alldata)
mostlikes <- alldata[with(alldata, order(-likes_count)), ]
View(mostlikes)

## Todas as localidades
library(sqldf)
names(alldata)
allloc <- sqldf("select distinct location_name from alldata")
allloc <- na.omit(allloc)
head(allloc, 20)
nrow(allloc)

# localidades com mais likes
loclikes <- sqldf("select location_name, sum(likes_count) as totlikes from alldata group by location_name")
loc <- loclikes[with(loclikes, order(-totlikes)), ]
loc <- na.omit(loc)
head(loc, 25)

# Localidade mais falada
loccomments <- sqldf("select location_name, sum(comments_count) as totcomm from alldata group by location_name")
loccomm <- loccomments[with(loccomments, order(-totcomm)), ]
loccomm <- na.omit(loccomm)
head(loccomm, 15)

# Localidade que ocorre com mais frequência
locations <- sqldf("select location_name, count(location_id) as locid from alldata group by location_name")
location <- locations[with(locations, order(-locid)), ]
location <- na.omit(location)
head(location,5)

# Com mais likes, por usuário
names(alldata)
userlikes <- sqldf("select username, sum(likes_count) as totlikes from alldata group by username")
user <- userlikes[with(userlikes, order(-totlikes)), ]
user <- na.omit(user)
head(user, 25)

# Wordcloud
library(wordcloud)
library(tm)

# Limpeza nos dados
options(warn=-1)
words <- strsplit(as.character(alldata$caption), " ")
words <- lapply(words, function(x) x[grep("^[A-Za-z0-9]+$", x)])
words <- unlist(words)
words <- tolower(words)
words <- words[-grep("^[rm]t$", words)]

# Removendo stop words
stopWords <- stopwords("en")
"%!in%" <- function(x,table) match(x,table, nomatch = 0) == 0
words <- words[words %!in% stopWords]

# Criando um objeto com a wordcloud
allwords <- as.data.frame(table(words))
wc <- wordcloud(allwords$words, allwords$Freq,
                random.order = FALSE,
                min.freq = 5,
                colors = brewer.pal(2, "Dark2"))

# Gravando a wordcloud
dev.copy(png,filename = "wordcloud.png", width = 600, height = 875);
dev.off ();


## Clustering
??fpc
library(fpc)
```

```
head(alldata)
data <- alldata

# Extraindo algumas colunas
cdata <- data.frame(data$type, data$comments_count, data$likes_count, data$user_id)

# Nomeando as colunas
colnames(cdata) <- c("type","comments","likes","user_id")

# Convertendo para integer (alguns algoritmos de clusters somente aceitam números como entrada)
cdata$user_id <- as.integer(cdata$user_id)
cdata$type <- as.integer(cdata$type)

# Visualizando os dados
View(data)
View(cdata)

# Estimando o número de clusters (automático)
?pamk
clusters <- pamk(cdata)
n <- clusters$nc
n

# Estimando o número de clusters (manual)
# Calculando a soma dos erros quadrados
wss <- (nrow(cdata)-1)*sum(apply(cdata,2,var))

# Buscando a soma dos erros quadrados dentro dos grupos
for (i in 2:25) wss[i] <- sum(kmeans(cdata, centers=i)$withinss)

# Plot dos clusters
# Logicamente, à medida que o número de clusters aumenta, a soma dos erros quadrados reduz.
# Se houver n objetos em um conjunto de dados, então n clusters resultaria em erro 0, mas idealmente
# precisamos parar em algum ponto. De acordo com as teorias, a taxa de diminuição na soma dos erros irá
# cair de repente em um ponto e que deve ser considerado como o número ideal de clusters.
# De acordo com o gráfico a seguir, o número ideal de cluster é 4.
plot(1:25, wss, type = "b", xlab = "Número de clusters", ylab = "Soma dos Quadrados Dentro dos grupos")

# Salvando o plot
dev.copy(png,filename = "clustering.png", width = 600, height = 875)
dev.off ();

## K-Means

# K-Means Cluster Analysis
?kmeans
fit <- kmeans(cdata, 4)
fit$cluster

# Número de elementos em cada cluster
table(fit$cluster)

# Médias dos clusters
aggregate(cdata, by = list(fit$cluster), FUN = mean)

# Observações em cada cluster
resultado <- cbind(cdata, clusterNum = fit$cluster)
View(resultado)

# Plot do cluster
library(fpc)
plotcluster(cdata, fit$cluster)
dev.copy(png, filename = "clusterPlot.png", width = 600, height = 875);
dev.off ();



## Sistema de Recomendação Baseado em Usuário
## Projeto 5 - Construindo um Sistema de Recomendação
library(data.table)

# Carregando o dataset com seguidores
userfollows <- read.csv("dataset4-userfollows.csv")
head(userfollows)
names(userfollows)
```

```
# Temos a variável precedente no conjunto de dados. Para construir o mecanismo de recomendação e fornecer
# recomendações aos usuários, precisamos apenas de duas colunas. Portanto, selecionamos essas duas colunas
# usando a função data.frame:
fdata <- data.frame(userfollows$users.i..1., userfollows$username)
colnames(fdata) <- c("user","follows")
head(fdata)

# Pivot dos dados
# Agora, temos de manipular o conjunto de dados de tal forma que os usuários se tornem a coluna e
# os usuários que seguem, se tornem as linhas. Assim, torna-se fácil calcular a correlação entre os usuários.
# Para girar os dados, precisamos usar a função dcast.data.table, que requer o pacote data.table.
pivoting <- data.table(fdata)
pivotdata <- dcast.data.table(pivoting, follows ~ user, fun.aggregate = length, value.var = "user")
write.csv(pivotdata, "dataset5-pivot-follows-temp.csv")

# Após deletar a coluna
 de índice e o usuário nulo

# Lendo os dados
data <- read.csv("dataset5-pivot-follows.csv")
head(data)
colnames(data)

# Removendo a coluna user
data.ubs <- (data[,!(names(data) %in% c("users"))])

# Função que calcula a distância entre 2 vetores
# Podemos calcular a similaridade dos usuários usando diferentes métodos.
# Em nosso caso, usaremos a técnica de similaridade de coseno para obter a pontuação de similaridade
# para todos os pares de usuários. Em nosso conjunto de dados, zero significa que o usuário não está seguindo.
# Se considerarmos essas linhas com zero, enquanto calculamos a similaridade usando correlação ou qualquer
# outra técnica, vamos acabar com uma saída tendenciosa que está longe da realidade.
# Assim, ao calcular a pontuação de similaridade, consideraremos somente as linhas não nulas.
# A seguinte função calcula a similaridade entre os usuários usando o método de similaridade de coseno.
getCosine <- function(x,y)
{
  dat <- cbind(x,y)
  f <- as.data.frame(dat)
  # Remove as linhas com zeros
  datn<- f[-which(rowSums(f==0)>0),]
  if(nrow(datn) > 2)
  {
    this.cosine <- sum(x*y) / (sqrt(sum(x*x)) * sqrt(sum(y*y)))
  }
  else
  {
    this.cosine <- 0
  }
  return(this.cosine)
}

# Agora, precisamos construir uma matriz de similaridade que nos dirá como os usuários são
# semelhantes entre si. Antes de computar a similaridade, vamos construir uma matriz vazia que pode
# ser usada para armazenar a similaridade:
data.ubs.similarity <- matrix(NA,
                              nrow = ncol(data.ubs),
                              ncol = ncol(data.ubs),
                              dimnames = list(colnames(data.ubs),
                                              colnames(data.ubs)))

# Aplicando o cálculo de similaridade a todas as colunas
# Agora podemos começar a substituir as células vazias na matriz de similaridade com a pontuação de
# similaridade real. No caso da similaridade de coseno, o intervalo será de -1 a + 1.
# O seguinte loop ajudará a calcular a similaridade entre todos os usuários.
# Se não houver dados suficientes para calcular a similaridade de acordo com nossa função,
# ela retornará zero. A instrução print no seguinte loop nos ajudará a entender o progresso do loop.
# Dependendo do conjunto de dados, o tempo necessário varia.
for(i in 1:ncol(data.ubs)) {
  # Loop em todas as colunas
  for(j in 1:ncol(data.ubs)) {
    # Calcula a similaridade e alimenta o dataframe
    data.ubs.similarity[i,j] <- getCosine(as.matrix(data.ubs[i]),as.matrix(data.ubs[j]))
  }
  print(i)
}

# Converte a matriz de similaridade em dataframe
data.ubs.similarity <- as.data.frame(data.ubs.similarity)
```

```
# Replace NA com 0
data.ubs.similarity[is.na(data.ubs.similarity)] <- 0
head(data.ubs.similarity)

# Obtendo os 10 vizinhos de cada usuário
data.neighbours <- matrix(NA,
                          nrow = ncol(data.ubs.similarity),
                          ncol = 11,
                          dimnames = list(colnames(data.ubs.similarity)))

# Gerando as recomendações
for(i in 1:ncol(data.ubs))
{
  # Evitando valores zero
  n <- length(data.ubs.similarity[,i])
  thres <- sort(data.ubs.similarity[,i],partial=n-10)[n-10]
  if(thres > 0.020)
  {
    # Selecionando as recomendações Top 10
    data.neighbours[i,] <- (t(head(n=11,rownames(data.ubs.similarity[order(data.ubs.similarity[,i],decreasing=TRUE),][i]))))
  }
  else
  {
    data.neighbours[i,] <- ""
  }
}

# Visualizando as recomendações
# No código anterior, pegamos um usuário de cada vez e, em seguida, classificamos a pontuação de similaridade
# do usuário com todos os outros usuários, de modo que o par com a maior similaridade venha primeiro.
# Então, paramos apenas filtrando os 10 primeiros para cada um dos usuários. Isso é recomendado para nós.
# Podemos ver as recomendações dadas para os usuários::
View(data.neighbours)

# Gravando as recomendações
write.csv(data.neighbours, "Recomendacoes.csv")
```

Quizz

Se você for como a grande maioria dos brasileiros e passar a maior parte do seu dia on-line, seja pelo computador ou pelo seu celular, podemos dizer que as mídias sociais são de suma importância, não só na sua vida, como na vida de milhares de produtos e marcas que almejam ficar mais perto de você.

O que faz das mídias sociais um lugar tão interessante para a marca ou produto de uma empresa é que o público está lá.

Devido ao grande tamanho dos dados de mídias sociais, não é possível obter um conjunto de dados adequadamente rotulado para treinar um algoritmo supervisionado de aprendizado de máquina. Sem os dados apropriados, não há maneira de julgar a precisão de qualquer algoritmo de classificação.

O Social Data Mining ou mineração de mídia social é uma análise sistemática de informações geradas a partir de mídias sociais.

O Social Data Mining pode ajudar a identificar os potenciais clientes com base nas suas atividades online e com base nas atividades online dos seus amigos.