Guia Completo para Definição de Estatística de Modelos e Algoritmos de Machine Learning utilizados em Projetos de Data Science para Negócios (com códigos em R e Python)

Março de 2017

Prof. Geanderson Lenz (@geanderson)

Data Scientist
Doutorando em Gestão Aplicada dos Dados pela Universidade de Coimbra
Mestre em Estratégia e Métodos Quantitativos (PUCRS)
Bacharel em Administração (UniRitter Fapa)

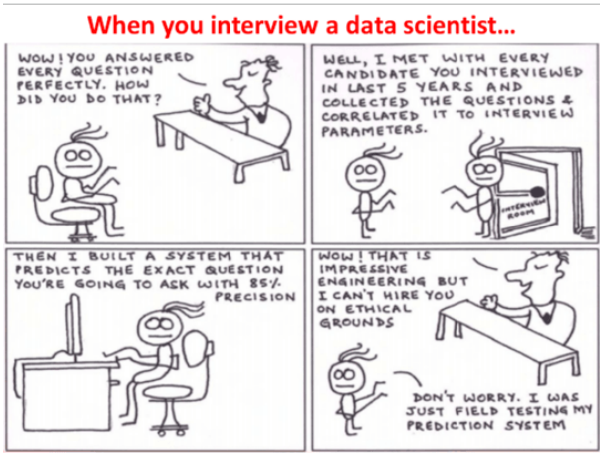
@geanderson

github.com/geandersonlenz

linkedin.com/in/geandersonlenz/

INTRODUÇÃO

Estamos vivendo a era das empresas direcionadas por dados. Os próximos anos mudaram completamente a forma que nos relacionamos com a tecnologia. Algoritmos são direcionadores de estratégia. Como um cientista de dados, considero um apogeu vivenciarmos a profusão de ferramentas e frameworks open source destinados ao uso do data science. Democracia de uso é uma premissa básica para que qualquer organização possa beneficiar-se das tecnologias como meio. Nos negócios, estamos vivendo tempos onde agilidade supera o capital, onde a flexibilidade supera a estrutura rígida, onde a execução supera a estratégia. Para difundir o uso do data science para negócios no Brasil, escrevo esse guia com a intenção de ser um catalisador, uma ponte que viabiliza a criação de uma rede ampla, sem centralização e com possibilidades infinitas de conexão. Aproveitem!



Sumário

APRENDIZAGEM SUPERVISIONADA

São algoritmos que fazem previsão baseados em exemplos, eles olham para padrões em seus rótulos de valores (labels). Tabelas com colunas e seus nomes são um exemplo comum de dados supervisionados.

> *“It can use any information that might be relevant—the day of the week, the season, the company's financial data, the type of industry, the presence of disruptive geopolicitical events—and each algorithm looks for different types of patterns. After the algorithm has found the best pattern it can, it uses that pattern to make predictions for unlabeled testing data.”* (Microsoft)

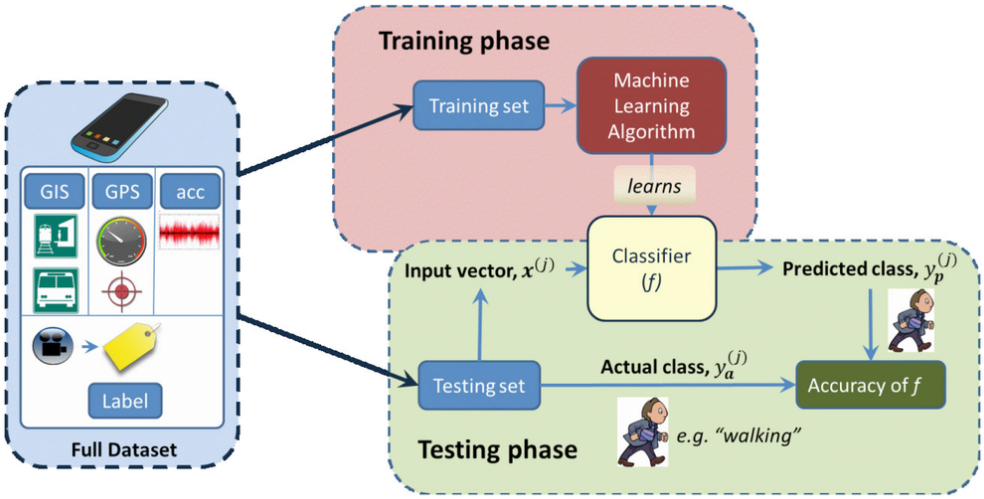
Estes algoritmos são baseados em variáveis dependentes (outcomes) nos quais são feitas predições a partir de variáveis independentes (predictors). Usando essas variaveis, geramos que uma função que mapeia os resultados desejados. O treinamento desse processo continua até o modelo atingir o nível de acuracidade desejada nos dados que foram treinados.

Exemplos:
- Regressão Linear
- Árvores de Decisão
- Random Forest
- K-Nearest Neighbors,
- Regressão Logística.

Classificação: quando os dados são utilizados para predizer uma categoria, o aprendizado supervisionado também é chamado de classificação. O reconhecimento de imagem como uma foto de um “cachorro” ou “gato”. Para classificar um e-mail como “spam” ou não, quando temos duas opções, é chamado de algoritmo de duas classes ou binominal.

Regressão: quando um valor está sendo predito com utilização de variável dependente e variáveis independentes, criando uma função.

Detecção de Anomalias: o objetivo é identificar pontos dos dados que são não usuais. Em detecção de fraudes, por exemplo, gastos não usuais de cartão de crédito fora de um padrão são suspeitos. A abordagem passa pela aprendizagem das atividades normais e identificar qualquer significante diferença.



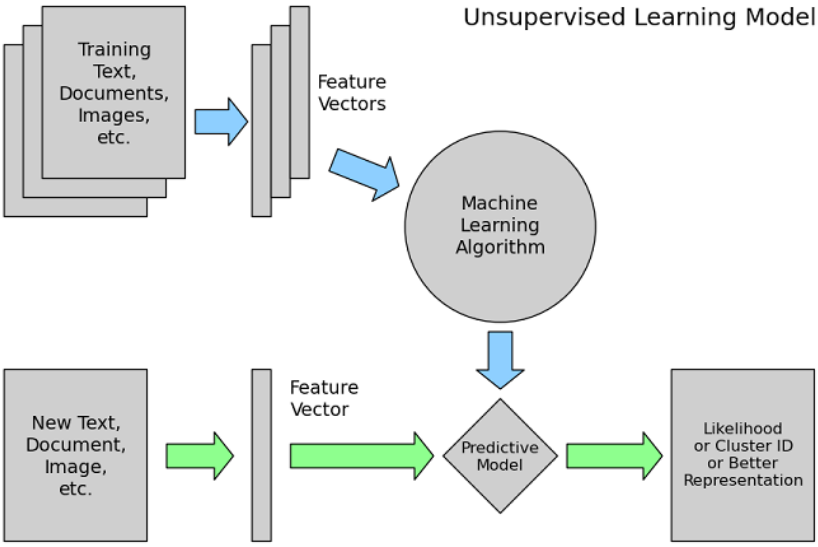
Aprendizagem Supervisionada

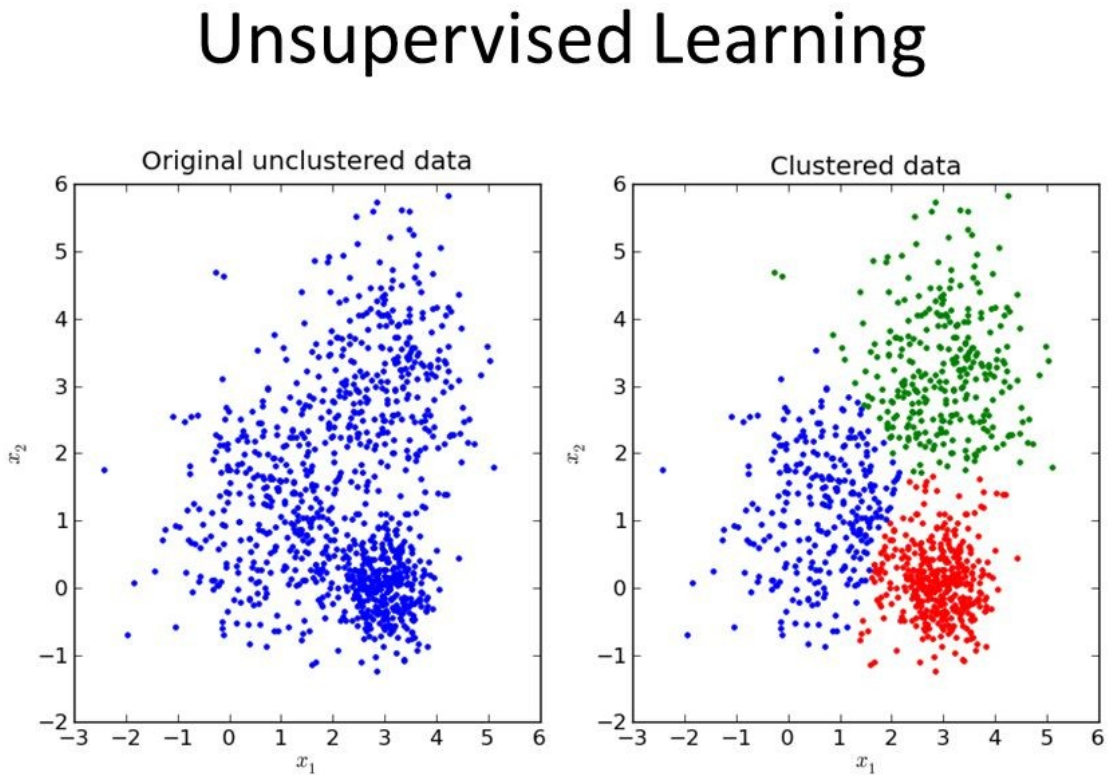
APRENDIZAGEM NÃO SUPERVISIONADA

Em aprendizagem não supervisionada, os dados não possuem rótulos associados a eles. Assim, não temos nenhum alvo ou variável de saída para predizer ou estimar. O bojetivo é organizar os dados de alguma forma ou descrever a estrutura. É usado para clustering (agrupar) clientes em diferentes grupos de intervenção específica. Isso significa agrupar em clusters ou encontrar diferentes jeitos de olhar dados complexo que simplesmente aparecem ou são organizados.

Exemplos:

1. Gaussian mixture models
2. Manifold learning
3. Clustering
4. Biclustering
5. Decomposing signals in components (matrix factorization problems)
6. Covariance estimation
7. Novelty and Outlier Detection
8. Density Estimation
9. Neural network models (unsupervised)
10. K-Means





REINFORCEMENT LEARNING

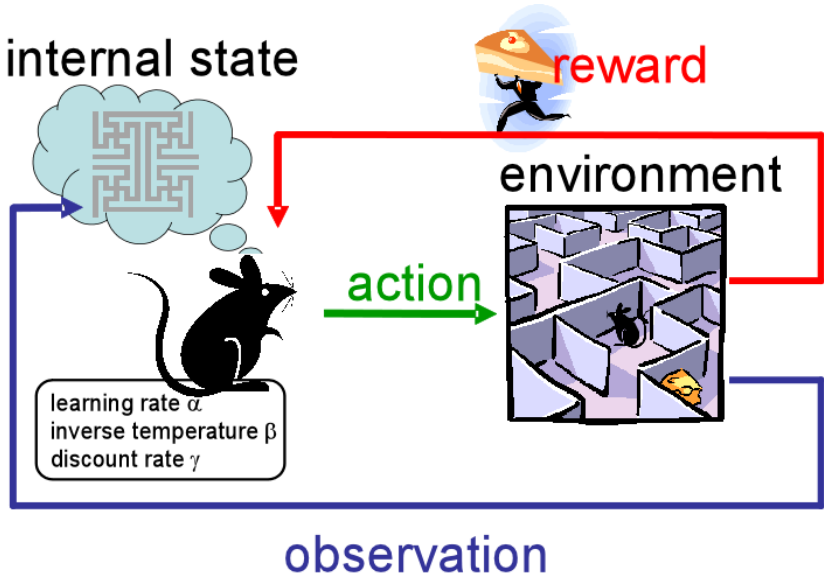
Em RL, o algoritmo escolhe uma ação em resposta a cada ponto de dado. A máquina é exposta a um ambiente onde ela fica treinando continuamente usando tentativa e erro. O algoritmo de aprendizado recebe um sinal de recompensa em um curto período após começar o treinamento, indicando o quanto sua decisão foi positiva. A máquina aprende a partir de experiencias do passado e tenta capturar o melhor conhecimento possível para fazer decisões de negócios com alto grau de acuracidade.

Uma aplicação comum de RL é na robotica, onde o conjunto de sensores fazem a leitura de cada ponto por vez nos dados, e o algoritmo deve

escolher a próxima ação do robo. Essa utilização é muito aplicada a Internet das Coisas (IoT)

Exemplo:

Markov Decision Process



Reinforcement Learning Process

DRIVERS DE ESCOLHA DOS MODELOS DE MACHINE LEARNING

1. Acuracidade

Conseguir a maior acuracidade possível do modelo nem sempre é necessário. Algumas vezes uma aproximação já é adequada, dependendo do que se quer usar o projeto. Se for o caso de trabalhar com aproximação, o tempo de processamento cairá drasticamente, utilizando os métodos de aproximação. Outra vantagem dos métodos de aproximação é que eles naturalmente tendem a evitar o chamado “overfitting”.

> *Sobre-ajuste (do inglês: overfitting) é o termo, em aprendizagem de máquina, estatística e afins, para quando o modelo estatístico se ajusta em demasiado ao conjunto de dados/amostra. É comum que a amostra apresente desvios causados por erros de medição ou fatores aleatórios, ocorre o sobre-ajuste quando o modelo se ajusta a estes. Um modelo sobre-ajustado apresenta alta precisão quando testado com seu conjunto de dados porém tal modelo não é uma boa representação da realidade e por isso deve ser evitado. (Wikipedia, revisado)*

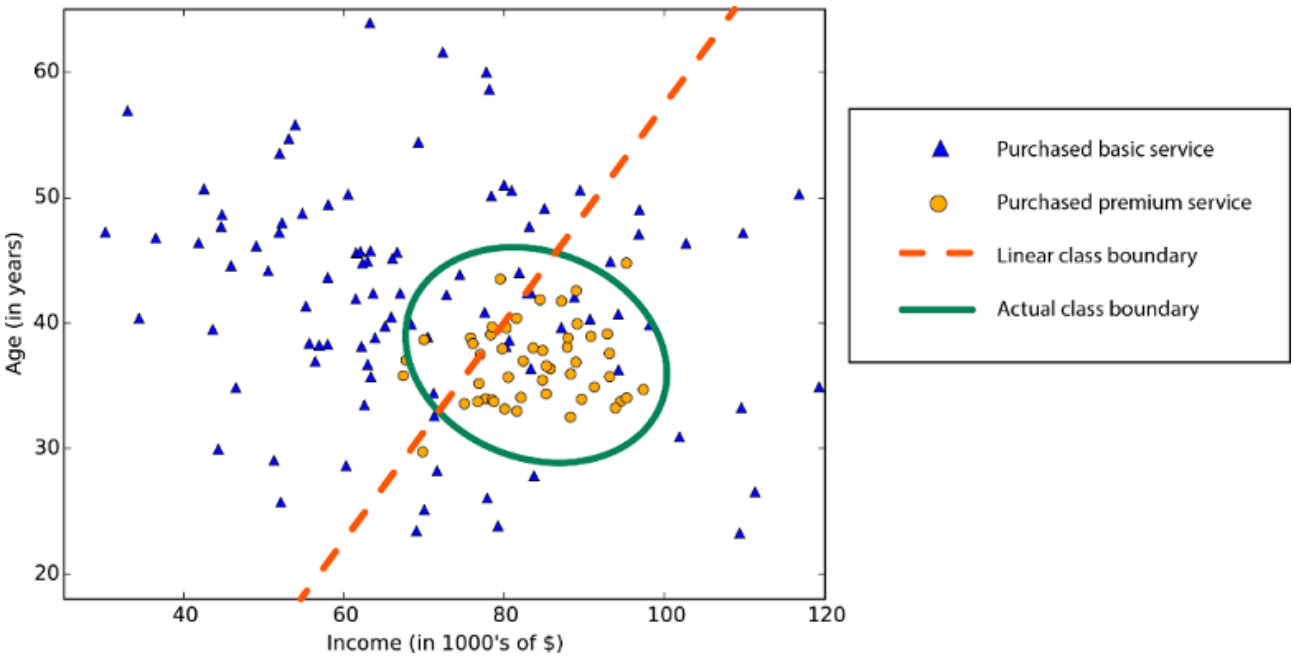
A acuracidade por si só pode ser um engano sobre a real utilidade do modelo, que mesmo tendo uma baixa acuracidade, pode ter um grande poder preditivo, ou vice versa. Isso nos leva ao paradoxo da acuracidade.

2. Training time

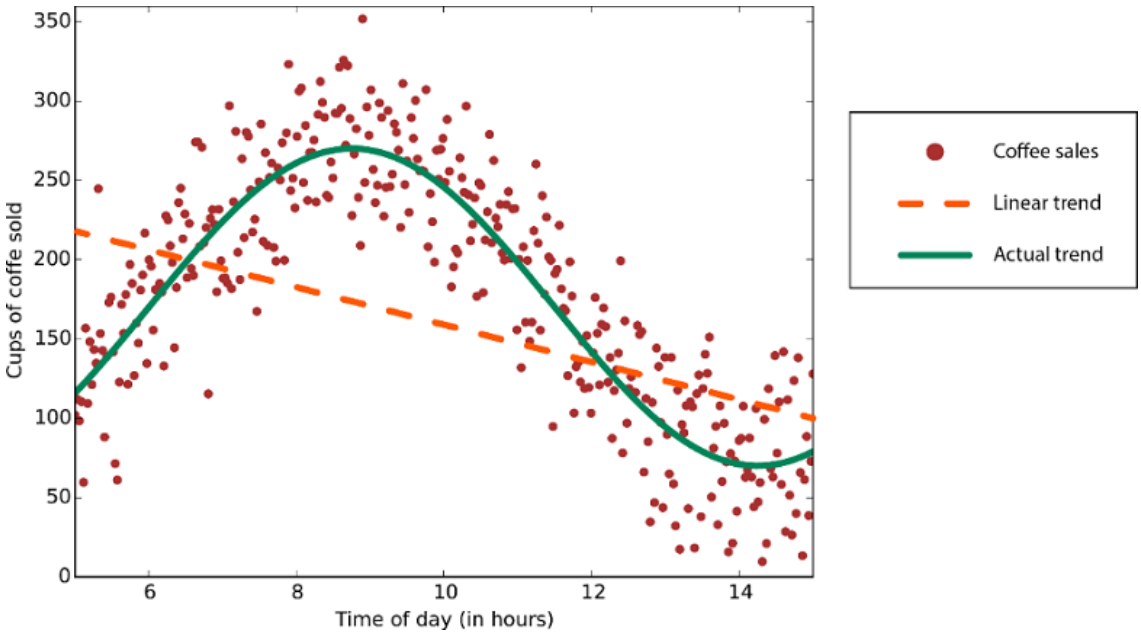
O número de minutos ou horas necessárias para treinar um modelo possui uma grande variação entre os algoritmos. O tempo de treinamento é acompanhado de perto pela acuracidade, um acompanha o outro. Adicionalmente, alguns algoritmos são mais sensíveis ao número de pontos de dados. Quando o tempo é limitado isso pode direcionar a escolha do algoritmo, em grandes data sets.

3. Linearidade

Algoritmos com Classificação linear assumem que as classes podem ser separadas por uma linha reta (or its higher-dimensional analog). Inúmeros algoritmos de machine learning fazem uso da linearidade. Esses incluem regressão logística e support vector machines (SVM). Regressão linear assume que a tendencia dos dados é seguir uma linha reta. Essas premissas não são ruins para alguns problemas, porém, para outros podem derrubar a acuracidade do modelo.



Exemplo de Linearidade entre duas variáveis. Fonte: Microsoft



Exemplo de Série com Não Linearidade. Impacto na Acuracidade. Fonte: Microsoft

4. Número de Parâmetros

Os parâmetros são as maçanetas para o cientista de dados começar a ligar quando configura um algoritmo. Eles são números que afetam o comportamento dos algoritmos, como erro de tolerância ou número de interações, ou opções entre variantes de como os algoritmos se comportam. O tempo de treinamento e acuracidade do algoritmo pode algumas vezes tornar sensíveis para as configurações estarem corretas. Algoritmos com grande número de parâmetros requerem mais tentativa e erro para encontrar a combinação adequada.

Algoritmos com inúmeros parâmetros indicam grande flexibilidade, isso pode indicar o encontro de acuracidade pertinente. É preciso encontrar a correta combinação entre a configuração de parâmetros.

5. Número de Características

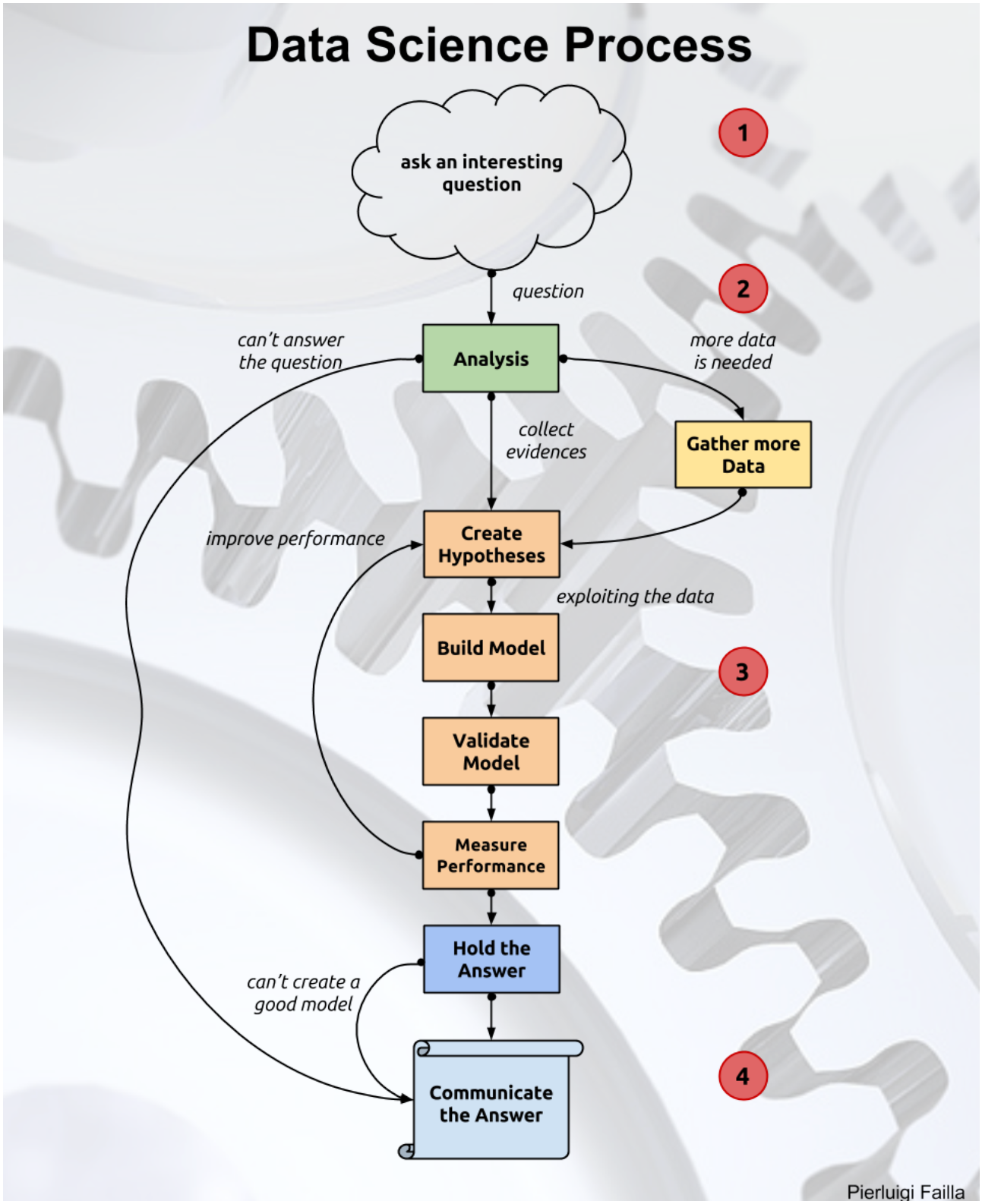
Para certos tipos de dados, o número de características pode ser largamente comparada com o número de pontos de dados. Esse é o caso de genéticos ou textos.

O grande número de características pode tornar o aprendizado do algoritmos reduzido, realizando o tempo de treinamento muito longo. Support Vector Machines são particularmente enquadrados nesse caso.

6. Cross Validation

É uma técnica para estimar a performance de um modelo preditivo. Também chamado de rotação de estimação, é uma forma de acessar como os resultados da análise estatística generalizam para o data set independente. É utilizado para configurar onde o objetivo é a predição, e um quer estimar quão acurado é um modelo preditivo irá se comportar na prática. Um round de cross-validation envolve particionar uma amostra de dados em subconjuntos complementares, training set, e validar a análise em outro subconjunto. Para reduzir a variabilidade, múltiplos rounds de cross-validation são performados usando diferentes partições, e a validação dos resultados são as médias dos rounds executados.

WORKFLOW DO PROJETO DE DATA SCIENCE PARA NEGÓCIOS



Faça perguntas interessantes...isso quer dizer que todo processo de data science precisa estar guiado por questões que ajudem os negócios nos seguintes objetivos;

a) como esses dados podem aumentar nossa receita?

b) como esses dados podem aumentar nossa participação de mercado?

c) como esses dados podem aumentar nossa produtividade?

d) como esses dados podem diminuir nossos custos?

e) como esses dados podem maximizar nossos investimentos?

f) como esses dados podem nos orientar sobre estratégias para pessoas?

g) como esses dados podem evitar dispêndios na cadeia de suprimentos?

h) Como esses dados podem melhorar nossa relação com os clientes e oferecer produtos e serviços de excelência?

Possuindo as perguntas corretas, vamos para a parte onde as hipóteses são declaradas para verificarmos a sua probabilidade de ocorrência e os efeitos que as mesmas teriam sobre nossa base de dados.

		Situação real	
		H_0 é verdadeira	H_0 é falsa
Nossa Decisão	Rejeitar H_0	***Erro Tipo I*** (Rejeitar H_0, quando H_0 é verdadeira)	Decisão correta
	Não Rejeitar H_0	Decisão correta	***Erro Tipo II*** (Não Rejeitar H_0, quando H_0 é falsa)

Fonte: Testes de Hipóteses. UFSCAR.

Um sumário dos principais passos que podem ser usados sistematicamente para qualquer teste de hipóteses é apresentado aqui, ou seja:

(a) Fixe a hipótese H0 a ser testada e a alternativa H1;

(b) Use a teoria estatística e as informações disponíveis para decidir qual estatística (estimador) será usada para testar a hipótese H0, obtendo-se suas propriedades (distribuição, estimativa, erro padrão);

(c) Fixe a probabilidade de cometer o erro tipo I e use este valor para construir a RC (região crítica). Lembre-se que a RC é construída para a estatística definida no passo (a), usando os valores hipotetizados por H0;

(d) Use as informações da amostra para calcular o valor da estatística do teste; e
(e) Se o valor da estatística calculado com os dados da amostra não pertencer à RC, não rejeite H0; caso contrário, rejeite H0.

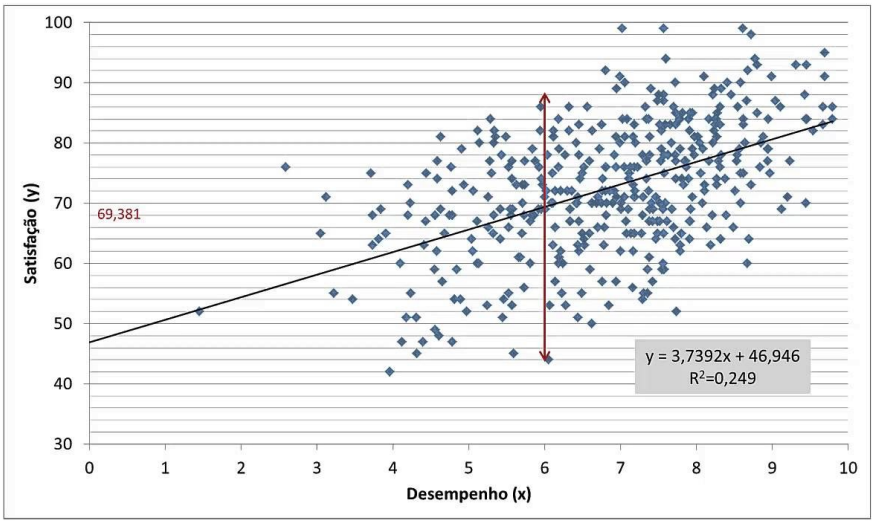
Principais Testes de Hipóteses:
Teste T
Teste Z
Distribuição t de Student
Normalização
Valor-p
Análise de variância

O próximo passo é a construção do modelo (Build Model), fazendo o treinamento de um subconjunto dos dados para verificar os padrões que serão encontrados. Nessa etapa, será destinado uma parte dos dados para criar o modelo, que geralmente é uma função que determina a predição dos dados ou sua classificação. Após a conclusão do processo, o treinamento dos dados irá ocorrer novamente para melhorar sua acuracidade e precisão nos objetivos que ele pretende atingir.
Verificando que o modelo está adequado a proposta, ocorre a mensuração da performance, utilizando os indicadores que verificamos anteriormente na seção de direcionadores, junto com precisão, densidade e recall.
O processo finaliza com a Comunicação dos Dados aplicados ao modelo, onde se realizam as análises e aplica-se o StoryTelling do projeto de Data Science. Além das competências de hacking, estatística, business, comunicar corretamente o que se está inferindo é uma das habilidades do cientista de dados para negócios. Explicar corretamente qual o impacto dos modelos nos negócios. O que a empresa vai ganhar com isso? Precisa ser respondido.

ALGORITMOS DE MACHINE LEARNING

Regressão Linear

É um dos algoritmos mais utilizados em negócios, pela sua versatilidade e amplitude. Ele estima valores reais, baseados em variáveis contínuas. Quanto maior for a correlação entre as variáveis, mais as mesmas se explicam. O relacionamento entre variáveis dependentes e independentes é dados pela melhor linha de ajuste entre as variáveis. Pode ser simples ou múltipla. É representada pela equação linear Y= a *X + b. Alguns exemplos de aplicação: preço de venda, cálculo de frete, tributos, folha de pagamento, taxas de juros.



Python

```
# Python Code - Import Library
from sklearn import linear_model
#Load Train and Test datasets
#Identify feature and response variable(s) and values must be numeric and numpy arrays
x_train=input_variables_values_training_datasets
y_train=target_variables_values_training_datasets
x_test=input_variables_values_test_datasets
# Create linear regression object
linear = linear_model.LinearRegression()
# Train the model using the training sets and check score
linear.fit(x_train, y_train)
linear.score(x_train, y_train)
#Equation coefficient and Intercept
print('Coefficient: \n', linear.coef_)
print('Intercept: \n', linear.intercept_)
#Predict Output
```

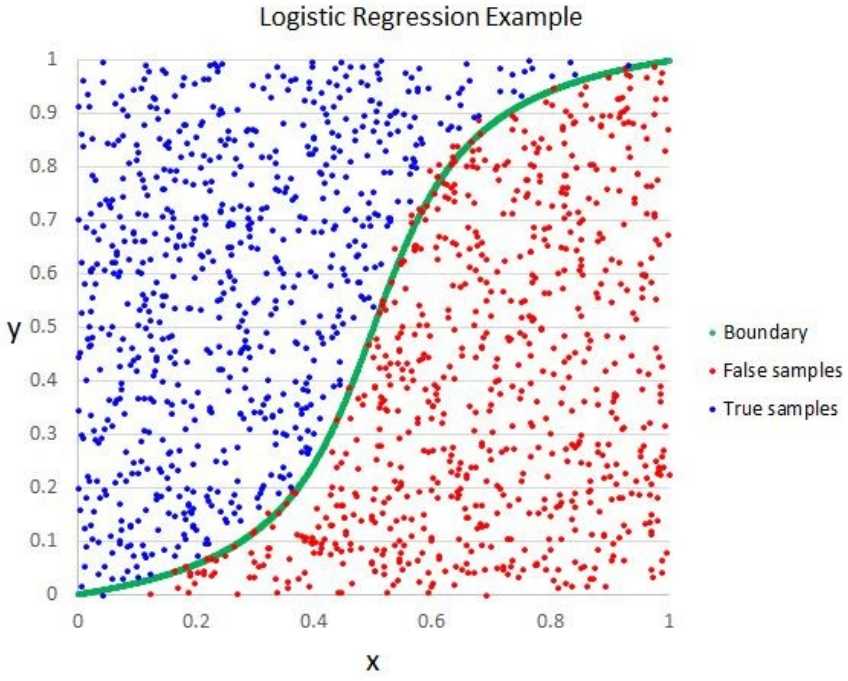
```
predicted= linear.predict(x_test)
```

R

```
#Load Train and Test datasets
#Identify feature and response variable(s) and values must be numeric
x_train <- input_variables_values_training_datasets
y_train <- target_variables_values_training_datasets
x_test <- input_variables_values_test_datasets
x <- cbind(x_train,y_train)
# Train the model using the training sets and check score
linear <- lm(y_train ~ ., data = x)
summary(linear)
#Predict Output
predicted= predict(linear,x_test)
```

Regressão Logística

Apesar do nome regressão, o algoritmo logístico é uma ferramenta para classificação de duas classes ou multiclasses, onde a curva de modelagem é S, ao invés da linha reta utilizada na regressão linear. É rapida e simples. O fato de utilizar a a curva S faz um enquadramento natural para divisão dos dados em grupos. É utilizada para setimar valores discretos, como binários (0 ou 1), sim ou não, verdadeiro ou falso, baseado em um dados conjunto de variáveis independentes. Ela prediz a probabilidade de ocorrência de um evento ajustado na função logística.



Python

```
#Import Library
from sklearn.linear_model import LogisticRegression
#Assumed you have, X (predictor) and Y (target) for training data set and x_test(predictor) of test_dataset
# Create logistic regression object
model = LogisticRegression()
# Train the model using the training sets and check score
model.fit(X, y)
model.score(X, y)
#Equation coefficient and Intercept
print('Coefficient: \n', model.coef_)
print('Intercept: \n', model.intercept_)
#Predict Output
predicted= model.predict(x_test)
```

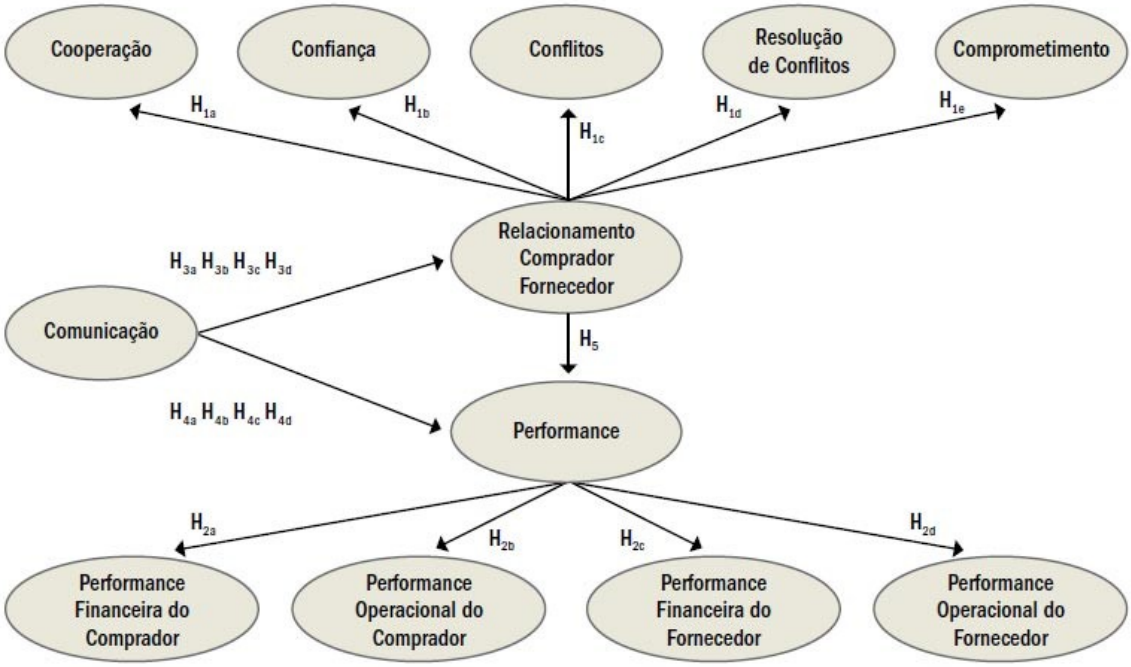
R

```
x <- cbind(x_train,y_train)
# Train the model using the training sets and check score
logistic <- glm(y_train ~ ., data = x,family='binomial')
summary(logistic)
#Predict Output
predicted= predict(logistic,x_test)
```

Equações Estruturais

Conhecida em inglês como Structural Equation Modelling (SEM) é um modelo de regressão multi equação, onde existe uma variável dependente e outras independentes, que por si, se tornam também variáveis dependentes, gerando outras equações.

Figura 1 – **Modelo conceitual da dimensão comunicação**

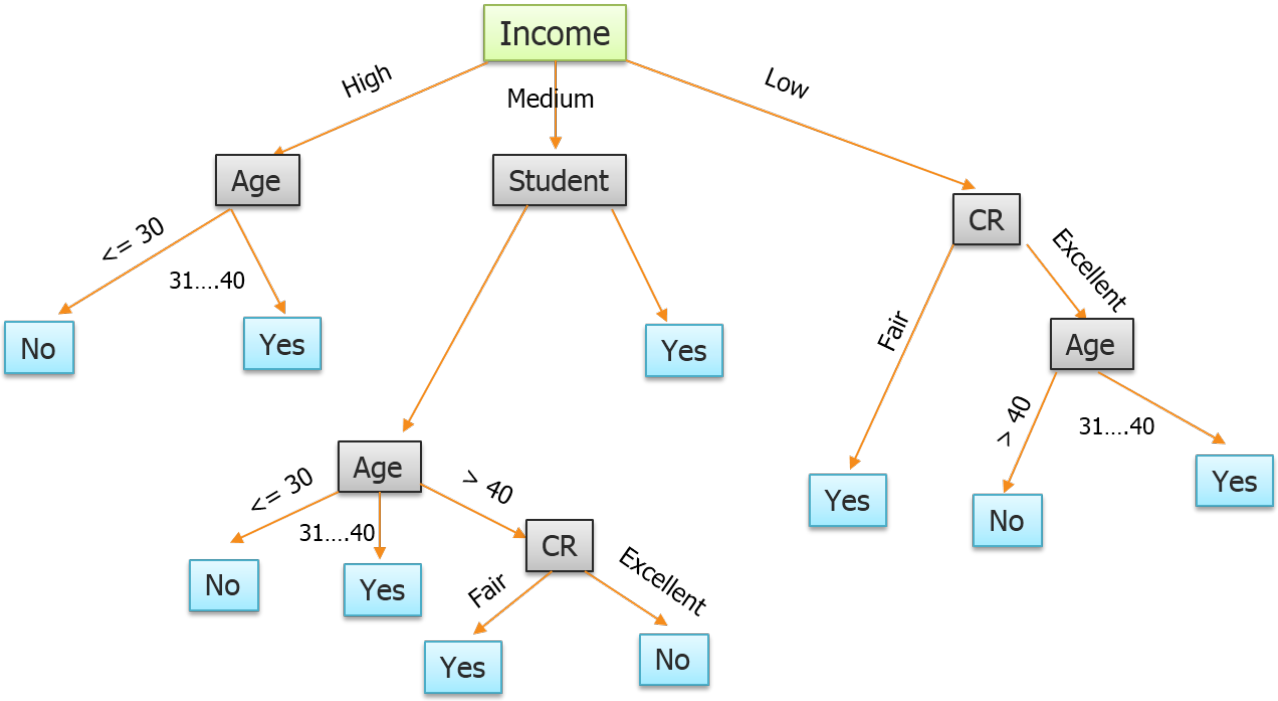


Diferente do modelo tradicional de regressão, na modelagem de equações estruturais, a variável de resposta de uma equação aparece como preditor em outra equação, e as variáveis influenciam umas as outras reciprocamente, diretamente, ou por meio das outras variáveis intermediárias. Uma de sua larga utilidade é no marketing, pois os relacionamentos entre variáveis é complexo e não linear. Em um projeto, podemos trabalhar com valor, marca, satisfação, desejo, emoções, lealdade, entre outros construtos, todos de forma integrada.

Arvore de Decisão (Decision Tree)

Árvores, florestas e selvas. Árvores de deicsão (regression, two-class, and multiclass), Selvas de decisão (two-class and multiclass), and árvores de decisão impulsionadas (regression and two-class) todas são baseadas em árvores de decisão, existem muitas variantes, porém todas baseadas na mesma coisa, subdividir as características em regiões com o mesmo rótulo.

É um tipo de algoritmo de aprendizagem supervisionada que é utilizado em grande parte dos poblemas de classificação. O algoritmo funciona com variáveis dependentes categóricas e contínuas, onde a população é colocada em duas ou mais conjuntos homogêneos. Isso é feito nos mais significantes atributos das variáveis indepedentes para fazer istinção entre os grupos. Sugestão de leitura: [Decision Tree Simplified]



Codigo Python

```
#Import Library
#Import other necessary libraries like pandas, numpy...
from sklearn import tree
#Assumed you have, X (predictor) and Y (target) for training data set and x_test(predictor) of test_dataset
# Create tree object
```

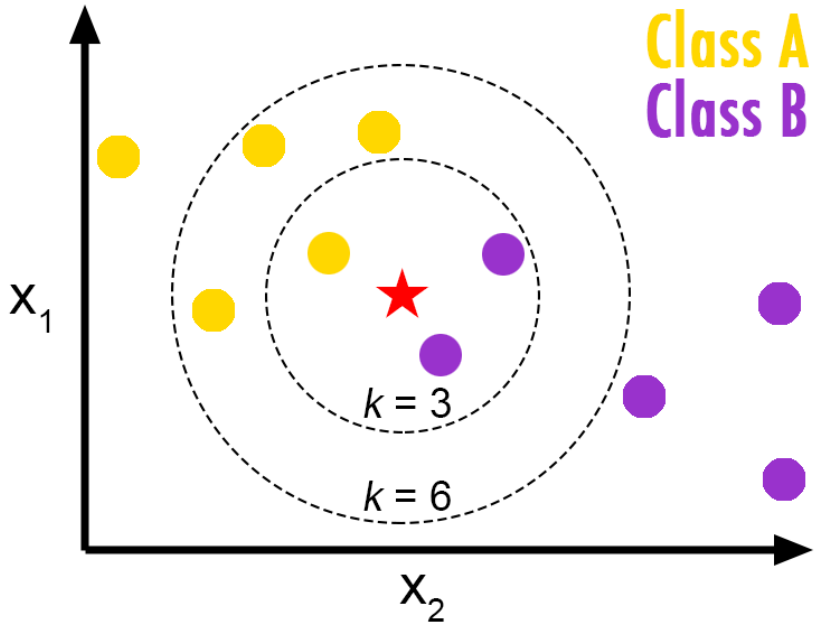
```
model = tree.DecisionTreeClassifier(criterion='gini') # for
classification, here you can change the algorithm as gini or
entropy (information gain) by default it is gini
# model = tree.DecisionTreeRegressor() for regression
# Train the model using the training sets and check score
model.fit(X, y)
model.score(X, y)
#Predict Output
predicted= model.predict(x_test)
```

Código R

```
library(rpart)
x <- cbind(x_train,y_train)
# grow tree
fit <- rpart(y_train ~ ., data = x,method="class")
summary(fit)
#Predict Output
predicted= predict(fit,x_test)
```

KNN (K- Nearest Neighbors)

Vizinhos próximos, esse algoritmo pode ser utilizado tanto para classificação quanto para regressão. Ele é largamente utilizado para problemas de classificação. Ele armazena todos os casos e classifica os novos pelos votos da maioria da vizinhança K. Para isso, é utilizada a função de distancia entre os pontos. A função de distancia pode ser Euclidiana, Manhattan, Minkowski and Hamming distance. As tres primeiras funções são usadas para variáveis contínuas e a quarta para variáveis categóricas. Se o K = 1, então o caso analisado é colocado na classe da sua vizinhança próxima. Essa escolha do K é um desafio enquanto monta-se a modelagem KNN.



Código Python

```
#Import Library
from sklearn.neighbors import KNeighborsClassifier
#Assumed you have, X (predictor) and Y (target) for training data set and x_test(predictor) of test_dataset
# Create KNeighbors classifier object model
KNeighborsClassifier(n_neighbors=6) # default value for n_neighbors is 5
# Train the model using the training sets and check score
model.fit(X, y)
#Predict Output
predicted= model.predict(x_test)
```

Código R

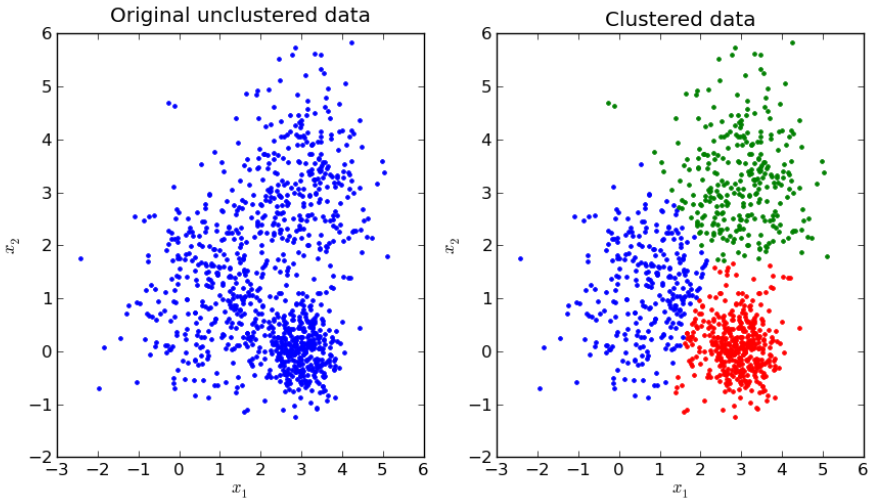
```
library(knn)
x <- cbind(x_train,y_train)
# Fitting model
fit <-knn(y_train ~ ., data = x,k=5)
summary(fit)
#Predict Output
predicted= predict(fit,x_test)
```

K-Means

É um algoritmo de aprendizagem não supervisionada no qual soluciona um problema de cluster. É um procedimento que segue um simples e fácil caminho para classificar um dado conjunto de dados por meio de um certo número de clusters. Os pontos dos dados dentro de um cluster são homogêneos e heterogêneos nos seus pares de grupos.

Como determinar o valor do K

> *“Em K-means, temos clusters e cada cluster tem seu próprio centroide. A soma do quadrado da diferença entre o centróide e os pontos de dados dentro de um conjunto constitui dentro da soma do valor quadrado para esse conjunto. Além disso, quando a soma de valores quadrados para todos os clusters são adicionados, torna-se total na soma do valor quadrado para a solução de cluster.*
>
> *Sabemos que à medida que o número de clusters aumenta, esse valor continua diminuindo, mas se você plotar o resultado, você pode ver que a soma da distância ao quadrado diminui acentuadamente até algum valor de k, e então muito mais lentamente depois disso. Aqui, podemos encontrar o número ótimo de cluster.” (SUNIL RAY, 2015)*



Código Python

```
#Import Library

from sklearn.cluster import KMeans

#Assumed you have, X (attributes) for training data set and
x_test(attributes) of test_dataset
```

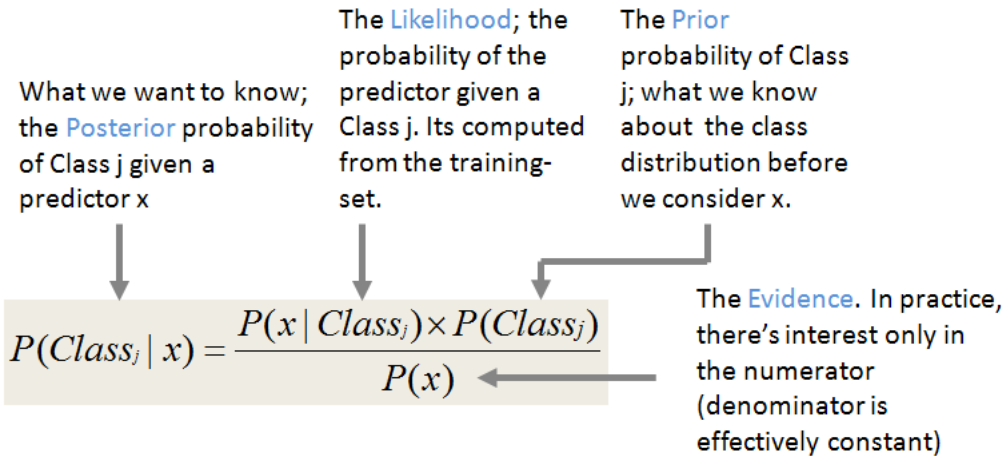
```
# Create KNeighbors classifier object model
k_means = KMeans(n_clusters=3, random_state=0)
# Train the model using the training sets and check score
model.fit(X)
#Predict Output
predicted= model.predict(x_test)
```

Código em R

```
library(cluster)
fit <- kmeans(X, 3) # 5 cluster solution
```

Naive Bayes

É uma técnica de classificação baseada no teorema bayesiano que assume a independencia entre os preditores. Em termos simples, a Naive Bayes assume a presença de uma caracteristica particular em uma classe não é relacionada com a presença de nenhuma outra característica. NB é fácil de construir e particularmente útil para largos data sets. Apesar da sua simplicidade, Naive Bayes tem uma grande performance considerada sofisticada dentro dos métodos de classificação.



Applying the independence assumption

$$P(x \mid Class_j) = P(x_1 \mid Class_j) \times P(x_2 \mid Class_j) \times \ldots \times P(x_k \mid Class_j)$$

Substituting the independence assumption, we derive the Posterior probability of Class j given a new instance x′ as...

$$P(Class_j \mid x') = P(x'_1 \mid Class_j) \times P(x'_2 \mid Class_j) \times \ldots \times P(x'_k \mid Class_j) \times P(Class_j)$$

Código Python

```
#Import Library
from sklearn.naive_bayes import GaussianNB

#Assumed you have, X (predictor) and Y (target) for training data set and x_test(predictor) of test_dataset

# Create SVM classification object model = GaussianNB() # there is other distribution for multinomial classes like Bernoulli Naive Bayes, Refer link

# Train the model using the training sets and check score

model.fit(X, y)

#Predict Output

predicted= model.predict(x_test)
```

Código em R

```
library(e1071)
x <- cbind(x_train,y_train)

# Fitting model

fit <-naiveBayes(y_train ~ ., data = x)

summary(fit)

#Predict Output

predicted= predict(fit,x_test)
```

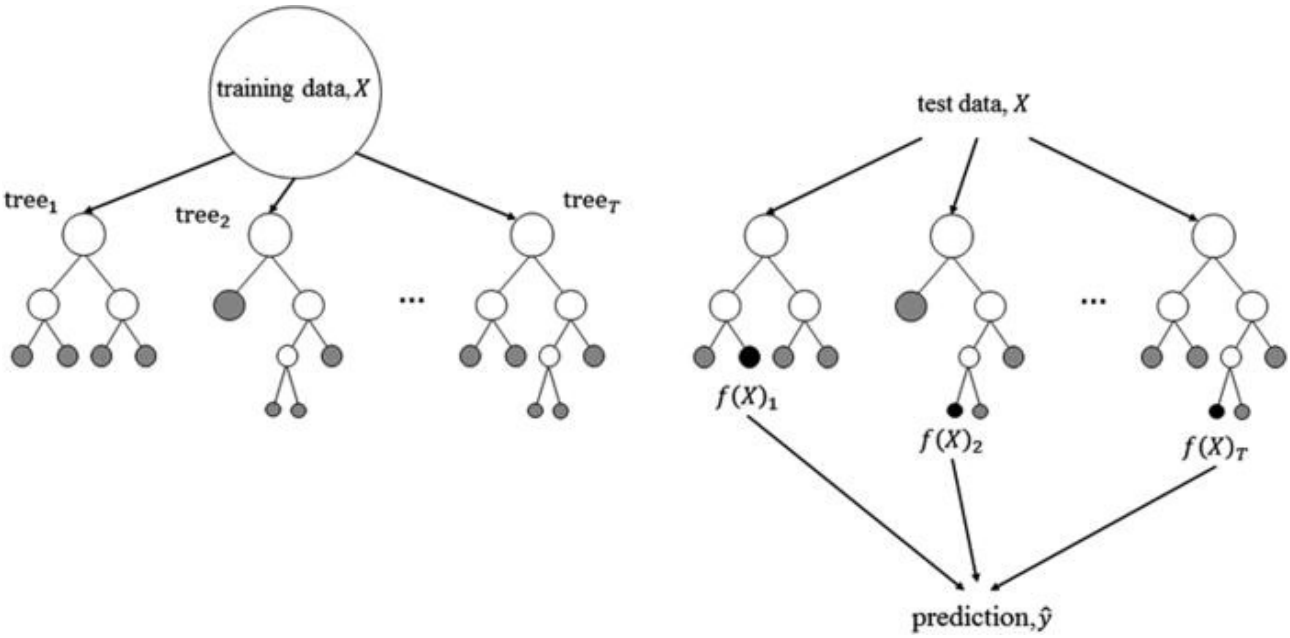
Random Forest

A floresta aleatória é um termo da marca registrada para um conjunto de árvores da decisão. Em RF, nos temos uma coleção de árvores de decisão, conhecidas como florestas. Para classificar um novo objeto baseado em atributos, cada árvore dá uma classificação e nós damos os votos que cada árvores recebe para cada classe. A floresta escolhe a classificação que possui mais votos.

"Cada árvore é plantada e cultivada da seguinte forma:

Se o número de casos no conjunto de treinamento for N, então a amostra de N casos é tomada ao acaso mas com substituição. Esta amostra será o conjunto de treinamento para o cultivo da árvore.

Se houver M variáveis de entrada, um número m << M é especificado de tal forma que em cada nó, m variáveis são selecionadas aleatoriamente para fora do M ea melhor divisão nestes m é usada para dividir o nó. O valor de m é mantido constante durante o crescimento da floresta. Cada árvore é cultivada na maior extensão possível. Não há poda."



Codigo em Python

```
#Import Library
from sklearn.ensemble import RandomForestClassifier
#Assumed you have, X (predictor) and Y (target) for training data set and x_test(predictor) of test_dataset
# Create Random Forest object
model= RandomForestClassifier()
# Train the model using the training sets and check score
model.fit(X, y)
#Predict Output
predicted= model.predict(x_test)
```

Código em R

```
library(randomForest)
x <- cbind(x_train,y_train)
# Fitting model
fit <- randomForest(Species ~ ., x,ntree=500)
summary(fit)
#Predict Output
predicted= predict(fit,x_test)
```

Guia de Escolha de Algoritmos

APLICAÇÃO	ALGORITMO SUGERIDO
Previsão de Vendas **Tendências de Custos**	Regressão Linear Simples Regressão Linear Múltipla
Segmentação de Mercado	K-NN Random Forest
Detecção de Anomalias	Naive Bayes K-Means
Cadeias de Suprimentos	Equações Estruturais Regressão Logística
Escore de Crédito	Supervised Vector Machine Gradient Boost Method

REFERENCIAS

How to choose algorithms for Microsoft Azure Machine Learning
https://docs.microsoft.com/en-us/azure/machine-learning/machine-learning-algorithm-choice

Essentials of Machine Learning Algorithms (with Python and R Codes)
https://www.analyticsvidhya.com/blog/2015/08/common-machine-learning-algorithms/

Ten Simple Rules for Effective Statistical Practice
http://www.datasciencecentral.com/profiles/blogs/ten-simple-rules-for-effective-statistical-practice

HAIR, J. F., ANDERSON, R. E., TATHAM, R. L., BLACK, W. C. Análise multivariada de dados. São Paulo: Bookman, 2010.

Classification Accuracy is Not Enough: More Performance Measures You Can Use
http://machinelearningmastery.com/classification-accuracy-is-not-enough-more-performance-measures-you-can-use/