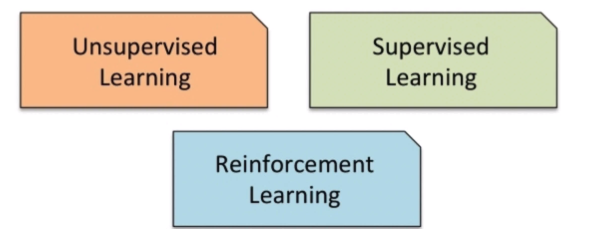
Machine Learning com Linguagem Python Scikit-learn

Considerar uma carreira em Machine Learning e aprender tudo que for possível sobre esse assunto é uma das decisões mais inteligente que você pode tomar na carreira profissional.

Machine Learning, ou seja, a aplicação de algoritmos e ciência para **extrair informações de dados**, é um dos campos mais espetaculares da ciência da computação atualmente.

* Processo de Machine Learning
* Biblioteca Scikit-learn
* Coleta, Análise Exploratória e Pré-Processamento
* Feature Selection
* Algoritmos de Machine Learning - Classificação
* Algoritmos de Machine Learning – Regressão
* Métodos Ensemble
* Algoritmo XGBoost e Kaggle

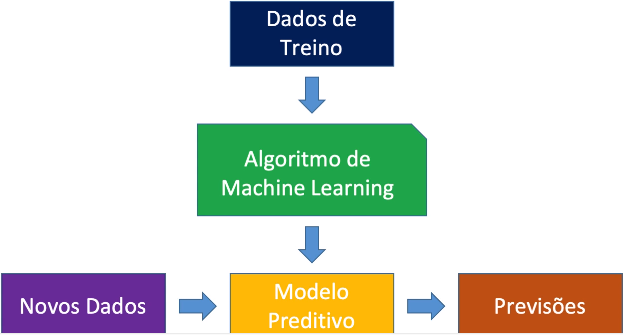
Algoritmos de Machine Learning são um elemento codificado capaz de programar computadores para que façam determinadas ações sozinhos. Alguns algoritmos de Machine learning estão classificados em 3 categorias principais:



Os algoritmos de cada categoria possuem procedimentos diferentes para realizar o aprendizado com os dados, portanto antes de trabalhar com **Machine Learning** precisamos escolher **qual algoritmo** iremos trabalhar de acordo com o **problema de negócio** que estamos tentando resolver.

Aprendizagem Supervisionada:Começamos pelos dados de **treino.** Para que o aprendizado ocorra precisamos de dados independentemente de qual seja o problema, para na sequência **alimentar o algoritmo.** Alimentar uma sequência de etapas, de instruções matemáticas e/ou estatísticas. Bem como as redes neurais que buscam simular o cérebro humano, aplicam os mesmos métodos matemáticas para solucionar problemas.

Uma vez que o algoritmo é **apresentado aos dados**, ele começa a **aprender o relacionamento entre os dados.** Entregamos as características de um carro para o algoritmo, e dizemos que aquilo é um automóvel, ou seja, entregamos dados de entrada sendo as **características** e o dado de saída ou rótulo, o **resultado** carro. O algoritmo vai aprendendo o **relacionamento entre entrada e saída.** Uma vez que o aprendizado ocorra, é **criado o modelo preditivo.** O modelo é o resultado do processo de aprendizagem do algoritmo.



Uma vez o **modelo preditivo** criado, basta apresentar **novas entradas de dados** ao modelo, ou seja, somente as “características”, para que assim seja capaz de prever as saídas.

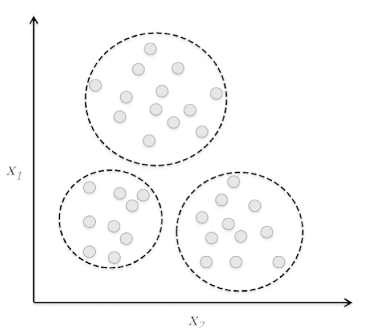
Quando trabalhamos com aprendizagem supervisionada temos basicamente **2 tipos de algoritmos:**



Usamos algoritmos de regressão quando nosso objetivo é prever um valor numérico. Prever o valor da casa em determinado bairro, prever o volume de vendas para o próximo período.

Em outras situações queremos **prever uma classe**, se uma pessoa tem ou não uma doença. A resposta final é um simples sim ou não. Queremos prever se uma pessoa deve ou não receber crédito do banco com base no seu histórico.

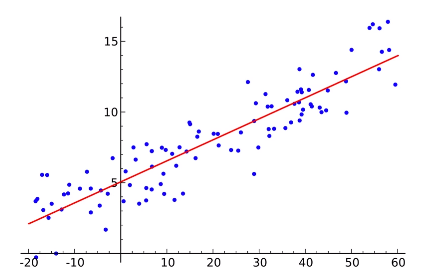
Na grande maioria das tarefas de Machine Learning trabalharemos em algoritmos de regressão ou classificação. Em níveis mais avançados trabalhamos com **aprendizagem não supervisionada** e em casos muito específicos com **aprendizagem por reforço.**

Aprendizagem Não Supervisionada: na aprendizagem não supervisionada, temos os dados de entrada (características) mas não temos as saídas. O algoritmo aprende o relacionamento nos dados e gera agrupamentos (clusters) por características muito similares.

Aprendizagem por Reforço: a máquina é exposta a um ambiente onde ela se treina usando **tentativa e erro.** A máquina aprende a partir das **experiências passadas** e tenta capturar melhor conhecimento possível para tomar decisões negócios mais precisas. Um bom exemplo de aprendizado por reforço é o **processo de decisão Markov.** Em robótica, os robôs em geral aprendem através de aprendizagem por reforço.

Regressão Linear

Um dos algoritmos de Machine Learning mais simples que existem, é a Regressão Linear. Cujo objetivo é prever um valor numérico. É utilizada para estimar valores reais **com base em variáveis contínuas**. Estabelecemos uma **relação entre variáveis** dependentes e independentes, ajusta a melhor linha de relacionamento entre essas variáveis. A variável **independente fica no eixo X**, enquanto a variável **dependente fica no eixo Y**. O objetivo nesse método é encontrar a linha de regressão para realizar previsões. Na linha encontraremos os **valores correspondentes de Y.**



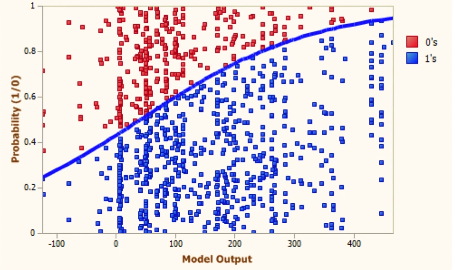
X poderia ser um número de quartos em uma casa e Y o valor da casa.

**Com base no valor da variável dependente, consigo prever o valor da variável independente.**

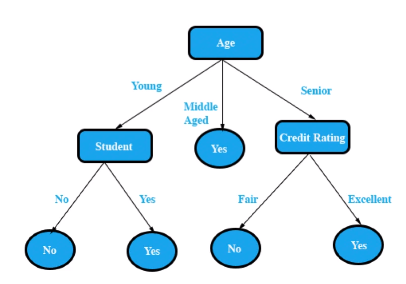
Regressão Logística

Um dos algoritmos mais importantes de Machine Learning, inclusive para quem trabalha com **Inteligência Artificial.** Apesar de levar no nome “regressão”, trata-se de um **algoritmo de Classificação** e de aprendizagem **supervisionada.**

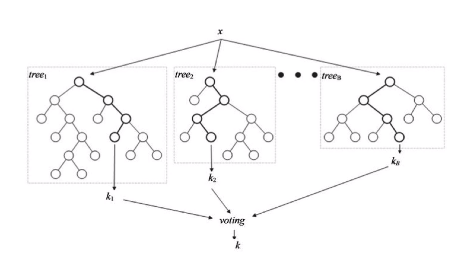
Neste caso o objetivo é prever uma classe, como neste gráfico onde temos os dados em azul e os dados em vermelho. Vermelhos representam 0 e azuis representam o valor 1. O que nos queremos é **encontrar a linha que separa os dados das duas categorias.** De modo que ao entregar novos dados de entrada ao modelo, ele seja capaz de fazer a classificação, se o tal dado de entrada pertence à categoria azul ou vermelha.



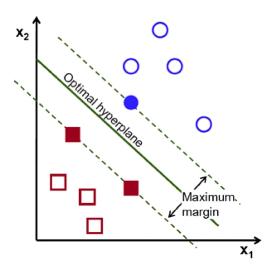
Árvores de Decisão: este algoritmo nada mais é que **uma sequência de etapas**, a partir da primeira variável o algoritmo vai tomando decisões até que na parte final ele seja capaz de fazer uma classificação. Portanto temos um conjunto de variáveis de entrada, uma variável de saída como a possibilidade de “sim ou não”, o que é um problema de classificação, e a **árvore de decisão encontrará o melhor caminho para chegar à classificação final.** Algoritmo bastante poderoso, com precisão relevante e boa compreensão.



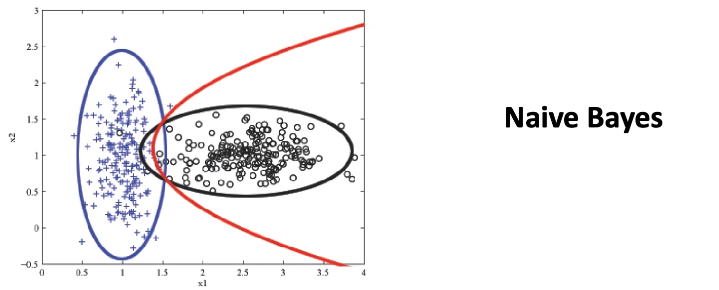
Random Forest: Se é possível ter uma árvore, é possível ter uma floresta. Colocar **vários algoritmos de árvore de decisão** **competindo juntos.** Alimentar com dados de entrada, esses vários algoritmos de árvore trabalharão entre si para encontrar o melhor caminho para uma saída e então no final, realizar uma **espécie de votação pela melhor situação possível.**



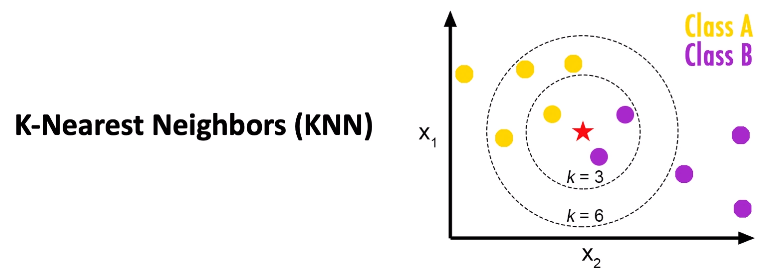
Support Vector Machines: É um algoritmo que pode ser usado para **problemas de regressão ou de classificação.** É possível criar um classificador ou criar um regressor. O que há de interessante neste modelo é trabalhar com **dados não linearmente separáveis.** Imagine que tenhamos um conjunto de dados com duas variáveis e **claramente tenhamos uma separação linear desses dados, limites entre eles.** Se isso realmente existir no conjunto podemos usar a regressão linear sem maiores dificuldades, a regressão linear por si só consegue fazer essa divisão linear nos dados, mas em algumas situações as variáveis **não são linearmente separáveis.** Para estes casos, o SVM cria uma **outra dimensão para os dados**, ou seja, uma terceira dimensão é criada onde os dados são separados, voltamos de dimensão e apresentamos o **classificador.** Geralmente um algoritmo muito preciso.



Naive Bayes: Assume a independência entre os preditores, as variáveis preditoras. É um **algoritmo probabilístico,** utiliza a essência da teoria da probabilidade. Uma acurácia razoável, mas de difícil ajuste.

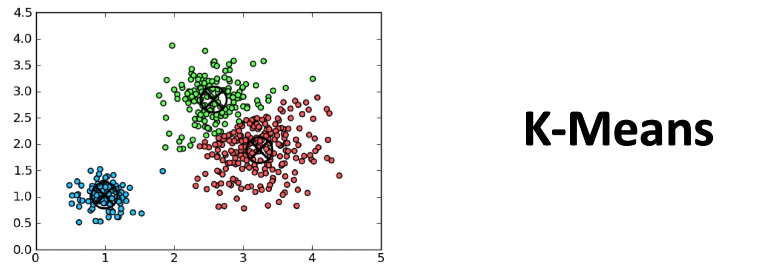


K-Nearest Neighboors (KNN): é considerado um dos algoritmos mais simples em Machine Learning. É um algoritmo utilizado para tarefas de **classificação**, mas também para tarefas de regressão dependendo do parâmetro utilizados. É um algoritmo que toma como base a **distância entre os pontos de dados**, uma distância matemática. Por conta de sua simplicidade, muitos não o reconhecem como um modelo de Machine Learning.

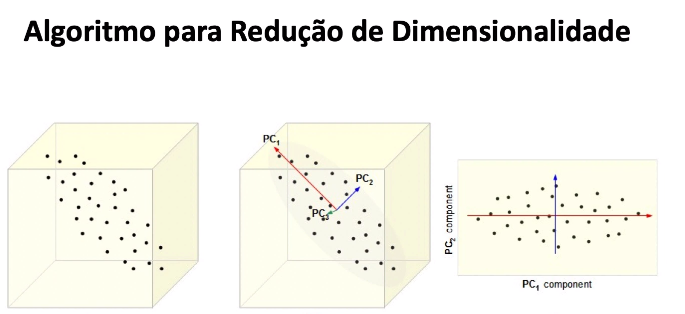


K-Means: este algoritmo é o **principal representante dos algoritmos de aprendizagem Não-Supervisionada.** Quando não temos dados de saída. O que ele faz é agrupar os dados, buscando entre seus relacionamentos os dados que mais se aproximam, criando os clusters.

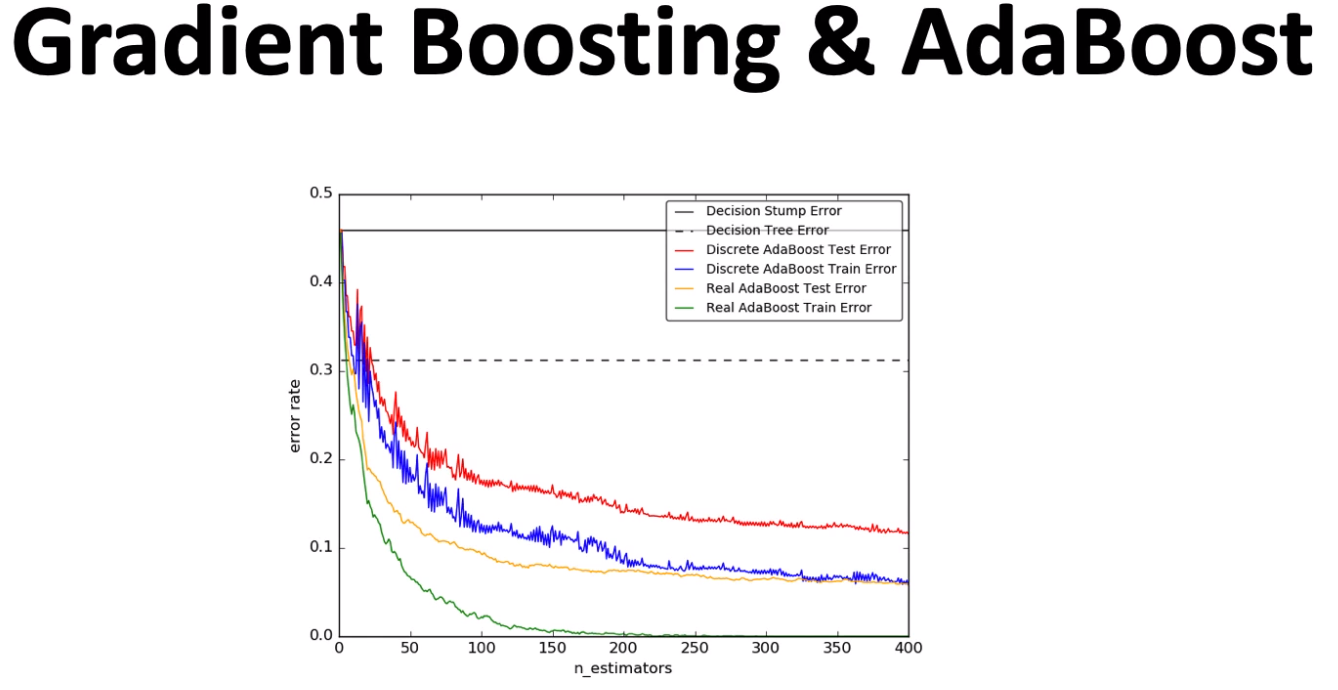
Algoritmo muito útil para **segmentação de clientes.**



Algoritmo para Redução de Dimensionalidade: imagine um dataset com 300 colunas | 300 variáveis. É de grande complexidade treinar um algoritmo com tantas variáveis, sendo necessário **reduzir a dimensionalidade criando Componentes principais.** Cada componente contém informação de um **grupo de variáveis.** Supondo que ao escolher 50 dessas variáveis e coloque em **um componente,** outras 70 variáveis e coloque em um **segundo componente** e assim por diante. Portanto, ao invés de treinar o algoritmo com 300 variáveis, ele é treinado com 3, 4 ou 5 componentes... Cada componente nada mais é que um **resumo vetorial das variáveis que fazem parte daquele componente.**



Gradient Boosting & AdaBoost: algoritmo de alta precisão, porém mais difíceis de treinar e exigindo mais capacidade computacional.



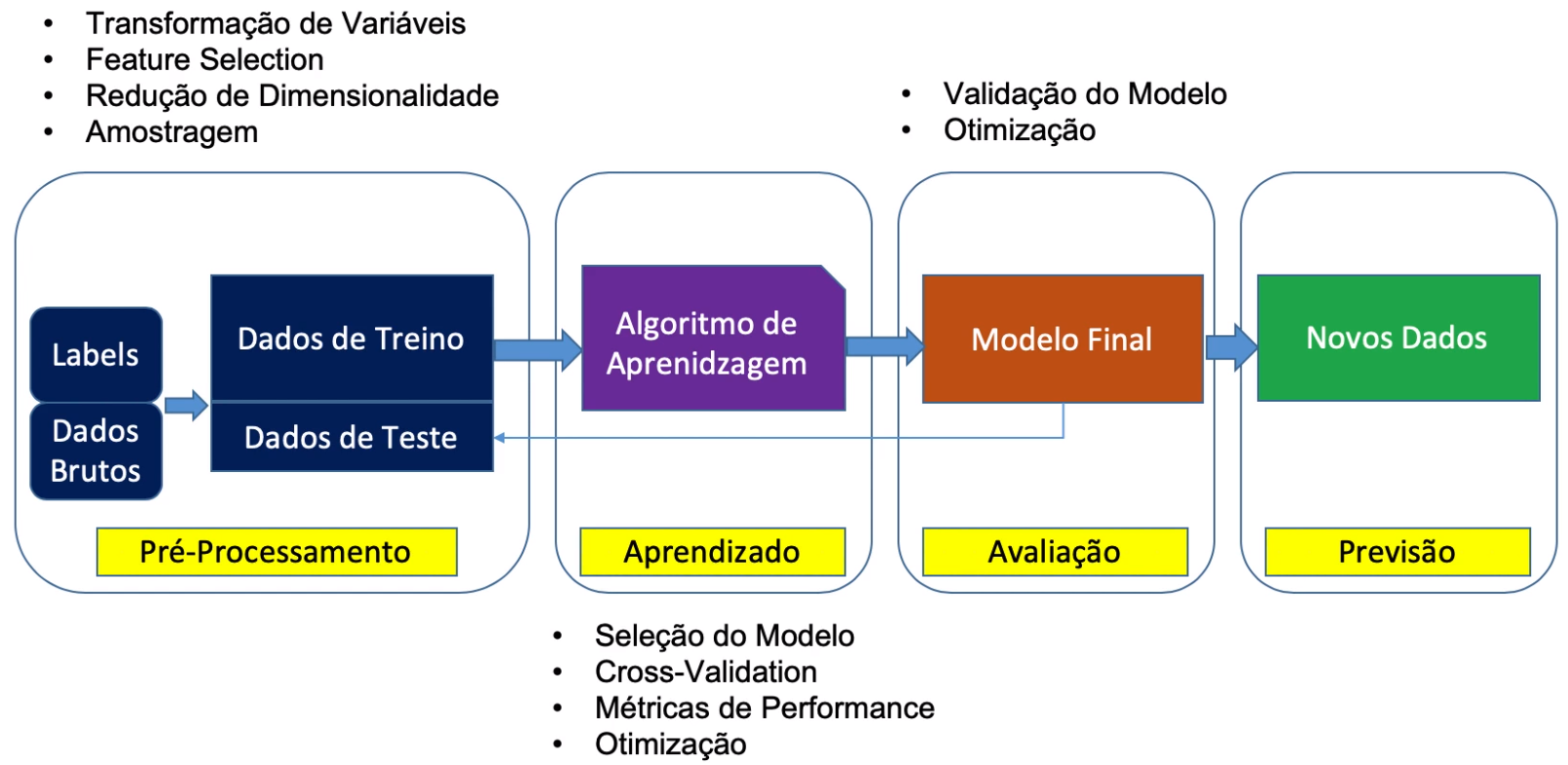
Processo de Construção de Modelos de Machine Learning

Cada etapa deste processo requer diferentes ferramentas, diferentes técnicas, diferentes procedimentos. Basicamente temos **quatro etapas principais.** Um ou outros algoritmos talvez requeira algum pequeno ajuste neste processo, principalmente **algoritmo de aprendizagem não-supervisionada,** mesmo assim, **a grande maioria dos algoritmos são de aprendizagem supervisionada.**

Na fase de pré-processamento nós vamos preparar os dados, organizar os dados de saída target, os dados brutos que são nossas variáveis preditoras, coletá-los de uma fonte, pré-processar os dados e dividi-los entre **Dados de Treino e Dados de Teste.** Algumas das atividades de Pré-Processamento.

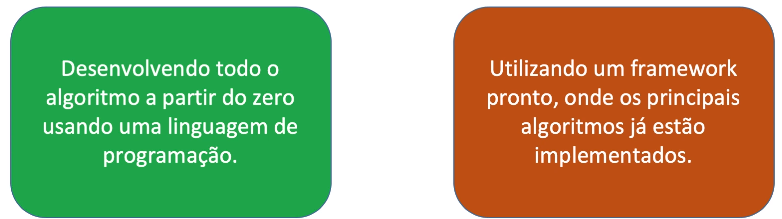
Uma vez com os Dados de Teste e Dados de Treino, nós usamos este para o **aprendizado.** Para treinar o algoritmo de Machine Learning. Nessa etapa de aprendizado do processo nós vamos realizar a **seleção do modelo,** escolher um dentre os 60 disponíveis. Escolher o melhor para a atividade proposta.

Depois de treinar o algoritmo temos um modelo, porém não criaremos apenas uma versão do modelo, mas sim no mínimo de meia dúzia de modelos com diferentes parâmetros e ajustes e por fim selecionar o mais aderente. Podemos também aplicar **cross-validation**, uma forma de tornar o **processo mais preciso**, usar também **métricas de performance para comparar versões do modelo** que criaremos durante o aprendizado. O processo de **otimização** envolve inclusive várias técnicas matemáticas.



Soluções de Machine Learning

Podemos construir modelos de Machine Learning de duas formas principais:



Para que possamos desenvolver rum algoritmo a partir do zero é necessária muita experiência **em Machine Learning, matemática, estatística e excelência em programação.**

As principais soluções de programação para Machine Learning são:

* Python
* Linguagem R
* Scala
* Java
* Javascript
* Go
* C++ / C#

Principais Frameworks para Machine Learning

* Scikit-learn (Python)
* Caret(R)
* TensorFlow(Python, R, Java, C++)
* Apache Mahout (Python, Java)
* Spark Milib (Scala, Java, Python, R)
* H20 (Java, Python)
* Weka (Java, Python)
* PyTorch, CNTK, MXNet (Python, C++, Java)

**A linguagem** Python oferece duas vantagens principais sobre todas as outras soluções. Primeiro, por se tratar de uma linguagem de uso geral, ela pode ser usada com qualquer uma destas soluções. Segundo, Python possui uma das mais poderosas soluções gratuitas de Machine Learning como o pydata Stack **Scikit-learn.**



O que é Normalização? Quando aplicar?

A normalização é uma técnica frequentemente aplicada à preparação dos dados em aprendizado de máquina. O objetivo da normalização é alterar os valores das colunas numéricas no conjunto de dados para uma **escala comum**, sem distorcer as diferenças nos intervalos de valores. Não precisamos aplicar normalização a todo conjunto de dados. É necessário apenas quando os recursos (variáveis) tiverem intervalos diferentes.

Por exemplo, considere o conjunto de dados contendo dois recursos, idade (x1) e receita (x2). Onde a faixa etária varia de 0 a 100 anos, enquanto a renda varia de 0 a 20.000 ou mais.

**A renda é cerca de 1.000 vezes maior do que a idade** e com uma variação de valores muito maior. Então, esses dois recursos estão em **intervalos muito diferentes**. Quando fazemos análises adicionais, como regressão linear multivariada, por exemplo, a renda atribuída influenciará muito mais o resultado devido ao seu valor maior. E isso causa problemas durante o treinamento do algoritmo.

A normalização também é chamada simplesmente de **Scaler Min-Max** e basicamente reduz o intervalo dos dados de forma que o intervalo seja fixo entre 0 e 1 (ou -1 a 1, se houver valores negativos).

Funciona melhor para casos em que a padronização (que veremos no próximo item de aprendizagem) pode não funcionar tão bem. **Se a distribuição não for gaussiana normal ou o desvio padrão for muito pequeno, o Scaler Min-Max funciona melhor.**

Quando a Normalização é Importante?

A normalização é principalmente **necessária no caso de algoritmos que usam medidas de distância como clustering**, sistemas de recomendação que usam semelhança de cosseno, etc. Isto é feito de forma que uma variável que está em uma escala maior não afete o resultado apenas porque está em uma escala maior. Abaixo listamos alguns **algoritmos de Machine Learning que requerem a normalização dos dados**

1. KNN com medida de distância euclidiana se quiser que todos os recursos contribuam igualmente no modelo.

2. Regressão Logística, SVM, Perceptrons, Redes Neurais.

3. K-Means

4. Análise discriminante linear, análise de componentes principais, análise de componentes principais do kernel.

Classificadores baseados em modelo gráfico, como Fisher LDA ou Naive Bayes, bem como Árvores de Decisão e métodos baseados em árvore, como Random Forest, **são invariantes ao dimensionamento de recursos**, mas ainda assim pode ser uma boa ideia redimensionar os dados.

A normalização eliminará a capacidade de interpretação do modelo e, portanto, dependerá, em última instância, da necessidade do negócio.

O que é Padronização? Quando aplicar?

Padronização (ou normalização do escore Z ou em inglês Standardization ou ainda Standard Scaler) é o processo de redimensionamento dos recursos (variáveis) para que eles tenham as propriedades de uma distribuição normal com μ = 0 e σ = 1, onde μ é a média e σ é o desvio padrão da média. É amplamente utilizado em SVMs, **regressão logística e redes neurais.**

Normalização x Padronização - Aplicação em **Variáveis Quantitativas**

Conforme vimos nas aulas anteriores, a Normalização transforma os dados em um intervalo, digamos entre 0 e 1 ou 1 e 10, de forma que os **números estejam na mesma escala**. Por exemplo, podemos converter os dados de centímetros para metros para que tenhamos todos na mesma escala. A Normalização pode ser formulada como:

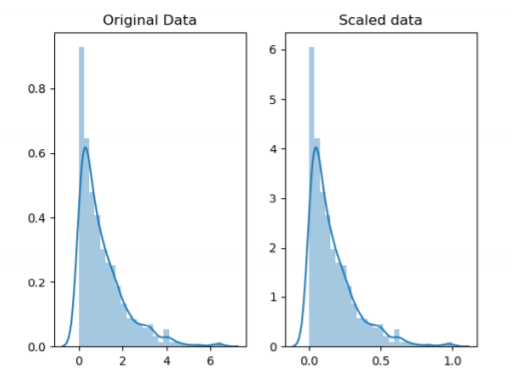
x <- (x - min (x)) / (max (x) - min (x))

**A Padronização significa transformar os dados de tal forma que eles tenham média zero e desvio padrão igual a 1**. Portanto, aqui temos os dados em **escala de forma padronizada**, de modo que a distribuição seja aproximadamente uma distribuição normal, sendo representado da seguinte forma:

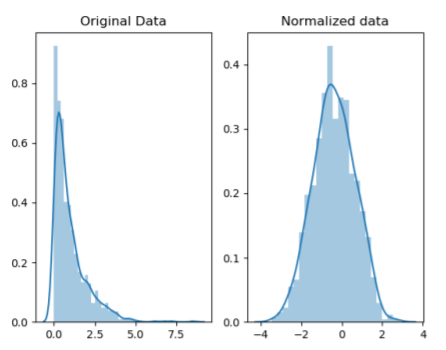
x <- (x - mean (x)) / sd (x)

Ambas as técnicas têm suas desvantagens. Se você tiver valores outliers em seu conjunto de dados, a Normalização dos dados certamente aumentará os dados "normais" para um intervalo muito pequeno. E, geralmente, a maioria dos conjuntos de dados tem outliers.

Ao usar a Padronização, seus novos dados não são limitados (ao contrário da Normalização). Portanto, a Normalização é geralmente evitada quando o conjunto de dados tem outliers (desde que inclua o valor máximo). Nesses casos, preferimos a Padronização. Os gráficos abaixo resumem as diferenças quando aplicamos Normalização e Padronização:



Observe no eixo x do gráfico acima como a escala dos dados é diferente, embora a distribuição dos dados seja a mesma. Isso é o que chamamos de Normalização.



Observe no eixo x do gráfico acima como a distribuição dos dados **agora segue uma distribuição normal depois que aplicamos a Padronização**. Uma distribuição normal é caracterizada por média 0 e desvio padrão 1.

1. A Normalização torna o treinamento menos sensível à escala de recursos, para que possamos resolver melhor os coeficientes.

2**. O uso de um método de Normalização melhorará a análise de múltiplos modelos**.

3. A Normalização assegurará que um problema de convergência não tenha uma variância massiva, tornando a otimização viável.

4**. A Padronização tende a tornar o processo de treinamento bem melhor, porque a condição numérica dos problemas de otimização é melhorada.**