MCTI FUTURO

FUTURO DO TRABALHO, TRABALHO DO FUTURO

FUNDAMENTOS EM MACHINE LEARNING

Apostila do aluno

Instituto de Tecnologia

MCTI FUTURO

FUTURO DO TRABALHO, TRABALHO DO FUTURO

Softex

GOVERNO FEDERAL

BRASIL

UNIÃO E RECONSTRUÇÃO

Jane Piantoni – Analista de Capacitação Técnica

Autora da apostila

Larissa Jessica Alves – Analista de Suporte Pedagógico

Revisão da apostila

Fit Instituto de Tecnologia

Sorocaba, novembro de 2021

Autor

Jane possui graduação em Tecnologia em Processamento de Dados pela Faculdade de Tecnologia de Sorocaba (2001), MBA em Marketing (2009), Gestão Financeira pela ESAMC (2010) e mestrado em Ciências da Computação pela UFSCar Sorocaba (2016). Tem experiência profissional nas áreas de Governança e Gestão da Tecnologia da Informação, Data Analysis, Docência no Ensino Técnico e Superior. Pesquisadora nas áreas Inteligência Artificial - Aprendizado de Máquina e Educação à Distância.

APRESENTAÇÃO

A presente apostila é um instrumento teórico que complementa o curso intermediário de capacitação em Inteligência Artificial, executado pelo FIT-Instituto de Tecnologia. Nela, veremos os Fundamentos de Machine Learning, uma subdivisão da Inteligência Artificial, em especial, vamos conhecer os problemas de Classificação, Regressão, Agrupamento e Otimização, e também, a estrutura de alguns algoritmos.

Vamos aplicar na prática, técnicas de pré-processamento de dados e implementar e avaliar algoritmos de Machine Learning, utilizando a linguagem de programação Python e suas principais bibliotecas.

Este material é baseado em artigos científicos, periódicos, revistas e livros científicos. Recomendo ao aluno que realize os exercícios propostos e acesse os materiais indicados nas referências bibliográficas para aprofundar a leitura desse material e complementar o que foi lido aqui.

Desejo a você, prezado aluno, que tenha um excelente curso!

Boa Leitura!

Sumário

1 Fundamentos em Machine Learning

Machine Learning (ML) ou aprendizado de máquina, é uma subdivisão da inteligência artificial que trata do processo de indução de uma hipótese (ou aproximação de função) a partir da experiência passada utilizando recursos computacionais.

Existem diversas definições de ML na literatura, entre elas, a definição apresentada por Tom Mitchell (1977):

> "A capacidade de melhorar o desempenho na realização de alguma tarefa por meio da experiência. Diz-se que um programa de computador aprende com a experiência E com relação a alguma classe de tarefas T e medida de desempenho P, se seu desempenho nas tarefas em T, conforme medido por P, melhora com a experiência E. "

Exemplo: jogar damas

E = a experiência de jogar muitos jogos de damas

T = a tarefa de jogar damas

P = a probabilidade de o programa vencer o próximo jogo.

Nas últimas décadas, com a crescente complexidade dos problemas a serem tratados computacionalmente e do volume de dados gerados por diferentes setores, tornou-se clara a necessidade de ferramentas computacionais mais sofisticadas, que fossem mais autônomas, reduzindo a necessidade de intervenção humana e dependência de especialistas. Essas técnicas deveriam ser capazes de criar por si próprias, a partir da experiência adquirida, uma hipótese ou função, capaz de resolver o probllema que se deseja tratar (Faceli et al., 2011).

Shalev-Shwartz (2014) refere-se ao termo Machine Learning como sendo à detecção automática de padrões em dados.

Neste curso, veremos problemas de aprendizado de máquina que podem ser atribuídos a uma de duas classificações gerais:

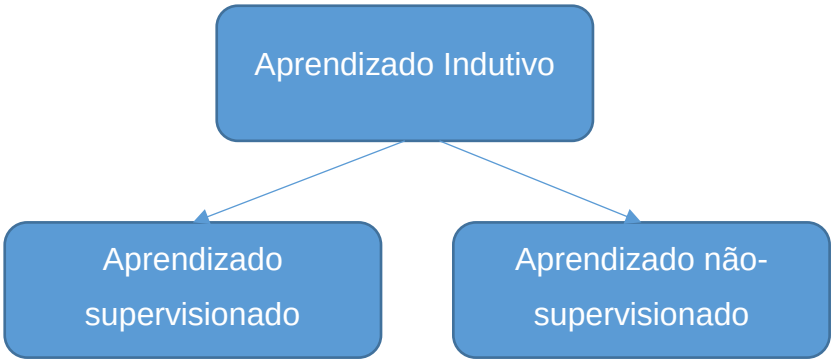

Figura 1 – Hierarquia de aprendizado. Fonte: Faceli et al.,2011.

Na aprendizagem supervisionada, ensinamos o computador a fazer alguma coisa, já na aprendizagem não-supervisionada, deixamos que ele aprenda por si só. A seguir detalharemos cada uma destas aprendizagens.

1.1. Aprendizado supervisionado

Na aprendizagem supervisionada, recebe-se um conjunto de dados a qual já sabemos como deve ser a saída correta, tendo a ideia de que existe uma relação entre a entrada *input e a saída output* (Ng, 2012).

Os problemas de aprendizagem supervisionada podem ser categorizados em problemas de **"Regressão**" e **"Classificação".** Em um problema de regressão, tenta-se prever os resultados em uma saída contínua, o que significa que o objetivo é mapear as variáveis de entrada para alguma função contínua. Em um problema de classificação, tenta-se prever os resultados em uma saída discreta. Em outras palavras, o objetivo pe mapear as variáveis de entrada em categorias discretas.

Exemplos:

- **Regressão** - Com os dados sobre o tamanho das casas no mercado imobiliário, tente prever seu preço. O preço em função do tamanho é uma produção contínua, portanto, este é um problema de regressão.

- **Classificação** – Diante dos sintomas de um paciente, como por exemplo, temperatura corporal, ocorrência de manchas na pele, entre outras, podemos classificar o paciente como doente ou saudável.

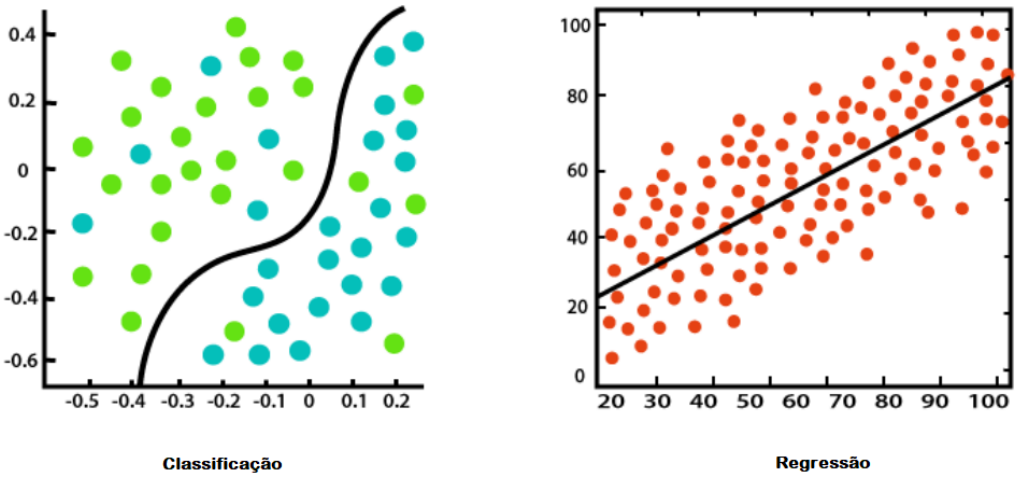

Figura 2 – Classificação e Regressão. Fonte: JavaTpoint.

1.2. Aprendizado não-supervisionado

O aprendizado não-supervisionado permite abordar problemas com pouca ou nenhuma ideia de como devem ser nossos resultados. Trata-se de um tipo de aprendizado de máquina no qual não é fornecido com nenhum rótulo pré-atribuído ou pontuação para os dados de treinamento.

No processo de aprendizado busca-se identificar regularidades entre os dados a fim de agrupá-los ou organizá-los em função das similaridades que apresentam entre si (Ng, 2012).

Exemplos:

- **Agrupamento** - pegue uma coleção de 1.000.000 de genes diferentes e encontre uma maneira de agrupar automaticamente esses genes em grupos que são de alguma forma semelhantes ou relacionados por variáveis diferentes, como tempo de vida, localização, funções e assim por diante.

1.3. Avaliação do Conhecimento 1

Uma empresa construiu um conjunto de dados com todas as transações comerciais que realizou nos últimos 5 anos, nesses dados podemos encontrar informações sobre volume de compras mensais e produtos mais consumidos, periodicidade da compra, histórico financeiro do cliente, entre outras informações. A equipe de produto gostaria de criar e oferecer novos produtos para grupos de clientes com perfis similares. Qual o tipo de técnica você utilizaria para atender a necessidade da equipe de produtos?

(a) Agruparia os clientes apenas pelo volume mensal de compras utilizando SQL.
(b) Utilizaria métodos de Aprendizado Supervisionado.
(c) Utilizaria métodos de Aprendizado Não-Supervisionado.
(d) Esses dados não trazem informações suficientes para atender a necessidade do time de produto.

2 Exploração e Pré-Processamento de Dados

Técnicas de pré-processamento de dados são frequentemente utilizadas para tornar os conjuntos de dados mais adequados para uso de algoritmos de ML.

Vários algoritmos de ML têm dificuldade em utilizar os dados no seu formato original. Para tratar desse problema, são realizadas transformações nos dados originais antes que eles sejam utilizados pelo algoritmo (Faceli et al., 2011).

A tarefa de se realizar a análise exploratória de dados é normalmente trabalhosa, chega-se a gastar até cerca de 80% do tempo no processo de limpeza e preparação dos dados que posteriormente serão utilizados (Dasu e Johnson, 2003).

2.1. Limpeza e transformação

O processo de limpeza de dados ou *data cleansing* envolve a preparação de dados para serem utilizados em projetos de ML, através de atividades que removem ou modificam dados incorretos, incompletos, irrelevantes, duplicados ou formatados incorretamente. Esses dados além de muitas vezes não serem necessários ou relevantes, também podem afetar o processo de aprendizado de alguns algoritmos de ML.

Transformação de dados ou *data transformation* refere-se ao processo de conversão de dados para um formato de interesse dos métodos de ML. Pode-se dizer que a transformação de dados é o mapeamento e a conversão de dados de um formato para outro.

2.2. Medidas de dispersão

As medidas de dispersão são parâmetros estatísticos usados para determinar o grau de variabilidade dos dados de um conjunto de valores, dentre elas, podemos destacar a variância e desvio padrão.

A variância mede a distância de cada valor no conjunto de dados em relação a média.

Exemplo:

```
import numpy as np
list = [2, 4, 4, 4, 5, 5, 7, 9]
print(np.var(list))
```

Já o desvio padrão, é a medida de dispersão em torno da média populacional. Ou seja, o desvio padrão indica o quanto um conjunto de dados é uniforme. Quanto mais próximo de 0 for o desvio padrão, mais homogêneo são os dados.

Exemplo:

```
import numpy as np
list = [2, 4, 4, 4, 5, 5, 7, 9]
print(np.std(list))
```

2.3. Visualização

Uma das formas mais simples de ilustrar a distribuição de um conjunto de valores de uma variável é o uso de histogramas. Neste tipo de gráfico, no eixo horizontal, temos o conjunto (ou intervalos) de valores observados, enquanto que no eixo vertical, apresenta-se a frequência de ocorrência de cada valor (ou valores dentre de um intervalo) presente na amostra analisada.

Exemplo:

```
import numpy as np
import matplotlib.pyplot as pyplt
valores = np.array([2, 5, 7, 3, 4, 6, 5, 2, 6, 5, 5, 3, 7, 7])
pyplt.hist(valores, bins='auto')
pyplt.title('Histograma')
pyplt.ylabel('Frequencia')
pyplt.xlabel('Valores')
pyplt.show()
```

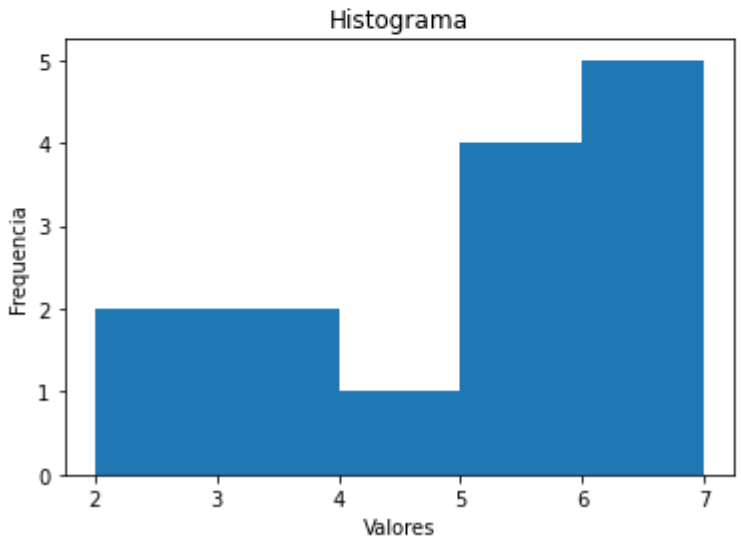

Figura 3 – Visualização Histograma. Fonte: Matplotlib.org.

2.4. Outliers

Em análise de dados, pode-se dizer que outliers são amostras que estão excepcionalmente distantes do fluxo principal dos dados. Tais valores discrepantes podem ter sido causados por diversos motivos, como por exemplo:

- Erro de medição ou entrada
- Corrupção de dados
- Observação de valores discrepantes verdadeiros

Não há uma maneira precisa de definir e identificar outliers em geral devido às especificidades de cada conjunto de dados. No entanto, pode-se

utilizadar métodos estatísticos para identificar observações que parecem ser raras ou improváveis no dados disponíveis.

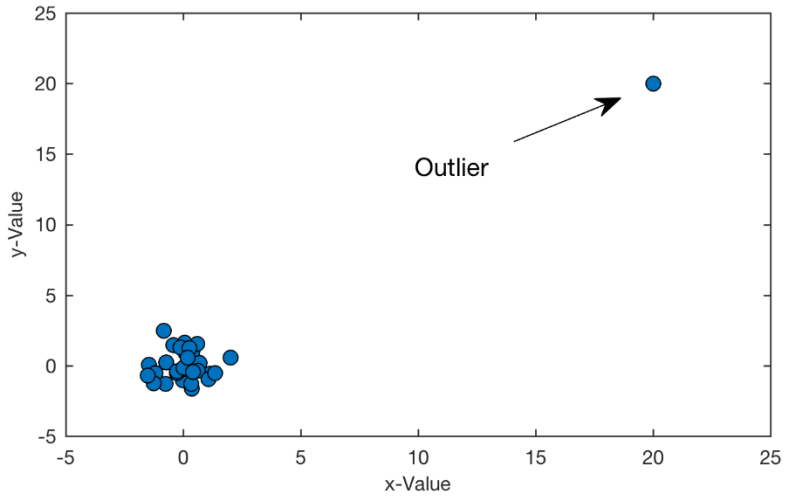

Figura 4 – Exemplo de Outlier. Fonte: Medium Anujkumar 2020.

2.5. Desbalanceamento

O desbalanceados de dados pode ser definido como uma pequena incidência de uma categoria dentro de um conjunto de dados (classe minoritária) em comparação com as demais categorias (classes majoritárias).

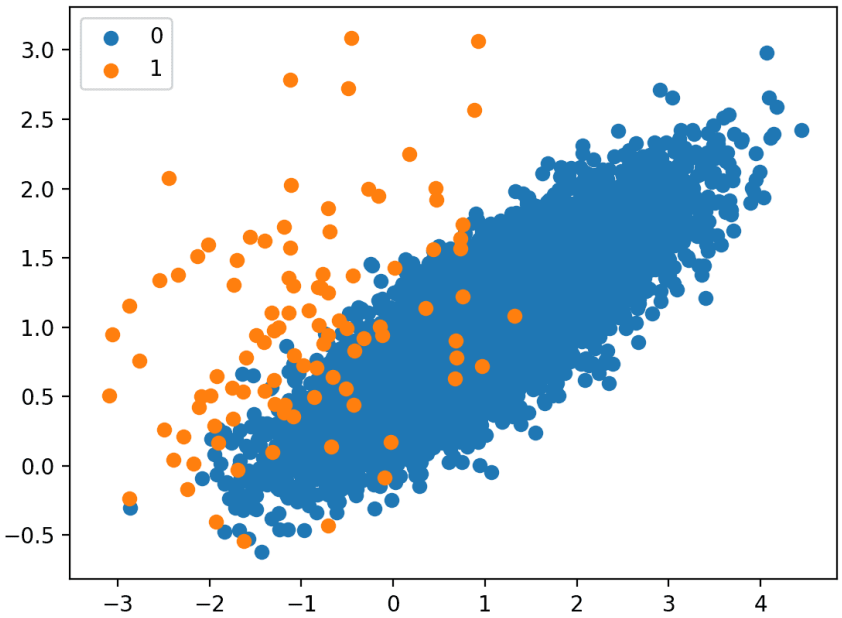

Figura 5 – Dados desbalanceados. Fonte: Machine Learning Mastery, 2020.

O desbalanceamento nos dados mostra-se presente nos dados reais em diversos campos do conhecimento, não sendo incomum encontra-los regularmente e em contextos variados, como por exemplo, em dados de diagnóstico de uma doença rara ou incomum, onde a maioria dos dados coletados apresentam um resultado negativo.

O não tratamento do problema de desbalanceamento afetará negativamente o desempenho do modelo de ML. Apesar de não existir um método universal e ótimo, existem diversas abordagens para lidar com dados desbalanceados.

Uma abordagem simples que tem sido amplamente utilizada é a abordagem ***Sampling***. Trata-se de um pré-processamento dos dados, que busca minimizar as discrepâncias entre as classes por meio de uma reamostragem do conjunto de dados de treinamento. Alguns exemplos dessa técnicas:

- Criar novas observações da classe minoritária a partir das informações contidas nos dados originais. Essa geração de novas entradas pode ser feita aleatoriamente.

- Reduzir o desbalanceamento do conjunto de dados de treinamento focando na classe majoritária. Ou seja, diminuir aleatoriamente entradas da classe com maior número de ocorrências para equilibrar o número de amostras em cada classe.

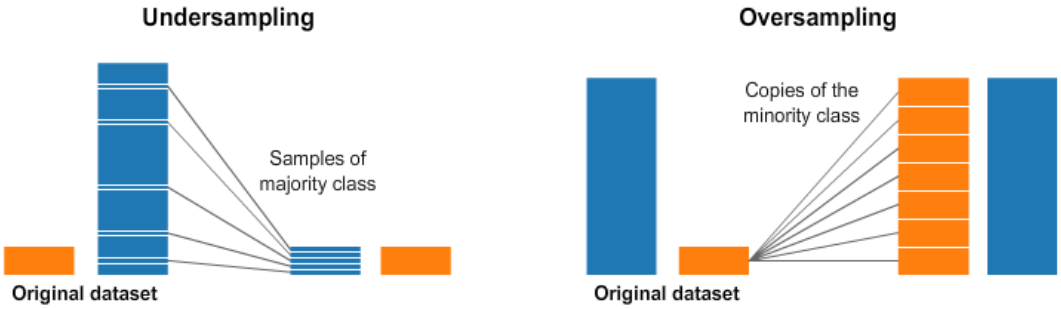

Figura 6 – Técnicas de sampling – reamostragem. Fonte: Alencar, 2018.

Na construção de um modelo de ML para classificação, por exemplo, a consequência do desequilíbrio entre as classes do conjunto de treinamento pode levar o modelo a classificar erroneamente as novas observações. Irá responder muito bem entradas para as classes majoritárias, mas terá um desempenho inferior para as minoriárias.

2.6. Redução de dimensionalidade

Um dos principais desafios de ML está relacionado a alta dimensionalidade do dados, trata-se da quantidade de características (atributos) presente nos dados, e que pode afetar o processo de aprendizagem dos algoritmos, principalmente em modelos preditivos, muitos atributos podem confundir o classificador ou regressor.

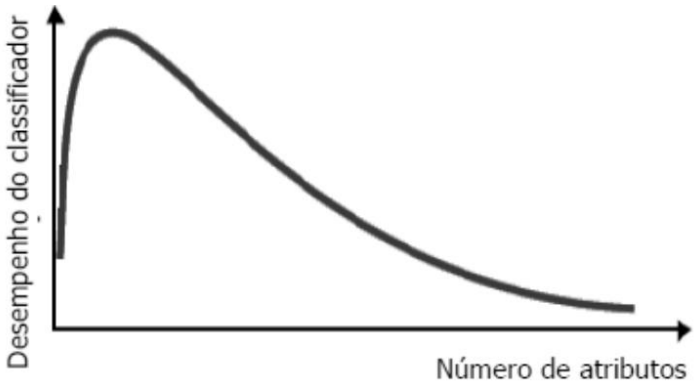

Figura 7 – Desempenho do classificador vs número de atributos. Fonte: Backes, 2018.

Para efetuar redução de dimensionalidade, pode-se aplicar algumas técnicas, como por exemplo:

- Seleção e a extração de atributos, onde busca-se diminuir a dimensão dos dados, removendo atributos redundantes ou irrelevantes que possam atrapalhar o processo de aprendizado.

- Utilização do algoritmo PCA (do inglês - Principal Component Analisys), que fornece um mapeamento de um espaço com N dimensões (em que N é o número de variáveis originais) para um espaço com M dimensões (onde M é tipicamente muito menor do que N).

Caso a redução de dimensionalidade ocorra de maneira excessida, o classificador pode perder o poder de discriminação. Torna-se necessário analisar a variação do comportamento do classificador com a dimensionalidade, realizando testes de remoção, de maneira a estimar a dimensionalidade ideal para determinado classificador e conjunto de dados.

2.7. Avaliação do conhecimento 2

Um modelo de ML de classificação apresentou resultados abaixo do esperado em seu processo de aprendizagem. Ao explorar o problema, o cientista de dados realizou uma análise exploratória nos dados, e percebeu

que no conjunto de dados de treinamento uma das classes era majoritária, representando 85% do total de amostras de treinamento. Qual técnica de pré-processamento de dados o cientista de dados poderia utilizar para melhorar a performance do aprendizado:

(a) Redução da dimensionalidade do conjunto de dados.
(b) Sampling ou reamostragem.
(c) Remoção de outliers.
(d) Não será possível realizar nenhum tipo de pré-processamento com este conjunto de treinamento.

3 Modelos Preditivos e Descritivos

Um algoritmo de ML preditivo é uma função que, dado um conjunto de dados com exemplos rotulados, constroi um estimador. Este rótulo, também chamado de classe, apresenta valores de um domínio conhecido. Para os conjuntos de dados com valores nominais, utiliza-se os métodos de classificação, caso o conjunto de dados apresente valores infinitos e ordenados, utiliza-se os métodos de regressão.

Já uma algoritmo de aprendizado descritivo, ou não-supervisionado, se refere à identificação de informações relevantes nos dados sem a presença de uma elemento externo para guiar o aprendizado, ou seja, não existem atribuitos que rotulam os dados. Dessa forma, o aprendizado reside na identificação das propriedades intrínsecas aos dados de entrada, cujo objetivo é encontrar padrões ou tendências, que auxiliem no entendimento dos dados. Os métodos de aprendizado descritivo podem ser divididos em sumarização, associação e agrupamento (Faceli et al., 2011).

Neste curso, iremos explorar alguns dos métodos de classificação, regressão linear e agrupamento (ou *clustering*), utilizando em nossas atividades práticas a biblioteca Scikit-Learn do Python.

3.1. Introdução ao Scikit-Learn

Scikit-learn, ou sklearn, é uma biblioteca de ML de código aberto que oferece suporte ao aprendizado supervisionado e não-supervisionado. Esta biblioteca também fornece várias ferramentas para ajuste de modelo, pré-processamento de dados, seleção e avaliação de modelo, bem como outros utilitários.

O Scikit-learn oferece vários métodos importantes para aprendizado de máquina e é projetado para operar com as bibliotecas numéricas e científicas do python, como NumPy e SciPy.

Ele fornece uma grande lista de ferramentas e algoritmos, divididos em 6 categorias, entre elas, Classificação, Regressão, Clustering, Redução da dimensionalidade, Seleção de modelos e Pré-processamento.

3.1.1 Interface do estimador

Todos os algoritmos implementados no scikit-learn, sejam algoritmos de pré-processamento, aprendizado supervisionado ou aprendizado não supervisionado, são considerados classes. Tais classes são chamadas de estimadores no scikit-learn.

Exemplo da classe de regressão linear:

```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model
```

A classe do estimador contém o algoritmo e também armazena o modelo que é aprendido a partir dos dados usando o algoritmo.

Pode-se definir quaisquer parâmetros do modelo ao construir o objeto de modelo. Esses parâmetros incluem regularização, controle de complexidade, número de clusters a serem encontrados, etc.

Mais informações podem ser obtidas no guia online de referência em https://scikit-learn.org/stable/

3.1.2 Método de ajuste

Todos os estimadores possuem um método de ajuste, que é usado para construir o modelo.

O método **fit** sempre requer como primeiro argumento os dados X, representados como uma matriz, onde cada linha representa um único ponto de dados.

```
estimator.fit(x_train, [y_train])
```

Os algoritmos supervisionados também requerem um **y** argumento, que contem os valores alvo (rótulos) para regressão ou classificação (ou seja, os rótulos de saída ou respostas conhecidas).

3.1.3 Método de previsão

```
estimator.predict(X_test)
```

Para criar uma previsão na forma de uma nova saída, como **y**, pode-se utilizar o método **predict**, exemplo:

```
y_pred = model.predict(X)
```

3.1.4 Método de pontuação

Os modelos supervisionados possuem um método de **score** que permite obter uma avaliação do desempenho do modelo:

```
estimator.score(X_test, y_test)
```

3.2. Classificação

O problema de classificação tem sido um dos mais estudados em aprendizado supervisionado, pode-se dizer, que um algoritmo de classificação é um procedimento que toma como entrada os dados exemplos de treinamento, com um rótulo e produz como saída um classificador (Sifuentes, 2019).

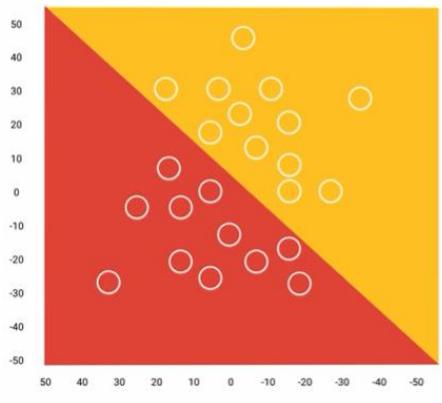

Figura 8 – Exemplo de classificação. Fonte: Cloud AI Adventures, 2017.

Neste curso, vamos conhecer os métodos de classificação K-Vizinhos Próximos (do inglês *K-Nearest Neighbors* – K-NN) e o Naive Bayes, um modelo probabilístico.

3.2.1 Algoritmo K-NN

O algoritmo K-NN localiza os vizinhos mais próximos em torno de uma nova observação de dados (desconhecida), ele calcula a distância de todos os pontos nas proximidades dos dados desconhecidos e seleciona aqueles com as distâncias mais curtas. Por isso, trata-se de uma abordagem baseada na distância.

A escolha do valor de ***k*** mais apropriado é definida como um parâmetro do algoritmo. Frequentemente, o valor utilizado para ***k*** é pequeno e ímpar. Em problemas de classificação, não é usual utilizar k = 2 ou valores pares, para evitar empates (Faceli et al., 2011).

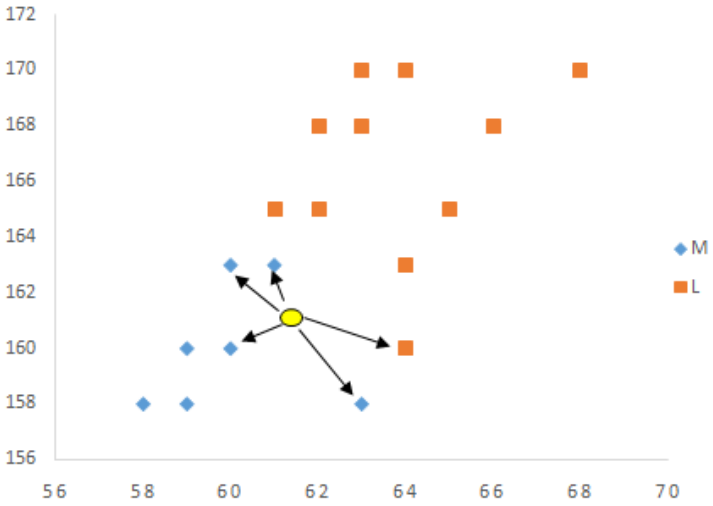

Figura 9 – K-NN. Fonte: Bha, 2017.

Utiliza-se da função de cálculo de distância (por exemplo, Euclidiana, Manhattan, etc.) para encontrar a proximidade entre a nova amostra e os casos de treinamento já conhecidos. Sendo este, um ponto fraco do algoritmo em termos de custo computacional com grandes conjuntos de dados.

Ou seja, o algoritmo é eficiente para dados com baixa dimensionalidade e com um conjunto de treinamento não muito grande.

Exemplo: **sklearn.neighbors**

```
from sklearn.neighbors import KNeighborsClassifier

classificador_knn = KNeighborsClassifier(n_neighbors = 5)
```

Parâmetro **k** (vizinhos mais próximos) = 5

3.2.2 Algoritmo probabilístico - Naive Bayes

O algoritmo Naive Bayes é um método de classificação que baseia-se na probabilidade de que um evento ocorra, ou seja, onde é determinada a probabilidade de um dado elemento, possuindo um conjunto de atributos, pertencer a uma classe específica.

Sendo baseado no teorema de Bayes:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Partindo do princípio de que, dado um conjunto contendo grupos divididos por valores e atributos é possível predizer em qual grupo uma nova instância pertence. Ou seja, a probabilidade de **A** dado **B** → dado um conjunto de evidências B, qual é a probabilidade da hipótese A estar em B. O algoritmo também utiliza o pressuposto de independência entre as variáveis de entrada.

O classificador Naive Bayes é conhecido por sua simplicidade e eficiência, além de se aprensentar robusto à presença de ruídos e atributos irrelevantes. Um dos exemplos de aplicação mais comum deste é a tarefa de identificar se um determinado e-mail é um *spam* ou não.

Exemplo: **sklearn.naive_bayes**

```
from sklearn.naive_bayes import GaussianNB

classificador_NB = GaussianNB()
```

Neste exemplo, utilizando Gaussian Naive Bayes, que oferece suporte a recursos e modelos de valor contínuo, distribuição gaussiana (normal).

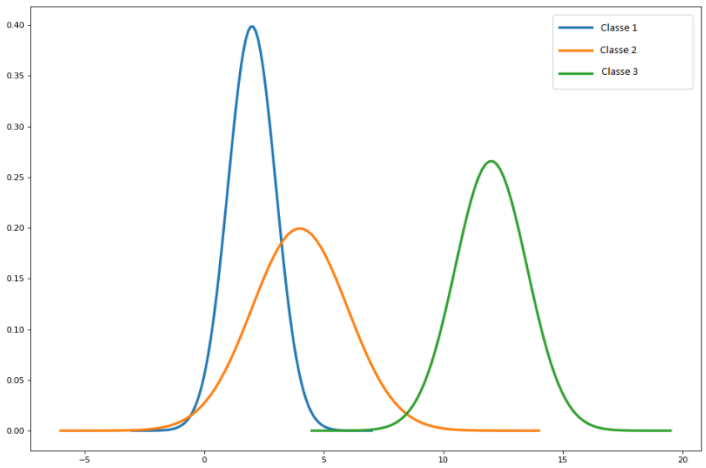

Figura 10 – Exemplo Gaussian Naive Bayes. Fonte: Eloquent Arduino, 2020.

3.3. Regressão Linear

A análise de regressão consiste na obtenção de uma equação que tenta explicar a variação de uma variável dependente pela variação dos níveis das variáveis independentes (Peternelli, 2009).

Neste curso, vamos estudar a regressão linear, que consiste em identificar uma equação linear que permita prever o valor da variável dependente em função dos valores conhecidos das variáveis independentes (Moreira, 2009).

Exemplos:

Regressão linear simples - com uma variável independente

- variável dependente → vendas
- variável independente → despesas com propaganda

Regressão linear múltipla - duas ou mais variáveis independentes

- variável dependente → preço do imóvel
- variáveis independentes → área, nº de quartos, nº de banheiros, localização, entre outros.

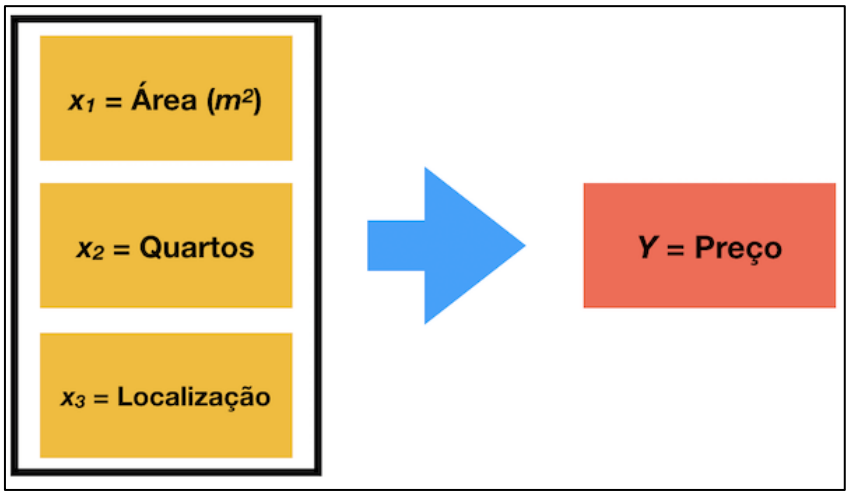

Figura 11 – Exemplo regressão múltipla linear. Fonte: Sigmoidal.ai.

Exemplo: **sklearn.linear_model**

```
from sklearn.linear_model import LinearRegression

regressaolin = LinearRegression()
```

3.4. Agrupamento

A análise de agrupamento consiste em uma técnica que tem como objetivo encontrar uma estrutura de grupos (*clusters*) nos dados, em que os objetos pertenecentes em cada *cluster* compartilham alguma característica ou sejam de alguma maneira similares.

Nesta técnica, não se tem informações a respeito de possíveis classificações dos dados, por isso, existem diversos algoritmos de agrupamento que apresentam diferentes formas de explorar e verificar as estruturas presentes em um conjunto de dados (Faceli et al., 2011).

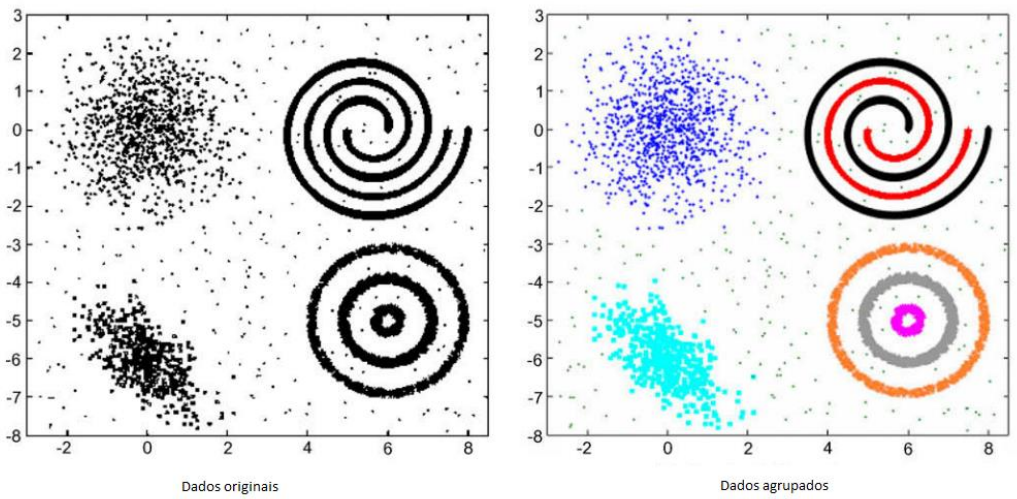

Figura 12 – Agrupamento de dados 7 *clusters*. Fonte: Jain, 2010.

3.4.1. Algoritmo K-means

O algoritmo *k-means* é um método de agrupamento particional que divide os objetos do conjunto de dados em grupos (*clusters*) não sobrepostos. Em

outras palavras, nenhum objeto pode ser membro de mais de um *cluster* e cada *cluster* deve ter pelo menos um objeto.

Para execução, requer que o usuário especifique o número de *clusters*, através do parâmetro ***k***. O algoritmo utiliza de um processo iterativo para atribuir subconjuntos de pontos de dados em ***k*** clusters.

Define-se também o *k-means* como um algoritmo não determinístico, pois pode produzir resultados diferentes em diferentes execuções (inicialização aleatória).

Algumas características desse método, funciona bem quando os clusters têm uma forma esférica, porém, não são adequados para clusters com formas complexas e tamanhos diferentes

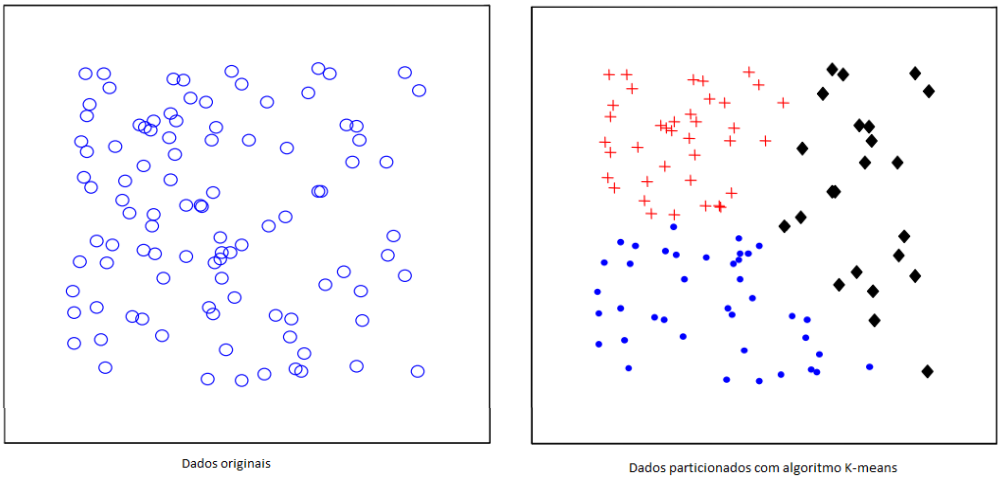

Figura 13 – Agrupamento com algoritmo k-means k=3. Fonte: Jain, 2010.

O processo de agrupamento inicia-se ao selecionar aleatoriamente ***k*** pontos referente ao número de clusters indicados, esses pontos irão determinar o centro de um *cluster*.

Os próximos passos do algoritmo ocorre em duas etapas, também denominada de maximização da expectativa. A etapa de expectativa atribui cada ponto de dados a seu centróide mais próximo. Em seguida, a etapa de maximização, calcula a média de todos os pontos para cada cluster e define o novo centróide.

Através deste processo iterativo, a etapa de maximização da expectativa é repetida até que as posições dos centróides atinjam a convergência e permaneçam inalteradas.

Exemplo: **sklearn.cluster.KMeans**

```
from sklearn.cluster import KMeans

kmeans = KMeans(n_clusters=3)
```

3.5. Otimização

O problema de otimização pode ser definido como um conjunto de processos em que busca minimizar ou maximizar uma determinada função, denominada função objetivo, obtendo assim um melhor aproveitamento dos recursos disponíveis.

3.5.1 Gradiente Descendente

O gradiente descendente é uma estratégia de otimização muito popular utilizado nos processos de ML e no aprendizado profundo. É baseado em uma função convexa, sendo utilizado para encontrar os valores dos parâmetros de uma função (coeficientes) que minimizam uma função de custo tanto quanto possível.

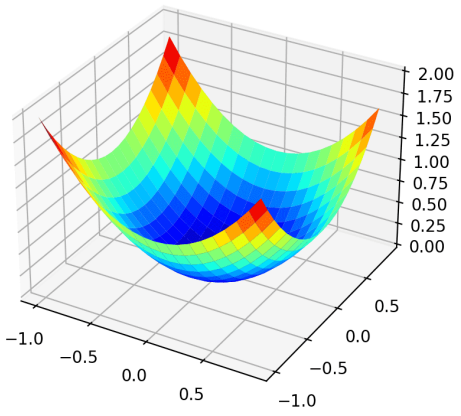

Figura 13 – Gradiente. Fonte: Brownlee, 2021.

Um gradiente mede a mudança em todos os pesos em relação à mudança no erro. Pode-se pensar em um gradiente como a inclinação de uma função. Quanto maior o gradiente, mais acentuada será a inclinação e mais rápido um modelo pode aprender. Mas se a inclinação for zero, o modelo para de aprender. Em termos matemáticos, um gradiente é uma derivada parcial em relação às suas entradas.

O tamanho dos degraus que o gradiente leva na direção do mínimo local é determinado pela taxa de aprendizado, que calcula quão rápido ou lento será o percurso em direção aos pesos ideais. Quando a descida do gradiente não pode mais diminuir a função de custo e permanece mais ou menos no mesmo nível, ela convergiu.

Exemplo: **sklearn.linear_model.SGDRegressor**

```
from sklearn.linear_model import SGDRegressor

gradient_desc = SGDRegressor (max_iter = 20)
```

3.6. Avaliação do Conhecimento 3

Um gerente industrial gostaria de realizar o controle de uma maneira mais eficiente das manutenções preventivas das máquinas e equipamentos da fábrica. Diariamente são coletados dados com a carga de trabalho de cada equipamento, também estão disponíveis os dados com todo histórico de manutenções realizadas nos últimos 3 anos. Dada essas informações, qual método de ML você indicaria para definir em que momento a manutenção devará ocorrer em um determinado equipamento?

[[
Preditivo - Classificação
Preditivo - Regressão Linear
Descritivo - Agrupamento
Otimização - Gradiente
]]

4 Avaliação de modelos de Machine Learning

Em ML diversas técnicas de avaliação são utilizadas para realizar medições de performance e desempenho dos modelos. A análise das informações obtidas na avaliação ajuda na definição de ajustes necessários nos hiperparâmetros e também, na identificação de necessidade de tratamento nos dados.

4.1. Matriz de confusão

Na validação de problemas de classificação, usa-se a matriz de confusão para analisar a proporção de classificações corretas.

As linhas da matriz indicam a classe prevista pelo classificador enquanto os índices da coluna indicam qual a classe o classificador deveria indicar:

Valor previsto \ Valor real	Positivo	Negativo
Positivo	Verdadeiro Positivo	Falso Positivo
Negativo	Falso Negativo	Verdadeiro negativo

Figura 15 – Matriz de Confusão. Fonte: Datarisk.io.

Verdadeiros Positivos (VP): é a quantidade de valores que o modelo previu corretamente para cada classe, ou seja, a quantidade de valores que o modelo previu que realmente pertenciam a classe prevista.

Verdadeiros Negativos (VN): é a quantidade de valores que o modelo previu corretamente para as outras classes, ou seja, a quantidade de valores que o modelo previu que realmente pertenciam a classe as outras classes.

Falsos Positivos (FP): é a quantidade de valores que o modelo previu de forma errônea para cada classe, ou seja, a quantidade de valores que o modelo previu como sendo de uma classe mas que na realidade não pertenciam a ela.

Falsos Negativos (FN): é a quantidade de valores que o modelo previu como sendo de outras classes de forma, ou seja, a quantidade de valores que o modelo previu como sendo de outras classes mas que na realidade não pertenciam a elas.

Exemplo: **sklearn.metrics.confusion_matrix**

```
from sklearn.metrics import confusion_matrix

confusion_matrix(true_label, pred_label)
```

A partir da matriz de confusão é possível calcular outras métricas de interesse de validação do desempenho dos modelos:

- **Acurácia** - indica uma performance geral do modelo. Dentre todas as classificações, quantas o modelo classificou corretamente.

$$acurácia = \frac{VP + VN}{VP + FN + VN + FP}$$

- **Precisão** - dentre todas as classificações Verdadeiros Positivos que o modelo fez, quantas estão corretas.

$$precisão = \frac{VP}{VP + FP}$$

- **Recall ou Revocação** - dentre todas as situações de classe Positivo como valor esperado, quantas estão corretas.

$$sensibilidade = \frac{VP}{VP + FN}$$

- **F1-Score** - média harmônica entre precisão e recall.

$$f1 = 2 * \frac{precisão * sensibilidade}{precisão + sensibilidade}$$

Exemplo: **sklearn.metrics.classification_report**

```
from sklearn.metrics import classification_report
y_true = [0, 1, 2, 2, 2]
y_pred = [0, 0, 2, 2, 1]
target_names = ['classe a', 'classe b', 'classe c']
print(classification_report(y_true, y_pred, target_names=target_names))
```

```
              precision    recall  f1-score   support

    classe a       0.50      1.00      0.67         1
    classe b       0.00      0.00      0.00         1
    classe c       1.00      0.67      0.80         3

    accuracy                           0.60         5
   macro avg       0.50      0.56      0.49         5
weighted avg       0.70      0.60      0.61         5
```

4.2. Rand Index (ajustado)

O índice Rand é utilizado para validação de agrupamento calculando-se a similaridade entre dois *clusters*. Varia no intervalo [-1,1], e valores menores ou próximos a 0 indicam que a semelhança entre as partições se deve ao acaso, o valor 1 indica que as partições são idênticas (Faceli et al., 2011).

Exemplo: **sklearn.metrics.adjusted_rand_score**

```
from sklearn.metrics.cluster import adjusted_rand_score

adjusted_rand_score([0, 0, 1, 1], [0, 0, 1, 1])
```

5 Overfitting e Underfitting

Uma das principais causas do baixo desempenho dos métodos de ML está relacionada aos problemas de *Overfitting* ou *Underfitting* nos dados. Mas antes de entendermos como esses problemas ocorrem, precisamos compreender ou relembrar de alguns conceitos:

- **Generalização**

No aprendizado supervisionado, busca-se construir um modelo a partir dos dados de treinamento que sejam capazes de fazer previsões precisas sobre dados novos que tenham as mesmas características do conjunto de treinamento que usamos.

Se um modelo é capaz de fazer previsões precisas sobre dados que ainda não conhece, podemos dizer que ele é capaz de generalizar a partir do conjunto de dados de treinamento.

- **Ruídos**

O ruído, refere-se à informação irrelevante ou aleatoriedade em um conjunto de dados.

- **Viés e variância**

A incapacidade de um modelo de capturar a verdadeira relação entre variáveis e o objeto a ser predito é o que chamamos de viés ou *bias* em inglês.

Por outro lado, se há um viés muito pequeno o modelo fica tão ajustado aos dados de treinamento que quando é usado com dados diferentes acaba errando muito. Aqui entra o conceito de variância.

É natural pensar que quanto melhor o modelo descreve os dados de treinamento, melhor ele é. Porém, o objetivo primordial não é descrever os dados, mas usá-los para fazer previsões (ou inferências) sobre dados ainda não observados (conhecidos).

- **Overfitting e underfitting**

Obter a complexidade certa de um modelo é um dos principais desafios no desenvolvimento de qualquer tipo de modelo baseado em estatísticas.

Overfitting refere-se a um modelo que modela os dados de treinamento muito bem, tendo um bom desempenho no conjunto de dados de treinamento, mas não em uma amostra de validação com dados ainda não “conhecidos”.

Underfitting refere-se a um modelo que não pode modelar os dados de treinamento nem generalizar para novos dados. Este modelo falha em aprender o problema e tem um desempenho ruim no conjunto de dados de treinamento, e também não tem um bom desempenho em uma amostra de validação.

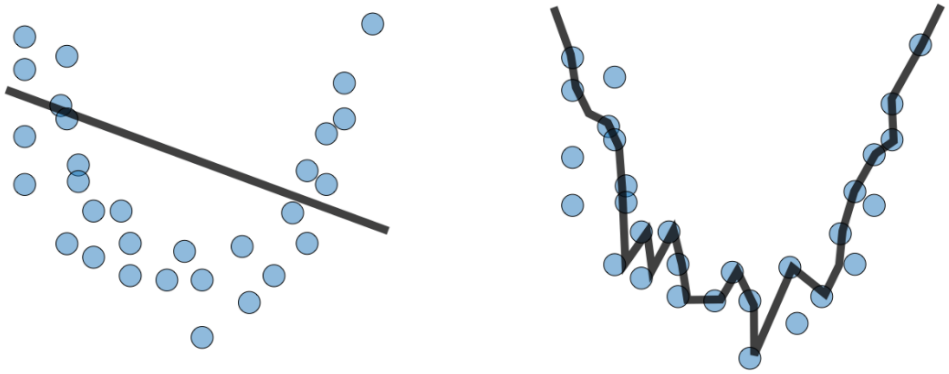

Figura 16 – Underfitting e Overfitting. Fonte: AMIDI, 2018.

Um dos principais desafios do overfitting em ML é que não podemos saber como nosso modelo irá performar com novos dados antes de realmente testá-lo.

Caso o modelo tenha um melhor desempenho no conjunto de treinamento do que no conjunto de validação, provavelmente está apresendando overfitting.

Existem algumas técnica usadas para evitar o overfitting, porém, são abordagens que precisam ser testadas junto ao modelo, pois podem não ser indicadas para todos os casos:

- Aplicar a validação cruzada também chamada de CV (Cross-Validation). Trata-se do particionamento do conjunto de dados em subconjuntos de dados de treinamento, sendo os subconjuntos restantes empregados na validação do modelo.

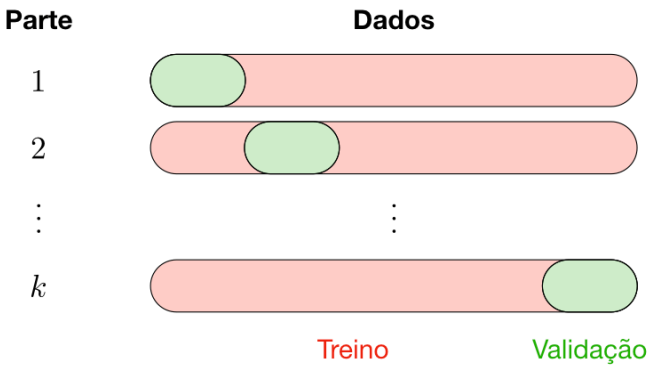

Figura 17 – Validação Cruzada - k-fold. Fonte: AMIDI, 2018.

- Utilizar maior volume de dados de treinamento
- Diminuir a complexidade do modelo, simplificar o modelo reduzindo o número de atributos ou parâmetros
- Tratar os dados retirando ruído dos mesmos

Já os desafios do underfitting podem ser tratados utilizando algumas das ações abaixo:

- Aumentar o tamanho ou o número de parâmetros no modelo
- Aumentar a complexidade ou tipo do modelo
- Aumentar o tempo de treinamento

O intuito é obter o que chamamos de *good fit* ou bom ajuste nos dados, quando um modelo aprende adequadamente com o conjunto de dados de treinamento e consegue uma boa generalização.

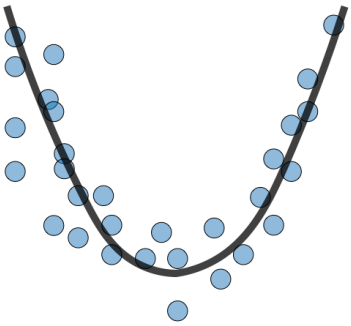

Figura 18 – Modelo com bom ajuste nos dados. Fonte: AMIDI, 2018.

5.1. Avaliação do Conhecimento 4

O modelo não consegue encontrar relações entre as variáveis, o desempenho do modelo já é ruim nos dados de treinamento. Esta afirmação descreve qual cenário?

(a) Outliers
(b) Underfitting
(c) Goodfit
(d) Overfitting

6 Plataformas em nuvem de Machine Learning

Pode-se dizer que a computação em nuvem é a entrega de recursos de Tecnologia da Informação sob demanda por meio da Internet com definição de preço de pagamento conforme o uso. Ao invés de adquirir equipamentos e realizar a sustenção de ambientes com datacenters e servidores físicos, pode-se utilizar os serviços de tecnologia, como capacidade computacional, armazenamento de dados, ferramentas da análise de dados e ML, conforme a necessidade, usando um provedor de nuvem.

Atualmente, os principais provedores de serviços em nuvem são:

- Azure - https://azure.microsoft.com/
- AWS - https://aws.amazon.com/
- Google Cloud – GCP - https://cloud.google.com/

Essas plataformas oferecem serviços de Inteligência Artificial e Machine Learning como serviço (PaaS), recursos computacionais específicos para problemas mais complexos como Deep Learning, Visão Computacional, entre outros, podem ser contratados individualmente e utilizados de acordo com a necessidade.

De acordo com o estudo realizada pela Nucleus Research (2019), dentre os principais benefícios da utilização das plataformas em nuvem, pode-se destacar:

- Redução de custos evitando despesas com hardware, pessoal e energia.
- Capacidade de colaborar e trabalhar em equipes distribuídas. Os modelos implantados na nuvem podem ficar acessíveis a todos os usuários com permissões, independentemente da localização física.
- Capacidade de aproveitar recursos e ferramentas suplementares da plataforma;
- Segurança e disponibilidade da infraestrutura;

- Fluxo de trabalho de ML totalmente gerenciado de ponta a ponta - desde a limpeza dos dados até o treinamento, a criação e a implantação de modelos.

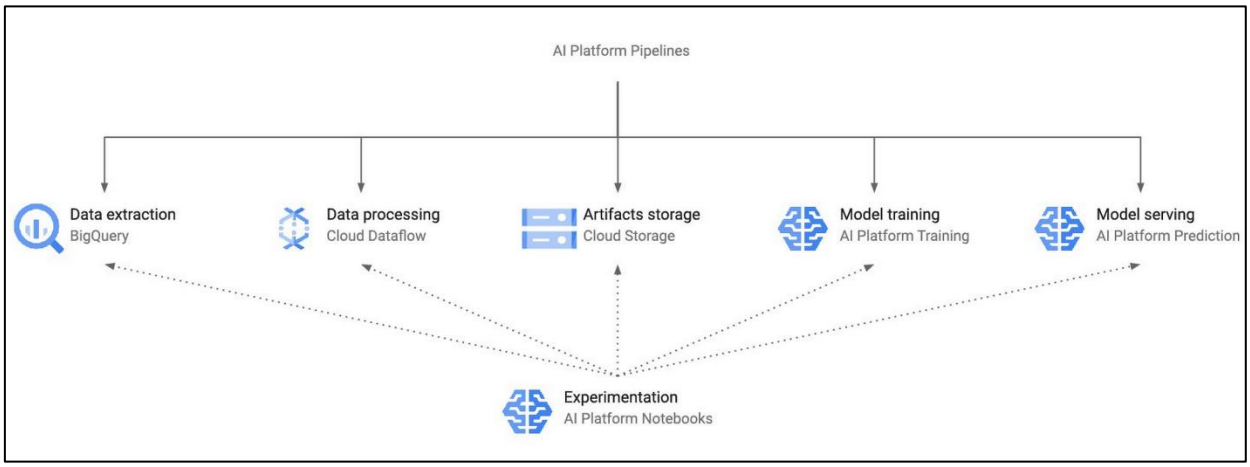

Figura 19 – Exemplo de fluxo de trabalho ML em nuvem. Fonte: GCP.

7 Projeto Final

Nesta atividade avaliativa você deverá demonstrar os conhecimentos adquiridos no curso aplicando-os em uma tarefa de aprendizado supervisionado.

Você deverá utilizar a linguagem Python e bibliotecas como Numpy, Matplotlib, Scikit-learn, entre outras, para realizar as atividades propostas:

Notebook: Projeto Final Machine Learning.ipynb
(https://colab.research.google.com/drive/1OyxYlbEcxKm43ndb8DV5EuvHY-KhZRs7?usp=sharing)

1. Realizar a análise exploratória dos dados e aplicar técnicas de pré-processamento se necessário;
2. Implementar ao menos um modelo de aprendizado supervisionado;
3. Implementar ao menos uma técnica de validação do desempenho do modelo;
4. Realizar análise (dissertativa) sobre o desempenho do modelo escolhido, exemplo: indicar se ocorreu overfitting ou onderfitting, indicar se foi necessário realizar algum pré-processamento nos dados, comentar sobre a técnica de validação utilizada, etc.

Enviar o arquivo notebook (***.ipynb)*** contendo seu projeto final.

Conclusão

Neste curso de Fundamentos em Machine Learning pudemos estudar e conhecer algumas das técnicas de ML nos desafios do aprendizado supervisionado e não-supervisionado.

Inicialmente, exploramos técnicas de pré-processamento de dados, que são de extrema importância para garantir um desempenho dos modelos de ML no desafio de aprendizagem - generalização dados.Implementamos alguns dos algoritmos clássicos da literatura em problemas de classificação, regressão linear e agrupamento dos dados. Por fim, avaliamos o desempenho dos algoritmos de ML utilizando métricas e técnicas de validação.

Em nossas atividades práticas, utilizamos a biblioteca de ML para Python Scikit-learn, recomendo que você explore a documentação oficial online para alavancar ainda mais os seus estudos. Os próximos curso da trilha tratarão de conceitos importantes e atuais de ML, como as redes neurais, visão computacional, entre outras.

Link: https://scikit-learn.org/stable/user_guide.html

Referências

AMIDI, Shervine. Aprendizado de máquina: CS 229. Stanfort University, 2018. Disponível em: https://stanford.edu/~shervine/. Acesso em: 20 out. 2021.

ANIL K., Jain. Data clustering: 50 years beyond K-means. Pattern Recognition Letters. 2010. Disponível em: https://www.sciencedirect.com/science/article/pii/S0167865509002323. Acesso em: 20 out. 2021.

BHA, Deepanshu. K-Nearest Neighbors. ListenData, 2017. Disponível em: https://www.listendata.com/search/label/Machine%20Learning. Acesso em: 20 out. 2021.

DASU, T.; JOHNSON, T. Exploratory data mining and data cleaning. John Wiley & Sons, 2003. v. 479. 14

DUA, Dheeru; GRAFF, Casey. UCI Machine Learning Repository. Irvine, USA: University of California School of Information and Computer Sciences, 2019. Disponível em: http://archive.ics.uci.edu/ml. Acesso em: 20 out. 2021.

FACELI, Katti; LORENA, Ana Carolina; GAMA, João; C. P. L. F. DE CARVALHO, André. Inteligência Artificial: Uma Abordagem de Aprendizado de Máquina. 1ª . ed. Rio de Janeiro: LTC, 2011. ISBN 978-8521618805.

MACHADO, Izabel F.; DRIEMEIER, Larissa. Avaliação do Comportamento Mecânico de Materiais Utilizando uma Abordagem de ML. Escola Politécnica da USP, 2020. Disponível em: https://edisciplinas.usp.br/pluginfile.php/5728638/mod_resource/content/1/Aula 01_Introd_ML.pdf. Acesso em: 20 out. 2021.

MIRKIN, Boris. Core Concepts in Data Analysis: Summarization, Correlation and Visualization. Springer-Verlag London, 2011. ISBN 978-0-85729-287-2.

MOREIRA, José Francisco P. Revisão de modelos de regressão linear. Instituto de Matemática e Estatística: UERJ, 2009. Disponível em: https://professorjf.webs.com/aula%20regress%C3%83%C2%A3o%20linear.pdf. Acesso em: 20 out. 2021.

NG, Andrew. Machine Learning. Stanford University, 2012. Disponível em: https://online.stanford.edu/courses/cs229-machine-learning. Acesso em: 20 out. 2021.

PEDREGOSA et al., SCIKIT-LEARN: Machine Learning in Python, JMLR 12, pp. 2825-2830, 2011.

PETERNELLI, Luiz Alexandre. Regressão linear e correlação. Departamento de Estatística: Universidade Federal de Viçosa, 2009. Disponível em: http://www.dpi.ufv.br/~peternelli/inf162.www.16032004/materiais/CAPITULO9.pdf. Acesso em: 20 out. 2021.

RUSSELL, Stuart; NORVIG, Peter. Artificial Intelligence: A Modern Approach. 4ª. ed. [S. l.]: Pearson Education Limited, 2021. ISBN 978-1292401133.

SHALEV-SHWARTZ , Shai; BEN-DAVID, Shai. Understanding Machine Learning: From Theory to Algorithms. New York, USA: Cambridge University Press, 2014. ISBN 978-1107057135.

SIFUENTES B., Maria Rita. A utilização de algoritmos de aprendizado de máquina em problemas de classificação. ICMC USP, São Carlos. 2019. Disponível em: https://teses.usp.br/teses/disponiveis/55/55137/tde-25032019-141126/publico/MariaRitaSifuentesBatista_revisada.pdf. Acesso em: 20 out. 2021.

CONTROLE DE REVISÃO DO DOCUMENTO / *DOCUMENT REVISION CONTROL*

Revisão	Descrição	Razão	Autor	Data
A	-	Revisão inicial	Larissa Alves	16/11/2021
B	Alteração do código de controle da qualidade de versão B para versão C. E numeração de página.	Atualização	Vitória Souza	24/08/2022
C	Alteração de logotipia do rodapé do documento	Atualização	Jane Piantoni/Vitória Souza	18/01/2023

MCTI FUTURO

FUTURO DO TRABALHO, TRABALHO DO FUTURO

Bom curso

Instituto de Tecnologia

MCTI FUTURO

FUTURO DO TRABALHO, TRABALHO DO FUTURO

Softex

GOVERNO FEDERAL

BRASIL

UNIÃO E RECONSTRUÇÃO