Curso Rápido:

Introdução à Ciência de Dados com Python

Welington Junio

Conteúdo

- Conceitos básicos de **Ciência de Dados.**
- Conceitos e Prática.
  1. **Preparação e Manipulação dos Dados.**
  2. **Visualização dos Dados.**
  3. **Predição.**

Ferramenta: Google Colab + Python.

Ciência de Dados

- Área que combina **estatística**, **programação** e **conhecimento de domínio** para extrair perspectivas a partir de dados e utilizá-los para **tomada de decisão** e **previsão.**

Ciência de Dados

Objetivos:

- Transformar dados brutos em informações úteis e acionáveis.
- Perspectivas e visões valiosas para empresas ou órgãos.
- Tomada de decisão, reduzir custos.
- Resolver problemas mundiais e locais.
- Previsões, contenção.

Ciência de Dados

Aplicações:

- **Negócios:** Precificação dinâmica, análise de clientes, detecção de fraudes.
- **Saúde:** Diagnóstico médico assistido por IA, análise de prontuários, otimização de processos hospitalares.
- **Finanças:** Algoritmos de investimento, detecção de anomalias em transações.
- **Marketing:** Recomendação de produtos, segmentação de clientes.
- **Esportes:** Análise de desempenho de jogadores, otimização de táticas.

Conteúdo +

Atividade Prática

Diabets Dataset

1. Preparação e Manipulação dos Dados

- Processo de extração, limpeza e transformação - procurar, ajustar, organizar e modificar dados - de dados brutos em um formato adequado para análise.

- Dados limpos e bem preparados garantem análises precisas, confiáveis, reduz tempo de esforço(manual e computacional) e a modelagem tem o desempenho otimizado.

1. Preparação e Manipulação dos Dados

COLETAR DADOS

Buscar dados de fontes diversas (bancos de dados, APIs, arquivos, web scraping).

↓

PROCESSAR DADOS

Limpeza, transformação e organização dos dados.

↓

ANÁLISE DESCRITIVA

Uso de descrições para entender características e formatações dos dados.

1. Preparação e Manipulação dos Dados

- Facilita a interpretação visual de gráficos.
- Minimiza a possibilidade de conclusões equivocadas devido a dados incorretos.
- Modelos treinados com dados de qualidade apresentam resultados superiores.

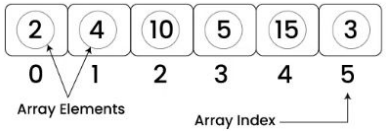
1. Preparação e Manipulação dos Dados

Antes de vermos as bibliotecas, alguns conceitos:

- CSV - Formato de armazenamento de arquivo separados por vírgula

```
Date,Country,Units,Revenue
2019-01-08,USA,343,15461.36
2019-01-04,Panama,93,4681.26
2019-01-07,Panama,42,2220.36
2019-01-16,Brazil,103,1853.78
2019-01-17,USA,28,286.3
2019-01-24,Canada,372,24826.98
2019-01-26,Canada,61,1592.42
2019-01-28,Canada,264,3228.11
2019-01-13,Canada,27,257.97
2019-01-28,Brazil,323,3024.25
```

- Array -> Vetor - Valores em uma única variável



- DF -> Dataframe - Estrutura bidimensional para trabalhar com os dados

	country	continent	year	lifeExp	pop	gdpPercap
0	Afghanistan	Asia	1952	28.801	8425333	779.445314
1	Afghanistan	Asia	1957	30.332	9240934	820.853030
2	Afghanistan	Asia	1962	31.997	10267083	853.100710
3	Afghanistan	Asia	1967	34.020	11537966	836.197138
4	Afghanistan	Asia	1972	36.088	13079460	739.981106

1. Preparação e Manipulação dos Dados

Biblioteca	Função	Descrição
Pandas	read_csv()	Carrega arquivos CSV para um DF.
	head()	Exibe as primeiras linhas de um DF.
	groupby()	Agrupa dados para operações de agregação.
NumPy	np.array()	Cria arrays multidimensionais.
	np.mean()	Calcula a média dos elementos de um array.
	np.arange()	Gera sequências numéricas com intervalos definidos.

1. Preparação e Manipulação dos Dados - Prática

Dataset: Diabetes - Informações sobre pacientes mulheres que possuem ou não diabetes.

```
df = pd.read_csv("diabetes.csv")
print("Cabeçalho do Dataset:")
df.head()
```

```
Cabeçalho do Dataset:
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

1. Preparação e Manipulação dos Dados - Prática

```
print("\nInformações do Dataset:")
df.info()
print("\nEstatísticas Descritivas:")
df.describe()
```

```
Informações do Dataset:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

- 768 Registros.
- Colunas detalhas.
- Tipos dos registros.

1. Preparação e Manipulação dos Dados - Prática

count	mean	std	min	25%	50%	75%	max
contagem	média	desvio padrão	menor valor	primeiro quartil	mediana	terceiro quartil	maior valor

Estatísticas Descritivas:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

2. Visualização dos Dados

- É a representação gráfica de informações e dados. Ela ajuda a transformar números e fatos em uma forma visual que facilita a compreensão.
- Auxilia na identificação de padrões, tendências e outliers que não são facilmente perceptíveis em dados brutos.
- Usar gráficos para entender o comportamento dos dados antes de qualquer modelagem.

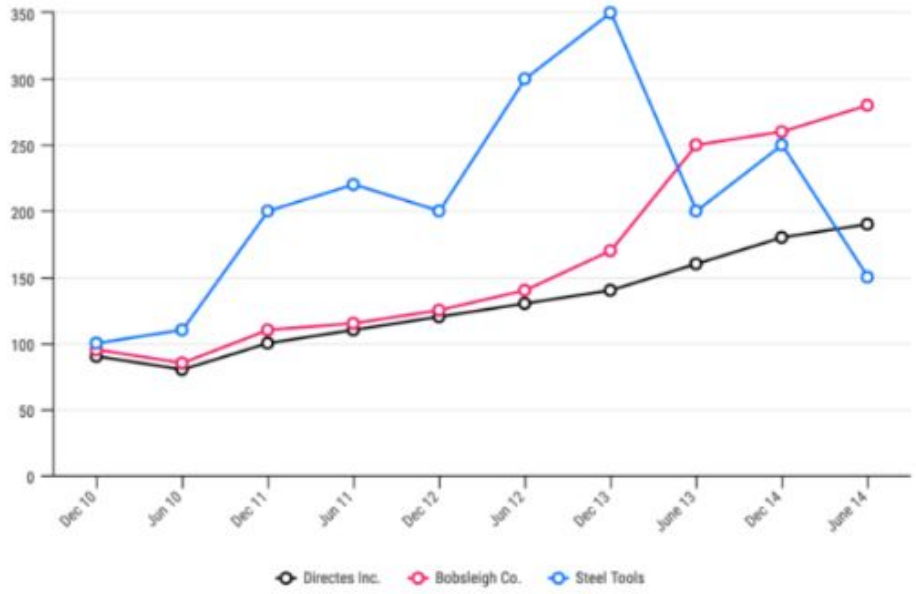
2. Visualização dos Dados

Biblioteca	Função	Descrição
Seaborn	histplot()	Plota histogramas.
	scatterplot()	Cria gráficos de dispersão para análise visual.
	heatmap()	Exibe matrizes de dados com mapeamento de cores.
Matplotlib	plt.plot()	Cria gráficos de linha.
	plt.scatter()	Plota pontos em um gráfico de dispersão.
	plt.show()	Renderiza e exibe o gráfico na tela.

2. Visualização dos Dados

Alguns Tipos de Gráficos

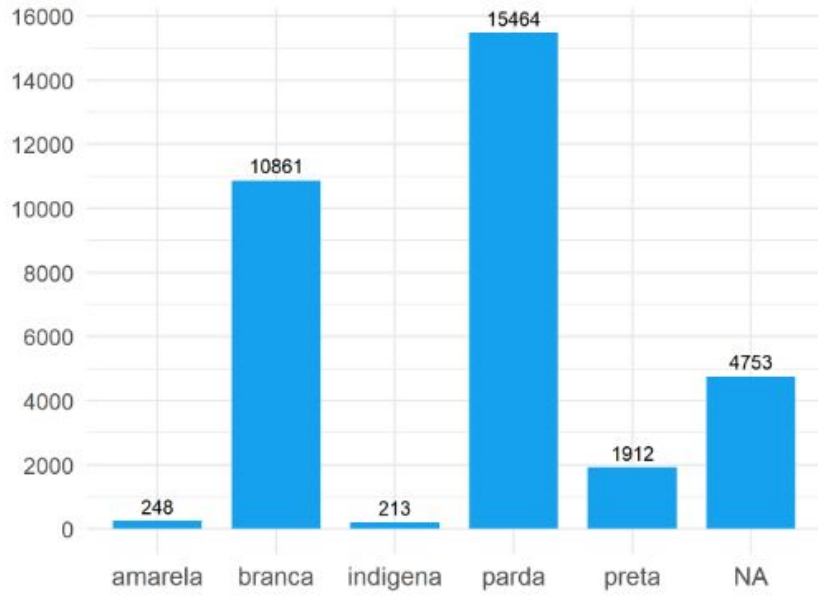
- **Gráfico de Linhas:** Ideal para mostrar tendências ao longo do tempo.



2. Visualização dos Dados

Alguns Tipos de Gráficos

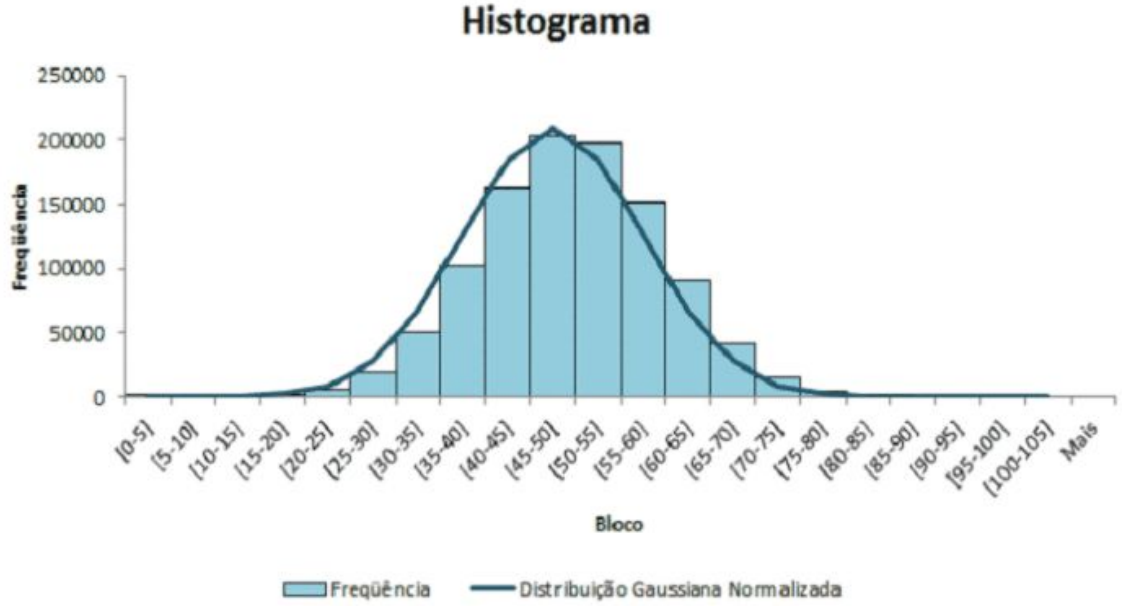
- **Gráfico de Barras:** Utilizado para comparar categorias ou grupos.



2. Visualização dos Dados

Alguns Tipos de Gráficos

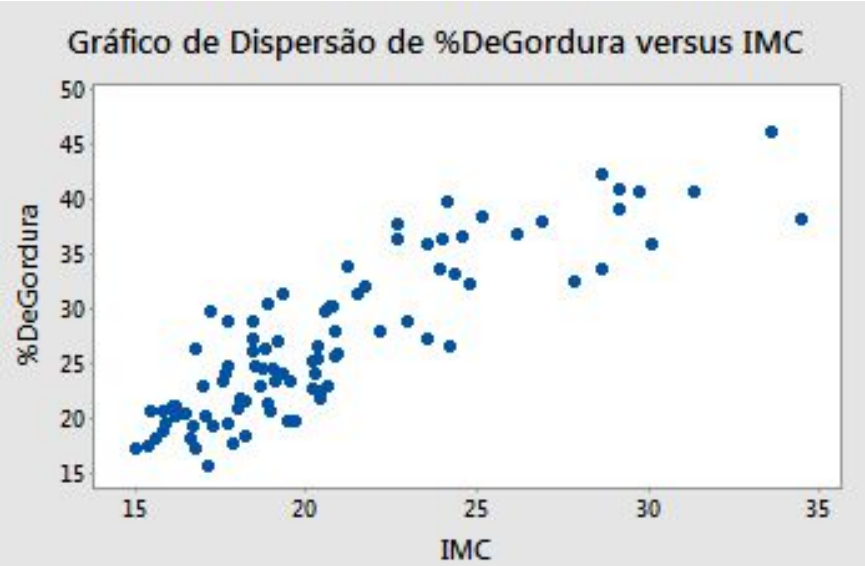
- **Histograma:** Mostra a distribuição de uma variável contínua.



2. Visualização dos Dados

Alguns Tipos de Gráficos

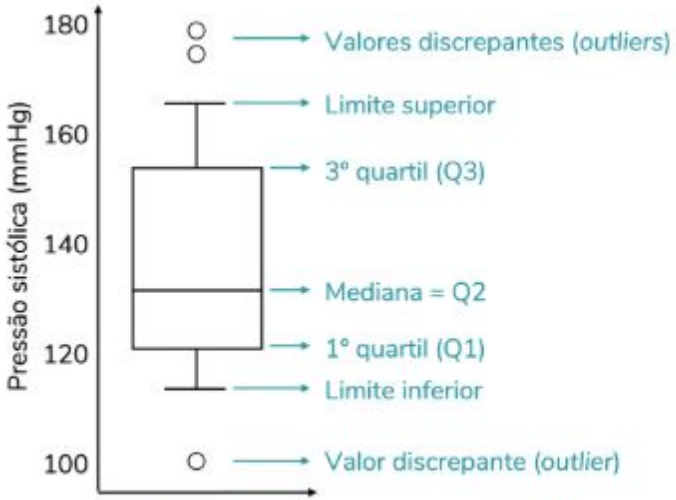
- **Gráfico de Dispersão:** Exibe a relação entre duas variáveis numéricas.



2. Visualização dos Dados

Alguns Tipos de Gráficos

- **Boxplot:** Resume a distribuição de uma variável, destacando medianas e outliers.



Elementos que compõem um boxplot.

2. Visualização dos Dados

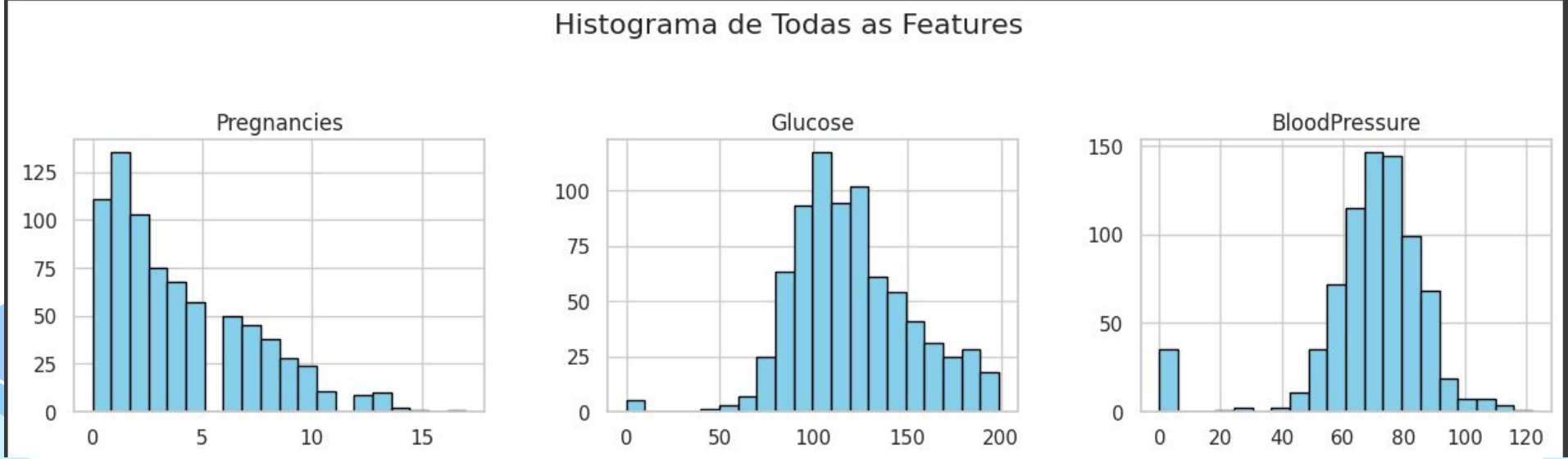
2. Visualização dos Dados

Melhores Práticas na Visualização de Dados

- **Clareza e Simplicidade:** Evitar excessos que possam confundir o público.
- **Escolha Adequada de Cores:** Utilizar paletas que sejam acessíveis e que realcem as informações importantes.
- **Contextualização:** Incluir títulos, legendas e rótulos que tornem o gráfico autoexplicativo.

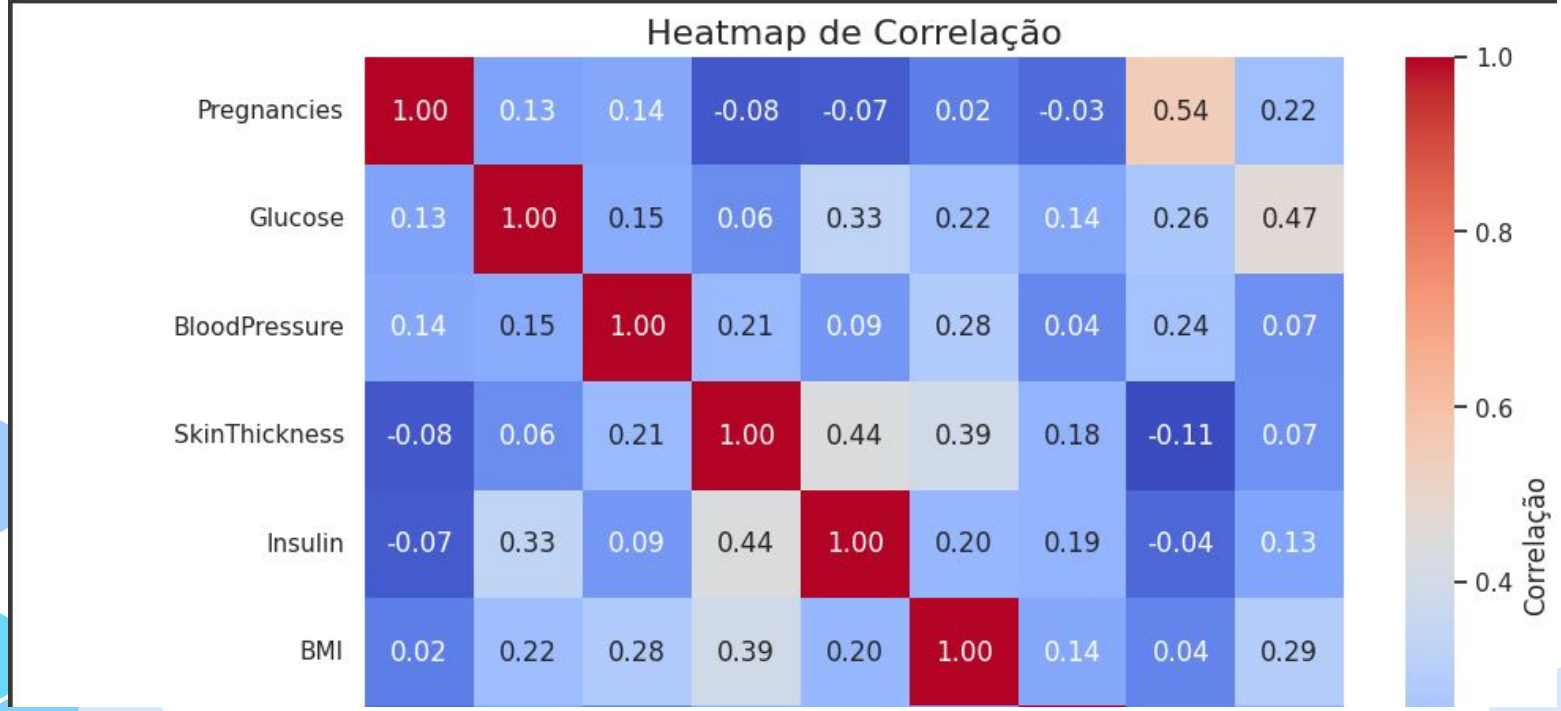
2. Visualização dos Dados - Prática

```
df.hist(bins=20, figsize=(15,10), color='skyblue', edgecolor='black')
plt.suptitle("Histograma de Todas as Features", fontsize=16)
plt.xlabel("Valor", fontsize=12)
plt.ylabel("Frequência", fontsize=12)
plt.show()
```



2. Visualização dos Dados - Prática

```
corr = df.corr()
plt.figure(figsize=(10,8))
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt=".2f", cbar_kws={'label': 'Correlação'})
plt.title("Heatmap de Correlação", fontsize=16)
plt.show()
```



3. Predição

- **Predição**: refere-se ao uso de **modelos estatísticos** ou de **machine learning** para prever resultados futuros com base em dados históricos.

- Permite antecipar eventos e tomar decisões informadas.

- Aplicada em diversos contextos: saúde, finanças, marketing, etc.

3. Predição

Machine Learning

- É um campo da inteligência artificial que permite que sistemas aprendam a partir de dados, identifiquem padrões e façam previsões ou decisões sem serem explicitamente programados para isso.

3. Predição

Tipo: Aprendizado Supervisionado

- Tipo de aprendizado de máquina onde o modelo é treinado com um conjunto de **dados rotulado**.
- **Dados rotulado**: Contém exemplos com as características de entrada e o rótulo associado.
- O modelo aprende a partir dados de exemplos e tenta generalizar para fazer previsões sobre dados novos, ainda não vistos.

3. Predição

Tipos:

- **Classificação**: Quando o rótulo a ser predito é categórico (ex: "diabetes sim" ou "diabetes não").

- **Regressão**: Quando o rótulo a ser predito é numérico (ex: prever a temperatura ou preço de uma casa).

Trabalharemos com a Classificação

3. Predição

K-Nearest Neighbors (KNN)

- Algoritmo de aprendizado supervisionado utilizado para classificação (e, em alguns casos, regressão).

- Baseia-se em uma ideia simples: para classificar ou prever uma nova amostra, o algoritmo verifica os **K** vizinhos mais próximos no conjunto de treinamento e atribui a classe ou o valor mais comum entre esses **vizinhos**.

3. Predição

Como funciona o **K**:

K: É o número de vizinhos mais próximos que serão considerados para determinar a classe ou valor de um novo dado.

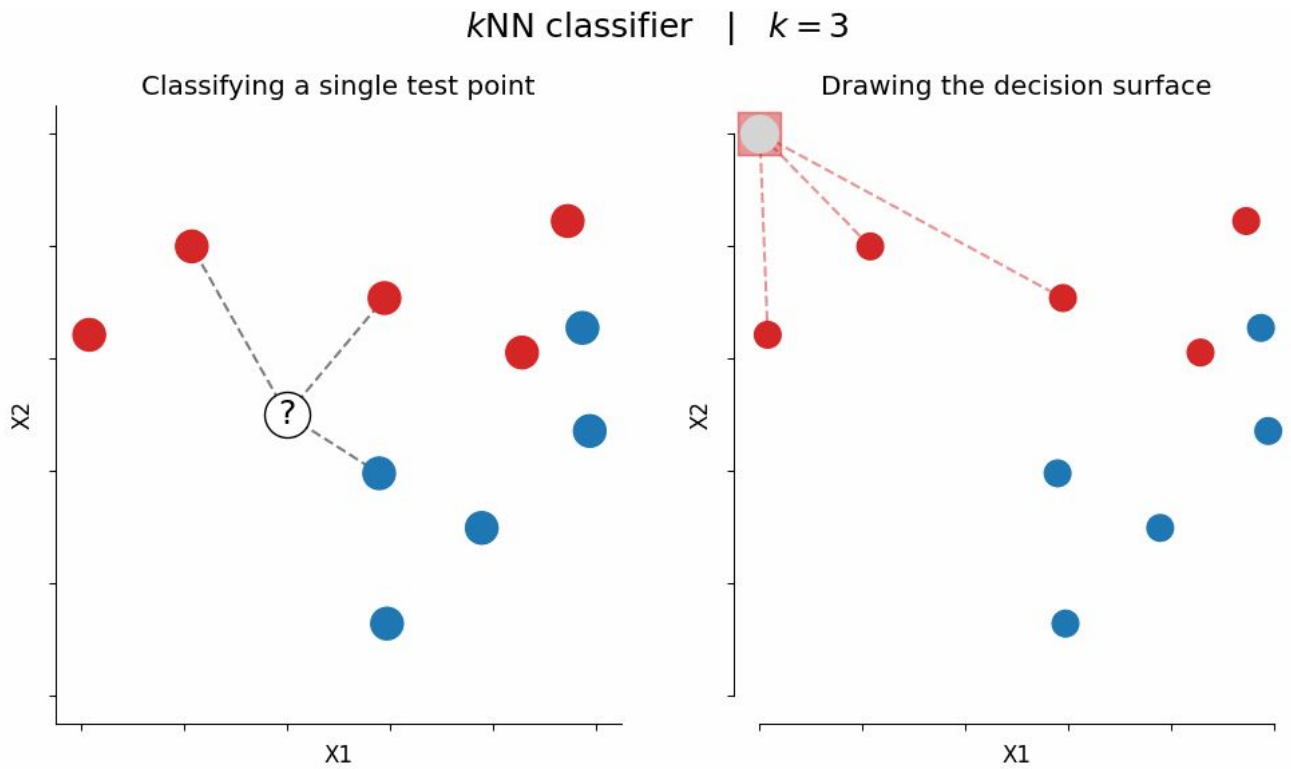
- Se K = 1, o modelo apenas olha o vizinho mais próximo e assume a sua classe.
- Se K = 3, o modelo olha os 3 vizinhos mais próximos e decide com base na maioria.
- Se K = 5, ele olha para os 5 vizinhos mais próximos, e assim por diante.

3. Predição

Como funciona o KNN:

1. **Escolher o valor de K:** O número de vizinhos a serem considerados.
2. **Calcular a distância:** Para cada ponto de entrada (exemplo não rotulado), o KNN calcula a distância até todos os outros pontos no conjunto de dados. A distância mais comum utilizada é a distância **Euclidiana**.
3. **Selecionar os K vizinhos mais próximos:** A partir do cálculo das distâncias, o algoritmo seleciona os **K** pontos mais próximos.
4. **Classificar:** O **KNN** retorna a classe mais comum entre os K vizinhos.
    - Se a maioria dos K vizinhos tiver o rótulo "Diabetes", o modelo prediz "Diabetes" para esse ponto de dados.

3. Predição


kNN classifier | k = 3
Classifying a single test point
Drawing the decision surface
X2
X1
?
X2
X1

3. Predição - Prática

```
# Dividindo em variáveis independentes (X) e dependente (y)
X = df.drop(columns=["Outcome"])  # Variáveis independentes
y = df["Outcome"]  # Variável dependente

# Dividindo os dados em treino e teste
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Normalizando os dados (necessário para o KNN)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Treinando o modelo KNN
knn = KNeighborsClassifier(n_neighbors=5)  # Usando K=5 como exemplo
knn.fit(X_train_scaled, y_train)

# Fazendo previsões
y_pred = knn.predict(X_test_scaled)
```

- Separação dos dados em conjunto de treinamento e teste (ex.: 70% treino, 30% teste).
- Uso de StandardScaler do scikit-learn para normalizar dados.
- Treinamento do KNN.

3. Predição - Prática

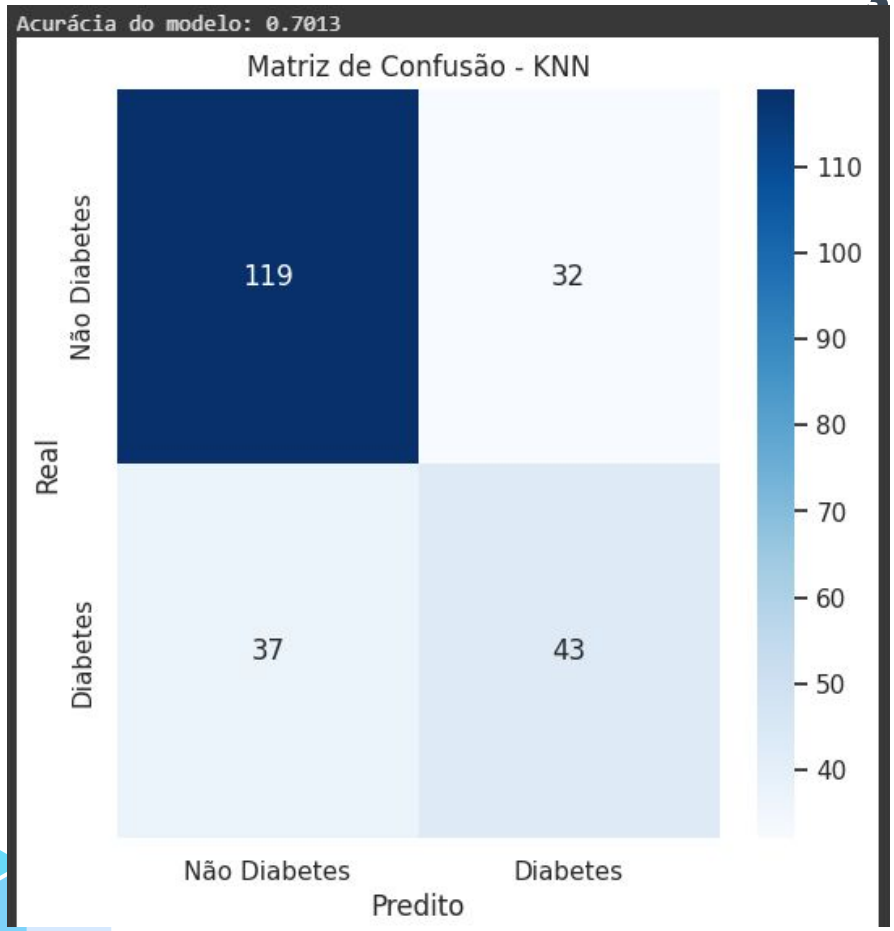
```python
# Avaliando o modelo: Acurácia
accuracy = accuracy_score(y_test, y_pred)
print(f"Acurácia do modelo: {accuracy:.4f}")

# Matriz de Confusão
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6,6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=["Não Diabetes", "Diabetes"], yticklabels=["Não Diabetes", "Diabetes"])
plt.title("Matriz de Confusão - KNN")
plt.xlabel("Predito")
plt.ylabel("Real")
plt.show()

# Exibindo a acurácia e matriz de confusão
print("Matriz de Confusão:")
print(cm)
```

Geração da Matriz de Confusão.

3. Predição



- Acurácia e Matriz de Confusão plotada.

3. Predição

Previsão

```
# Criar o vetor de entrada com esses valores (os valores devem estar na ordem correta)
entrada = [[pregnancies, glucose, blood_pressure, skin_thickness, insulin, bmi, diabetes_pedigree_function, age]]

# Normalizar a entrada (para ficar no mesmo padrão das variáveis de treino)
entrada_normalizada = scaler.transform(entrada)

# Fazer a previsão
previsao = knn.predict(entrada_normalizada)

# Mostrar o resultado
if previsao == 0:
    print("A previsão é: Não tem diabetes.")
else:
    print("A previsão é: Tem diabetes.")
```

- Prevendo a possibilidade de uma certa mulher, ter diabetes:

```
Digite o valor de Pregnancies: 2
Digite o valor de Glucose: 123
Digite o valor de BloodPressure: 120
Digite o valor de SkinThickness: 140
Digite o valor de Insulin: 23
Digite o valor de BMI: 123
Digite o valor de DiabetesPedigreeFunction: 12
Digite o valor de Age: 21
A previsão é: Tem diabetes.
```

Exercícios

Acesse o código do Google Colab e faça uma cópia (Caso queira salvar):

https://bit.ly/41nKMLj

Execute e agora resolva os seguintes exercícios:

1. **Parte 1 - Preparação e Manipulação de Dados**. Pesquise e gere novas estatísticas descritivas.
   Exemplo: Liste pacientes sem diabetes, e uma média agrupada por outra coluna (poucos valores e mais exatos).

2. **Parte 2 - Visualização dos Dados**. Pesquise e gere novos gráficos de sua preferência.
   Exemplo: Gráfico de pizza de outcome e gráfico de dispersão entre idade e gravidezes.

3. **Parte 3 - Predição.** Verificar se a diminuição ou aumento do valor k leva a uma melhora ou piora no modelo, com base em acurácia e matriz de confusão. Interprete os resultados.
   Exemplo: Para k = 3 e k = 10.

Obrigado