

SEGREDOS DA ANÁLISE DE DADOS COM PYTHON

DOMINE A FORÇA DOS DADOS



Aprenda quais são alguns comandos importantes e muito utilizados na hora de construir as análises dos dados.

ADRIANO SANTOS



Segredos da Análise de Dados

Dominando os Dados com a Linguagem Python

A análise de dados com Python é uma habilidade essencial no mundo da programação e da ciência de dados. Este ebook revela alguns dos segredos mais valiosos para dominar essa arte, sempre com exemplos práticos e reais.



01

Importe as Bibliotecas Certas

Para começar qualquer análise de dados em Python, é crucial importar as bibliotecas certas. As mais comuns são `pandas`, `numpy` e `matplotlib`..



Importe as Bibliotecas Certas

Dominando os Dados com a Linguagem Python

No código a seguir, está sendo importado as bibliotecas **Pandas** como **pd**, **NumPy** como **np** e **Matplotlib.pyplot** como **plt**. Essas bibliotecas são amplamente utilizadas para manipulação de dados, operações numéricas e visualização gráfica em Python, respectivamente.



Importando as bibliotecas necessárias

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```



02

Carregue e Explore os Dados

Carregar os dados corretamente é o primeiro passo para qualquer análise. Use pandas para ler arquivos CSV e explorar o dataset.



Carregue e Explore os Dados

Dominando os Dados com a Linguagem Python

No código a seguir, está sendo utilizada a biblioteca `pandas` (importada como `pd`) para ler um arquivo CSV chamado `'dataset.csv'` através da função `read_csv`, armazenando os dados na variável `dados`. Em seguida, a função `display` exibe as primeiras 10 linhas do conjunto de dados usando o método `head(10)`, que é aplicado à variável `dados`.



Carregando e visualizando os Dados

```
dados = pd.read_csv('dataset.csv')  
display(dados.head(10))
```



03

Limpeza de Dados

Dados sujos podem comprometer a análise.
Remova valores nulos e trate dados
inconsistentes.



Limpeza de Dados

Dominando os Dados com a Linguagem Python

No código a seguir, está sendo utilizado o método **dropna()** para remover linhas com valores nulos do **DataFrame dados**. Em seguida, valores faltantes na coluna 'Quantidade' são substituídos pela média dessa coluna usando o método **fillna()** com o parâmetro **mean()**. Por fim, as primeiras dez linhas do dataset são exibidas com o método **head(10)** e a **função display()**

Limpando e tratando os dados

```
dados = dados.dropna()  
dados['Quantidade'] = dados['Quantidade']  
                        .fillna(dados['Quantidade']  
                        .mean())  
display(dados.head(10))
```



04

Análise Exploratória de Dados (EDA)

Entenda melhor os dados com análise exploratória. Use descrições estatísticas e visualizações.



Análise Exploratória de Dados (EDA)

Dominando os Dados com a Linguagem Python

No código a seguir, está sendo realizada uma análise estatística e visualização de dados. Primeiramente, a função `display` é utilizada para exibir o resumo estatístico do DataFrame `dados` através do método `describe()`, que fornece estatísticas descritivas como média, mediana, desvio padrão, entre outros. Em seguida, utilizando a biblioteca `matplotlib`, é criada uma visualização da distribuição da coluna `Quantidade` do DataFrame. A função `plt.hist` gera um histograma desta coluna e, por fim, `plt.show()` exibe o gráfico resultante.



Análise Exploratória de Dados (EDA)

```
display(dados.describe())  
plt.hist(dados['Quantidade'])  
plt.show()
```



05

Manipulação de Dados

Transforme e manipule os dados para
prepará-los para a análise.



Manipulação de Dados

Dominando os Dados com a Linguagem Python

No código a seguir, está sendo utilizada a função `np.where` da biblioteca **NumPy** para criar uma nova coluna chamada `'nova_coluna'` no **dataframe** `'dados'`, onde os valores são atribuídos como `'Alto'` se a `'Quantidade'` for maior que 10, caso contrário, `'Baixo'`. Em seguida, os dados são filtrados para incluir apenas as linhas onde a `'Quantidade'` é maior que 50, resultando no **dataframe** `'dados filtrados'`. Finalmente, as primeiras 10 linhas do **dataframe** `'dados'` são exibidas utilizando a função `display`.

```
Manipulando os Dados

dados['nova_coluna'] = np.where(
    dados['Quantidade'] > 10, 'Alto', 'Baixo'
)
dados_filtrados = dados[dados['Quantidade'] > 50]
display(dados.head(10))
```



06

Visualização de Dados

Visualize os dados para extrair insights valiosos.



Visualização de Dados

Dominando os Dados com a Linguagem Python

No código a seguir, está sendo utilizado a biblioteca `matplotlib.pyplot` para criar dois tipos de gráficos. Primeiro, um gráfico de dispersão é gerado usando a função `plt.scatter()`, que recebe os parâmetros `dados['Valor_Venda']` e `dados['Quantidade']`, seguidos pela customização dos eixos com `plt.xlabel()` e `plt.ylabel()`, e o título do gráfico com `plt.title()`. Em seguida, `plt.show()` exibe o gráfico. O segundo gráfico é um gráfico de barras gerado a partir do método `value_counts()` aplicado à coluna `Categoria` do DataFrame `dados`, que é então plotado com a função `plot(kind='bar')`. O título é definido com `plt.title()`, e `plt.show()` é novamente utilizado para exibir o gráfico.

```
Manipulando os Dados

# Gráfico de dispersão
plt.scatter(dados['Valor_Venda'], dados['Quantidade'])
plt.xlabel('Valor_Venda 1')
plt.ylabel('Quantidade 2')
plt.title('Gráfico de Dispersão')
plt.show()

# Gráfico de barras
dados['Categoria'].value_counts().plot(kind='bar')
plt.title('Distribuição de Valores')
plt.show()
```

07

Modelagem e Predição

Use bibliotecas como **scikit-learn** para criar
modelos preditivos.



Modelagem e Predição

Dominando os Dados com a Linguagem Python

No código a seguir, está sendo usada a biblioteca **scikit-learn** para realizar uma regressão linear. Primeiramente, a função **train_test_split** é utilizada para dividir os dados em conjuntos de treino e teste, com 20% dos dados reservados para teste. Em seguida, a classe **LinearRegression** é instanciada e o método **fit** é chamado para treinar o modelo com os dados de treino. Finalmente, o método **predict** faz previsões com o conjunto de teste e as previsões são impressas na tela..

```
Importando outras Bibliotecas Necessárias

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

# Separar dados em treino e teste
X = dados[['Valor_Venda']]
y = dados['Quantidade']
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

# Treinar um modelo de regressão linear
modelo = LinearRegression()
modelo.fit(X_train, y_train)

# Fazer previsões
previsoes = modelo.predict(X_test)
print(previsoes)
```

08

Conclusão



Conclusão

Dominando os Dados com a Linguagem Python

Dominar a análise de dados com Python envolve conhecer as bibliotecas certas, carregar e limpar os dados adequadamente, explorar e visualizar dados, e criar modelos preditivos. Com os exemplos práticos fornecidos, você está pronto para começar a aplicar esses segredos em seus próprios projetos de análise de dados.



AGRADECIMENTOS



OBRIGADO POR LER ATÉ AQUI



Este livro eletrônico foi criado usando inteligência artificial e formatado por um ser humano. As instruções detalhadas estão disponíveis no meu GitHub.

Este material foi desenvolvido com propósitos educativos e não passou por uma revisão cuidadosa feita por humanos, podendo conter erros originados de uma inteligência artificial.



<https://github.com/Adriano1976/python-data-analysis-secrets>



Autor



Adriano Santos

[GitHub](#) | [LinkedIn](#) | [Instagram](#) |



