UNIVERSIDADE REGIONAL INTEGRADA DO ALTO URUGUAI E DAS MISSÕES CAMPUS DE ERECHIM
DEPARTAMENTO DAS ENGENHARIAS E CIÊNCIA DA COMPUTAÇÃO

RICARDO BALEN BAIOCCO

SOFTWARE DE MINERAÇÃO DE DADOS PARA ANALISAR CONJUNTO DE DADOS COM DATASET TITANIC

ERECHIM - RS
2024

1. Introdução

A pesquisa e implementação do trabalho se deu na escolha da linguagem Python, utilizando um dataset do Titanic encontrado no Kaggle. Tema escolhido para se desafiar a conhecer esta nova linguagem e que ainda não havia estudado e tive interesse.

2. Bibliotecas utilizadas

- pandas para manipular os dados;
- train_test_split para divisão dos dados;
- RandomForestClassifier para classificação;
- accuracy_score para avaliação do modelo;
- KMeans para agrupamento;
- LinearRegression para regressão;
- StandardScaler para normalização;
- matplotlib e seaborn para visualização dos dados.

3. Técnicas

Classificação: é uma técnica utilizada com objetivo de prever rótulos categóricos como sobrevivência no nosso DataSet do Titanic, onde mostra os sobreviventes e não sobreviventes. É usada quando se tem rótulos conhecidos e deseja prever valores novos.

Agrupamento: como o próprio nome já diz responsável por agrupar os dados que não são rotulados nos clusters com base nas características similares. Usado pra explorar os padrões definidos nos dados sem supervisão, dividir os grupos dos passageiros com base na idade e tarifa.

Regressão: objetivo é adivinhar um valor numérico ou categórico e é usada quando a variável dependente é contínua ou categórica.

4. Implementação

Código aplicado no arquivo main.py

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.cluster import KMeans
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
import seaborn as sns
```

Importando as Bibliotecas usadas.

```python
# Função para carregar o dataset
def load_data(file_path):  1 usage
    data = pd.read_csv(file_path)
    return data

# Carregar o dataset Titanic
DATA_FILE = 'data/Titanic-Dataset.csv'
data = load_data(DATA_FILE)
```

Função para carregar e diretório do arquivo do DataSet.

```python
# Limpeza e preparação dos dados
data = data.drop(['Name', 'Ticket', 'Cabin'], axis=1)
data['Age'].fillna(data['Age'].mean(), inplace=True)
data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)

# Codificar variáveis categóricas
data = pd.get_dummies(data, columns=['Sex'], drop_first=True)  #
data['Embarked'] = data['Embarked'].apply(lambda x: 1 if x == 'S' else 0)
```

Inicia um pré-processamento, ajustando as colunas irrelevantes removendo as mesmas, preenche valores ausentes na coluna e transforma as categorias em valores numéricos, possibilitando a interpretação posterior.

```python
# Variáveis independentes e dependente
X = data[['Pclass', 'Age', 'SibSp', 'Sex_male', 'Embarked']]
y = data['Survived']

# Dividir os dados em conjuntos de treino e teste
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Separa os dados definindo “x “para representar variáveis independentes e “y” como variável alvo que indica se o passageiro sobreviveu ou não.

CLASSIFICAÇÃO

```
# Classificação com Floresta Aleatória
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# Fazer previsões
predictions = model.predict(X_test)
```

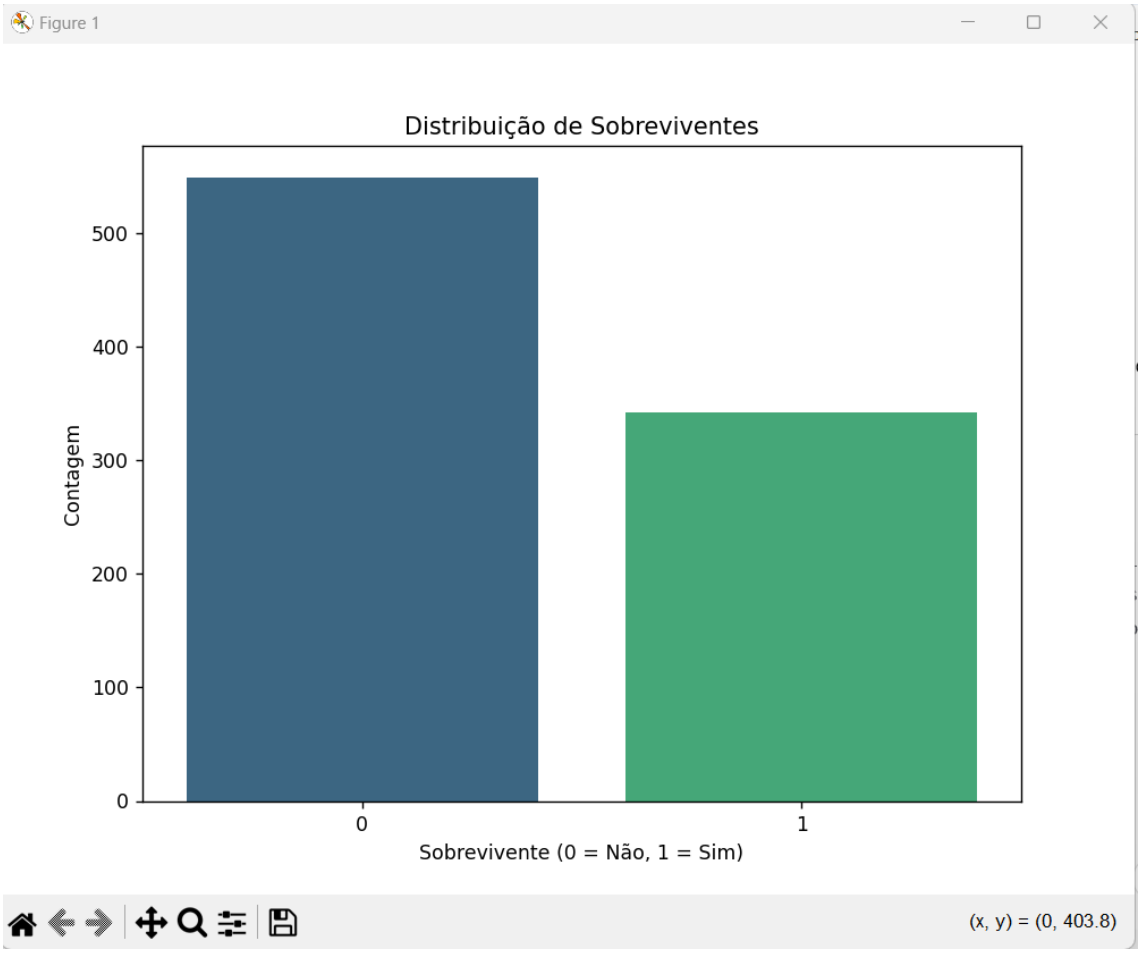
Implementação da técnica de Classificação, utilizando várias árvores de decisão pra fazer a variação. Além disso implementado um model com dados de teste para fazer previsões, armazenando em predictions.

```
# Avaliar o modelo
accuracy = accuracy_score(y_test, predictions)
print(f'Acurácia do modelo de classificação: {accuracy:.2f}')
```

Calcula a precisão do modelo imprimindo o valor.

```
# Visualização da distribuição de sobreviventes
plt.figure(figsize=(8, 6))
sns.countplot(x='Survived', data=data, palette='viridis')
plt.title('Distribuição de Sobreviventes')
plt.xlabel('Sobrevivente (0 = Não, 1 = Sim)')
plt.ylabel('Contagem')
plt.show()
```

Montagem e impressão do gráfico.



Impressão do gráfico mostrando resultados da Técnica Classificação.

AGRUPAMENTO

```
# Normalizar os dados para o agrupamento
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

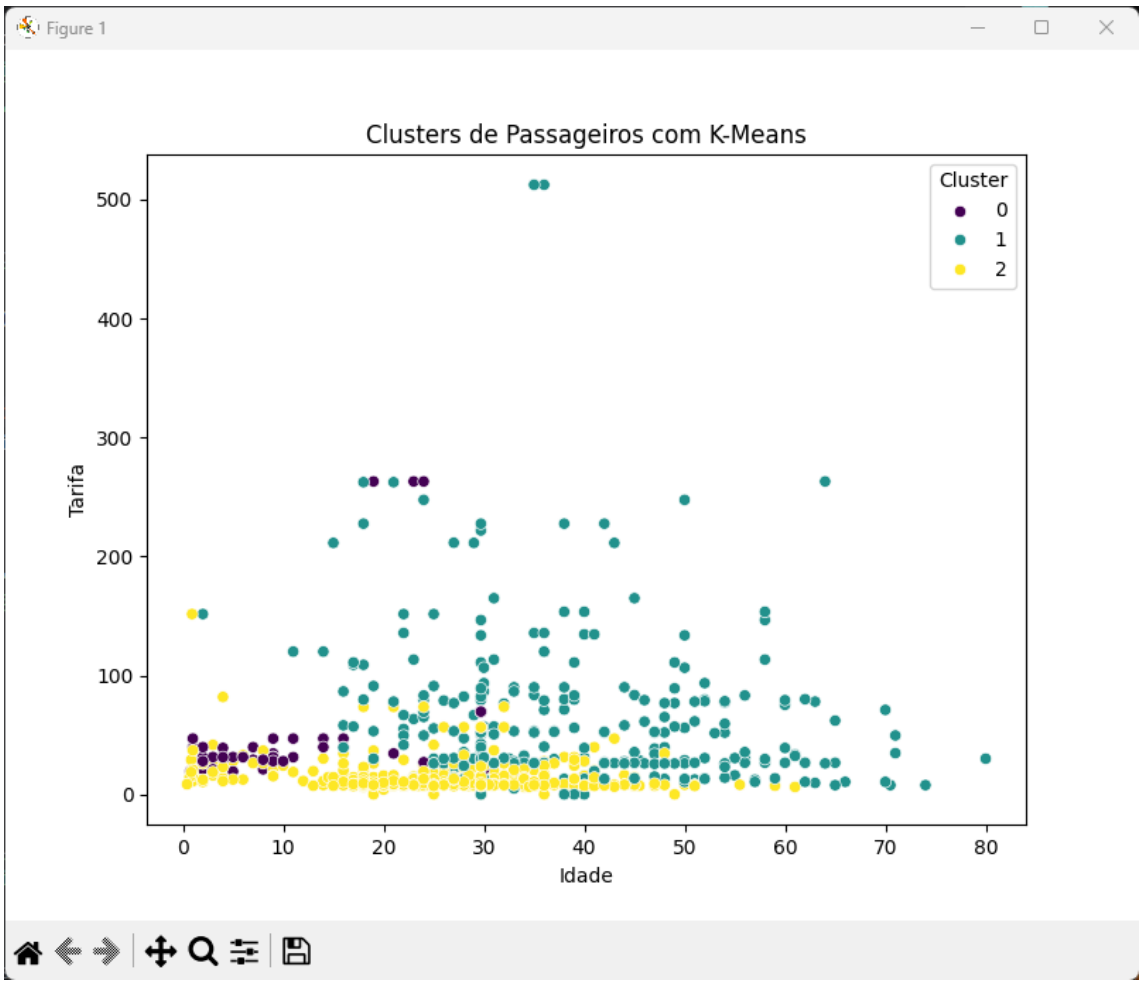
Normaliza os dados para realizar o agrupamento.

```
# Aplicar K-Means
kmeans = KMeans(n_clusters=3, random_state=42)
data['Cluster'] = kmeans.fit_predict(X_scaled)
```

Escolhe 3 clusters para explorar os grupos.

```python
# Visualizar os clusters
plt.figure(figsize=(8, 6))
sns.scatterplot(x='Age', y='Fare', hue='Cluster', data=data, palette='viridis')
plt.title('Clusters de Passageiros com K-Means')
plt.xlabel('Idade')
plt.ylabel('Tarifa')
plt.legend(title='Cluster')
plt.show()
```

Montagem a impressão do gráfico.



Impressão do gráfico mostrando resultados da Técnica de Agrupamento com gráfico de passageiros com base em características como idade e tarifa.

REGRESSÃO

```python
# Prever a tarifa  usando outras variáveis
X_reg = data[['Pclass', 'Age', 'SibSp', 'Sex_male', 'Embarked']]
y_reg = data['Fare']
```

Prevendo as tarifas usando outras variáveis.

```
# Dividir os dados em treino e teste
X_train_reg, X_test_reg, y_train_reg, y_test_reg = train_test_split(X_reg, y_reg, test_size=0.2, random_state=42)
```

Divide os dados para dados de teste.

```
# Cria e treina o modelo de regressão
reg_model = LinearRegression()
reg_model.fit(X_train_reg, y_train_reg)
```

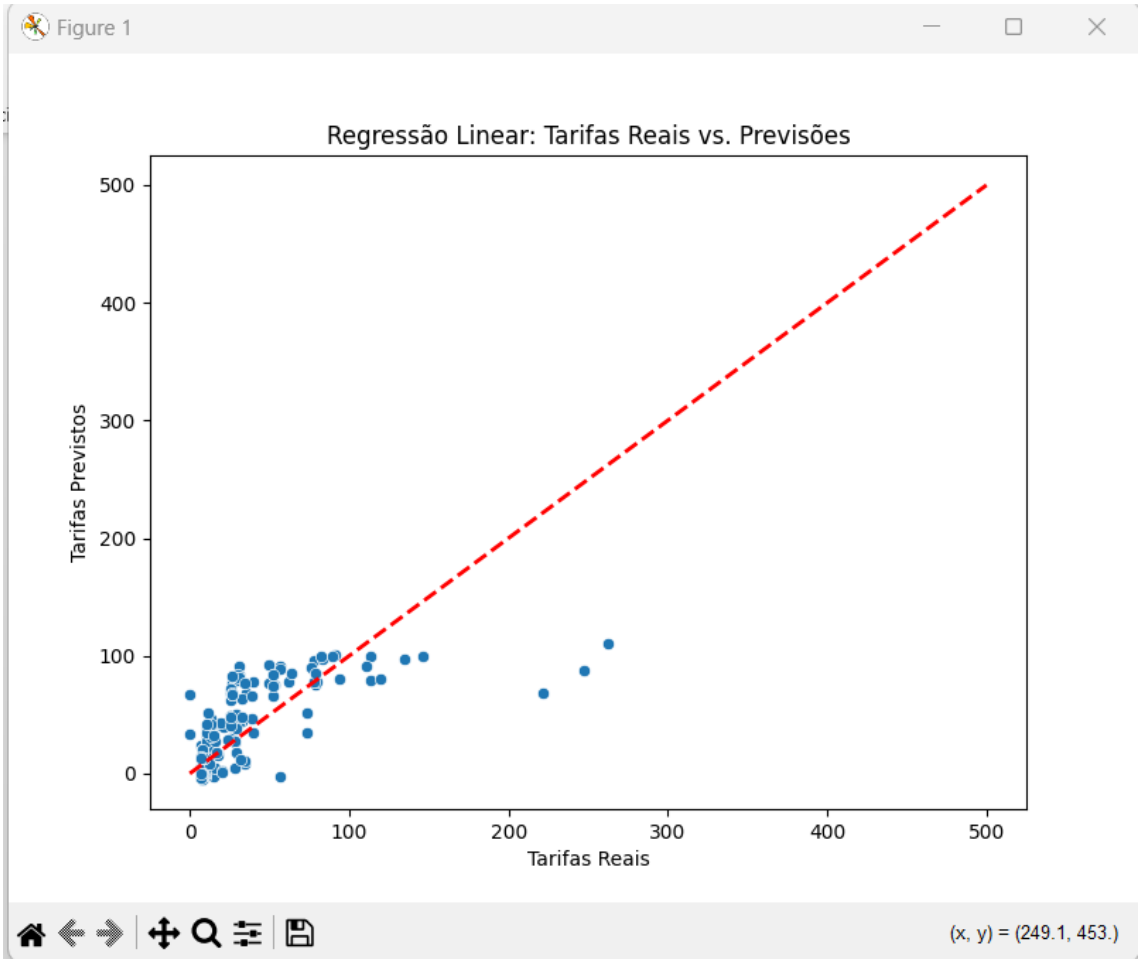
Aplica os dados de testes no modelo de regressão.

```
# Prever tarifas
y_pred_reg = reg_model.predict(X_test_reg)
```

Prever as tarifas.

```
# Visualizar a relação entre valores reais e previstos
plt.figure(figsize=(8, 6))
sns.scatterplot(x=y_test_reg, y=y_pred_reg)
plt.title('Regressão Linear: Tarifas Reais vs. Previsões')
plt.xlabel('Tarifas Reais')
plt.ylabel('Tarifas Previstos')
plt.plot([0, 500], [0, 500], '--', color='red', linewidth=2)
plt.show()
```

Montagem a impressão do gráfico.



Impressão do gráfico mostrando resultados da Técnica de Regressão com gráficos de comparação entre valores reais e previstos.

5. Conclusão

A implementação das três técnicas em um conjunto só, é essencial pois a classificação fornece uma entrega direta, o agrupamento ajuda a entender os padrões não supervisionados, e a regressão fornece conhecimentos adicionais sobre relações contínuas ou categóricas, portanto, a implementação das três técnicas oferece uma análise mais profunda e visualmente com uma interface gráfica atrativa para o mesmo conjunto de dados.

6. Anexos

- DataSet: aqui;

- Repositório dos arquivos do projeto no Drive: aqui;