Universidade de Trás-os-Montes e Alto Douro

Ciência dos Dados

Protocolo 1 Versão A

2022 – 2023

Autores:

Patrício Simões, 73407

José Rocha, 74392

Hugo Mansilha, 72957

Simão Lopes, 70623

Índice

1. Introdução

No decorrer deste trabalho foram propostos vários problemas, neste relatório encontram-se explicitas as soluções aos respetivos, tal como o progresso de realização para chegar para tal e a sua análise de dados de resultados.

Com base no Dataset "Life Expectancy (WHO) Fixed", este protocolo desenvolve um conjunto de inúmeras tarefas baseadas na linguagem Python e recorrendo a técnicas de machine learning, para analisar problemas da vida real, apresentando as suas soluções em vários gráficos para à posteriori analise de dados.

1.1 Enquadramento Teórico

O mean absolute error (MAE) é a média do valor absoluto das diferenças entre as previsões e os valores reais. O mean squared error (MSE) é a média dos quadrados das diferenças entre as previsões e os valores reais. O R-squared (R^2) é uma medida da proporção da variância nos dados que é explicada pelo modelo.

Regressão linear simples é um modelo estatístico usado para descrever a relação entre uma variável dependente (também chamada de variável resposta ou variável Y) e uma única variável independente (também chamada de variável explanatória ou variável X). O objetivo da regressão linear simples é encontrar a melhor reta que representa a relação entre as duas variáveis. A reta é definida pela equação Y = a + bX, onde a é o intercepto e b é o coeficiente angular da reta.

A regressão polinomial é uma extensão da regressão linear simples, onde a relação entre as variáveis é modelada usando uma equação polinomial em vez de uma reta. Ou seja, a regressão polinomial tenta ajustar uma curva que melhor se ajusta aos dados em vez de uma reta. Isso é útil quando a relação entre as variáveis não é linear. Por exemplo, se os dados seguirem uma curva em forma de U, uma regressão polinomial pode ser usada para modelar essa relação.

A regressão linear múltipla é uma extensão da regressão linear simples, onde a relação entre uma variável dependente Y e várias variáveis independentes X é modelada. Essas variáveis independentes são chamadas de preditores ou variáveis explicativas. A regressão linear múltipla tenta ajustar uma equação de plano ou hiperplano que melhor se ajusta aos dados, em vez de uma reta, para explicar a relação entre as variáveis. Isso é útil quando a relação entre a variável dependente e as variáveis independentes é complexa e pode ser explicada por mais de uma variável.

2. Desenvolvimento

2.1 Exercícios

2.1.1 Exercício 1

```python
""" 1. Carregue o ficheiro .csv para um DataFrame, e de seguida crie um novo
DataFrame com apenas a informação da Região "Africa". Grave este novo
DataFrame num novo ficheiro .csv."""

import pandas as pd

# Le o ficheiro
life_expectancy = pd.read_csv("../Life-Expectancy-Data-Updated.csv")

#Seleciona só a região de Africa
life_expectancy = life_expectancy.loc[life_expectancy.Region == 'Africa']

#Escreve um novo ficheiro com a nova informação
life_expectancy.to_csv("africa_data.csv", index=False)
```

Código 1: Exercício 1

Este código utiliza a biblioteca Pandas para carregar um arquivo CSV em um DataFrame, de seguida filtra os dados pela região "Africa" e, posteriormente, grava os dados filtrados em um novo arquivo CSV.

O resultado final é um novo arquivo CSV chamado "africa_data.csv" que contem apenas as linhas do arquivo original onde a região é "Africa".

2.2.2 Exercício 2

```
""" 2. A partir do novo DataFrame, faça um gráfico que lhe permita visualizar
convenientemente a evolução das mortes de crianças menores de cinco anos por
1000 habitantes ("Under_five_deaths") nos países Angola, Cabo Verde, Guinea e
Mozambique. """

import pandas as pd
pd.plotting.register_matplotlib_converters()
import matplotlib.pyplot as plt
import seaborn as sns


#Carrega os dados
df = pd.read_csv('../Exercicio_1/africa_data.csv')

#Filtra para os países selecionados
paises = ['Angola', 'Cabo Verde', 'Guinea', 'Mozambique']
df_filtrado = df.loc[df.Country.isin(paises)]

#Cria o gráfico
sns.lineplot(data=df_filtrado, x='Year', y='Under_five_deaths', hue='Country')
plt.title('Mortes de Crianças Menores de 5 anos em Angola, Cabo Verde, Guiné e Moçambique', fontsize = 8)
plt.xlabel('Ano')
plt.ylabel('Mortes de Crianças Menores de 5 anos por 1000 habitantes', fontsize = 7)
plt.show()
```

Código 2: Exercício 2

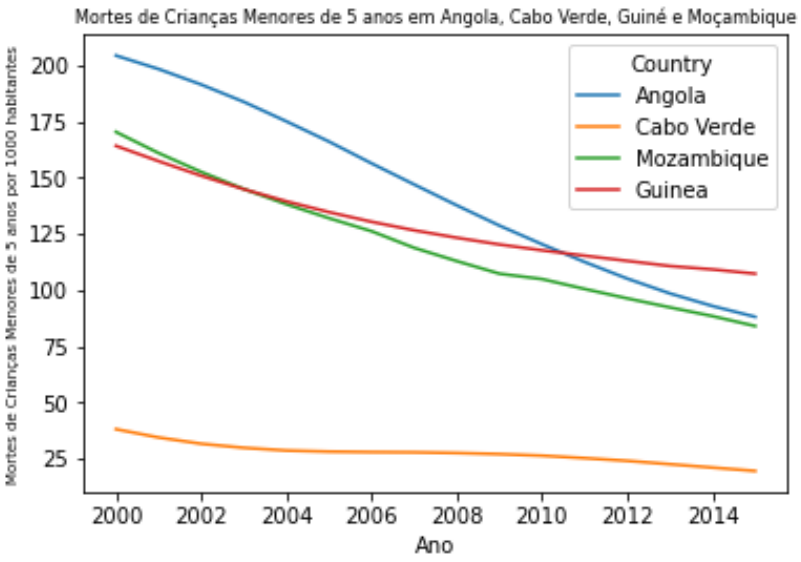

Gráfico 1: Morte de crianças Menores de 5 anos em Angola, Cabo Verde, Guiné e Moçambique

Este exercício pretende a análise da evolução das mortes de crianças menores de cinco anos por 1000 habitantes em Angola, Cabo Verde, Guiné e Moçambique.

Analisamos a evolução das mortes de crianças menores de cinco anos por 1000 habitantes em quatro países africanos: Angola, Cabo Verde, Guiné e Moçambique. Utilizamos um conjunto de dados disponível em um arquivo CSV chamado "africa_data.csv" e utilizamos as bibliotecas

pandas, matplotlib e seaborn em Python para realizar a análise e criar um gráfico que permita visualizar convenientemente os resultados.

Carregamento dos dados: Utilizamos a biblioteca pandas para carregar os dados do arquivo CSV em um DataFrame. O caminho do arquivo foi especificado como "../Exercicio_1/africa_data.csv".

Filtragem dos países: Definimos uma lista chamada "paises" contendo os nomes dos países de interesse: Angola, Cabo Verde, Guiné e Moçambique. Em seguida, utilizamos a função "loc()" do pandas para filtrar o DataFrame original e incluir apenas as linhas correspondentes a esses países. O resultado foi armazenado em um novo DataFrame chamado "df_filtrado".

Criação do gráfico: Utilizamos a biblioteca seaborn para criar um gráfico de linhas que representa a evolução das mortes de crianças menores de cinco anos por 1000 habitantes ao longo dos anos. Utilizamos o DataFrame "df_filtrado" como fonte de dados, onde o eixo x do gráfico foi definido como "Year" e o eixo y como "Under_five_deaths". A variável categórica "Country" foi utilizada como argumento "hue" para colorir as linhas do gráfico de acordo com o país.

Resultados: O gráfico gerado permite visualizar a evolução das mortes de crianças menores de cinco anos por 1000 habitantes nos países Angola, Cabo Verde, Guiné e Moçambique ao longo dos anos. O gráfico de linhas mostra as tendências de mortalidade nesses países, permitindo identificar possíveis padrões ou variações ao longo do tempo.

2.2.3 Exercício 3

```
""" Usando a biblioteca Matplotlib,
crie um gráfico circular ('pie chart') que represente a
média nos anos 2000 a 2015 da população total em milhões ("Population_mln"),
nos países Ghana, Kenya, Morocco e Nigeria. Coloque as legendas adequadas. """

import pandas as pd
import matplotlib.pyplot as plt

# Carregar o arquivo CSV para um DataFrame do Pandas
df = pd.read_csv("../Life-Expectancy-Data-Updated.csv")

# Filtrar o DataFrame para os países e anos de interesse
countries = ["Ghana", "Kenya", "Morocco", "Nigeria"]
years = range(2000, 2016)

df_filtrado = df.loc[df.Country.isin(countries) & df.Year.isin(years)]

# Calcular a média da população total em milhões para cada país
pop_means = df_filtrado.groupby("Country")["Population_mln"].mean()

# Criar o gráfico circular
plt.pie(pop_means, labels=pop_means.index, autopct=lambda x: f"{x:.1f} M", )
plt.title("Média da População Total em Milhões (2000-2015)")

#Para modificar a legenda
plt.legend(title="Países", loc= (1, 0))
```

Código 3 – Exercício 3

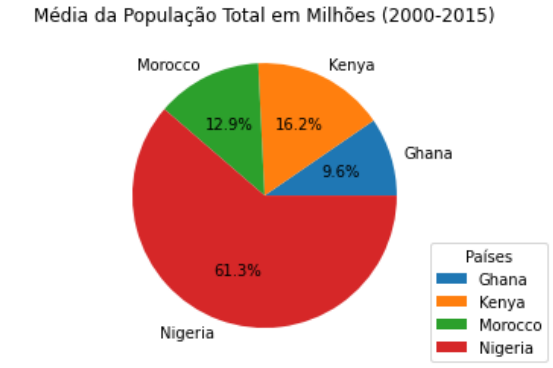

Gráfico 2 – Média da População Total em Milhões (%)

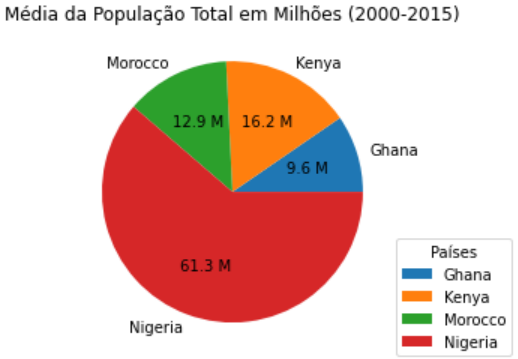

Gráfico 3 Média da População Total em Milhões (M)

O objetivo é representar a média da população total em milhões para quatro países (Gana, Quênia, Marrocos e Nigéria) ao longo dos anos de 2000 a 2015, com as legendas adequadas.

As bibliotecas Pandas e Matplotlib são importadas para realizar as operações de manipulação de dados e criação do gráfico, respectivamente.

Carregamento do arquivo CSV: O arquivo CSV contendo os dados é carregado para um DataFrame do Pandas, utilizando a função pd.read_csv(). O caminho do arquivo é especificado como "../Life-Expectancy-Data-Updated.csv".

Filtragem dos dados: O DataFrame é filtrado para incluir apenas as linhas que correspondem aos países (Gana, Quênia, Marrocos e Nigéria) e aos anos (2000 a 2015) de interesse. Isso é feito

utilizando a função .loc[] do Pandas em combinação com os métodos .isin() para verificar a presença dos valores desejados.

Cálculo da média da população: O método .groupby() do Pandas é utilizado para agrupar o DataFrame filtrado por país, e em seguida, a média da coluna "Population_mln" é calculada para cada grupo, utilizando o método .mean().

Criação do gráfico circular: A função plt.pie() do Matplotlib é utilizada para criar o gráfico circular. Os valores da média da população são passados como argumento, assim como as legendas dos países (obtidas a partir do índice do DataFrame) e a formatação dos valores exibidos no gráfico é definida usando a função autopct com uma função lambda para formatar os valores com uma casa decimal e a letra "M" para indicar milhões. O título do gráfico é definido com a função plt.title().

Modificação da legenda: A função plt.legend() é utilizada para modificar a legenda do gráfico. O título da legenda é definido como "Países" e a posição é ajustada para (1, 0) para posicioná-la no canto superior direito do gráfico.

Exibição do gráfico: A função plt.show() é utilizada para exibir o gráfico na saída do código.

2.2.4 Exercício 4

```python
"""
Crie uma função que, dado o nome do país da Região "Africa", apresente o ano em
que a esperança média de vida ("Life_expectancy") foi maior, bem como o
respetivo valor.
"""

import pandas as pd
import matplotlib.pyplot as plt

# Ler o arquivo CSV e carregá-lo em um DataFrame do Pandas
df = pd.read_csv('../Exercicio_1/africa_data.csv')

# Definir uma função que, dada o nome do país, apresente o ano em que a esperança média de vida foi a maior
def EMV(_country):
    # Filtrar o DataFrame para o país especificado
    df_country = df[df['Country'] == _country]

    # Encontrar a linha com o maior valor de esperança média de vida
    max_row = df_country.loc[df_country['Life_expectancy'].idxmax()]

    # Extrair o ano e o valor da esperança média de vida
    year = max_row['Year']
    life_expectancy = max_row['Life_expectancy']

    # Retornar o resultado como uma string
    return f"A maior esperança média de vida em {_country} foi de {life_expectancy:.1f} anos em {year}."

# Testar a função para o país "Gabão"
print(EMV("Gabon"))
```

Código 4 - Exercício 4

Carrega um arquivo csv contendo informações sobre a esperança média de vida em vários países africanos em diferentes anos. Em seguida, a função 'EMV' é definida, que recebe um parâmetro 'country' que representa o nome do país a ser analisado.

Dentro da função, filtra o DataFrame pelos registros do país especificado e, em seguida, encontra a linha com o maior valor de esperança média de vida, utilizando o método 'idxmax'. O 'idxmax' retorna o índice da linha com o valor máximo de 'Life_expectancy'.

Após encontrar a linha com o maior valor de esperança média de vida, a função extrai o ano e o valor correspondente e retorna uma string formatada com essas informações.

Por fim, testa a função para o país Gabão e imprime o resultado.

Em resumo, cria uma função simples que permite ao utilizador obter rapidamente informações sobre a esperança média de vida em um determinado país africano em um determinado ano.

2.2.5 Exercício 5

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load the data
df = pd.read_csv('../Exercicio_1/africa_data.csv')
df = df.dropna(subset=['GDP_per_capita', 'Life_expectancy'])

#So de dispersão por pais
sns.scatterplot(data=df, x="GDP_per_capita", y="Life_expectancy", hue="Country",
                style="Country" )

# Labels
plt.xlabel('PIB per capita em dólares')
plt.ylabel('Esperança média de vida')
plt.title('Relação entre PIB per capita e Esperança Média de Vida')
plt.legend(title="Países", loc= (1.1, -1))
plt.show()

# Com linha de regressão global
g = sns.regplot(x="GDP_per_capita", y="Life_expectancy", data=df,
                scatter_kws = {"color": "black", "alpha": 0.5},
                line_kws = {"color": "red"},
                ci = 99)
# Labels
plt.xlabel('PIB per capita em dólares')
plt.ylabel('Esperança média de vida')
plt.title('Relação entre PIB per capita e Esperança Média de Vida')
plt.show()
```

Código 5 - Exercício 5

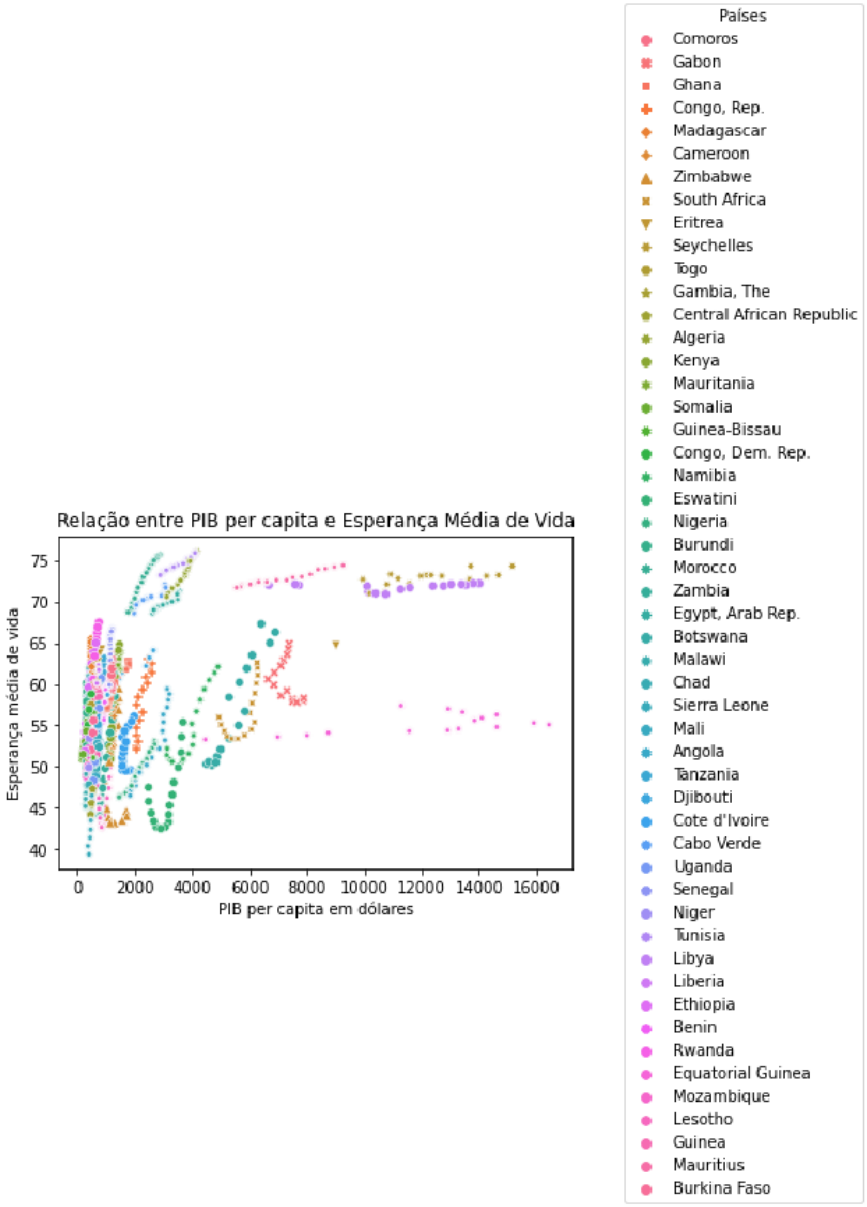

Gráfico 4 - Relação entre PIB per capita e Esperança Média de Vida Entre Países

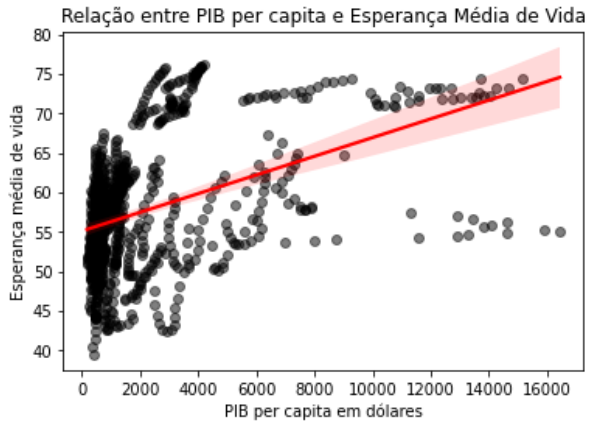

Gráfico 5 - Relação entre PIB per capita e Esperança Média de vida

Carrega dados de um arquivo csv, que contém informações sobre o PIB per capita em dólares e a esperança média de vida em países da África. Em seguida cria um gráfico de dispersão que permite visualizar a relação entre as duas variáveis, em que o eixo x representa o PIB per capita em dólares e o eixo y representa a esperança média de vida.

O gráfico de dispersão é personalizado para mostrar os dados de cada país individualmente, através da cor e do estilo do marcador. O código também adiciona rótulos para cada eixo, título do gráfico e legenda para identificar cada país no gráfico.

Além do gráfico de dispersão, há uma linha de regressão linear global para mostrar a tendência geral da relação entre o PIB per capita e a esperança média de vida. A linha de regressão é traçada usando a função 'regplot' do Seaborn.

A linha de regressão ajuda a visualizar a relação entre as variáveis e a prever a esperança média de vida com base no PIB per capita. O coeficiente de correlação de Pearson (r) é uma medida da força e direção dessa relação, e é impresso na linha de regressão.

Isto pode ser usado para analisar a relação entre o PIB per capita e a esperança média de vida em diferentes países ou regiões, desde que os dados sejam fornecidos na mesma estrutura de arquivo csv. A visualização do gráfico permite observar a distribuição dos dados e identificar possíveis padrões, além de destacar os países com desempenho melhor ou pior em termos de PIB per capita e esperança média de vida.

2.2.6 Exercicio 6

O objetivo deste exercício é explorar técnicas de Machine Learning que nos permita fazer uma previsão da esperança média de vida (“Life_expectancy”) no futuro. Onde nos deixaram usar quaisquer colunas do Dataset que achássemos conveniente. Este exercício foi dividido em 3 partes, numa regressão linear múltipla, regressão linear simples e numa regressão polinomial.

Para a resolução deste exercício também tivemos de importar bibliotecas “seaborn” e “sklearn” que são usadas para implementar gráficos mais avançados e aplicar técnicas de Machine Learning.

Regressão linear múltipla:

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
import seaborn as sns


# Carregar dados
df = pd.read_csv('../Exercicio_1/africa_data.csv')

df.drop(columns=['Economy_status_Developed', 'Economy_status_Developing'], inplace=True)

df = df[df['Country'] == 'Gabon']

# Sort data by year
df = df.sort_values('Year')

# Normalizar os dados
scaler = StandardScaler()
df_norm = pd.DataFrame(scaler.fit_transform(df.drop(columns=['Country', 'Region'])), columns=df.columns[2:])


# Gerar mapa de calor da matriz de correlação
corr = df_norm.corr()
plt.figure(figsize=(20, 8))
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title('Matriz de Correlação')
plt.show()
```

```
# Filtrar dados para ser so do Gabão
gabon_df = df[df['Country'] == 'Gabon']

# Selecionar features (X) e o target (y)
X = gabon_df[['Year', 'BMI', 'Schooling']]
y = gabon_df['Life_expectancy']

# Separar dados em treino e teste
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=46)

# Treine um modelo de regressão linear nos dados de treino
model = LinearRegression().fit(X_train, y_train)

# Predict Life Expectancy for 2020 and 2030
X_2020 = [[2020, gabon_df['BMI'].iloc[-1], gabon_df['Schooling'].iloc[-1]]]
X_2030 = [[2030, gabon_df['BMI'].iloc[-1], gabon_df['Schooling'].iloc[-1]]]
y_pred_2020 = model.predict(X_2020)[0]
y_pred_2030 = model.predict(X_2030)[0]

# Avaliar o modelo no conjunto de teste
y_pred = model.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Imprimir as métricas de avaliação e previsões
print('MAE:', mae)
print('MSE:', mse)
print('R^2:', r2)
print('Esperança de vida no Gabão em 2020:', y_pred_2020)
print('Esperança de vida no Gabão em 2030:', y_pred_2030)

# Treinar um modelo de regressão linear em todo o conjunto de dados
model = LinearRegression().fit(X, y)
```

```
# Prever a expectativa de vida para todos os anos
y_pred = model.predict(X)

# Traçar a evolução da expectativa de vida ao longo do tempo
plt.plot(X['Year'], y, '-o', color='blue')
plt.plot(X['Year'], y_pred, color='red')
plt.plot([2020, 2030], [y_pred_2020, y_pred_2030], 'xr', color='green')
plt.xlabel('Year')
plt.ylabel('Life Expectancy')
plt.title('Evolução da esperança de vida no Gabão')
plt.legend(['Atual', 'Linha Regressão Linear Múltipla', 'Valores previstos para 2020 e 2030'])
plt.xticks(range(2000, 2031, 5))
plt.xlim(2000, 2032)
plt.show()
```

Código 6 - Exercício 6 Regressão linear múltipla:

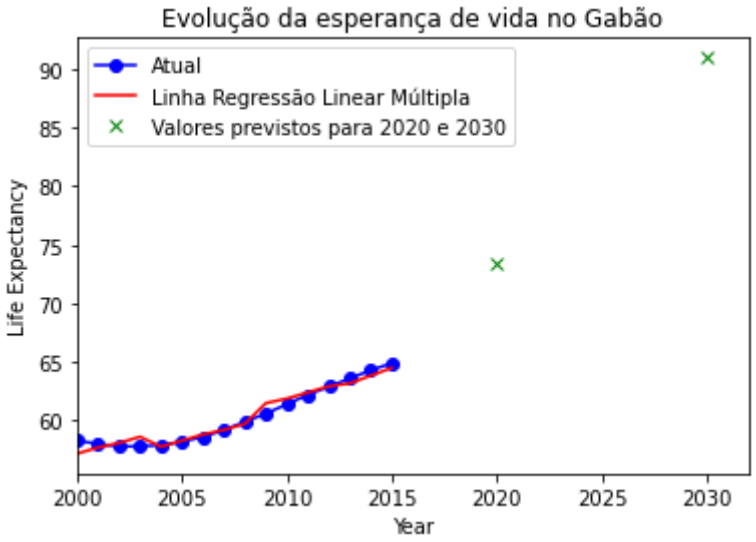

Gráfico 7 – Evolução da esperança de vida no Gabão usando Regressão linear múltipla

Esse código tem como objetivo treinar um modelo de regressão linear múltipla para prever a expectativa de vida no Gabão ao longo do tempo, utilizando dados de índice de massa corporal (BMI), escolaridade (Schooling) e ano (Year).

O código começa importando diversas bibliotecas para ajudar na análise dos dados, tais como pandas para leitura e manipulação de dados, matplotlib e seaborn para plotagem de gráficos e sklearn para a criação do modelo de regressão linear e métricas de avaliação do modelo.

Em seguida, ele carrega os dados de um arquivo CSV e filtra apenas para os dados do Gabão. Depois, normaliza os dados utilizando StandardScaler, que transforma os dados para que eles tenham média zero e desvio padrão um. Isso é importante para que as diferentes variáveis estejam na mesma escala e possam ser comparadas de forma adequada.

O código então gera um mapa de calor da matriz de correlação dos dados normalizados para ajudar a entender a relação entre as variáveis. A matriz de correlação é uma tabela que mostra o coeficiente de correlação entre cada par de variáveis. Um valor próximo de 1 indica uma correlação positiva forte, enquanto um valor próximo de -1 indica uma correlação negativa forte. Um valor próximo de zero indica que as variáveis não estão correlacionadas.

Em seguida, o código seleciona as variáveis que serão utilizadas no modelo de regressão linear (Year, BMI e Schooling) e separa os dados em conjunto de treino e teste. É importante separar os dados em conjuntos de treino e teste para avaliar o desempenho do modelo em dados que ele não foi treinado.

O modelo de regressão linear é treinado utilizando os dados de treino e as métricas de avaliação (mean absolute error, mean squared error e R-squared) são calculadas utilizando os dados de teste.

Depois de treinar e avaliar o modelo, ele é utilizado para fazer previsões para a expectativa de vida no Gabão em 2020 e 2030. As previsões são baseadas nos valores de BMI e Schooling mais recentes e no ano de 2020 e 2030. Em seguida, o modelo é treinado em todos os dados disponíveis e utilizado para prever a expectativa de vida para todos os anos.

Por fim, criamos o gráfico a evolução da expectativa de vida ao longo do tempo, comparando os valores reais com as previsões do modelo e os valores previstos para 2020 e 2030.

Podemos analisar um pouco de *overfitting* no gráfico no uso deste modelo, o overfitting pode ocorrer por motivos como o modelo ser muito complexo, dados insuficientes, ruído excessivo nos dados, sendo que neste caso terá sido por só termos dados de 2000 a 2015, o que é bastante pouco.

Regressão linear simples:

```python
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
from sklearn.model_selection import train_test_split

# Load data from csv
df = pd.read_csv('../Exercicio_1/africa_data.csv')

# Filter data for Gabon only
gabon_df = df[df['Country'] == 'Gabon']

# Split data into features (X) and target (y)
X = gabon_df[['Year']]
y = gabon_df['Life_expectancy']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=46)

# Train a Linear Regression model on the training data
model = LinearRegression().fit(X_train, y_train)

# Predict Life Expectancy for 2020 and 2030
y_pred_2020 = model.predict([[2020]])[0]
y_pred_2030 = model.predict([[2030]])[0]

# Evaluate the model on the testing set
y_pred = model.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

```
# Print the evaluation metrics and predictions
print('MAE:', mae)
print('MSE:', mse)
print('R^2:', r2)
print('Esperança de vida no Gabão em 2020:', y_pred_2020)
print('Esperança de vida no Gabão em 2030:', y_pred_2030)

# Train a Linear Regression model on the entire dataset
model = LinearRegression().fit(X, y)

# Predict Life Expectancy for all years
y_pred = model.predict(X)

# Plot the Life Expectancy evolution over time
plt.plot(X, y, '-o', color='blue')
plt.plot(X, y_pred, color='red')
plt.plot([2020, 2030], [y_pred_2020, y_pred_2030], 'xr', color='green')
plt.xlabel('Year')
plt.ylabel('Life Expectancy')
plt.title('Evolução da esperança de vida no Gabão')
plt.legend(['Atual', 'Linha Regressão Linear', 'Valores previstos para 2020 e 2030'])
plt.xticks(range(2000, 2031, 5))
plt.xlim(2000, 2032)
plt.show()
```

Código 7 - Exercício 6 Regressao linear simples

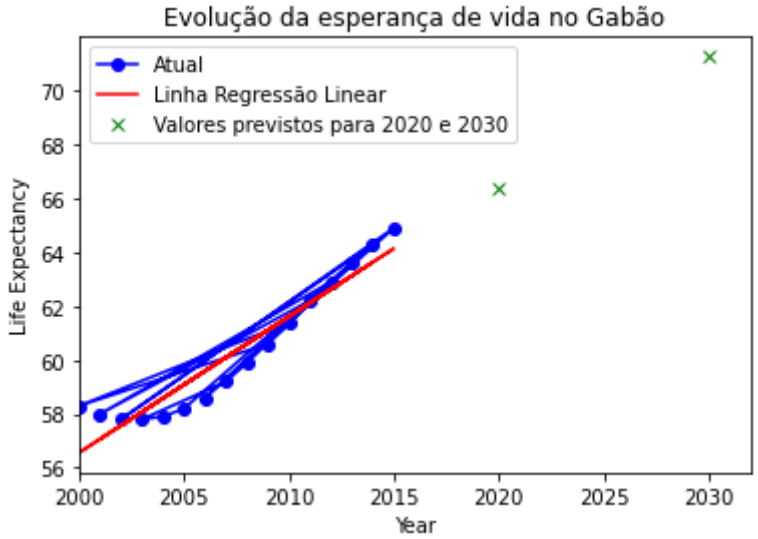

Gráfico 8 – Evolução da esperança de vida no Gabão usando Regressão linear simples

No modelo de regressão linear simples criamos as variaveis ‘x’ e ‘y’ para representar as caracteristicas ano e expectativa de vida do conjunto de dados. De seguida para os dados em conjuntos de treinamento usamos o metodo ‘train_test_split’ do scikit-learn, onde 80% dos dados são usados para treinar o modelo e 20% para usar ao testá-los. O modelo de regressao linear simples é treinado usando o método ‘LinearRegression().fit’ com as variáveis ‘X_train’ e ‘y_train’.

O método predict é usado para fazer previsões da expectativa de vida em Gabão para 2020 e 2030. Mais uma vez calculamos o ‘MAE’,’MSE’ e ‘R2’ usando o conjunto do teste.

Finalmente, um novo modelo de regressão linear é treinado usando o conjunto completo de dados, e as previsões para todos os anos são plotadas junto com os dados reais usando o Matplotlib. O resultado é um gráfico que mostra a evolução da expectativa de vida em Gabão ao longo dos anos, bem como as previsões para 2020 e 2030.

Podemos analisar um pouco de *underfitting* no gráfico no uso deste modelo ao contrário da regressão linear multipla, neste caso poderá ser por este modelo ser muito simples e também falta de dados.

Regressão polinomial:

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
from sklearn.model_selection import train_test_split

# Load data from csv
df = pd.read_csv('../Exercicio_1/africa_data.csv')

# Filter data for Gabon only
gabon_df = df[df['Country'] == 'Gabon']

# Split data into features (X) and target (y)
X = gabon_df[['Year']]
y = gabon_df['Life_expectancy']

# Create polynomial features
poly = PolynomialFeatures(degree=2)
X_poly = poly.fit_transform(X)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_poly, y, test_size=0.2, random_state=46)

# Train a Polynomial Regression model on the training data
model = LinearRegression().fit(X_train, y_train)

# Predict Life Expectancy for 2020 and 2030
X_pred_2020 = poly.transform([[2020]])
X_pred_2030 = poly.transform([[2030]])
y_pred_2020 = model.predict(X_pred_2020)[0]
y_pred_2030 = model.predict(X_pred_2030)[0]
```

```
# Evaluate the model on the testing set
y_pred = model.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Print the evaluation metrics and predictions
print('MAE:', mae)
print('MSE:', mse)
print('R^2:', r2)
print('Esperança de vida no Gabão em 2020:', y_pred_2020)
print('Esperança de vida no Gabão em 2030:', y_pred_2030)

# Train a Polynomial Regression model on the entire dataset
model = LinearRegression().fit(X_poly, y)

# Predict Life Expectancy for all years
X_all = np.array(range(2000, 2031)).reshape(-1, 1)
X_all_poly = poly.transform(X_all)
y_pred = model.predict(X_all_poly)

# Plot the Life Expectancy evolution over time
plt.plot(X, y, '-o', color='blue')
plt.plot(X_all, y_pred, color='red')
plt.plot([2020, 2030], [y_pred_2020, y_pred_2030], 'xr', color='green')
plt.xlabel('Year')
plt.ylabel('Life Expectancy')
plt.title('Evolução da esperança de vida no Gabão')
plt.legend(['Atual', 'Linha Regressão Polinomial de Grau 2', 'Valores previstos para 2020 e 2030'])
plt.xticks(range(2000, 2031, 5))
plt.xlim(2000, 2032)
plt.show()
```

Código 8 – - Exercício 6 Regressao polinomial

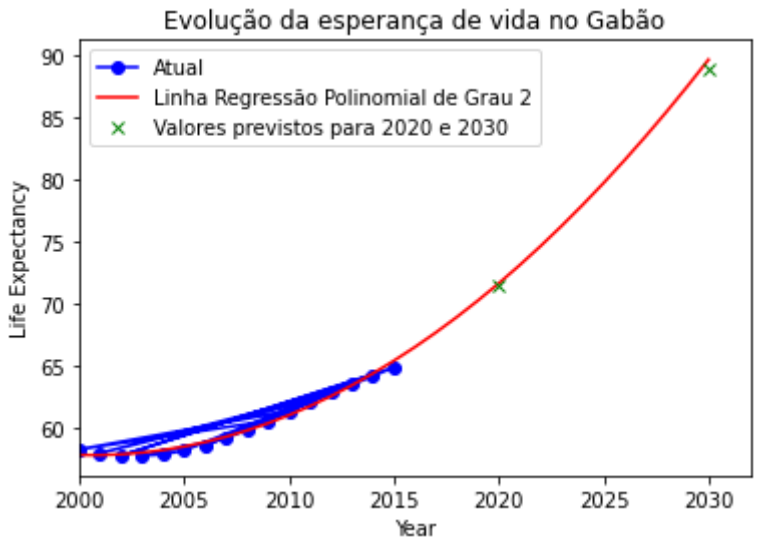

Gráfico 9 – Evolução da esperança de vida no Gabão usando Regressão polinomial

Neste modelo o código filtra os dados para incluir apenas informações sobre o Gabão e separa os dados em recursos 'X' e alvo 'y'. Ele então cria recursos polinomiais de 2º grau e divide os dados em conjuntos de treino e teste. Com essses recursos criamos um modelo de regressão polinomial de 2º grau para fazer as previsões da expectativa de vida no Gabão entre 2020 e 2030, avaliando também a 'MAE', 'MSE' e 'R2'. Para acabar traça um gráfico que mostra a evolução da expectativa de vida no Gabão ao longo do tempo, incluindo os valores reais, a linha de regressão e as previsões para 2020 e 2030.

No gráfico gerado por este modelo podemos observar que a curva gerada é a mais apropriada para o caso aqui apresentado, sendo este o modelo ideal, comparado com os outros dois usados.

3. Conclusão

Em conclusão no âmbito da cadeira Ciência dos dados, com a resolução deste protocolo 1, os vários membros do grupo conseguiram dotar-se na vertente do machine learning e python devido a esta introdução pelas propostas entregues, conseguimos realizar o pretendido na sua totalidade, apesar de dificuldades encontradas pelo caminho na parte da procura da resolução.

4.0 Anexos

4.1 Exercício 1

```
""" 1. Carregue o ficheiro.csv para um DataFrame, e de seguida crie um novo
DataFrame com apenas a informação da Região “Africa”. Grave este novo
DataFrame num novo ficheiro.csv."""

import pandas as pd

# Le o ficheiro
life_expectancy = pd.read_csv("../Life-Expectancy-Data-Updated.csv")

#Seleciona só a região de Africa
life_expectancy = life_expectancy.loc[life_expectancy.Region == 'Africa']

#Escreve um novo ficheiro com a nova informação
life_expectancy.to_csv("africa_data.csv", index=False)
```

4.2 Exercício 2

```
""" 2. A partir do novo DataFrame, faça um gráfico que lhe permita visualizar
convenientemente a evolução das mortes de crianças menores de cinco anos por
1000 habitantes (“Under_five_deaths”) nos países Angola, Cabo Verde, Guinea e
Mozambique. """

import pandas as pd
pd.plotting.register_matplotlib_converters()
import matplotlib.pyplot as plt
import seaborn as sns


#Carrega os dados
df = pd.read_csv('../Exercicio_1/africa_data.csv')

#Filtra para os países selecionados
paises = ['Angola', 'Cabo Verde', 'Guinea', 'Mozambique']
df_filtrado = df.loc[df.Country.isin(paises)]

#Cria o gráfico
sns.lineplot(data=df_filtrado, x='Year', y='Under_five_deaths', hue='Country')
plt.title('Mortes de Crianças Menores de 5 anos em Angola, Cabo Verde, Guiné e Moçambique',
fontsize = 8)
plt.xlabel('Ano')
plt.ylabel('Mortes de Crianças Menores de 5 anos por 1000 habitantes', fontsize = 7)
plt.show()
```

4.3 Exercício 3

```
""" Usando a biblioteca Matplotlib,
crie um gráfico circular ('pie chart') que represente a
média nos anos 2000 a 2015 da população total em milhões ("Population_mln"),
nos países Ghana, Kenya, Morocco e Nigeria. Coloque as legendas adequadas. """

import pandas as pd
import matplotlib.pyplot as plt

# Carregar o arquivo CSV para um DataFrame do Pandas
df = pd.read_csv("../Life-Expectancy-Data-Updated.csv")

# Filtrar o DataFrame para os países e anos de interesse
countries = ["Ghana", "Kenya", "Morocco", "Nigeria"]
years = range(2000, 2016)

df_filtrado = df.loc[df.Country.isin(countries) & df.Year.isin(years)]

# Calcular a média da população total em milhões para cada país
pop_means = df_filtrado.groupby("Country")["Population_mln"].mean()

# Criar o gráfico circular
plt.pie(pop_means, labels=pop_means.index, autopct=lambda x: f"{x:.1f} M", )
plt.title("Média da População Total em Milhões (2000-2015)")

#Para modificar a legenda
plt.legend(title="Países", loc= (1, 0))
plt.show()
```

4.4 Exercício 4

```
"""
Crie uma função que, dado o nome do país da Região “Africa”, apresente o ano em
que a esperança média de vida (“Life_expectancy”) foi maior, bem como o
respetivo valor.
"""

import pandas as pd
import matplotlib.pyplot as plt

# Ler o arquivo CSV e carregá-lo em um DataFrame do Pandas
df = pd.read_csv('../Exercicio_1/africa_data.csv')

# Definir uma função que, dada o nome do país, apresente o ano em que a esperança média de vida
foi a maior
def EMV(_country):
    # Filtrar o DataFrame para o país especificado
    df_country = df[df['Country'] == _country]

    # Encontrar a linha com o maior valor de esperança média de vida
    max_row = df_country.loc[df_country['Life_expectancy'].idxmax()]

    # Extrair o ano e o valor da esperança média de vida
    year = max_row['Year']
    life_expectancy = max_row['Life_expectancy']

    # Retornar o resultado como uma string
     return f"A maior esperança média de vida em {_country} foi de {life_expectancy:.1f} anos em
{year}."

# Testar a função para o país "Gabão"
print(EMV("Gabon"))
```

4.5 Exercício 5

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load the data
df = pd.read_csv('../Exercicio_1/africa_data.csv')
df = df.dropna(subset=['GDP_per_capita', 'Life_expectancy'])

#So de dispersão por pais
sns.scatterplot(data=df, x="GDP_per_capita", y="Life_expectancy", hue="Country",
                style="Country" )

# Labels
plt.xlabel('PIB per capita em dólares')
plt.ylabel('Esperança média de vida')
plt.title('Relação entre PIB per capita e Esperança Média de Vida')
plt.legend(title="Países", loc= (1.1, -1))
plt.show()

# Com linha de regressão global
g = sns.regplot(x="GDP_per_capita", y="Life_expectancy", data=df,
                scatter_kws = {"color": "black", "alpha": 0.5},
                line_kws = {"color": "red"},
                ci = 99)
# Labels
plt.xlabel('PIB per capita em dólares')
plt.ylabel('Esperança média de vida')
plt.title('Relação entre PIB per capita e Esperança Média de Vida')
plt.show()
```

4.6 Exercício 6

Regressão linear múltipla:

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error,
r2_score
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
import seaborn as sns


# Carregar dados
df = pd.read_csv('../Exercicio_1/africa_data.csv')

df.drop(columns=['Economy_status_Developed',
'Economy_status_Developing'], inplace=True)

df = df[df['Country'] == 'Gabon']

# Sort data by year
df = df.sort_values('Year')

# Normalizar os dados
scaler = StandardScaler()
df_norm = pd.DataFrame(scaler.fit_transform(df.drop(columns=['Country',
'Region'])), columns=df.columns[2:])


# Gerar mapa de calor da matriz de correlação
corr = df_norm.corr()
plt.figure(figsize=(20, 8))
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title('Matriz de Correlação')
plt.show()

# Filtrar dados para ser so do Gabão
gabon_df = df[df['Country'] == 'Gabon']

# Selecionar features (X) e o target (y)
X = gabon_df[['Year', 'BMI', 'Schooling']]
y = gabon_df['Life_expectancy']

# Separar dados em treino e teste
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=46)

# Treine um modelo de regressão linear nos dados de treino
model = LinearRegression().fit(X_train, y_train)

# Predict Life Expectancy for 2020 and 2030
X_2020 = [[2020, gabon_df['BMI'].iloc[-1], gabon_df['Schooling'].iloc[-
1]]]
X_2030 = [[2030, gabon_df['BMI'].iloc[-1], gabon_df['Schooling'].iloc[-
1]]]
y_pred_2020 = model.predict(X_2020)[0]
y_pred_2030 = model.predict(X_2030)[0]

# Avaliar o modelo no conjunto de teste
y_pred = model.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Imprimir as métricas de avaliação e previsões
print('MAE:', mae)
print('MSE:', mse)
print('R^2:', r2)
print('Esperança de vida no Gabão em 2020:', y_pred_2020)
print('Esperança de vida no Gabão em 2030:', y_pred_2030)

# Treinar um modelo de regressão linear em todo o conjunto de dados
model = LinearRegression().fit(X, y)


# Prever a expectativa de vida para todos os anos
y_pred = model.predict(X)

# Traçar a evolução da expectativa de vida ao longo do tempo
plt.plot(X['Year'], y, '-o', color='blue')
plt.plot(X['Year'], y_pred, color='red')
plt.plot([2020, 2030], [y_pred_2020, y_pred_2030], 'xr', color='green')
plt.xlabel('Year')
plt.ylabel('Life Expectancy')
plt.title('Evolução da esperança de vida no Gabão')
plt.legend(['Atual', 'Linha Regressão Linear Múltipla', 'Valores
previstos para 2020 e 2030'])
plt.xticks(range(2000, 2031, 5))
plt.xlim(2000, 2032)
plt.show()
```

Regressão linear simples:

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error,
r2_score
from sklearn.model_selection import train_test_split

# Load data from csv
df = pd.read_csv('../Exercicio_1/africa_data.csv')

# Filter data for Gabon only
gabon_df = df[df['Country'] == 'Gabon']

# Split data into features (X) and target (y)
X = gabon_df[['Year']]
y = gabon_df['Life_expectancy']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=46)

# Train a Linear Regression model on the training data
model = LinearRegression().fit(X_train, y_train)

# Predict Life Expectancy for 2020 and 2030
y_pred_2020 = model.predict([[2020]])[0]
y_pred_2030 = model.predict([[2030]])[0]

# Evaluate the model on the testing set
y_pred = model.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Print the evaluation metrics and predictions
print('MAE:', mae)
print('MSE:', mse)
print('R^2:', r2)
print('Esperança de vida no Gabão em 2020:', y_pred_2020)
print('Esperança de vida no Gabão em 2030:', y_pred_2030)

# Train a Linear Regression model on the entire dataset
model = LinearRegression().fit(X, y)

# Predict Life Expectancy for all years
```

```
y_pred = model.predict(X)

# Plot the Life Expectancy evolution over time
plt.plot(X, y, '-o', color='blue')
plt.plot(X, y_pred, color='red')
plt.plot([2020, 2030], [y_pred_2020, y_pred_2030], 'xr', color='green')
plt.xlabel('Year')
plt.ylabel('Life Expectancy')
plt.title('Evolução da esperança de vida no Gabão')
plt.legend(['Atual', 'Linha Regressão Linear', 'Valores previstos para
2020 e 2030'])
plt.xticks(range(2000, 2031, 5))
plt.xlim(2000, 2032)
plt.show()
```

Regressão polinomial:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import mean_absolute_error, mean_squared_error,
r2_score
from sklearn.model_selection import train_test_split

# Load data from csv
df = pd.read_csv('../Exercicio_1/africa_data.csv')

# Filter data for Gabon only
gabon_df = df[df['Country'] == 'Gabon']

# Split data into features (X) and target (y)
X = gabon_df[['Year']]
y = gabon_df['Life_expectancy']

# Create polynomial features
poly = PolynomialFeatures(degree=2)
X_poly = poly.fit_transform(X)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_poly, y,
test_size=0.2, random_state=46)

# Train a Polynomial Regression model on the training data
```

```
model = LinearRegression().fit(X_train, y_train)

# Predict Life Expectancy for 2020 and 2030
X_pred_2020 = poly.transform([[2020]])
X_pred_2030 = poly.transform([[2030]])
y_pred_2020 = model.predict(X_pred_2020)[0]
y_pred_2030 = model.predict(X_pred_2030)[0]

# Evaluate the model on the testing set
y_pred = model.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Print the evaluation metrics and predictions
print('MAE:', mae)
print('MSE:', mse)
print('R^2:', r2)
print('Esperança de vida no Gabão em 2020:', y_pred_2020)
print('Esperança de vida no Gabão em 2030:', y_pred_2030)

# Train a Polynomial Regression model on the entire dataset
model = LinearRegression().fit(X_poly, y)

# Predict Life Expectancy for all years
X_all = np.array(range(2000, 2031)).reshape(-1, 1)
X_all_poly = poly.transform(X_all)
y_pred = model.predict(X_all_poly)

# Plot the Life Expectancy evolution over time
plt.plot(X, y, '-o', color='blue')
plt.plot(X_all, y_pred, color='red')
plt.plot([2020, 2030], [y_pred_2020, y_pred_2030], 'xr', color='green')
plt.xlabel('Year')
plt.ylabel('Life Expectancy')
plt.title('Evolução da esperança de vida no Gabão')
plt.legend(['Atual', 'Linha Regressão Polinomial de Grau 2', 'Valores
previstos para 2020 e 2030'])
plt.xticks(range(2000, 2031, 5))
plt.xlim(2000, 2032)
plt.show()
```