

INTELIGÊNCIA ARTIFICIAL REGRESSÃO

Eduarda Tobias Fernandes - RA: 12522216514;

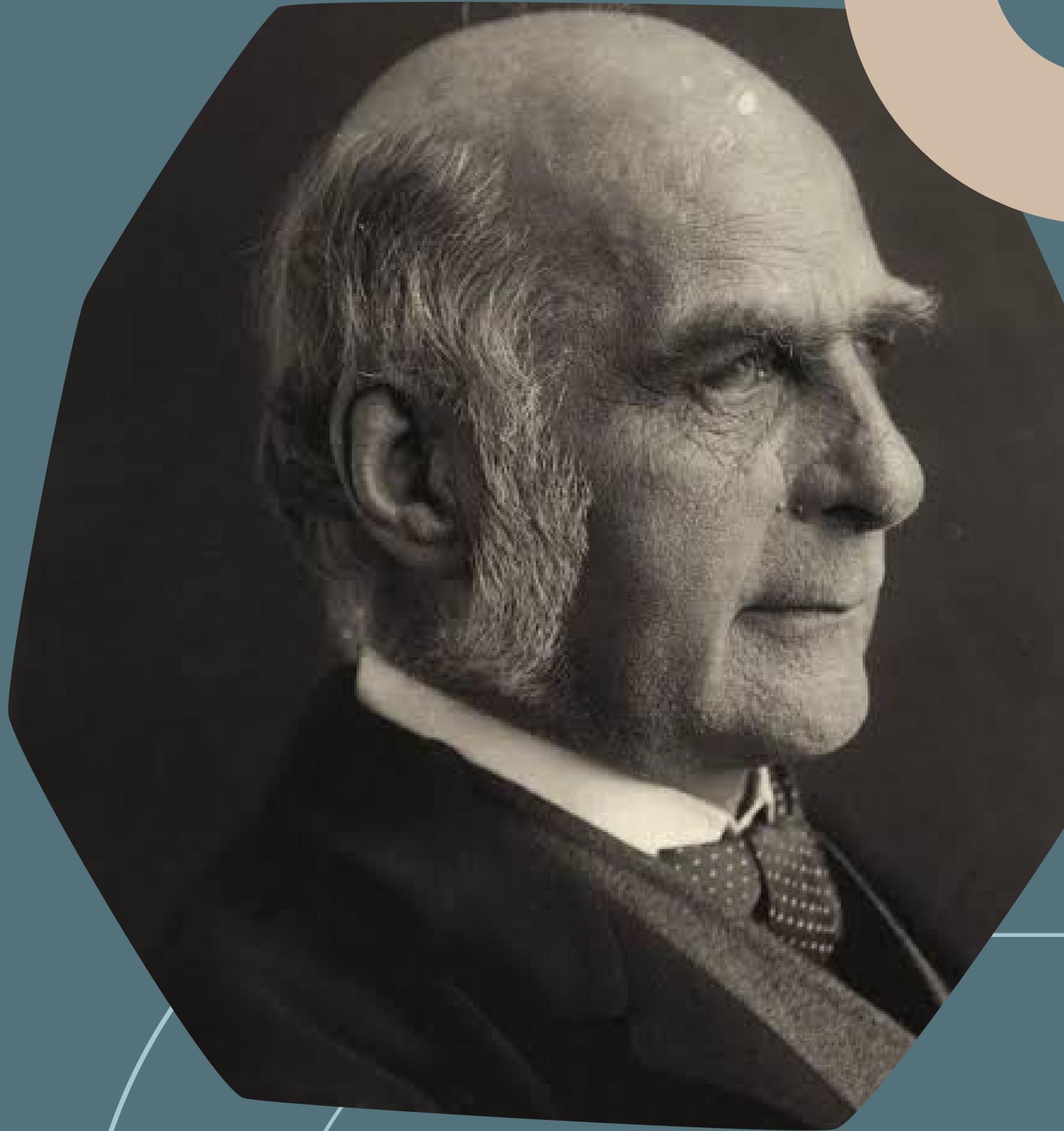
Jennifer Reis Sicherolli De Souza - RA: 125221110547;

Katerine Linda Witkoski - RA: 1252221362;

Nathalie Mori Taylor Zampieri - RA: 1252224816;

HISTÓRICO

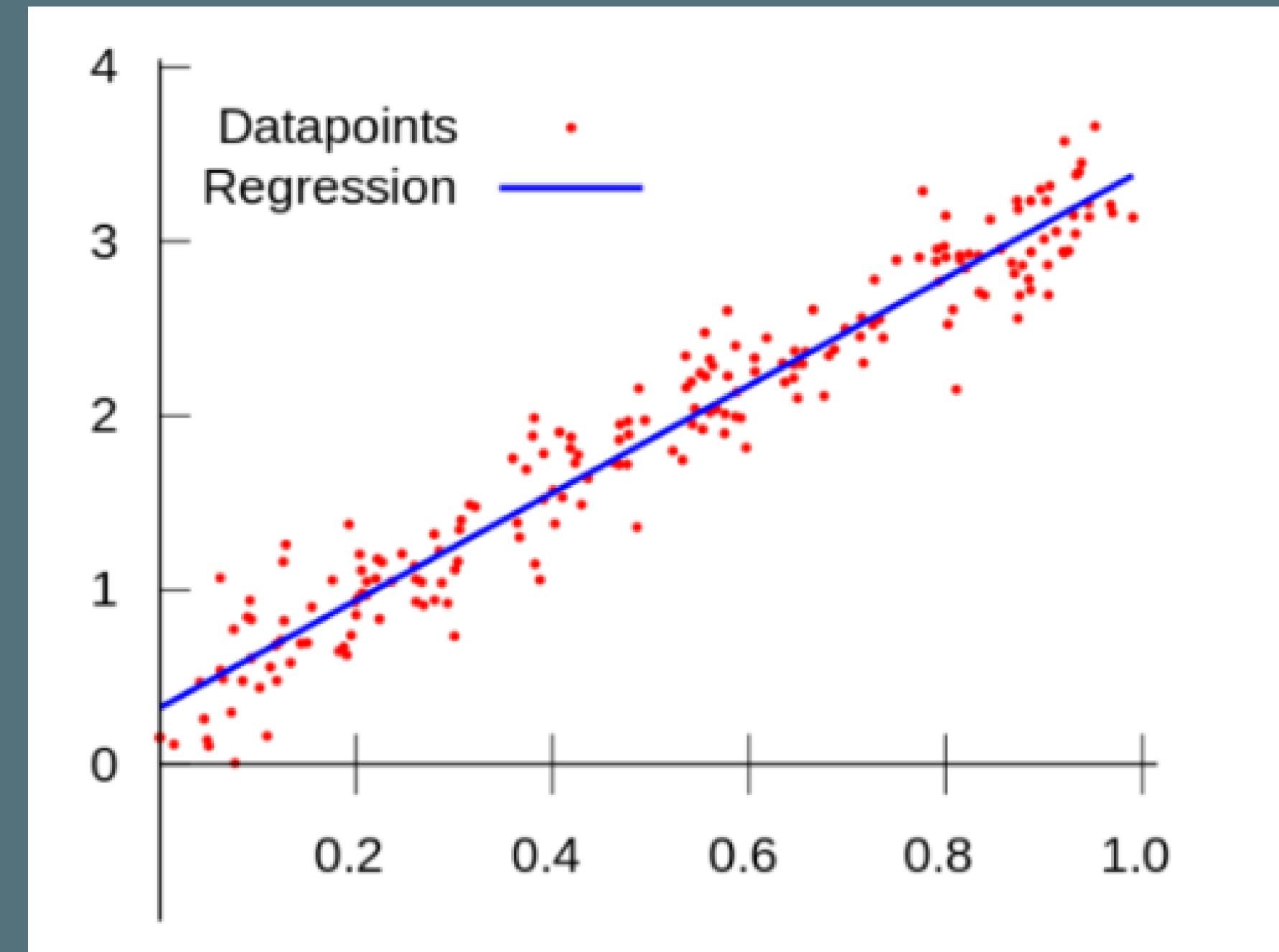
- introduzida por Sir Francis Galton
- altura media
- prever valores
- relação entre variável dependente e variável independente



CARACTERISTICAS

Regressão Linear:

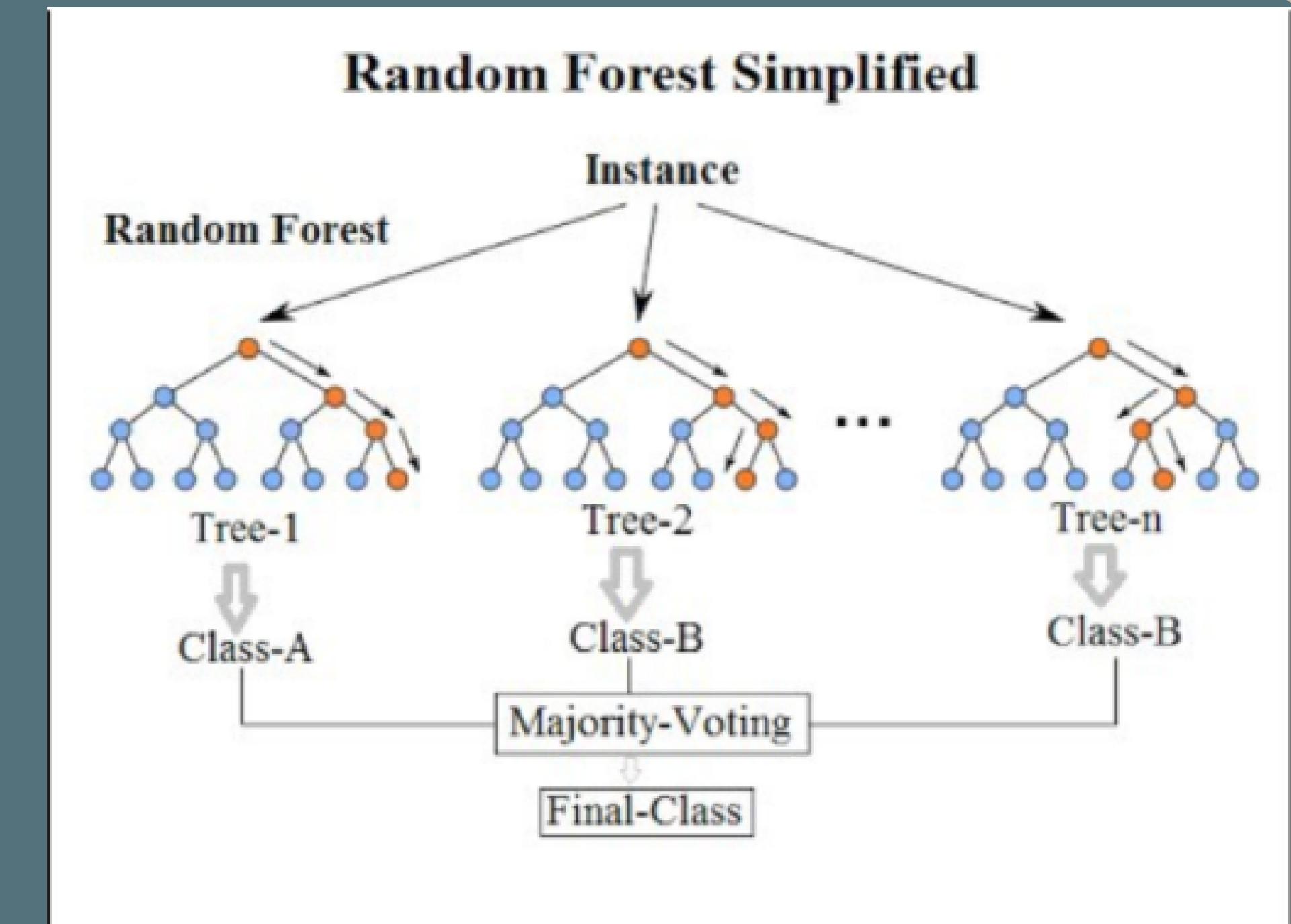
- relação entre a variável dependente e as variáveis independentes é linear.
- Regressão linear Simples:
- Regressão Linear Múltipla



CARACTERISTICAS

Regressão com Random Forest

- árvores de decisão
- variâncias reduzidas
- resultados mais consistentes



BANCO DE DADOS

- Kaggle Datasets
- expectativa de vida
- composta por 22 colunas, e 2938 linhas

```
174     informacao = df.info()
175     print(informacao)
```

PROBLEMAS SAÍDA CONSOLE DE DEPURAÇÃO TERMINAL PORTAS SQL CONSOLE GITLENS

```
0 > 0 ~/Doc/regressao > on 0 0 main !1 > /Users/katerinewitkoski/Documents/regressao/.venv/bin/python /Users/katerinewitkoski/Documents/regressao/LifeExpect.py
Index(['country', 'year', 'status', 'life_expectancy', 'adult_mortality',
       'infant_deaths', 'alcohol', 'percentage_expenditure', 'hepatitis_b',
       'measles', 'bmi', 'under-five_deaths', 'polio', 'total_expenditure',
       'diphtheria', 'hiv/aids', 'gdp', 'population', 'thinness_1-19_years',
       'thinness_5-9_years', 'income_composition_of_resources', 'schooling'],
      dtype='object')
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2938 entries, 0 to 2937
Data columns (total 22 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   country          2938 non-null    int64  
 1   year              2938 non-null    int64  
 2   status             2938 non-null    int64  
 3   life_expectancy   2938 non-null    float64 
 4   adult_mortality   2938 non-null    float64 
 5   infant_deaths    2938 non-null    int64  
 6   alcohol            2938 non-null    float64 
 7   percentage_expenditure  2938 non-null    float64 
 8   hepatitis_b       2938 non-null    float64 
 9   measles            2938 non-null    int64  
 10  bmi                2938 non-null    float64 
 11  under-five_deaths 2938 non-null    int64  
 12  polio               2938 non-null    float64 
 13  total_expenditure 2938 non-null    float64 
 14  diphtheria         2938 non-null    float64 
 15  hiv/aids           2938 non-null    float64 
 16  gdp                 2938 non-null    float64 
 17  population          2938 non-null    float64 
 18  thinness_1-19_years 2938 non-null    float64 
 19  thinness_5-9_years  2938 non-null    float64 
 20  income_composition_of_resources 2938 non-null    float64 
 21  schooling            2938 non-null    float64 
dtypes: float64(16), int64(6)
memory usage: 585.1 KB
None
```

FUNCIONAMENTO DO ALGORITMO

- métodos estatísticos
- relação entre as variáveis
- Base de dados:
- Análise Exploratória:
- Modelagem:
- Previsão:

```
< def regressao_linear(df):
    # TRAIN TEST SPLIT
    # Para a regressão as variáveis serão x (como um conjunto de todas as colunas menos o recurso alvo a ser observado) e Y (life_expectancy)
    X = df.drop(['life_expectancy'], axis =1)
    y = df['life_expectancy']

    # . Dividir os dados em conjuntos de treinamento e teste permite avaliar o desempenho do modelo em dados não vistos, ajudando a identificar se o
    # modelo está superestimando (overfitting) ou subestimando (underfitting) os dados. No nosso caso, 30% dos dados serão utilizados para teste e 70% para treinamento.

    X_train_org, X_test_org, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 0)

    # Criando uma instância MinMaxScaler (método de normalização que transforma os dados de um intervalo específico, geralmente entre 0 e 1.)

    scaler = MinMaxScaler()
    X_train = scaler.fit_transform(X_train_org)
    X_test = scaler.transform(X_test_org)

    print("X_train shape: ",X_train.shape)
    print("y_train shape: ",y_train.shape)
    print("X_test shape: ",X_test.shape)
    print("y_test shape",y_test.shape)

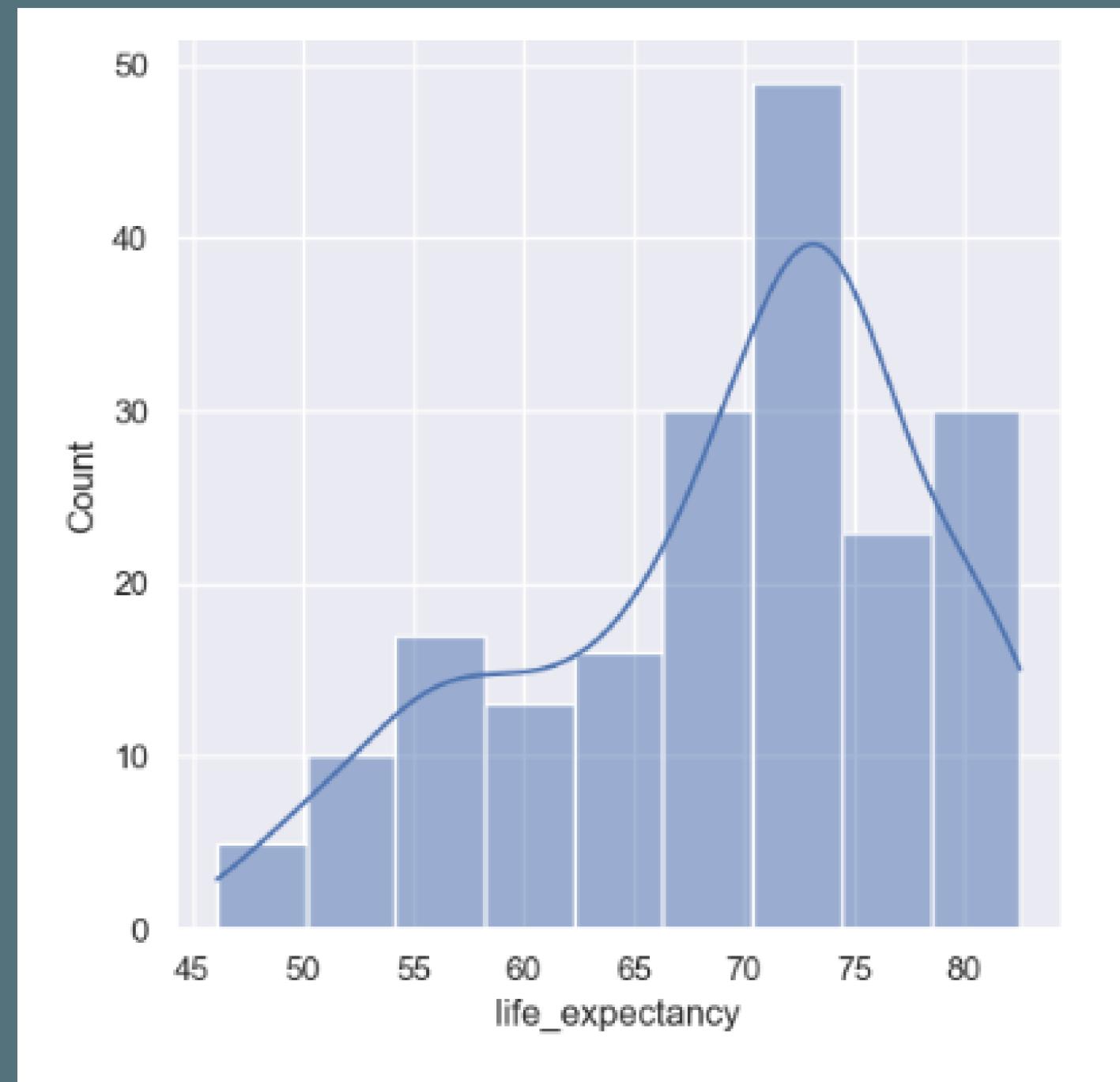
    # Em resultado vimos que 2056 são X_train e 882 são X_test

    # Criando uma Instância do Modelo de Regressão Linear
    lreg = LinearRegression()
    # Ajusta o modelo de regressão linear (lreg) aos dados de treinamento normalizados (X_train e y_train)
    lreg.fit(X_train, y_train)

    print('Train Score: {:.4f}'.format(lreg.score(X_train, y_train)))
    print('Test Score:{:.4f}'.format(lreg.score(X_test, y_test)))
```

CARACTERÍSTICAS DO ALGORITMO

- Previsão e estimativa
- Coeficientes
- Avaliação do Modelo
- Resíduos



Áreas de aplicação:

Economia



Ciências Sociais



Saúde



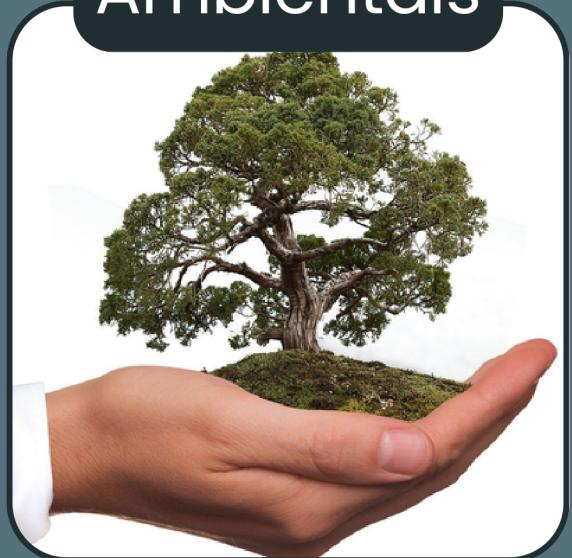
Marketing



Engenharia



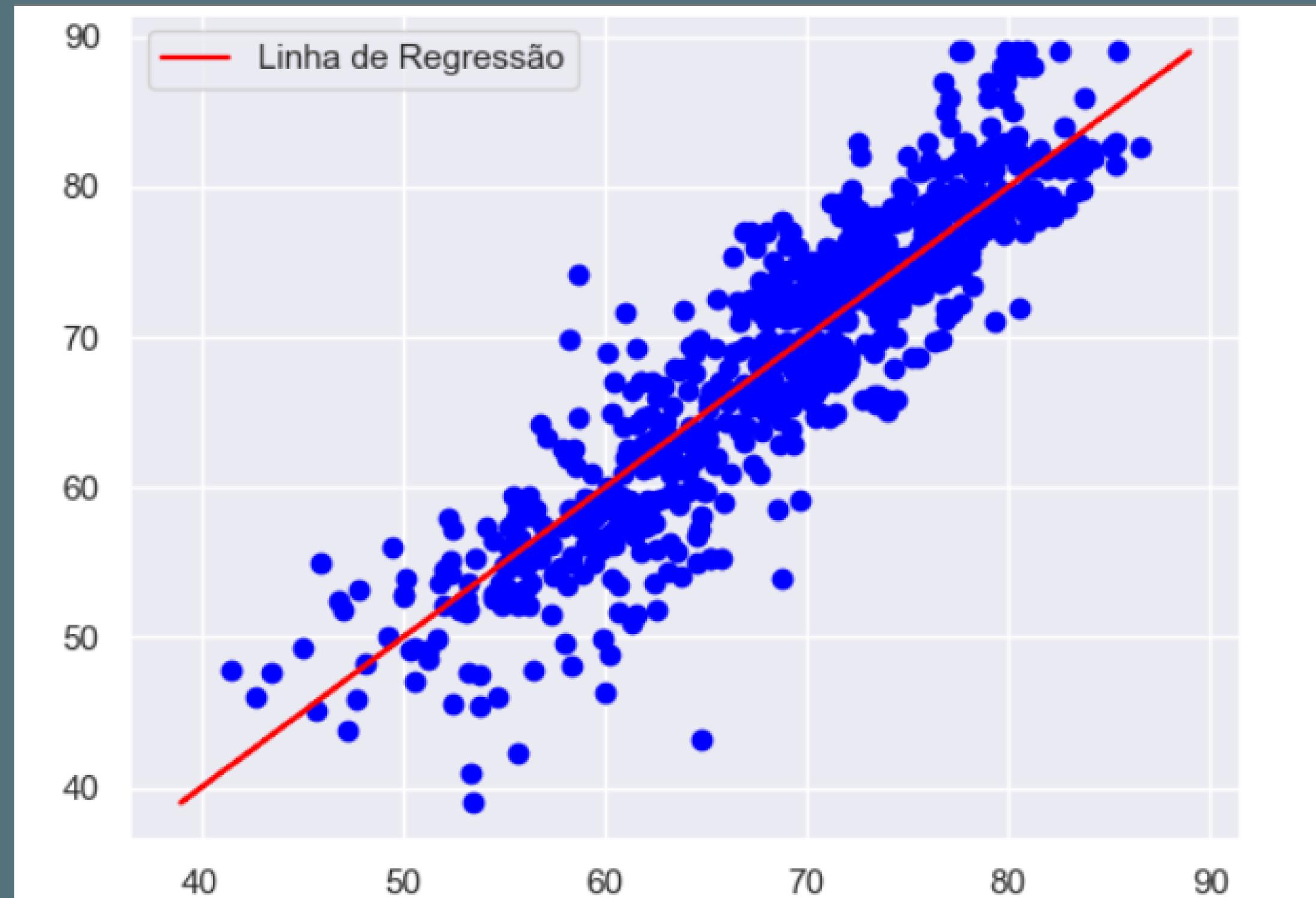
Ciências Ambientais



Finanças



Gráfico de regressão linear



implementação da regressão em python

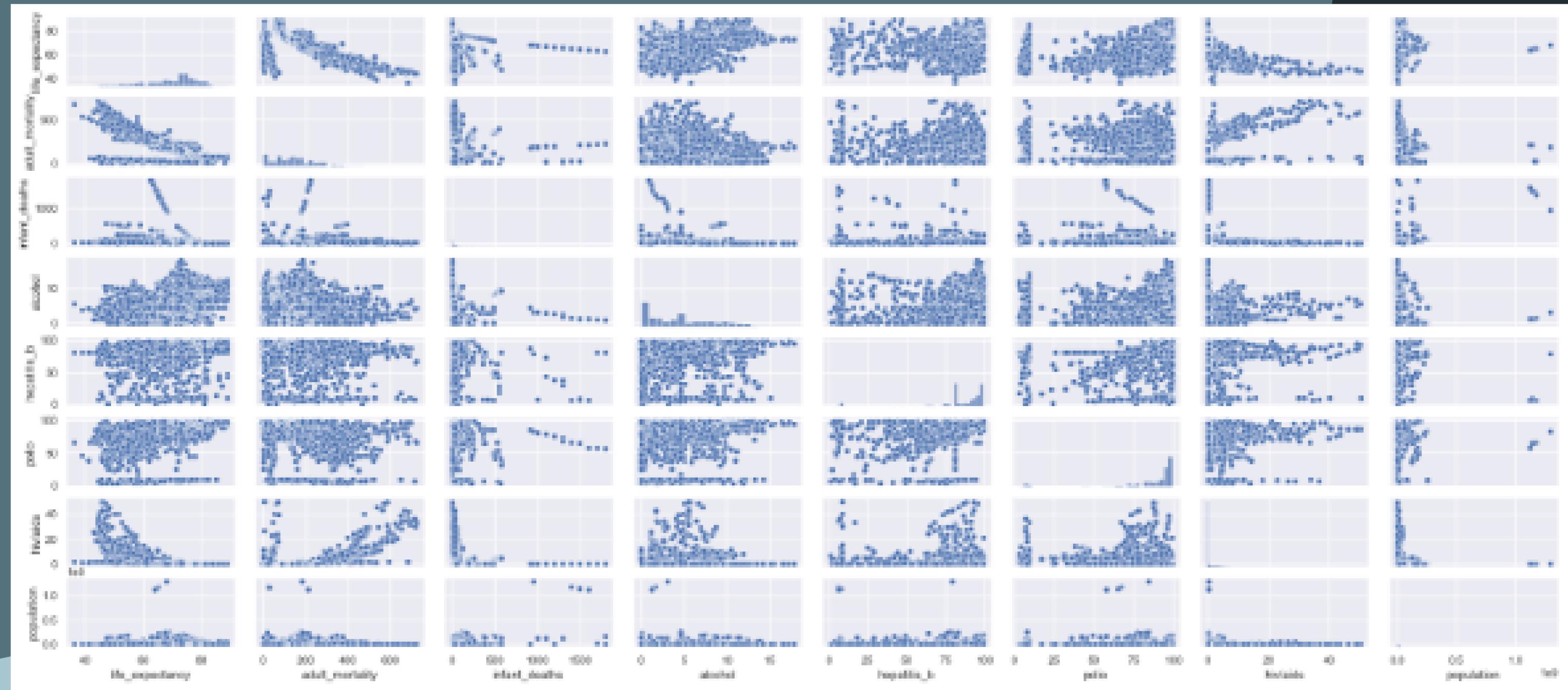


Gráfico de matriz de colunas

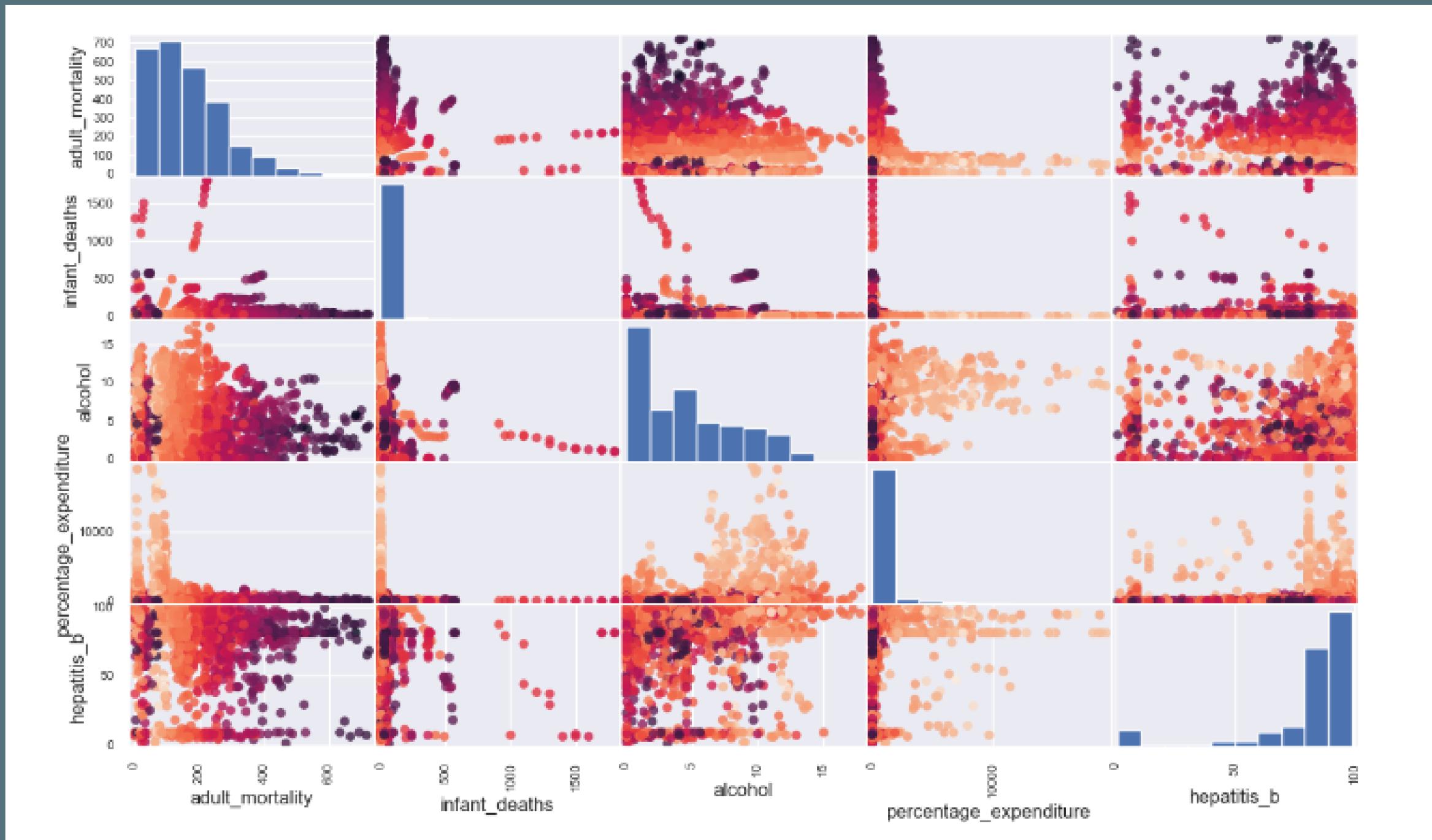


Gráfico de Matriz de correlação

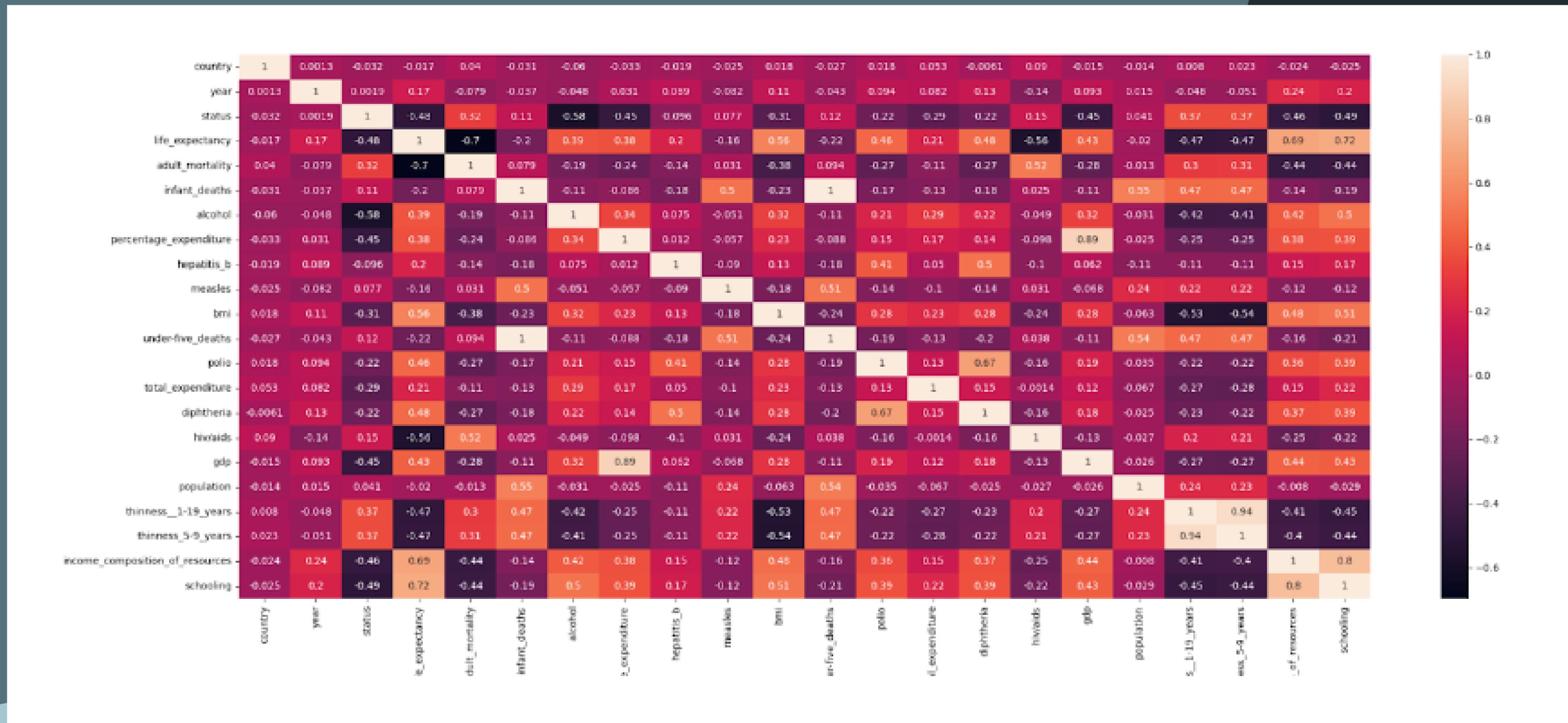


Gráfico de distribuição expectativa de vida

