Importação de bibliotecas que serão utilizadas

```
import pandas as pd
from sklearn.svm import LinearSVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
```

Importando base de dados parcialmente tratada.

Anteriormente fizemos o upload da base de dados para a pasta 'sample_data' do notebook.

Neste passo a base ainda está com as informações originais, apenas com as colunas desnecessárias excluidas.

Deixamos para tratar a base de dados no proprio código pois se trata de um arquivo csv muito pesado, o que ocasionaria um processo moroso e propicio a erros humanos ao tratar por exemplo via Excel.

Colunas existentes nesta etapa:

- sintomas
- comorbidades
- internacao
- internacao_uti
- sexo
- obito
- idade

```
df = pd.read_csv('/content/sample_data/dados.csv', sep = ';')
```

Tratando os dados da base

Distribuiremos todos os sintomas em colunas, ficando assim, uma coluna para cada sintoma.

Marcaremos 1 para positivo referente aquele sintoma e 0 para negativo.

No final do processo deletamos a coluna 'sintomas', pois, como já distribuimos os dados em outras colunas ela se faz desnecessária agora.

```
df['CANSACO'] = df['sintomas'].apply(lambda x: 1 if 'CANSACO' in str(x) else 0)
df['CEFALEIA'] = df['sintomas'].apply(lambda x: 1 if 'CEFALEIA' in str(x) else 0)
df['CONGESTAO_NASAL'] = df['sintomas'].apply(lambda x: 1 if 'CONGESTAO NASAL' in str(x) else 0)
df['CORIZA'] = df['sintomas'].apply(lambda x: 1 if 'CORIZA' in str(x) else 0)
df['DIARREIA'] = df['sintomas'].apply(lambda x: 1 if 'DIARREIA' in str(x) else 0)
df['DISPNEIA'] = df['sintomas'].apply(lambda x: 1 if 'DISPNEIA' in str(x) else 0)
df['DOR_NO_CORPO'] = df['sintomas'].apply(lambda x: 1 if 'DOR NO CORPO' in str(x) else 0)
df['DOR_DE_GARGANTA'] = df['sintomas'].apply(lambda x: 1 if 'DOR DE GARGANTA' in str(x) else 0)
df['FEBRE'] = df['sintomas'].apply(lambda x: 1 if 'FEBRE' in str(x) else 0)
df['TOSSE'] = df['sintomas'].apply(lambda x: 1 if 'TOSSE' in str(x) else 0)
df['MIALGIA'] = df['sintomas'].apply(lambda x: 1 if 'MIALGIA' in str(x) else 0)

del df['sintomas']
```

Faremos o mesmo processo para a coluna 'comorbidades'.

Distribuiremos todas as comorbidades em colunas, ficando assim, uma coluna para cada comorbidade.

Marcaremos 1 para positivo referente aquela comorbidade e 0 para negativo.

No final do processo deletamos a coluna 'comorbidades', pois, como já distribuimos os dados em outras colunas ela se faz desnecessária agora.

```
df['ASMA'] = df['comorbidades'].apply(lambda x: 1 if 'ASMA,' in str(x) else 0)
df['CANCER'] = df['comorbidades'].apply(lambda x: 1 if 'CANCER' in str(x) else 0)
df['DIABETES'] = df['comorbidades'].apply(lambda x: 1 if 'DIABETES' in str(x) else 0)
df['DOENCA_HEMATOLOGICA_CRONICA'] = df['comorbidades'].apply(lambda x: 1 if 'DOENCA HEMATOLOGICA CRONICA' in str(x) else 0)
df['DOENCA_CARDIOVASCULAR_CRONICA'] = df['comorbidades'].apply(lambda x: 1 if 'DOENCA CARDIOVASCULAR CRONICA' in str(x) else 0)
df['DOENCA_RENAL_CRONICA'] = df['comorbidades'].apply(lambda x: 1 if 'DOENCA RENAL CRONICA' in str(x) else 0)
df['DOENCA_NEUROLOGICA_CRONICA'] = df['comorbidades'].apply(lambda x: 1 if 'DOENCA NEUROLOGICA CRONICA' in str(x) else 0)
df['DOENCA_HEPATICA_CRONICA'] = df['comorbidades'].apply(lambda x: 1 if 'DOENCA HEPATICA CRONICA' in str(x) else 0)
df['DOENCA_PNEUMATICA_CRONICA'] = df['comorbidades'].apply(lambda x: 1 if 'DOENCA PNEUMATICA CRONICA' in str(x) else 0)
```

```
df['OBESIDADE'] = df['comorbidades'].apply(lambda x: 1 if 'OBESIDADE' in str(x) else 0)
df['HIPERTENSAO'] = df['comorbidades'].apply(lambda x: 1 if 'HIPERTENSAO' in str(x) else 0)
df['IMUNODEPRESSAO'] = df['comorbidades'].apply(lambda x: 1 if 'IMUNODEPRESSAO' in str(x) else 0)
df['SINDROME_DE_DOWN'] = df['comorbidades'].apply(lambda x: 1 if 'SINDROME DE DOWN' in str(x) else 0)

del df['comorbidades']
```

Agora iremos mapear os dados das colunas: 'internacao', 'internacao_uti' e 'obito'.

Marcaremos 1 para positivo referente aquele indicador e 0 para negativo.

Para facilitar o processo, criaremos um dicionário e depois mapearemos ele nas colunas utilizando a função '.map()' do pandas.

```
dic_internacao_obito = {'INTERNADO':1,
                        'NAO INTERNADO':0,
                        'INTERNADO UTI':1,
                        'NAO INTERNADO UTI':0,
                        'SIM':1,
                        'NAO':0,
                        }
df['internacao'] = df['internacao'].map(dic_internacao_obito)
df['internacao_uti'] = df['internacao_uti'].map(dic_internacao_obito)
df['obito'] = df['obito'].map(dic_internacao_obito)
```

Aplicando um modelo LinearSVC sobre a base

Separando a base em conjuntos de treinamento e teste

```
X_train, X_test, y_train, y_test = train_test_split(
    df.drop(['obito','sexo'], axis=1).fillna(0),
    df['obito'],
    test_size=0.2,
    random_state=42
)
```

Cria o modelo LinearSVC e ajusta aos dados de treinamento

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

```
/usr/local/lib/python3.10/dist-packages/sklearn/svm/_base.py:1244: ConvergenceWarn
  warnings.warn(
▾ LinearSVC
LinearSVC()
```

Faz a predição dos dados de teste e exibe um relatório de classificação

```
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred))
```

```
                  precision    recall  f1-score   support

               0       1.00      0.99      1.00    207285
               1       0.57      0.59      0.58      2430

        accuracy                           0.99    209715
       macro avg       0.78      0.79      0.79    209715
    weighted avg       0.99      0.99      0.99    209715
```

✓ 0s conclusão: 23:14