Regressão_Logística_em_Python

April 29, 2025

1 Regressão Logística em Python

2 Importar as bibliotecas

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

3 Inserir os dados

[ ]:
```
titanic = pd.read_csv('titanic_train.csv')
```

[ ]:
```
titanic.head()
```

[ ]:

	PassengerId	Survived	Pclass	…	Fare	Cabin	Embarked
0	1	0	3	…	7.2500	NaN	S
1	2	1	1	…	71.2833	C85	C
2	3	1	3	…	7.9250	NaN	S
3	4	1	1	…	53.1000	C123	S
4	5	0	3	…	8.0500	NaN	S

[5 rows x 12 columns]

[ ]:
```
titanic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
```

```
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

[ ]:
```
titanic.isnull()
```

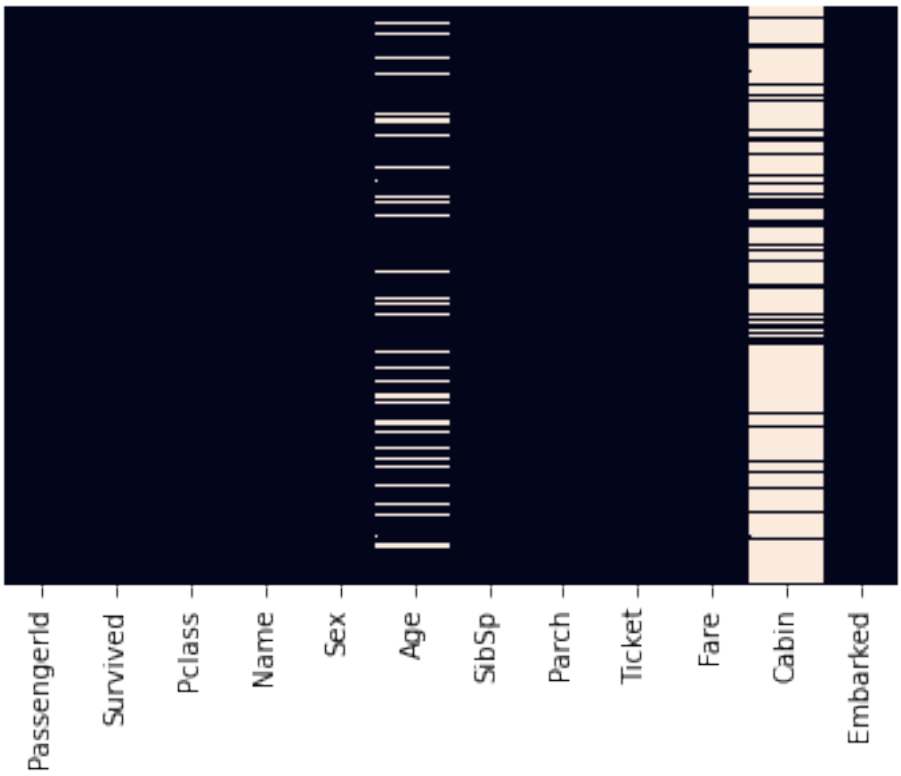
[ ]:

	PassengerId	Survived	Pclass	Name	…	Ticket	Fare	Cabin	Embarked
0	False	False	False	False	…	False	False	True	False
1	False	False	False	False	…	False	False	False	False
2	False	False	False	False	…	False	False	True	False
3	False	False	False	False	…	False	False	False	False
4	False	False	False	False	…	False	False	True	False
..	…	…	…	…	…	…	…	…	
886	False	False	False	False	…	False	False	True	False
887	False	False	False	False	…	False	False	False	False
888	False	False	False	False	…	False	False	True	False
889	False	False	False	False	…	False	False	False	False
890	False	False	False	False	…	False	False	True	False

[891 rows x 12 columns]

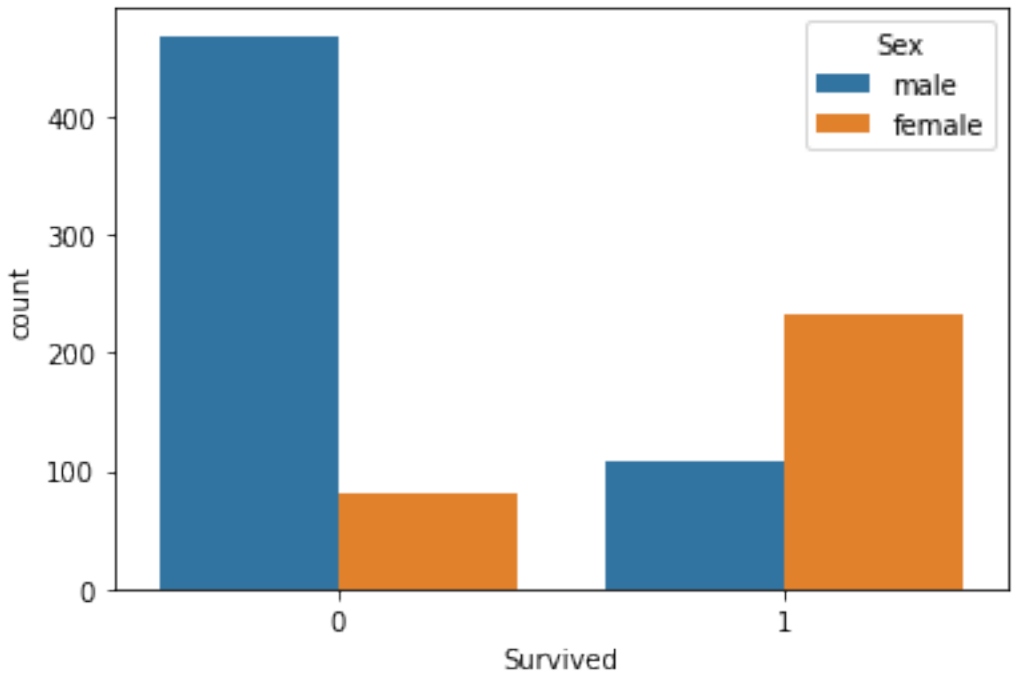
[ ]:
```
sns.heatmap(titanic.isnull(), yticklabels=False, cbar=False)
```

[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f74df41ca50>
```



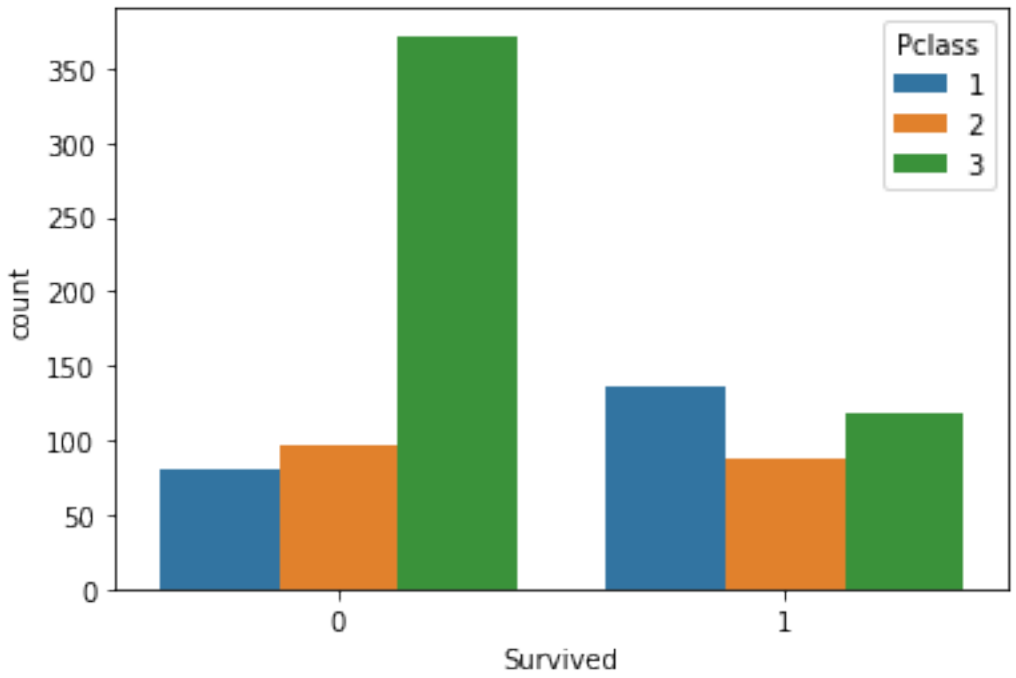
[ ]:
```
sns.countplot(x='Survived', hue='Sex', data=titanic)
```

[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f74d685f350>
```



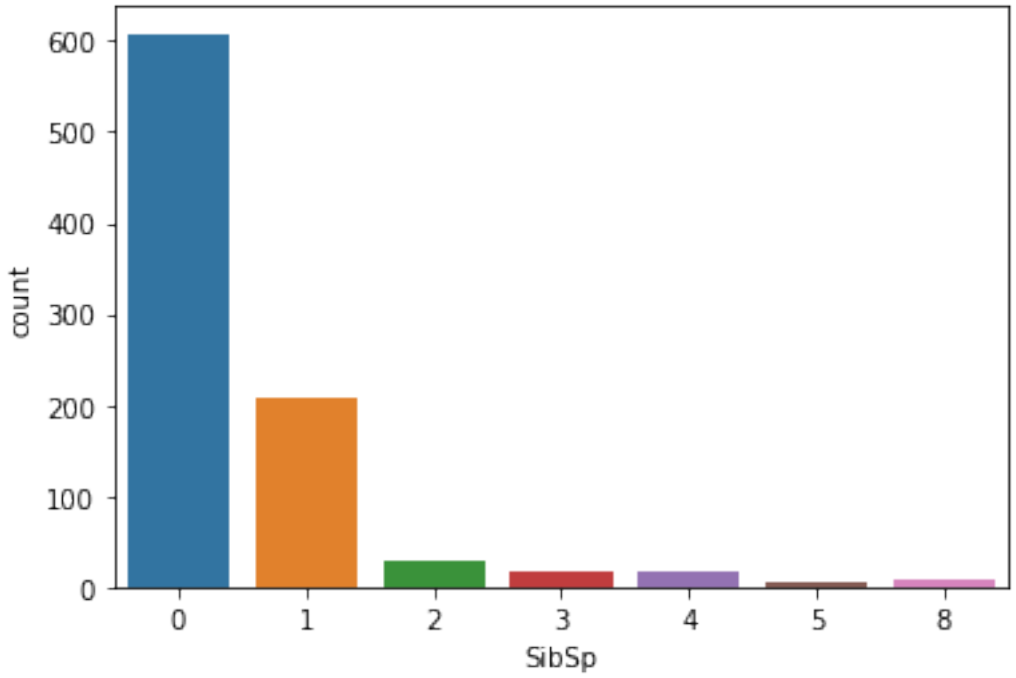
[ ]:
```
sns.countplot(x='Survived', hue='Pclass', data=titanic)
```

[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f74df41c0d0>
```



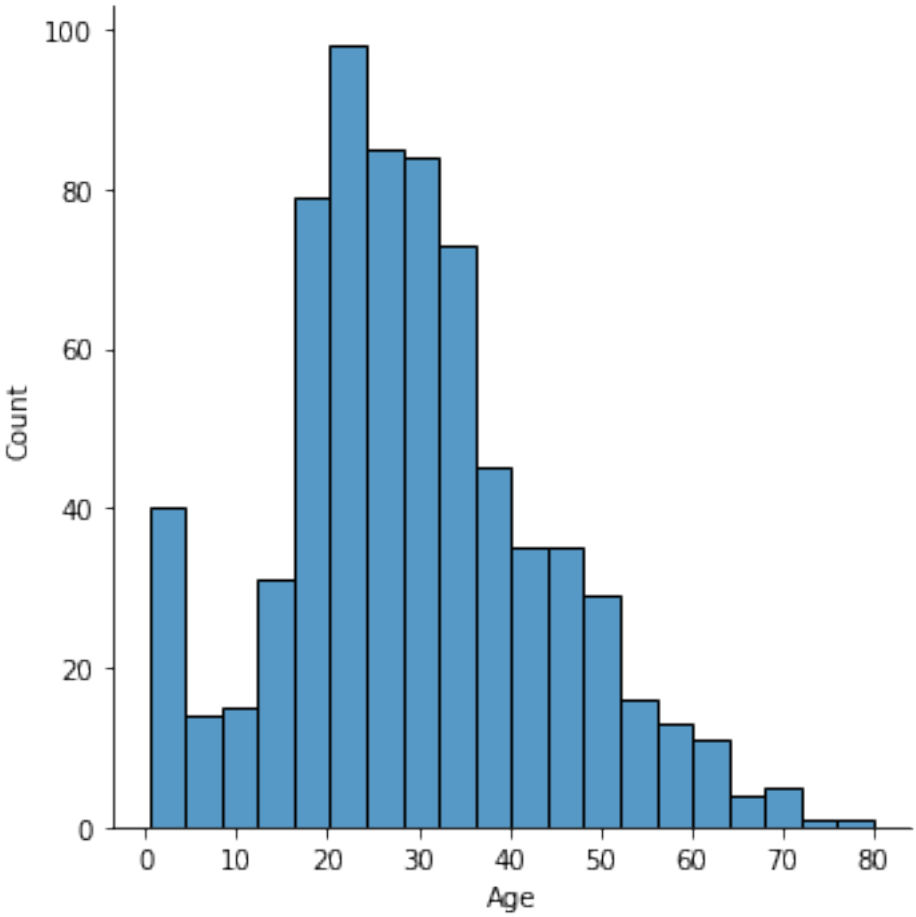
[ ]:
```
sns.countplot(x='SibSp', data=titanic)
```

[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f74d4f84910>
```



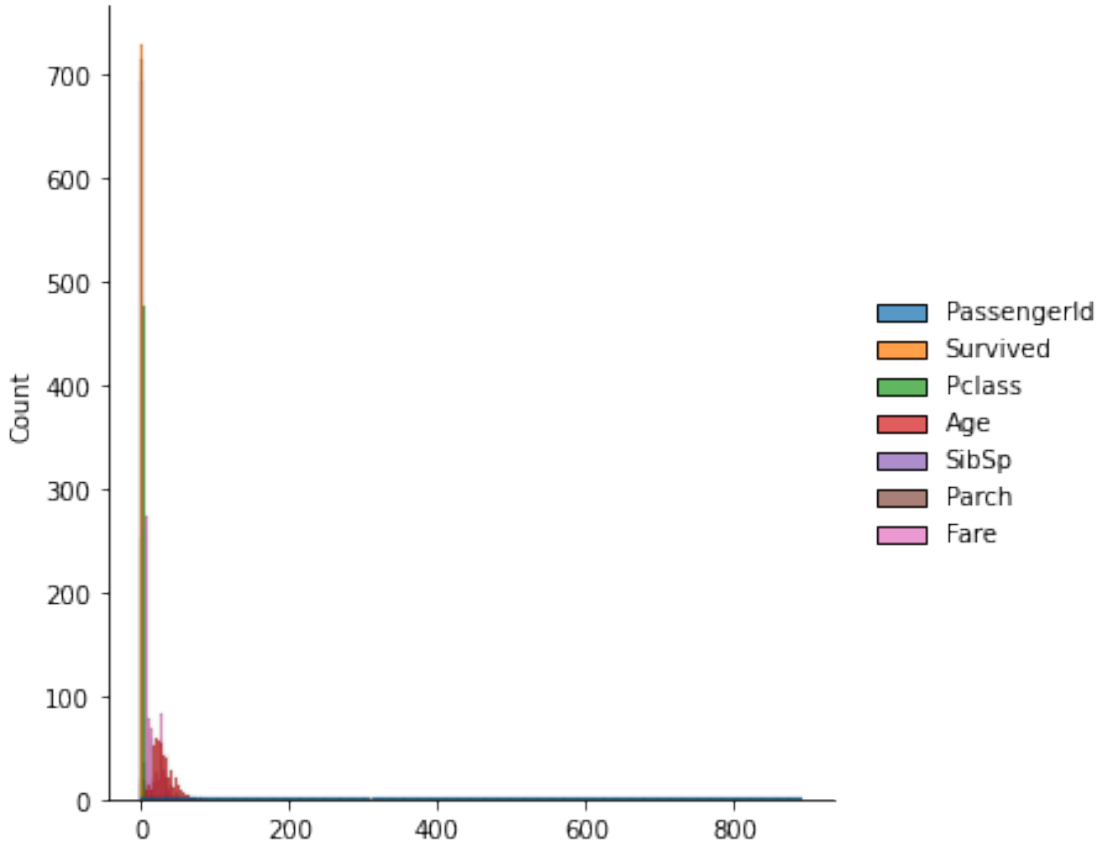
[ ]:
```
sns.displot(titanic['Age'].dropna())
```

[ ]:
```
<seaborn.axisgrid.FacetGrid at 0x7f74d508f990>
```



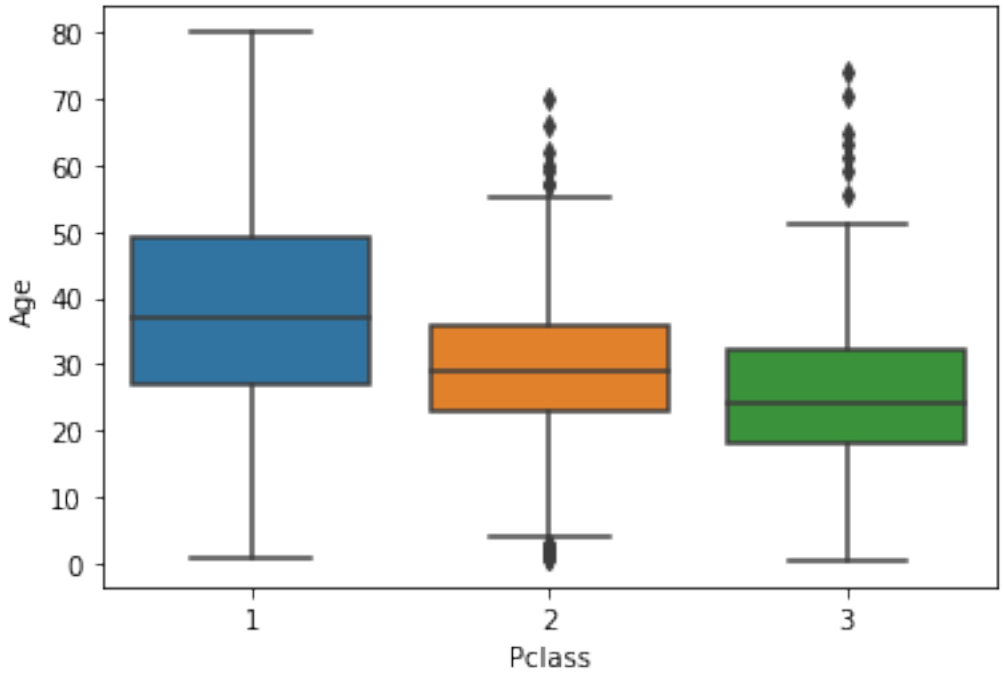
[ ]:
```
sns.displot(titanic[titanic['Fare']<50])
```

[ ]:
```
<seaborn.axisgrid.FacetGrid at 0x7f74df0fd750>
```



[ ]:
```
sns.boxplot(x='Pclass', y='Age', data=titanic)
```

[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f74d4d29650>
```

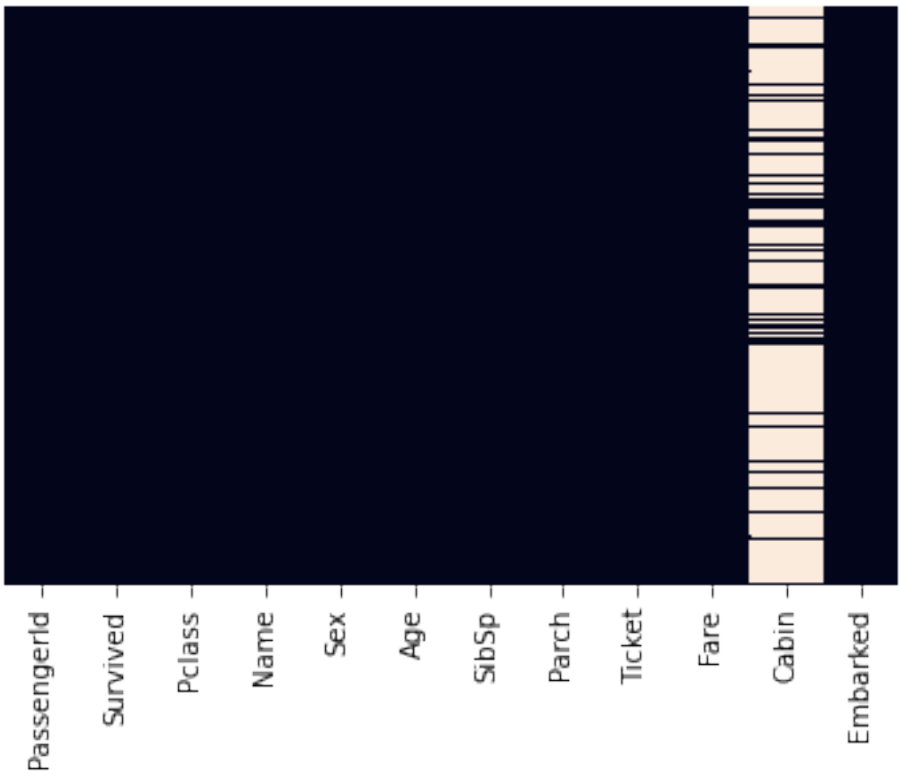


[ ]:
```
def input_idade(cols):
  Age = cols[0]
  Pclass = cols[1]

  if pd.isnull(Age):
    if Pclass == 1:
      return 39
    elif Pclass == 2:
      return 30
    else:
      return 25

  else:
    return Age
```

[ ]:
```
titanic['Age']= titanic[['Age', 'Pclass']].apply(input_idade, axis=1)
```

[ ]:
```
sns.heatmap(titanic.isnull(), yticklabels=False, cbar=False)
```

[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f74df453610>
```



[ ]:
```
titanic.drop('Cabin', axis=1, inplace=True)
```

[ ]:
```
titanic.head()
```

[ ]:

	PassengerId	Survived	Pclass	…	Ticket	Fare	Embarked
0	1	0	3	…	A/5 21171	7.2500	S
1	2	1	1	…	PC 17599	71.2833	C
2	3	1	3	…	STON/O2. 3101282	7.9250	S
3	4	1	1	…	113803	53.1000	S
4	5	0	3	…	373450	8.0500	S

[5 rows x 11 columns]

[ ]:
```
sexo=pd.get_dummies(titanic['Sex'],drop_first=True)
```

[ ]:
```
embarque=pd.get_dummies(titanic['Embarked'],drop_first=True)
```

[ ]:
```
titanic.drop(['Sex', 'Name', 'Ticket', 'Embarked'], axis=1, inplace=True)
```

[ ]:
```
titanic.head()
```

[ ]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
0	1	0	3	22.0	1	0	7.2500
1	2	1	1	38.0	1	0	71.2833
2	3	1	3	26.0	0	0	7.9250
3	4	1	1	35.0	1	0	53.1000
4	5	0	3	35.0	0	0	8.0500

[ ]:
```
titanic = pd.concat([titanic,sexo,embarque], axis=1)
```

[ ]:
```
titanic.head()
```

[ ]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	male	Q	S
0	1	0	3	22.0	1	0	7.2500	1	0	1
1	2	1	1	38.0	1	0	71.2833	0	0	0
2	3	1	3	26.0	0	0	7.9250	0	0	1
3	4	1	1	35.0	1	0	53.1000	0	0	1
4	5	0	3	35.0	0	0	8.0500	1	0	1

#Regressão Logistíca

[ ]:
```
from sklearn.model_selection import train_test_split
```

[ ]:
```
X_treino, X_teste, Y_treino, Y_teste =  train_test_split(titanic.
↪drop('Survived',axis=1), titanic['Survived'], train_size=0.7,␣
↪random_state=56)
```

[ ]:
```
from sklearn.linear_model import LogisticRegression
```

[ ]:
```
RL=LogisticRegression()
```

[ ]:
```
RL.fit(X_treino,Y_treino)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:940:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

[ ]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

[ ]:
```
preditor = RL.predict(X_teste)
```

[ ]:
```
from sklearn.metrics import  classification_report
from sklearn.metrics import confusion_matrix
```

[ ]:
```
print(classification_report(Y_teste,preditor))
```

```
              precision    recall  f1-score   support

           0       0.86      0.89      0.88       178
           1       0.76      0.72      0.74        90

    accuracy                           0.83       268
   macro avg       0.81      0.80      0.81       268
weighted avg       0.83      0.83      0.83       268
```

[ ]:
```
print(confusion_matrix(Y_teste, preditor))
```

```
[[158  20]
 [ 25  65]]
```