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

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
```

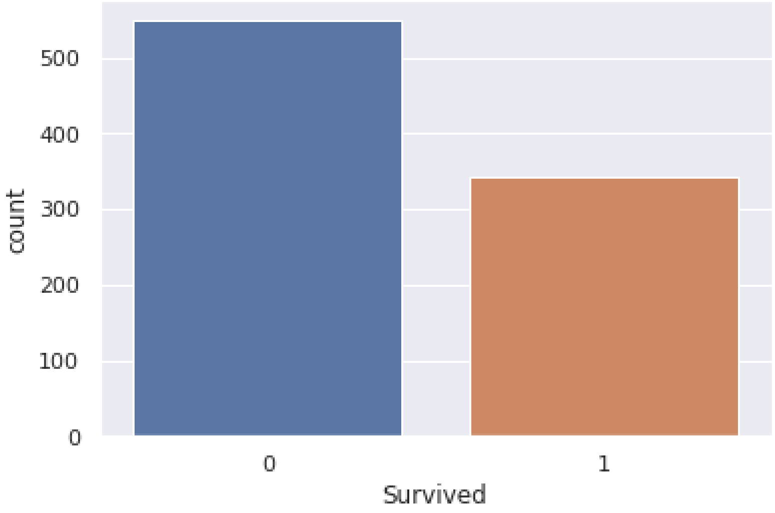
```
train = pd.read_csv('/content/train.csv')
test = pd.read_csv('/content/test.csv')
```

```
print('Number of passengers in train dataset: ' + str(len(train)))
```

```
    Number of passengers in train dataset: 891
```

```
sns.countplot(x = 'Survived', data = train)
```

```
    <matplotlib.axes._subplots.AxesSubplot at 0x7fdb53953310>
```

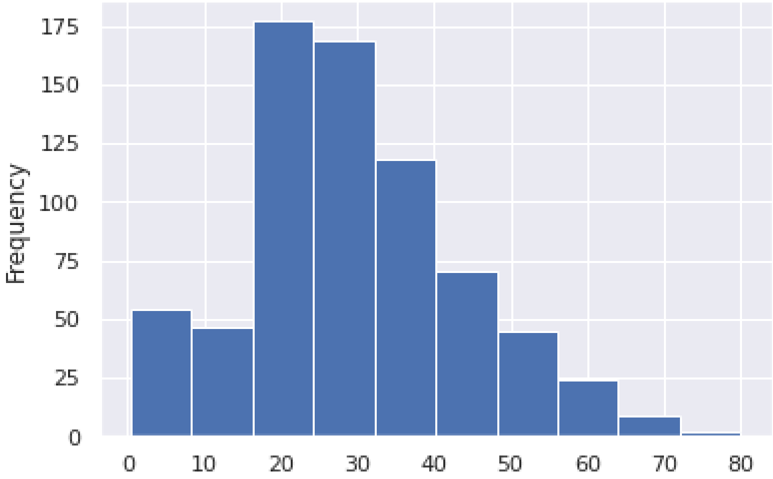


```
sns.countplot(x = 'Survived', hue = 'Pclass', data = train)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdb538ffed0>
```

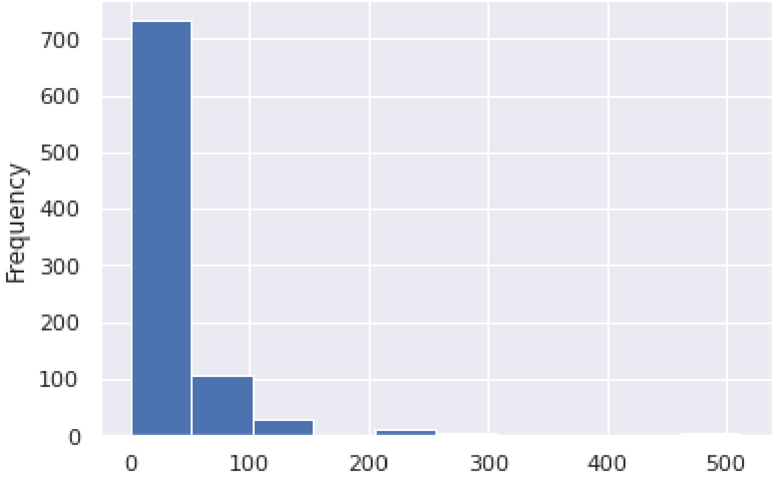
```
train['Age'].plot.hist()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdb5263c550>
```



```
train['Fare'].plot.hist()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdb5254a3d0>
```

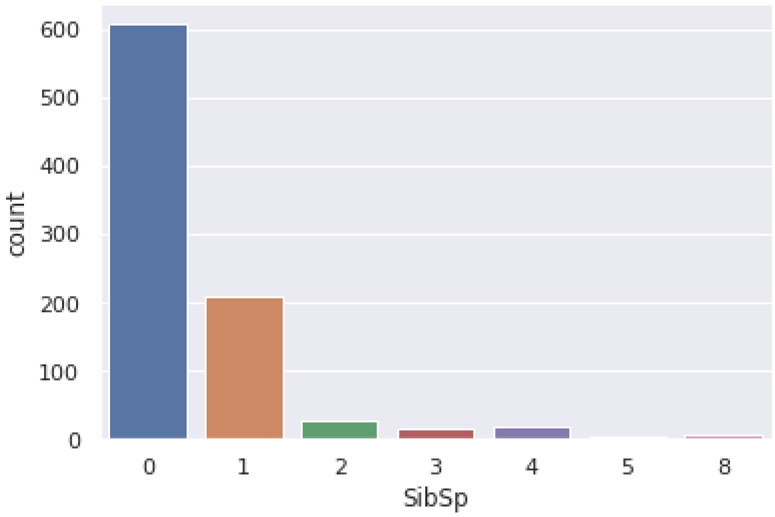


```
train.info()
```

```
<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
```

```
sns.countplot(x = 'SibSp', data = train)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdb524e4990>
```

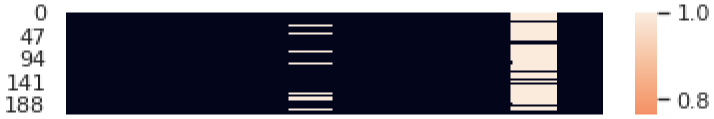


```
train.isnull().sum()
```

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

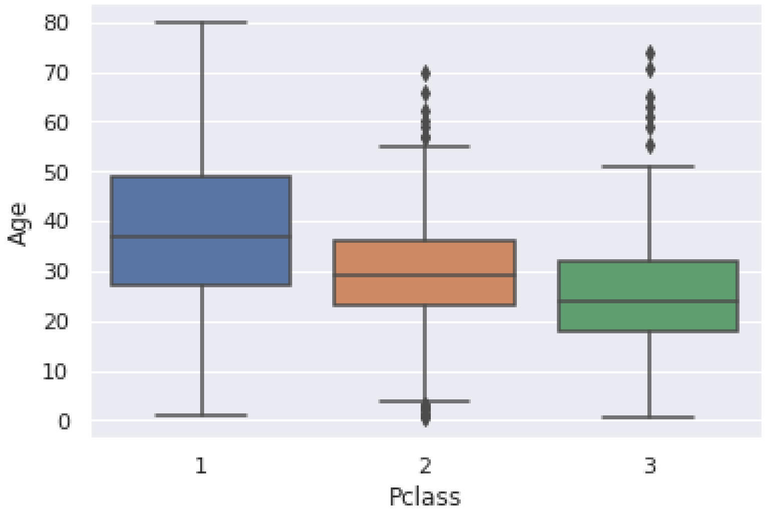
```
sns.heatmap(train.isnull())
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdb5244a4d0>
```



```
sns.boxplot(x = 'Pclass', y = 'Age', data = train)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdb5239ea90>
```



```
sex = pd.get_dummies(train['Sex'], drop_first = True)
embark = pd.get_dummies(train['Embarked'],drop_first=True)
pcl = pd.get_dummies(train['Pclass'],drop_first=True)
```

```
train = pd.concat([train,sex,embark,pcl],axis=1)
train.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs	female	38.0	1	0	PC 17599	71.

```
train.drop(['Pclass','Sex','Embarked','Cabin','PassengerId','Name','Ticket'],axis=1, inpla
train.head()
train.isnull().sum()
```

```
Survived      0
Age         177
SibSp         0
Parch         0
Fare          0
male          0
Q             0
S             0
2             0
3             0
dtype: int64
```

```
train_values = {'Age': round(np.mean(train['Age']))}
train = train.fillna(value = train_values)
train.head()
```

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

```
sex = pd.get_dummies(test['Sex'], drop_first = True)
embark = pd.get_dummies(test['Embarked'],drop_first=True)
pcl = pd.get_dummies(test['Pclass'],drop_first=True)
```

```
test = pd.concat([test,sex,embark,pcl],axis=1)
test.head()
```

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN

```
test_values = {'Age':round(np.mean(test['Age'])), 'Fare':round(np.mean(test['Fare']))}
test = test.fillna(value = test_values)
test.head()
```

	Age	SibSp	Parch	Fare	male	Q	S	2	3
0	34.5	0	0	7.8292	1	1	0	0	1
1	47.0	1	0	7.0000	0	0	1	0	1
2	62.0	0	0	9.6875	1	1	0	1	0
3	27.0	0	0	8.6625	1	0	1	0	1

```
X = train.drop('Survived',axis=1)
y = train['Survived']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)
```

```
logmodel = LogisticRegression(solver = 'liblinear')
```

```
logmodel.fit(X_train, y_train)
```

```
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='liblinear', tol=0.0001, verbose=0,
                   warm_start=False)
```

```
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='liblinear', tol=0.0001, verbose=0,
                   warm_start=False)
```

```
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='liblinear', tol=0.0001, verbose=0,
                   warm_start=False)
```

```
predections = logmodel.predict(X_test)
```

```
print(classification_report(y_test, predections))
```

```
              precision    recall  f1-score   support

           0       0.77      0.88      0.82       153
           1       0.81      0.65      0.72       115

    accuracy                           0.78       268
   macro avg       0.79      0.77      0.77       268
weighted avg       0.79      0.78      0.78       268
```

```
print(confusion_matrix(y_test, predections))
```

```
[[135  18]
```

```
 [ 40  75]]
```

```
print(accuracy_score(y_test, predections))
```

```
0.7835820895522388
```

```
test_predictions = logmodel.predict(test)
```

```
sub_file = pd.read_csv('/content/gender_submission.csv')
sub_file['Survived'] = test_predictions
sub_file.to_csv('submission.csv',index=False)
```

✓ 0s completed at 2:33 PM ● ✕