```
#loading the dataset
import pandas as pd


data = pd.read_csv("https://raw.githubusercontent.com/datasciencedojo/datasets/master/titanic


#checking the head of the data
data.head(2)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parc
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	
1	2	1	1	Cumings, Mrs. John Bradley	female	38.0	1	

```
#the size of our dataset
data.shape
```

```
(891, 12)
```

```
#overview of the data
data.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
```

```
#more descriptive information
data.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200

```
#ficing the missing data
#delete the missing rows
data = data.dropna(subset=['Embarked'])


# delete the whole cabin
data = data.drop("Cabin",axis=1)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-22-550d4e1fd11e> in <module>
      1 # delete the whole cabin
----> 2 data = data.drop("Cabin",axis=1)

 4 frames 

/usr/local/lib/python3.7/dist-packages/pandas/core/indexes/base.py in drop(self,
labels, errors)
   6015         if mask.any():
   6016             if errors != "ignore":
-> 6017                 raise KeyError(f"{labels[mask]} not found in axis")
   6018             indexer = indexer[~mask]
   6019         return self.delete(indexer)

KeyError: "['Cabin'] not found in axis"
```

SEARCH STACK OVERFLOW

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 889 entries, 0 to 890
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  889 non-null    int64  
 1   Survived     889 non-null    int64  
 2   Pclass       889 non-null    int64  
 3   Name         889 non-null    object 
 4   Sex          889 non-null    object 
 5   Age          889 non-null    float64
 6   SibSp        889 non-null    int64  
```

```
 7   Parch        889 non-null    int64
 8   Ticket       889 non-null    object
 9   Fare         889 non-null    float64
 10  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(4)
memory usage: 83.3+ KB
```

```
#fixing the null values with the mean data of the age
mean=data["Age"].mean()
data["Age"] = data["Age"].fillna(mean)
```

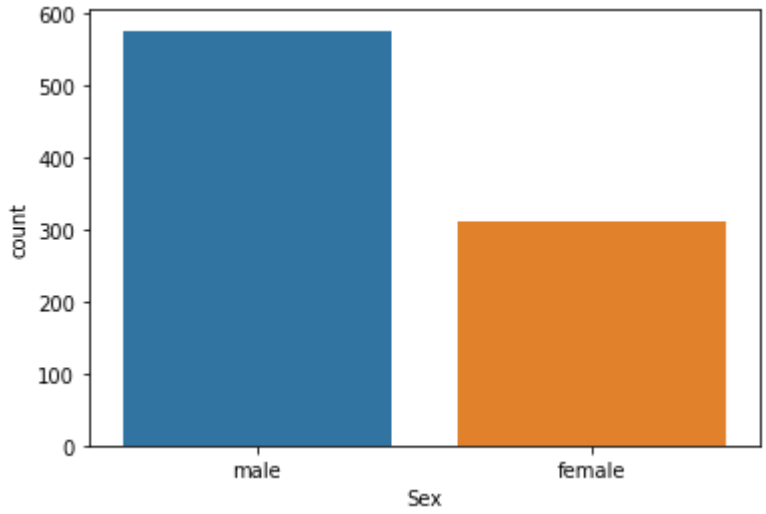
```
data.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Pa
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	

```
#visualizing the amount of male and female
import seaborn as sns
```

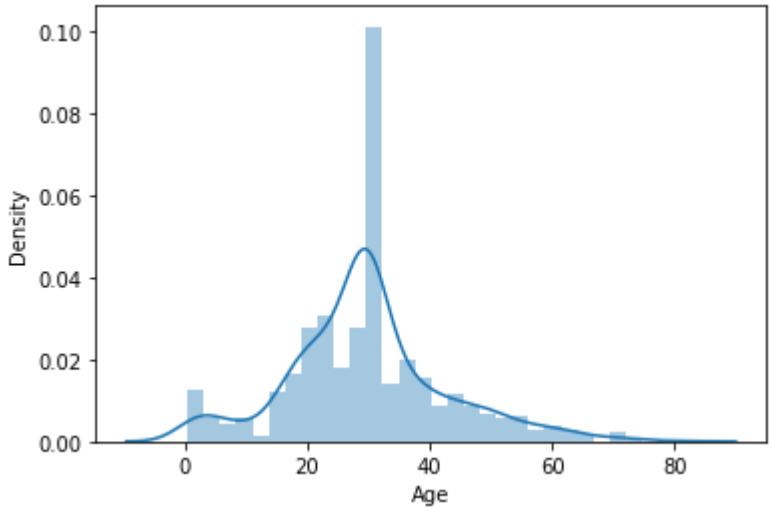
```
sns.countplot(x='Sex', data=data)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7feefc088490>
```



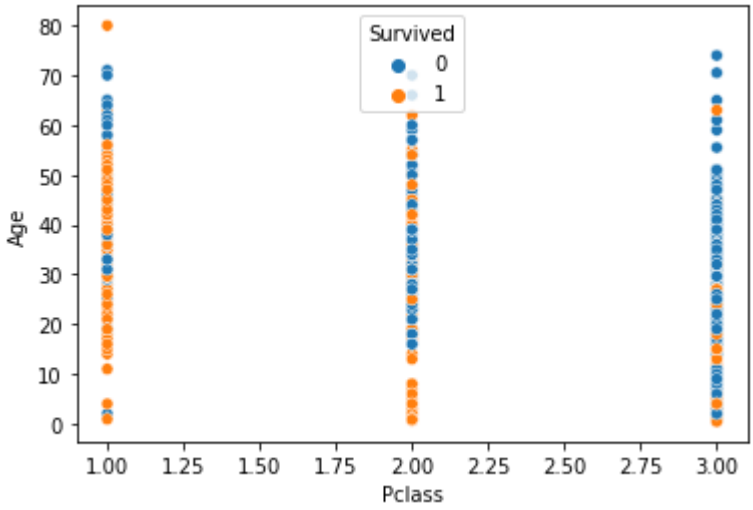

```
#visualizing the distribution of the ages
sns.distplot(data["Age"])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `di
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7feefba591d0>
```




```
#scatter plot
sns.scatterplot(data=data, x=data["Pclass"], y=data["Age"], hue=data["Survived"])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7feefb8e5c10>
```




```
#plotting the swarm plot so that i can count the ages better
sns.swarmplot(x="Pclass", y="Age", hue="Survived", data=data)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/categorical.py:1296: UserWarning: 11.7% o
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/categorical.py:1296: UserWarning: 6.5% of
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/categorical.py:1296: UserWarning: 43.8% o
  warnings.warn(msg, UserWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7feefb82f290>
```


```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 889 entries, 0 to 890
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  889 non-null    int64  
 1   Survived     889 non-null    int64  
 2   Pclass       889 non-null    int64  
 3   Name         889 non-null    object 
 4   Sex          889 non-null    object 
 5   Age          889 non-null    float64
 6   SibSp        889 non-null    int64  
 7   Parch        889 non-null    int64  
 8   Ticket       889 non-null    object 
 9   Fare         889 non-null    float64
 10  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(4)
memory usage: 115.6+ KB
```

```
data["PassengerId"].nunique()
```

```
889
```

```
data["Name"].nunique()
```

```
889
```

```
data["Ticket"].nunique()
```

```
680
```

```
#deleting the non related text like information
data = data.drop("PassengerId",axis=1)

data = data.drop("Name",axis=1)

data = data.drop("Ticket",axis=1)
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 889 entries, 0 to 890
Data columns (total 8 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Survived  889 non-null    int64  
 1   Pclass    889 non-null    int64  
 2   Sex       889 non-null    object 
 3   Age       889 non-null    float64
 4   SibSp     889 non-null    int64  
 5   Parch     889 non-null    int64  
 6   Fare      889 non-null    float64
 7   Embarked  889 non-null    object 
dtypes: float64(2), int64(4), object(2)
memory usage: 94.8+ KB
```

```
data["Embarked"]
```

```
0      S
1      C
2      S
3      S
4      S
      ..
886    S
887    S
888    S
889    C
890    Q
Name: Embarked, Length: 889, dtype: object
```

```
#lets convert the non numeric data to numeric
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
for col in ["Sex", "Embarked"]:
  le.fit(data[col])
  data[col]=le.transform(data[col])
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 889 entries, 0 to 890
Data columns (total 8 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Survived  889 non-null    int64  
 1   Pclass    889 non-null    int64  
 2   Sex       889 non-null    int64  
 3   Age       889 non-null    float64
```

```
 4   SibSp     889 non-null    int64
 5   Parch     889 non-null    int64
 6   Fare      889 non-null    float64
 7   Embarked  889 non-null    int64
dtypes: float64(2), int64(6)
memory usage: 94.8 KB
```

```
data.head()
```

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

```
data["Age"].max()
```

```
80.0
```

```
data["Fare"].max()
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-49-83b89a0e7807> in <module>
----> 1 data["Fare"].max()

IndexError: only integers, slices (`:`), ellipsis (`...`), numpy.newaxis (`None`) and
integer or boolean arrays are valid indices
```

SEARCH STACK OVERFLOW

```
#lets scale and normalize the data
from sklearn.preprocessing import MinMaxScaler

Scaler = MinMaxScaler()
Scaler.fit(data)
data = Scaler.transform(data)

print("Min ", data.min())
print("Max ", data.max())
```

```
Min  0.0
Max  1.0
```

```
#when you scale your data it will be conveeted to numpy array
```

```
data
```

```
array([[0.        , 1.        , 1.        , ..., 0.        , 0.01415106,
        1.        ],
       [1.        , 0.        , 0.        , ..., 0.        , 0.13913574,
        0.        ],
       [1.        , 1.        , 0.        , ..., 0.        , 0.01546857,
        1.        ],
       ...,
       [0.        , 1.        , 0.        , ..., 0.33333333, 0.04577135,
        1.        ],
       [1.        , 0.        , 1.        , ..., 0.        , 0.0585561 ,
        0.        ],
       [0.        , 1.        , 1.        , ..., 0.        , 0.01512699,
        0.5       ]])
```

```
#Lets create the train and test sets x=data[:,1:]--> : -> means all rows 1 denotes from row 1
from sklearn.model_selection import train_test_split

X=data[:,1:]
y=data[:,0]

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


#functions to check the accuracy and confusion matrix
from sklearn.metrics import confusion_matrix


#classify with linera regression
from sklearn.linear_model import LogisticRegression

classifier = LogisticRegression()
classifier.fit(X_train, y_train)
```

```
LogisticRegression()
```

```
#predict the output of the model
y_pred = classifier.predict(X_test)


y_pred
```

```
array([0., 0., 1., 0., 0., 1., 0., 0., 0., 0., 1., 0., 1., 1., 0., 0., 0.,
       0., 0., 0., 1., 0., 1., 0., 1., 0., 0., 0., 1., 1., 0., 1., 1., 1.,
       1., 0., 0., 1., 0., 0., 1., 1., 1., 1., 0., 1., 0., 1., 0., 1., 0.,
       0., 0., 0., 1., 0., 1., 0., 0., 1., 1., 1., 0., 1., 1., 0., 0., 0.,
       1., 1., 0., 1., 0., 1., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 1.,
       0., 0., 1., 1., 1., 0., 1., 1., 1., 0., 0., 0., 1., 0., 1., 0., 0.,
       0., 0., 1., 0., 0., 0., 0., 1., 1., 0., 1., 1., 1., 0., 0., 0., 0.,
       0., 1., 0., 0., 0., 1., 1., 0., 0., 1., 0., 1., 0., 0., 1., 0., 0.,
       1., 1., 1., 0., 1., 1., 0., 0., 1., 1., 1., 1., 0., 1., 1., 0., 0.,
```

```
       0., 1., 0., 1., 0., 0., 1., 1., 1., 1., 0., 0., 1., 0., 0., 0., 1.,
       0., 1., 1., 1., 1., 1., 0., 0.])
```

```
print(confusion_matrix(y_test,y_pred))
```

```
[[86 25]
 [13 54]]
```

```
from sklearn.matrics import classification_report
print(classification_report(y_test,y_pred))
```

```
---------------------------------------------------------------------------
ImportError                               Traceback (most recent call last)
<ipython-input-69-ba53eae911cd> in <module>
----> 1 from sklearn.linear_model import classification_report

ImportError: cannot import name 'classification_report' from 'sklearn.linear_model'
(/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/__init__.py)

---------------------------------------------------------------------------
NOTE: If your import is failing due to a missing package, you can
manually install dependencies using either !pip or !apt.

To view examples of installing some common dependencies, click the
"Open Examples" button below.
---------------------------------------------------------------------------
```

OPEN EXAMPLES SEARCH STACK OVERFLOW