TITANIC SURVIVA PREDICTION USING MACHINNE LEARING.

By: KRUPALI MISTARI.

Intoduction:

- The Titanic was a famous ship that sank in 1912 after hitting an iceberg. About 1,500 people died, and around 700 survived. Many factors, like lack of lifeboats and slow response, contributed to the high death toll.

- **The dataset I'm using here to train a titanic survival prediction model was downloaded from Kaggle. It contains data about all the main features that contribute to the titanic survival. So let's start this task by importing the necessary Python libraries and the dataset:**

In [1]:
```
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 accuracy_score
```

In [2]:
```
df=pd.read_csv("E:\Titanic-Dataset.csv")
df
```

Out[2]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q

891 rows × 12 columns

- **There are 9 columns in this dataset, so it is very important to check whether or not this dataset contains null values before going any further:**

```
: df.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
```

- **So this dataset doesn't have any null values, now let's look at some of the other important insights to get an idea of what kind of data we're dealing with:**

```
: df.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
```

- **Lets check head and tail of the dataset**
- **With The help of head() function we can able to check our first 5 columns from our dataset**
- **With The help of tail() function we can able to check our last 5 columns from our dataset**

```
df.head()
```

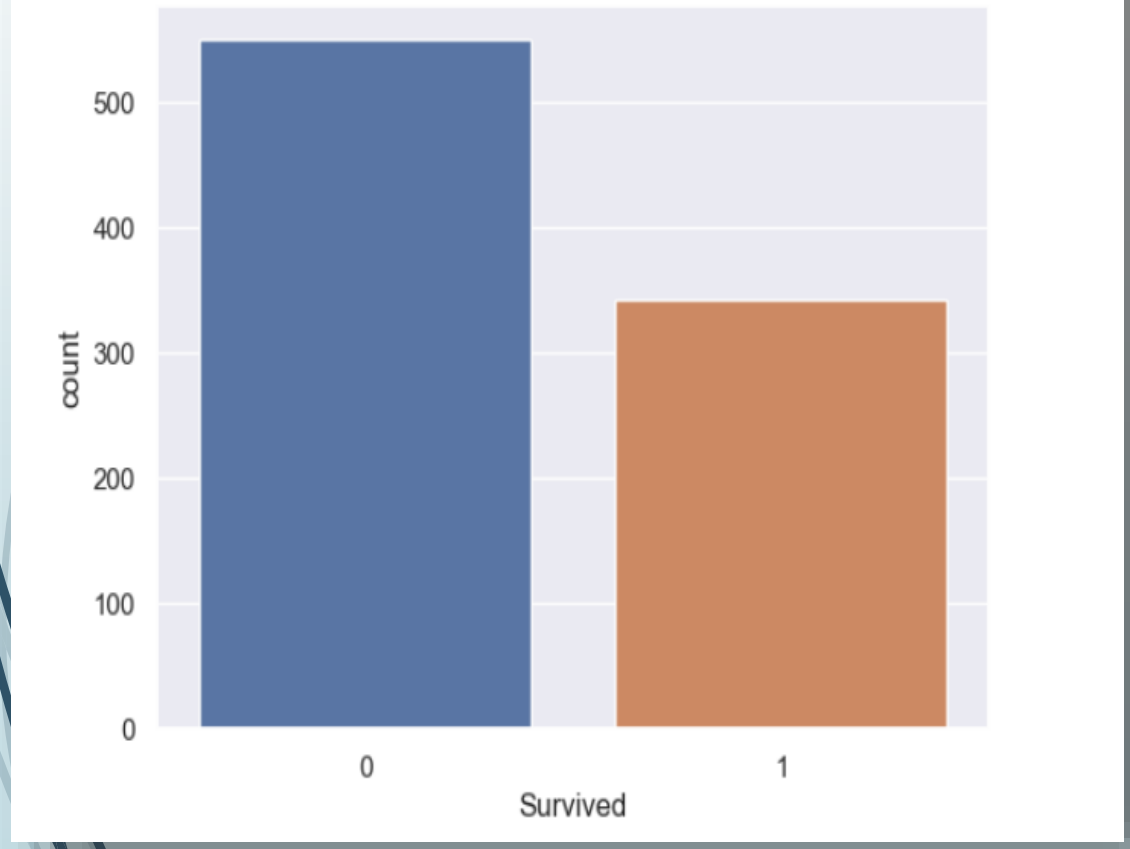
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

```
df.tail()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.00	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.00	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.45	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.00	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.75	NaN	Q

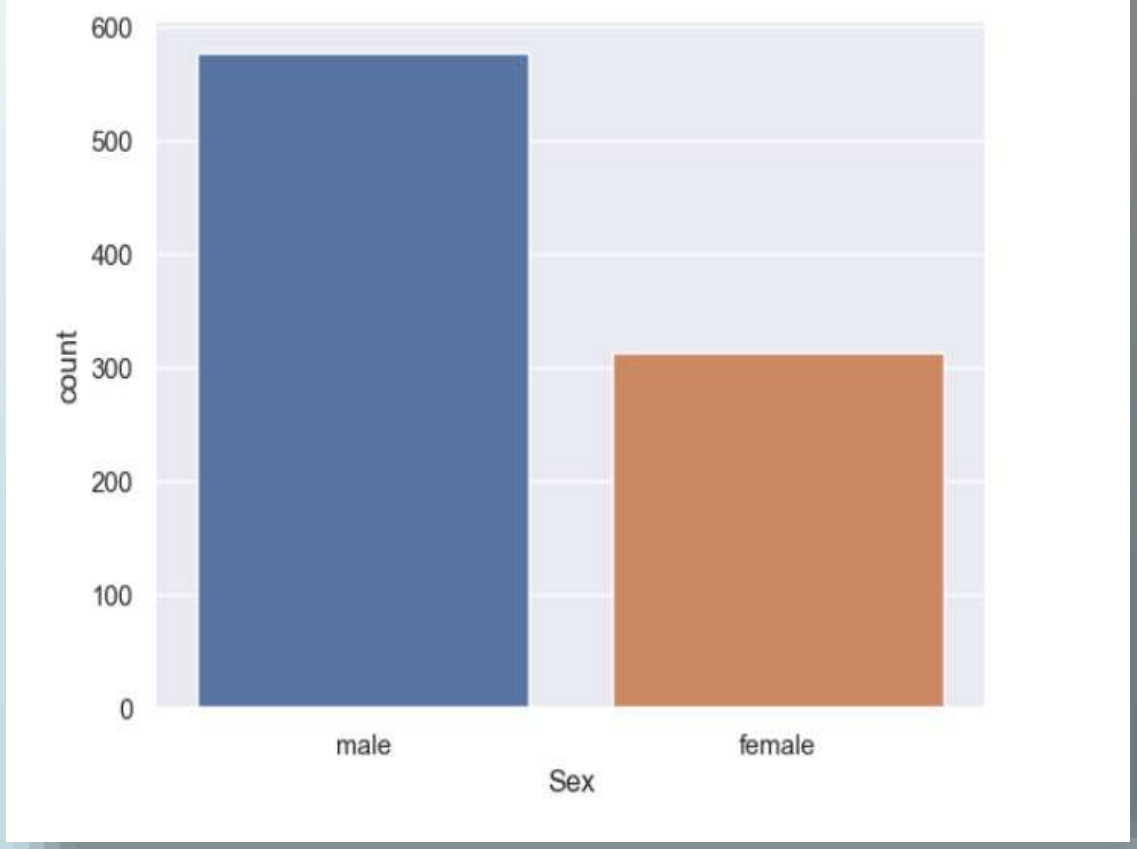
Making a count plot for “Survival ” columns

```
# making a count plot for "Survived" columns
sns.countplot(x='Survived', data=df)
plt.show()  # Display the plot
```



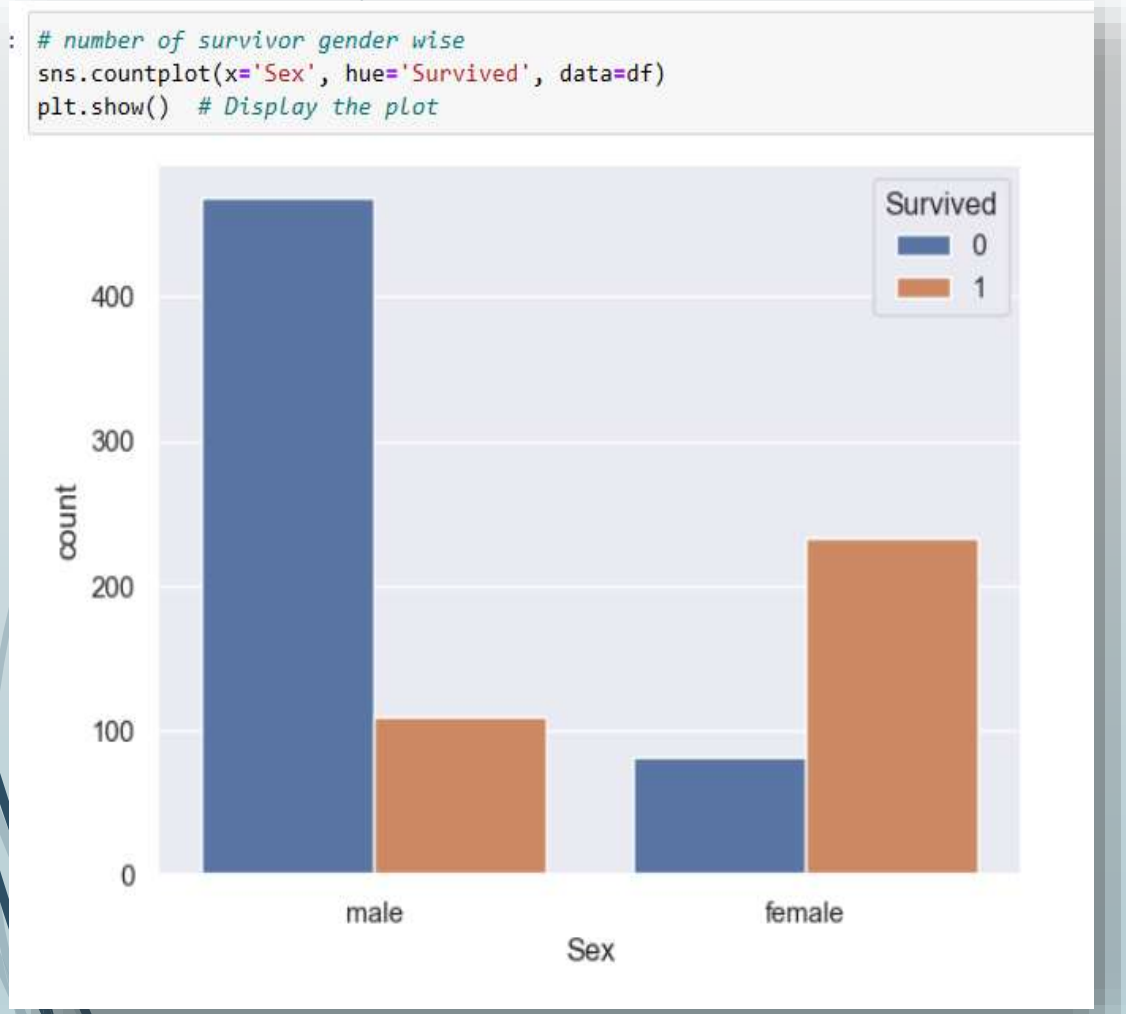
Making a count plot for “Sex” cloumns

```
# making a count plot for 'sex' columns
sns.countplot(x='Sex', data=df)
plt.show()  # Display the plot
```



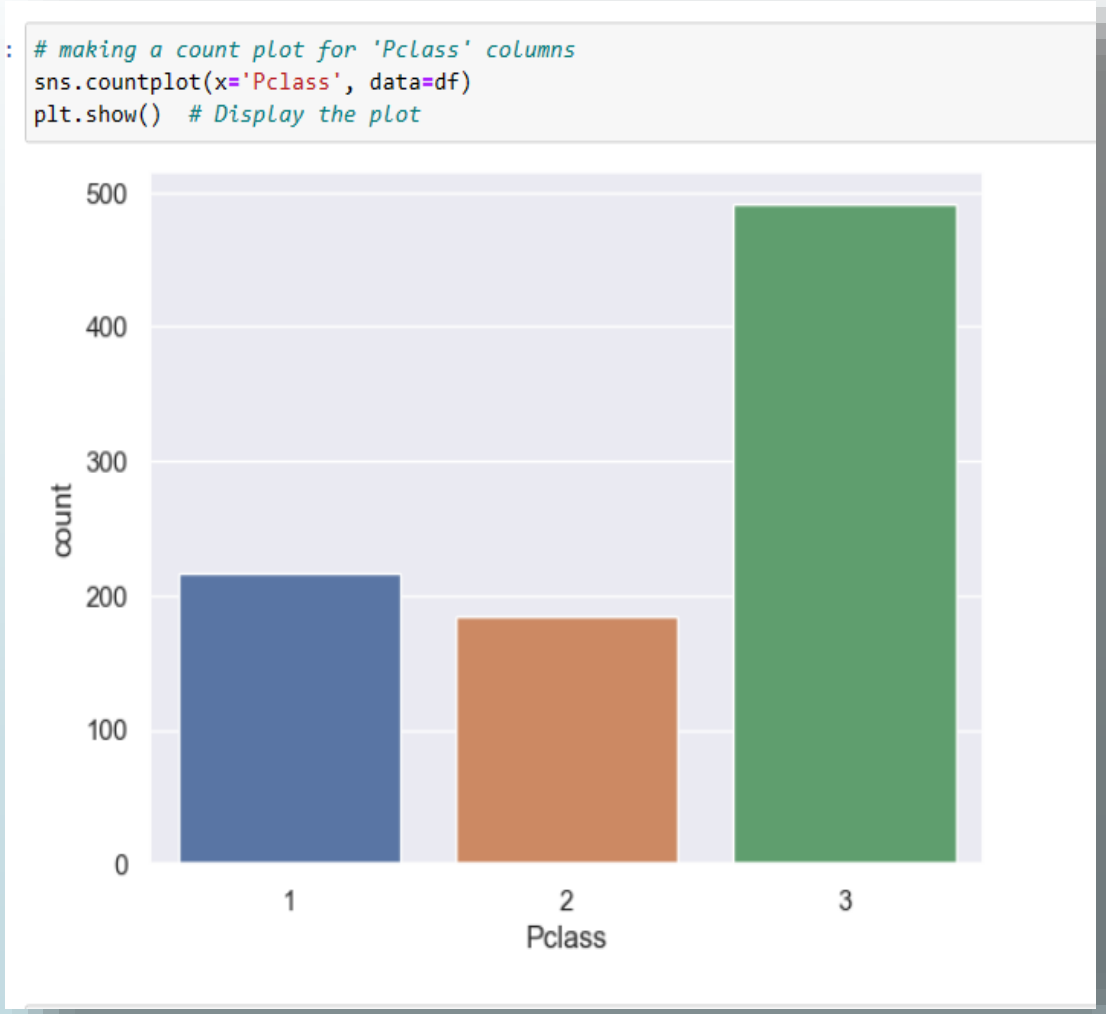
Number of survival gender wise

```
: # number of survivor gender wise
sns.countplot(x='Sex', hue='Survived', data=df)
plt.show()  # Display the plot
```



Making a count plot for “Pclass” columns

```
: # making a count plot for 'Pclass' columns
sns.countplot(x='Pclass', data=df)
plt.show()  # Display the plot
```



- **With The help of shape we can able to check over all shape of our dataset**

```
df.shape
```

```
(891, 12)
```

- **I will use Logistic Regression and the Random Forest clasification algorithm to train a Titanic Survival Prediction model. So let's split the data into training and test sets**

- **Logistic Regression:**

Logistic Regression in Machine LearningLogistic regression is a supervised machine learning algorithm mainly used for classification tasks where the goal is to predict the probability that an instance of belonging to a given class or not. It is a kind of statistical algorithm, which analyze the relationship between a set of independent variables and the dependent binary variables.

- **Random Forest Classification:**

Random Forest Classification is a supervised machine learning technique that builds a collection of decision trees during training and combines their outputs to classify data points into different classes or categories. It is known for its high accuracy, robustness against overfitting, and capability to handle complex datasets with various features.

```
model = LogisticRegression()
```

```
#training the Logistic Regression model with training data
model.fit(X_train, y_train)
```

```
C:\Users\admin\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:460: 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
  n_iter_i = _check_optimize_result(
```

```
LogisticRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
# accuracy on test data
X_test_prediction = model.predict(X_test)
```

```

test_data_accuracy = accuracy_score(y_test, X_test_prediction)

test_data_accuracy
```

```
0.8156424581005587
```

Random forest classification

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=49)

print('shape of x_train=',X_train.shape)
print('shape of y_train=',y_train.shape)
print('shape of x_test=',X_test.shape)
print('shape of y_test=',y_test.shape)
```

```
shape of x_train= (712, 7)
shape of y_train= (712,)
shape of x_test= (179, 7)
shape of y_test= (179,)
```

```
from sklearn.ensemble import RandomForestClassifier
```

```
classifier=RandomForestClassifier(n_estimators=100,criterion='gini')
classifier.fit(X_train,y_train)
```

```
RandomForestClassifier()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
classifier.score(X_test,y_test)
```

```
0.8603351955307262
```

THANK YOU