Pune Institute of Computer Technology

Department of Computer Engineering

(2022- 2023)

"Predict survival of people in the Titanic Shipwreck using Machine Learning"

Submitted to the

Savitribai Phule Pune University

In partial fulfilment for the award of the Degree of

Bachelor of Engineering

in

Computer Engineering

By

1)	**Medha Badamikar**	**41108**
2)	**Udayan Chavan**	**41117**
3)	**Tejas Deshpande**	**41122**

Under the guidance of

Prof. Vaishali Kandekar

Problem Statement

Build a machine learning model that predicts the type of people who survived the Titanic shipwreck using passenger data (Age, Gender, Socio-Economic class).

Objective

To implement classification using Logistic Regression and test accuracy of the trained model.

Theory

Machine Learning:

Machine learning (ML) is a field of inquiry devoted to understanding and building methods that 'learn', that is, methods that leverage data to improve performance on some set of tasks. It is seen as a part of artificial intelligence.

Machine learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so.

Machine learning algorithms are used in a wide variety of applications, such as in medicine, email filtering, speech recognition, and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

Logistic Regression:

This type of statistical model (also known as logit model) is often used for classification and predictive analytics. Logistic regression estimates the probability of an event occurring, such as voted or didn't vote, based on a given dataset of independent variables. Since the outcome is a probability, the dependent variable is bounded between 0 and 1. In logistic regression, a logit transformation is applied on the odds—that is, the probability of success divided by the probability of failure. This is also commonly known as the log odds, or the natural logarithm of odds.

Libraries used:

- **Numpy**: It is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.
- **Pandas**: It is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.
- **Seaborn**: It is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
- **Sklearn**: (Scikit-learn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support-vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

Titanic Dataset:

The dataset contains detailed information about passengers from the famous Titanic ship and the statistics of their survival. It has a total of 891 values that have been reduced to 400 for this particular application. It contains several columns such as survival, pclass, sex, age, sibsp, ticket, parch, fare, cabin and embarked that contain details about the passengers.

This detailed nature of the dataset makes it easier for analysing and drawing relations between various attributes of the passengers.

CODE:

LP-3: ML Mini-Project

Group Members:

- 41108: Medha Badamikar
- 41117: Udayan Chavan
- 41122: Tejas Deshpande

Build a machine learning model that predicts the type of people who survived the Titanic shipwreck using passenger data (Age, Gender, Socio-Economic class)

Importing all required dependencies

In [168]:
```
import numpy as np
import pandas as pd

import seaborn as sns

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

Reading the CSV file into a data frame

In [169]:
```
titanic_data = pd.read_csv('train.csv')
titanic_data.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

Preprocessing

Checking for empty cells

In [170]:
```
titanic_data.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
```

In [171]:
```
# Drop cabin column
titanic_data = titanic_data.drop(columns='Cabin', axis=1)
# Replace missing values in Age by mean value
titanic_data['Age'].fillna(titanic_data['Age'].mean(), inplace=True)

# Can't take mean of Embarked, replacing missing values by mode
print(titanic_data['Embarked'].mode())

titanic_data['Embarked'].fillna(titanic_data['Embarked'].mode()[0], inplace=True)
```

```
0    S
dtype: object
```

Rechecking for missing values

In [172]:
```
titanic_data.isnull().sum()
```

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

Reducing overall volume of dataframe

In [173]:
```
len(titanic_data)
```

```
891
```

In [174]:
```
titanic_data = titanic_data.iloc[:400,]
```

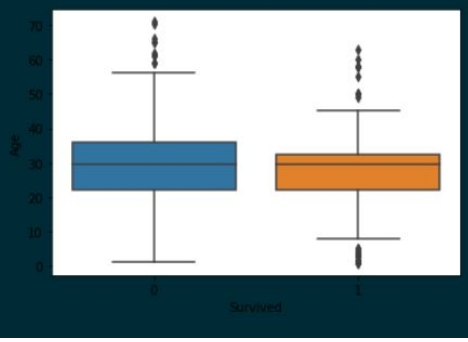
In [175]:
```
len(titanic_data)
```

```
400
```

Visualizing the available data

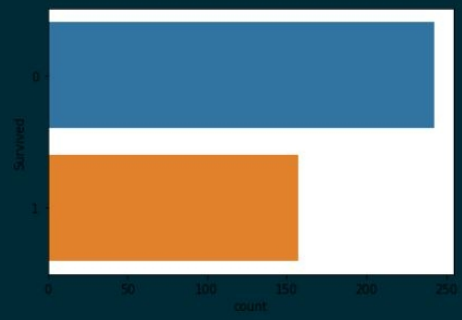
In [176]:
```
sns.boxplot(data=titanic_data, x="Survived", y="Age")
```

```
<AxesSubplot:xlabel='Survived', ylabel='Age'>
```



In [177]:
```
sns.countplot(data=titanic_data, y="Survived")
```

```
<AxesSubplot:xlabel='count', ylabel='Survived'>
```



Transforming into Categorical values

In [178]:
```
titanic_data.replace({'Sex':{'male':0,'female':1}}, inplace=True)
```

In [179]:
```
titanic_data.head()
```

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

Splitting data into feature and target variables

In [180]:
```
# Remove unnecessary columns
X = titanic_data.drop(columns = ['PassengerId','Name','Ticket','Survived','SibSp','Parch','Fare', 'Embarked
Y = titanic_data['Survived']
```

Splitting dataset into Training and Testing in 80:20
X_train: set of values from variable x
Y_train: output (alive/dead) for data from variable x
(Similar for X_test and Y_test)

In [181]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size=0.2, random_state=2)
```

In [182]:
```
print(X_train)
```

```
     Pclass  Sex        Age
205       3    1   2.000000
354       3    0  29.699118
3         1    1  35.000000
264       3    1  29.699118
194       1    1  44.000000
..      ...  ...        ...
299       1    1  50.000000
22        3    1  15.000000
72        2    0  21.000000
15        2    1  55.000000
168       1    0  29.699118

[320 rows x 3 columns]
```

In [183]:
```
print(Y_train)
```

```
205    0
354    0
3      1
264    0
194    1
      ..
299    1
22     1
72     0
15     1
168    0
Name: Survived, Length: 320, dtype: int64
```

Creating a Logistic Regression Model

In [184]:
```
model = LogisticRegression()
```

In [185]:
```
model.fit(X_train, Y_train)
```

```
LogisticRegression()
```

Creating a variable to store predictions

In [186]:
```
X_train_prediction = model.predict(X_train)
```

Checking accuracy of predictions

In [187]:
```
training_data_accuracy = accuracy_score(Y_train, X_train_prediction)
print('Accuracy score of training data : ', training_data_accuracy)
```

```
Accuracy score of training data :  0.83125
```

Checking accuracy with the test data

In [188]:
```
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(Y_test, X_test_prediction)
print('Accuracy score of test data : ', test_data_accuracy)
```

```
Accuracy score of test data :  0.7875
```

Checking for a random person

In [189]:
```
random_person = (3,1,85)
input_as_array = np.asarray(random_person)
print(input_as_array)
```

```
[ 3  1 85]
```

Reshaping values to match our model

In [190]:
```
reshaped_input_as_array = input_as_array.reshape(1,-1)
```

In [191]:
```
print(reshaped_input_as_array)
```

```
[[ 3  1 85]]
```

In [192]:
```
prediction = model.predict(reshaped_input_as_array)

if prediction[0]==0:
    print("Dead")
if prediction[0]==1:
    print("Alive")
```

```
Dead
```

Conclusion

Hence we successfully implemented Logistic Regression on the Titanic Dataset to create a model to calculate probability of a person surviving the Titanic shipwreck and tested it for accuracy.