

# VIVEKANAND EDUCATION SOCIETY'S INSTITUTE OF TECHNOLOGY

Hashu Advani Memorial Complex, Collector's Colony, R C Marg, Chembur, Mumbai-  
400074

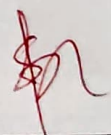


## Department of Artificial Intelligence and Data Science

**Subject:** ML

**Class:** D11AD

**Semester:** VI

Roll No.: 26	Name:  Dyotak Kachare		
Exp No.: 3	Title: <u>Logistic Regression</u>		
DOP:	31/1	DOS:	3/2
GRADE		LAB OUTCOME:  L02	SIGNATURE: 

Name: \_\_\_\_\_ Class & Div. : \_\_\_\_\_ Page No.: \_\_\_\_\_

Subject: \_\_\_\_\_ Topic: \_\_\_\_\_ Date: \_\_\_\_\_

### ML Experiment - 3

Aim.

Logistic regression

Theory -

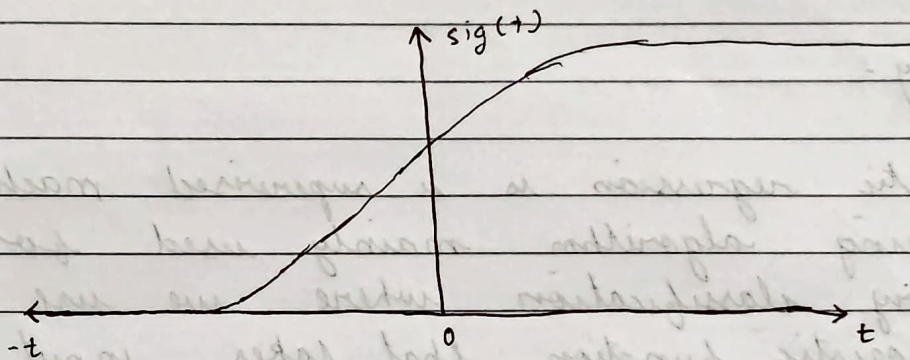
Logistic regression is a supervised machine learning algorithm mainly used for binary classification where we use a logistic function that takes input as independent variable and produce a probability value between 0 and 1.

- Logistic regression predicts the output of a categorical dependent variable. There for the outcome must be categorical or discrete value.
- Logistic regression is much similar to the linear regression except that how they are used. Linear regression is used for regression problem, whereas logistic regression solves classification problem.
- The curve from the logistic function indicated the likelihood of something such as true or false, survived or not survived.



Logistic function (sigmoid function):

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$



- It maps any real value into another value within the range of 0 and 1.
- In logistic regression, we use ~~the~~ a threshold value which defines the probability of either 0 or 1.

### Conclusion

Thus we have successfully implement logistic regression.

```
[47]: import numpy as np
import pandas as pd

from sklearn import preprocessing
import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.simplefilter(action='ignore')
```

```
[22]: # Loading train & test datasets:
train_df = pd.read_csv('./data/train.csv')
test_df = pd.read_csv('./data/test.csv')
```

```
[23]: train_df.head()
```

```
[23]: PassengerId  Survived  Pclass  \
0             1         0         3
1             2         1         1
2             3         1         3
3             4         1         1
4             5         0         3

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

   Parch    Ticket   Fare Cabin Embarked
0      0  A/5 21171   7.2500   NaN        S
1      0    PC 17599  71.2833   C85        C
2      0 STON/O2. 3101282   7.9250   NaN        S
3      0    113803  53.1000  C123        S
4      0    373450   8.0500   NaN        S
```

```
[24]: test_df.head()
```

```
[24]: PassengerId  Pclass                                Name    Sex  \
0           892         3                Kelly, Mr. James   male
1           893         3      Wilkes, Mrs. James (Ellen Needs) female
2           894         2          Myles, Mr. Thomas Francis   male
3           895         3                Wirz, Mr. Albert   male
4           896         3  Hirvonen, Mrs. Alexander (Helga E Lindqvist) female

   Age  SibSp  Parch    Ticket   Fare Cabin Embarked
0  34.5     0     0    330911   7.8292   NaN        Q
1  47.0     1     0    363272   7.0000   NaN        S
2  62.0     0     0    240276   9.6875   NaN        Q
3  27.0     0     0    315154   8.6625   NaN        S
4  22.0     1     1   3101298  12.2875   NaN        S
```

### 0.0.1 Preprocessing Dataset

```
[25]: train_df.isnull().sum()
```

```
[25]: 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
```

```
[26]: train_data = train_df.copy()
      train_data["Age"].fillna(train_df["Age"].median(skipna=True), inplace=True)
      train_data["Embarked"].fillna(train_df['Embarked'].value_counts().idxmax(),
      ↪inplace=True)
      train_data.drop('Cabin', axis=1, inplace=True)
```

```
[27]: train_data.isnull().sum()
```

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

```
[29]: train_data['TravelAlone']=np.where((train_data["SibSp"]+train_data["Parch"])>0, 0,
      ↪1)
      train_data.drop('SibSp', axis=1, inplace=True)
      train_data.drop('Parch', axis=1, inplace=True)
```

```
[30]: training=pd.get_dummies(train_data, columns=["Pclass","Embarked","Sex"])
      training.drop('Sex_female', axis=1, inplace=True)
      training.drop('PassengerId', axis=1, inplace=True)
      training.drop('Name', axis=1, inplace=True)
      training.drop('Ticket', axis=1, inplace=True)

      final_train = training
      final_train.head()
```

```
[30]:
```

	Survived	Age	Fare	TravelAlone	Pclass_1	Pclass_2	Pclass_3	\
0	0	22.0	7.2500	0	0	0	1	
1	1	38.0	71.2833	0	1	0	0	
2	1	26.0	7.9250	1	0	0	1	
3	1	35.0	53.1000	0	1	0	0	
4	0	35.0	8.0500	1	0	0	1	

	Embarked_C	Embarked_Q	Embarked_S	Sex_male
0	0	0	1	1
1	1	0	0	0
2	0	0	1	0
3	0	0	1	0
4	0	0	1	1

## 0.0.2 Traing Model

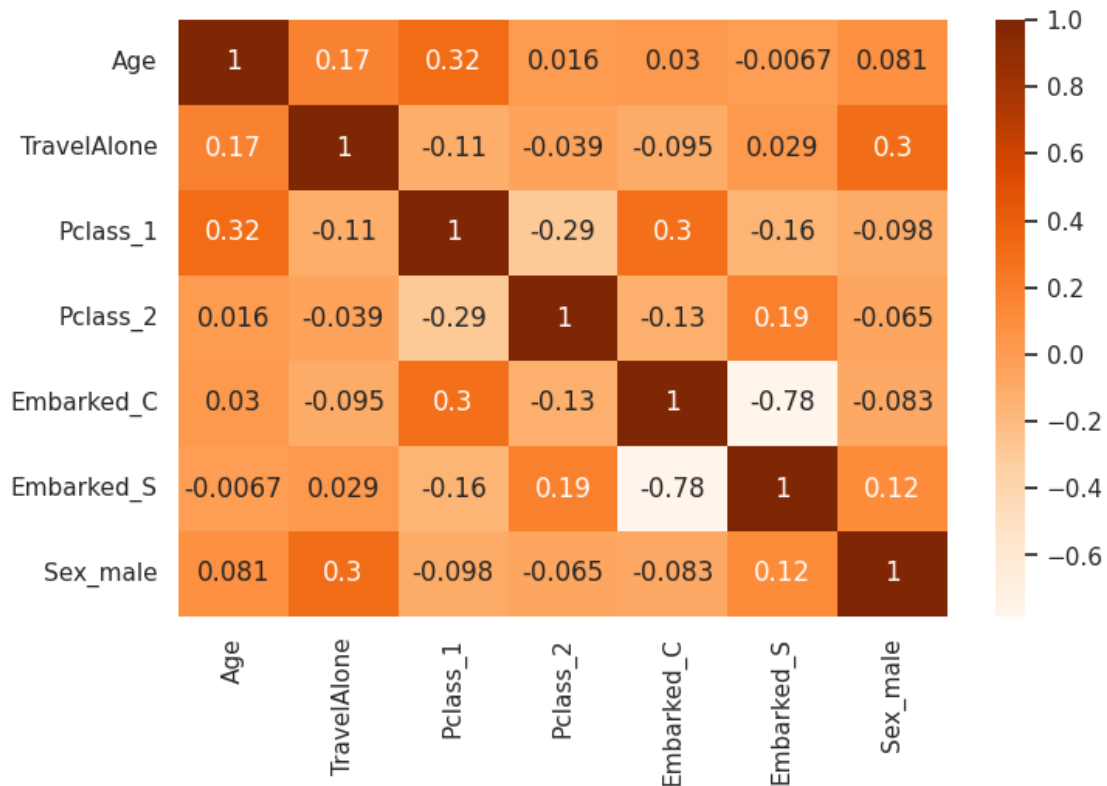
```
[41]:
```

```

Selected_features = ['Age', 'TravelAlone', 'Pclass_1', 'Pclass_2', 'Embarked_C',
                    'Embarked_S', 'Sex_male']
X = final_train[Selected_features]

plt.subplots(figsize=(8, 5))
sns.heatmap(X.corr(), annot=True, cmap="Oranges")
plt.show()

```



```
[37]: from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import accuracy_score, classification_report,
precision_score, recall_score
from sklearn.metrics import confusion_matrix, precision_recall_curve, roc_curve,
auc, log_loss

X = final_train[Selected_features]
y = final_train['Survived']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=2)

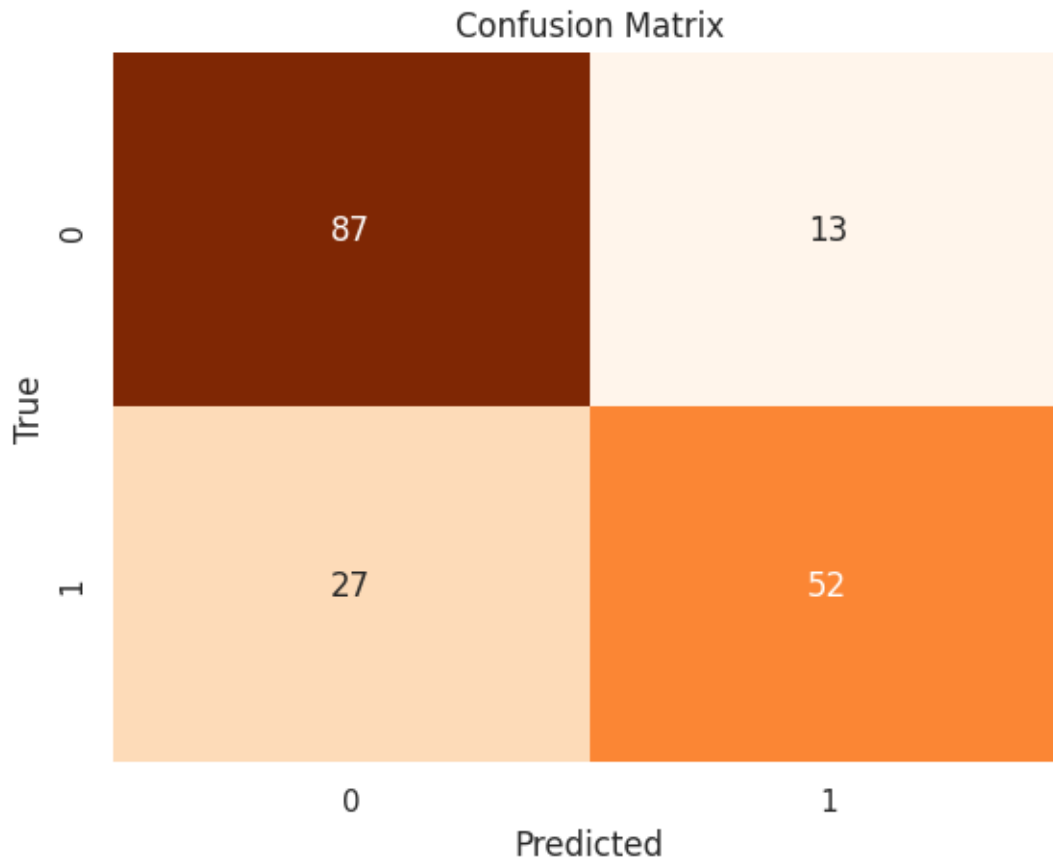
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
y_pred = logreg.predict(X_test)
```

```
[38]: print(f"Accuracy is {accuracy_score(y_test, y_pred)}")
```

Accuracy is 0.776536312849162

```
[40]: cm = confusion_matrix(y_test, y_pred)

sns.heatmap(cm, annot=True, fmt="d", cmap="Oranges", cbar=False)
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
```



```
[44]: from sklearn.metrics import roc_curve, auc

y_prob = logreg.predict_proba(X_test)[: , 1]

fpr, tpr, thresholds = roc_curve(y_test, y_prob)
roc_auc = auc(fpr, tpr)

plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='darkorange', lw=2, label='ROC curve (AUC = {:.2f})'.
        format(roc_auc))
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--', label='Random')
plt.xlabel('False Positive Rate (FPR)')
plt.ylabel('True Positive Rate (TPR)')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc='lower right')
plt.show()
```



