| |
|---|
| Experiment No. 2 |
| Analyze the Titanic Survival Dataset and apply appropriate regression technique |
| Date of Performance: 02/08/2023 |
| Date of Submission: 10/08/2023 |

**Aim:** Analyze the Titanic Survival Dataset and apply appropriate Regression Technique.

**Objective:** Able to perform various feature engineering tasks, apply logistic regression on the given dataset and maximize the accuracy.

**Theory:**

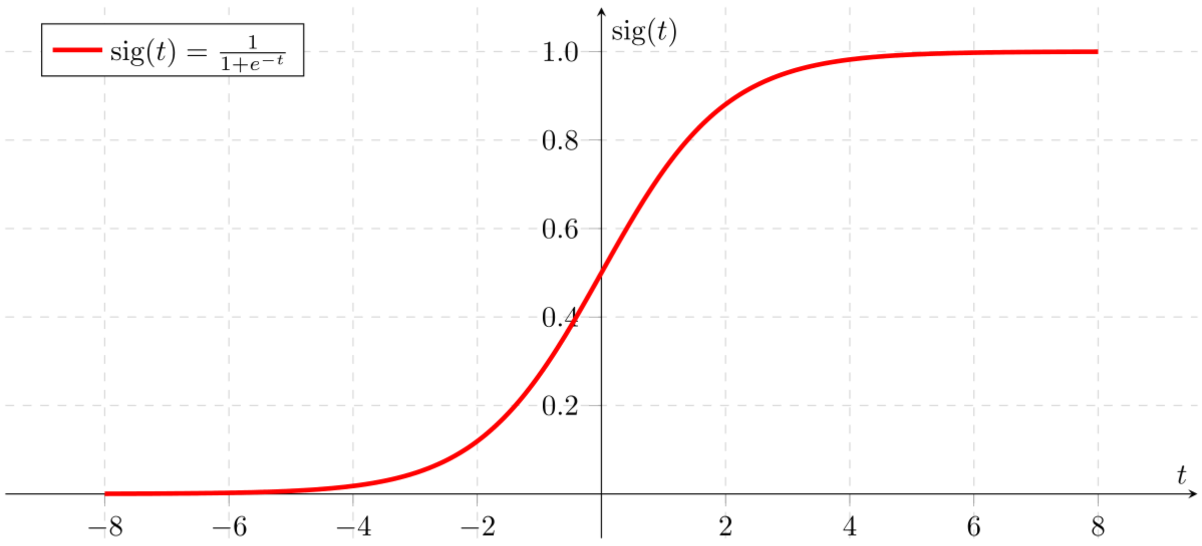
Logistic Regression was used in the biological sciences in early twentieth century. It was then used in many social science applications. Logistic Regression is used when the dependent variable(target) is categorical and is binary in nature. In order to perform binary classification the logistic regression techniques makes use of Sigmoid function.

For example,

To predict whether an email is spam (1) or (0)

Whether the tumor is malignant (1) or not (0)

Consider a scenario where we need to classify whether an email is spam or not. If we use linear regression for this problem, there is a need for setting up a threshold based on which classification can be done. Say if the actual class is malignant, predicted continuous value 0.4 and the threshold value is 0.5, the data point will be classified as not malignant which can lead to serious consequence in real time.




From this example, it can be inferred that linear regression is not suitable for classification problem. Linear regression is unbounded, and this brings logistic regression into picture. Their value strictly ranges from 0 to 1.

**Dataset:**

The sinking of the Titanic is one of the most infamous shipwrecks in history.

On April 15, 1912, during her maiden voyage, the widely considered “unsinkable” RMS Titanic sank after colliding with an iceberg. Unfortunately, there weren’t enough lifeboats for everyone onboard, resulting in the death of 1502 out of 2224 passengers and crew.

While there was some element of luck involved in surviving, it seems some groups of people were more likely to survive than others.

In this challenge, we ask you to build a predictive model that answers the question: “what sorts of people were more likely to survive?” using passenger data (ie name, age, gender, socio-economic class, etc).

| **Variable** | **Definition** | **Key** |
|---|---|---|
| survival | Survival | 0 = No, 1 = Yes |
| pclass | Ticket class | 1 = 1st, 2 = 2nd, 3 = 3rd |
| sex | Sex | |
| Age | Age in years | |
| sibsp | # of siblings / spouses aboard the Titanic | |
| parch | # of parents / children aboard the Titanic | |
| ticket | Ticket number | |
| fare | Passenger fare | |
| cabin | Cabin number | |
| embarked | Port of Embarkation | C = Cherbourg, Q = Queenstown, S = Southampton |

Variable Notes

pclass: A proxy for socio-economic status (SES)

1st = Upper, 2nd = Middle, 3rd = Lower

age: Age is fractional if less than 1. If the age is estimated, is it in the form of xx.5

sibsp: The dataset defines family relations in this way...,

Sibling = brother, sister, stepbrother, stepsister

Spouse = husband, wife (mistresses and fiancés were ignored)

parch: The dataset defines family relations in this way...

Parent = mother, father

Child = daughter, son, stepdaughter, stepson

Some children travelled only with a nanny, therefore parch=0 for them.

**Code:**

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb
```

```
df = pd.read_csv('./titanic.csv')
```

```
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
```

```
def impute_age(cols):
    Age = cols[0]
    Pclass = cols[1]

    if pd.isnull(Age):

        if Pclass == 1:
            return 37

        elif Pclass == 2:
            return 29

        else:
            return 24

    else:
        return Age
```

```
df['Age'] = df[['Age','Pclass']].apply(impute_age,axis=1)
```

```
df.isnull().sum()
```

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

```
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          891 non-null    int64  
 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(1), int64(6), object(5)
memory usage: 83.7+ KB
```

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

```
df.drop(['Sex','Embarked','Name','Ticket'],axis=1,inplace=True)
```

```
df = pd.concat([df,sex,embark],axis=1)
```

```
df.head(10)
```

| | PassengerId | Survived | Pclass | Age | SibSp | Parch | Fare | Cabin | male | Q | S |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 0 | 3 | 24 | 1 | 0 | 7.2500 | NaN | 1 | 0 | 1 |
| **1** | 2 | 1 | 1 | 37 | 1 | 0 | 71.2833 | C85 | 0 | 0 | 0 |
| **2** | 3 | 1 | 3 | 24 | 0 | 0 | 7.9250 | NaN | 0 | 0 | 1 |
| **3** | 4 | 1 | 1 | 37 | 1 | 0 | 53.1000 | C123 | 0 | 0 | 1 |
| **4** | 5 | 0 | 3 | 24 | 0 | 0 | 8.0500 | NaN | 1 | 0 | 1 |
| **5** | 6 | 0 | 3 | 24 | 0 | 0 | 8.4583 | NaN | 1 | 1 | 0 |
| **6** | 7 | 0 | 1 | 37 | 0 | 0 | 51.8625 | E46 | 1 | 0 | 1 |
| **7** | 8 | 0 | 3 | 24 | 3 | 1 | 21.0750 | NaN | 1 | 0 | 1 |
| **8** | 9 | 1 | 3 | 24 | 0 | 2 | 11.1333 | NaN | 0 | 0 | 1 |
| **9** | 10 | 1 | 2 | 29 | 1 | 0 | 30.0708 | NaN | 0 | 0 | 0 |

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 11 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   Age          891 non-null    int64  
 4   SibSp        891 non-null    int64  
 5   Parch        891 non-null    int64  
 6   Fare         891 non-null    float64
 7   Cabin        204 non-null    object 
 8   male         891 non-null    uint8  
 9   Q            891 non-null    uint8  
 10  S            891 non-null    uint8  
dtypes: float64(1), int64(6), object(1), uint8(3)
memory usage: 58.4+ KB
```

```
from sklearn.model_selection import train_test_split
```

```
features = df[['Pclass', 'Age','SibSp','Parch', 'Fare','male']]
target = df['Survived']
```

```
X_train, X_test, y_train, y_test = train_test_split(features,target, test_size=0.30,
                                                    random_state=2)
```

```
acc = []
model = []
```

```
from sklearn.linear_model import LogisticRegression
import sklearn.metrics as metrics
from sklearn.metrics import classification_report
LogReg = LogisticRegression(random_state=2)

LogReg.fit(X_train,y_train)

predicted_values = LogReg.predict(X_test)
```

```
x = metrics.accuracy_score(y_test, predicted_values)
acc.append(x)
model.append('Logistic Regression')
print("Logistic Regression's Accuracy is: ", x)

print(classification_report(y_test,predicted_values))
```

```
Logistic Regression's Accuracy is:  0.7910447761194029
              precision    recall  f1-score   support

           0       0.78      0.91      0.84       160
           1       0.82      0.62      0.71       108

    accuracy                           0.79       268
   macro avg       0.80      0.76      0.77       268
weighted avg       0.79      0.79      0.78       268
```

```
model
```

```
['Logistic Regression']
```

```
acc
```

```
[0.7910447761194029]
```

**Conclusion:**

1. What are features have been chosen to develop the model? Justify the features chosen to determine the survival of a passenger.

=> The features chosen to develop the model for determining the survival of a passenger are:

i) pclass (Passenger Class):

Justification: Higher passenger class values might indicate higher socio-economic status and potentially higher chances of survival.

ii) age (Age):

Justification: Age can be a critical factor in survival as children and elderly passengers might need more assistance and care during emergencies. Additionally, age-related priorities during evacuation might affect survival rates.

iii) sibsp (Number of Siblings/Spouses Aboard):

Justification: The presence of siblings or spouses could indicate potential assistance or family support during the disaster, affecting the passenger's survival chances.

iv) parch (Number of Parents/Children Aboard):

Justification: Similar to sibsp, the presence of parents or children could impact survival by providing familial support or requiring additional assistance during an evacuation.

v) Fare:

Justification: Fare might be an indicator of passenger class and socio-economic status. Higher fare payments might correlate with higher class and better access to safety measures.

vi) Male (Gender, Male):

Justification: Gender could influence survival chances due to the priority given to women and children during evacuation. "Male" is likely to represent gender, and it can help capture any gender-related patterns in survival rates.

**2.** Comment on the accuracy obtained.

⇨ The accuracy of 79.10% indicates that the model's predictions align with the actual outcomes in the dataset, and the precision, recall, and F1-score metrics provide additional insights into the model's performance for each class.