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

**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:**

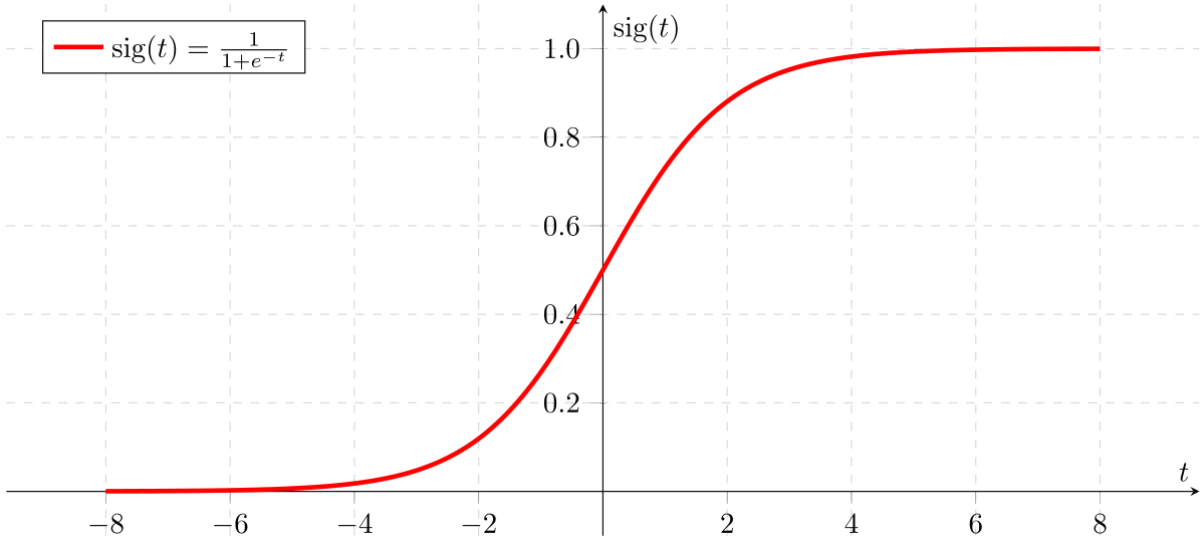
Logistic Regression was used in the biological sciences in early twentieth century. It was then used in many social science applications. Logistic Regression is used when the dependent variable(target) is categorical and is binary in nature. In order to perform binary classification the logistic regression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

**Conclusion:**

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

   In developing the model I have used “SibSp”,”Parch”,”Fare”,”Age” features from the dataset and dropped “Ticket”,”PassengerId”,”Name”,”Cabin” features as they had many null values and had no importance in the prediction of the cause.

2. Comment on the accuracy obtained.

   The accuracy obtained by using the sklearn model is 0.810055

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt



data = pd.read_csv("train.csv")
test = pd.read_csv("test.csv")
test_ids = test["PassengerId"]
def clean(data):
    data = data.drop(["Ticket", "PassengerId", "Name", "Cabin"], axis=1)

    cols = ["SibSp", "Parch", "Fare", "Age"]
    for col in cols:
        data[col].fillna(data[col].median(), inplace=True)

    data.Embarked.fillna("U", inplace=True)
    return data

data = clean(data)
test = clean(test)


data.head(5)
```

| | Survived | Pclass | Sex | Age | SibSp | Parch | Fare | Embarked |
|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 3 | male | 22.0 | 1 | 0 | 7.2500 | S |
| **1** | 1 | 1 | female | 38.0 | 1 | 0 | 71.2833 | C |
| **2** | 1 | 3 | female | 26.0 | 0 | 0 | 7.9250 | S |
| **3** | 1 | 1 | female | 35.0 | 1 | 0 | 53.1000 | S |
| **4** | 0 | 3 | male | 35.0 | 0 | 0 | 8.0500 | S |

```
from sklearn import preprocessing
le = preprocessing.LabelEncoder()
columns = ["Sex", "Embarked"]

for col in columns:
    data[col] = le.fit_transform(data[col])
    test[col] = le.transform(test[col])
    print(le.classes_)

data.head(5)
```

```
['female' 'male']
['C' 'Q' 'S' 'U']
```

| | 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 |

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

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

y = data["Survived"]
X = data.drop("Survived", axis=1)

X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
clf = LogisticRegression(random_state=0, max_iter=1000).fit(X_train, y_train)
```

```
predictions = clf.predict(X_val)
from sklearn.metrics import accuracy_score
accuracy_score(y_val, predictions)
```

```
0.8100558659217877
```

✓ 0s completed at 7:44 AM