Predicting the Survival of Titanic Passengers

Dataset: Titanic Dataset

It involves predicting the survival of Titanic Passengers(Y), from given different 11 parameters of passengers.

Titanic Dataset

- [Download (https://www.kaggle.com/c/titanic/data)](https://www.kaggle.com/c/titanic/data).
- [More Information (https://www.kaggle.com/c/titanic/data)](https://www.kaggle.com/c/titanic/data).

Q1: Why you want to apply regression on selected dataset? Discuss full story behind dataset.

Titanic data set contains 12 different columns as shown in code below. Name, PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin, Embarked. Among them Survived column is one dependent variable. which indicate that the passenger is survived or not.

So here the output or target value will be only two that the passenger is survived or not.

So here we can use Logistic Regression for two class classification.

Also, it is expected that as the number of claims increases, total claim amount will also increase.This is indicating application of linear regression.

Logistic Regression is one of the most simple and commonly used Machine Learning algorithms for two-class classification. It is easy to implement and can be used as the baseline for any binary classification problem. Its basic fundamental concepts are also constructive in deep learning. Logistic regression describes and estimates the relationship between one dependent binary variable and independent variables.

In [0]:

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

In [1]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3aietf%3awg%3aoauth%3a2.0%3aoob&response_type=code&scope=email%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdocs.test%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdrive%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdrive.photos.readonly%20https%3a%2f%2fwww.googleapis.com%2fauth%2fpeopleapi.readonly

Enter your authorization code:
··········
Mounted at /content/drive
```

In [7]:
```
train = pd.read_csv('/content/drive/My Drive/titanic_train.csv')
train.head()
```

Out[7]:

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

Q2: How many total observations in data?

There are total 712 observations in data from code below.

Q3: How many independent variables?

There are total eleven columns out of which one is dependent and ten are independent.

Q4: Which is dependent variable?

Survived is dependent variable which is indicating that passenger is survived or not.

In [8]:
```
train.drop('Cabin',axis=1,inplace=True)
train.dropna(inplace=True)
train.head()
train.shape
```

Out[8]: (712, 11)

Converting Categorical Features

In [9]:
```
sex = pd.get_dummies(train['Sex'],drop_first=True)
embark = pd.get_dummies(train['Embarked'],drop_first=True)
#drop the sex,embarked,name and tickets columns
train.drop(['Sex','Embarked','Name','Ticket'],axis=1,inplace=True)
#concatenate new sex and embark column to our train dataframe
train = pd.concat([train,sex,embark],axis=1)
#check the head of dataframe
train.head()
```

Out[9]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	male	Q	S
0	1	0	3	22.0	1	0	7.2500	1	0	1
1	2	1	1	38.0	1	0	71.2833	0	0	0
2	3	1	3	26.0	0	0	7.9250	0	0	1
3	4	1	1	35.0	1	0	53.1000	0	0	1
4	5	0	3	35.0	0	0	8.0500	1	0	1

Q5: Which are most useful variable in estimation? Prove using correlation.

Here, data has only one independent variable which has linear correlation with independent variable.

Understanding: Correlation is a statistical technique that can show whether and how strongly pairs of variables are related. For example, height and weight are related; taller people tend to be heavier than shorter people. Source: [https://www.surveysystem.com/correlation.htm (https://www.surveysystem.com/correlation.htm)](https://www.surveysystem.com/correlation.htm)

If there are more than one independent variable, not all independent variables contributes equally in estimation of dependent variable. This can be quatified using correlation between dependent and independent variable.

corr function is sklearn can be used to find correlation between variables. We can find correlation of each independent variable with dependent vatiable using loop, store them in a list/dataframe, sort them and finally decide which varible to use in delveloping model.

In [10]:

```
train.corr(method ='pearson')
```

Out[10]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	
PassengerId	1.000000	0.029526	-0.035609	0.033681	-0.082704	-0.011672	0.009655	0.02
Survived	0.029526	1.000000	-0.356462	-0.082446	-0.015523	0.095265	0.266100	-0.53
Pclass	-0.035609	-0.356462	1.000000	-0.365902	0.065187	0.023666	-0.552893	0.15
Age	0.033681	-0.082446	-0.365902	1.000000	-0.307351	-0.187896	0.093143	0.09
SibSp	-0.082704	-0.015523	0.065187	-0.307351	1.000000	0.383338	0.139860	-0.10
Parch	-0.011672	0.095265	0.023666	-0.187896	0.383338	1.000000	0.206624	-0.24
Fare	0.009655	0.266100	-0.552893	0.093143	0.139860	0.206624	1.000000	-0.18
male	0.024674	-0.536762	0.150826	0.099037	-0.106296	-0.249543	-0.182457	1.00
Q	-0.027045	-0.048966	0.131989	-0.021693	0.051331	-0.009417	-0.062346	-0.02
S	0.004605	-0.159015	0.197831	-0.025431	0.018968	0.013259	-0.250994	0.10

Logistic Regression basics

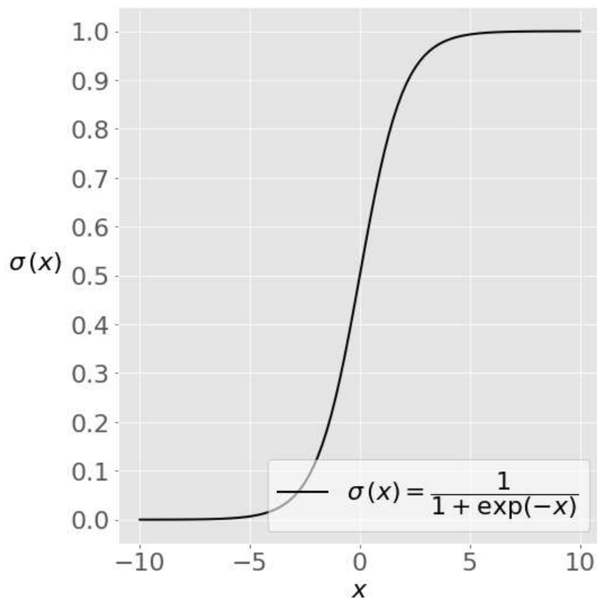
Logistic Regression Overview

Logistic regression is a fundamental classification technique. It belongs to the group of linear classifiers and is somewhat similar to polynomial and linear regression. Logistic regression is fast and relatively uncomplicated, and it's convenient for you to interpret the results. Although it's essentially a method for binary classification, it can also be applied to multiclass problems.

Math Prerequisites

You'll need an understanding of the sigmoid function and the natural logarithm function to understand what logistic regression is and how it works.

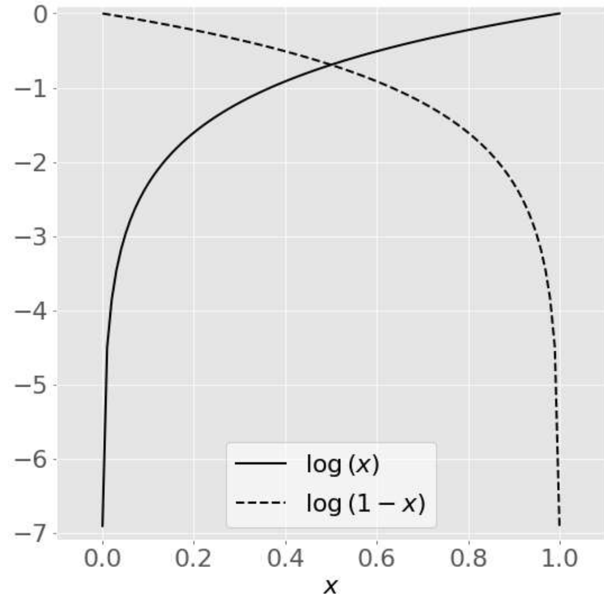
This image shows the sigmoid function (or S-shaped curve) of some variable x:



Sigmoid Function

The sigmoid function has values very close to either 0 or 1 across most of its domain. This fact makes it suitable for application in classification methods.

This image depicts the natural logarithm $\log(x)$ of some variable x, for values of x between 0 and 1:



Natural Logarithm

As x approaches zero, the natural logarithm of x drops towards negative infinity. When x = 1, log(x) is 0. The opposite is true for log(1 − x).

Note that you'll often find the natural logarithm denoted with ln instead of log. In Python, math.log(x) and numpy.log(x) represent the natural logarithm of x, so you'll follow this notation in this tutorial.

Classification Performance

Binary classification has four possible types of results:

True negatives: correctly predicted negatives (zeros) True positives: correctly predicted positives (ones) False negatives: incorrectly predicted negatives (zeros) False positives: incorrectly predicted positives (ones) You usually evaluate the performance of your classifier by comparing the actual and predicted outputsand counting the correct and incorrect predictions.

The most straightforward indicator of classification accuracy is the ratio of the number of correct predictions to

Building a Logistic Regression model Using Sklearn Library

In [0]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(train.drop('Survived',axis
=1),
           train['Survived'], test_size=0.30,
            random_state=101)
```

Training and Predicting

In [15]:
```
from sklearn.linear_model import LogisticRegression
#create an instance and fit the model
logmodel = LogisticRegression(max_iter=1000,)
logmodel.fit(X_train, y_train)
```

Out[15]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=1000,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

In [0]:
```
#predictions
Predictions = logmodel.predict(X_test)
```

Q6: Quantify goodness of your model and discuss steps taken for improvement (Accuracy, Confusion matrics,F-measure).

Ans: Accuracy is defined as: (fraction of correct predictions): correct predictions / total number of data points

A confusion matrix is a table that is often used to describe the performance of a classification model (or “classifier”) on a set of test data for which the true values are known.

The F1 Score is the 2*((precision*recall)/(precision+recall)). It is also called the F Score or the F Measure

Model Evaluation

In [0]:
```
from sklearn.metrics import classification_report , confusion_matrix,mean_squa
red_error,r2_score
print(classification_report(y_test,Predictions))
```

```
              precision    recall  f1-score   support

           0       0.80      0.82      0.81       128
           1       0.72      0.70      0.71        86

    accuracy                           0.77       214
   macro avg       0.76      0.76      0.76       214
weighted avg       0.77      0.77      0.77       214
```

confusion matrix:

In [0]:
```
print(confusion_matrix(y_test, Predictions))
```

```
[[105  23]
 [ 26  60]]
```