

Titanic data analysis and machine learning from disaster

“The **sinking of the RMS *Titanic*** occurred on the night of 14 April through to the morning of 15 April 1912 in the [North Atlantic Ocean](#), four days into the ship's [maiden voyage](#) from [Southampton](#) to [New York City](#). The largest [passenger liner](#) in service at the time, [Titanic](#) had an estimated 2,224 people on board when she struck an [iceberg](#) at around 23:40 (ship's time)^[a] on Sunday, 14 April 1912. Her sinking two hours and forty minutes later at 02:20 (05:18 [GMT](#)) on Monday, 15 April resulted in the deaths of more than 1,500 people, which made it one of the deadliest [peacetime maritime disasters](#) in history.”

- From Wikipedia, the free encyclopedia

Project by :

Jyotsana Yadav

Email: jy82398n@pace.edu

This project is based on the prediction of the survival of passengers based on set of data. Analyzing the data so as to retrieve the accurate results on best of my abilities. I will be using the dataset provided by Kaggle "Titanic: Machine learning from disaster".

The data is comprised of two set of files: **Test and training**

The fields are as follows:

- ✓ PassengerId
- ✓ Pclass (Passenger class 1 = 1st, 2 = 2nd, 3 = 3rd)
- ✓ Name
- ✓ Sex
- ✓ Age
- ✓ Sibsp (Passengers sibling or spouse on the board)
- ✓ Parch (Passengers parents or children on board)
- ✓ Ticket
- ✓ Fare
- ✓ Cabin
- ✓ Embarked (Port C = Cherbourg, Q = Queenstown, S = Southampton)

Where I will be using Pclass, sex, age as the parameters to find the rate of survival of the passengers.

On the basis of the training data I will be working on the two possible scenarios /Hypothesis:

H0: The passenger not survived the disaster

H1: The passenger survived the disaster.

Platform

I used Ananconda jupyter notebook to run my program.

The screenshot shows a Jupyter Notebook titled 'titanic_dataset (autosaved)' running on a local host. The code in the first cell imports necessary libraries: numpy, pandas, csv, and sklearn's model_selection, linear_model, and metrics modules. The second cell reads the 'train.csv' file into a DataFrame and displays the first 10 rows. The output shows a table with columns: PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin, and Embarked.

```
In [ ]: import numpy as np
import pandas as pd
import csv
from pandas import *
from sklearn import model_selection
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

In [56]: train_df = pd.read_csv("C:/Users/jyots/Desktop/jyots/bigdata/bigdataproject/titanic/train.csv")

train_df.head(10)
```

Out[56]:

	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	Cummings, 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
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	517429	51.8625	E46	C

Programming language and algorithm

I used Python as the programming language and imported below libraries to help my model to predict better.

import numpy as np

import pandas as pd

import csv

from pandas import *

from sklearn import model_selection

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

Used Logistic Regression algorithm for the prediction since my values will be yes/no (0/1)

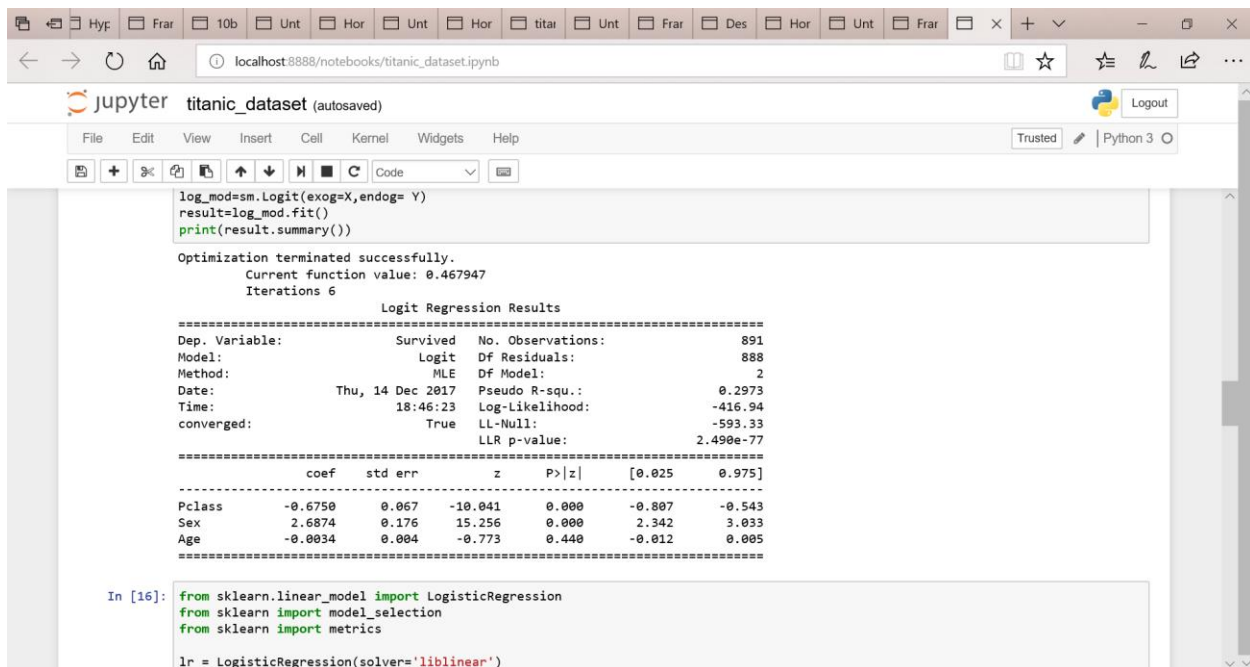
Logistic Regression for Machine Learning

Logistic regression is a technique borrowed by machine learning from the field of statistics.

It is a method for binary classification problems (problems with two class values). Logistic regression is used where the dependent variable is Binary. That means the dependent variable can take only two possible values such as “Yes or No. Independent factors or variables can be categorical or numerical variables.

Since in our problem we know that survival be of only two values so it was good or may one of the best methods to use logistic regression.

Summary result



The screenshot shows a Jupyter Notebook interface with a file explorer at the top displaying various files. The notebook is titled "titanic_dataset (autosaved)". The code cell contains the following Python code:

```
log_mod=sm.Logit(exog=X, endog= Y)
result=log_mod.fit()
print(result.summary())
```

The output of the code is as follows:

```
Optimization terminated successfully.
Current function value: 0.467947
Iterations 6
```

Logit Regression Results

	Dep. Variable:	Survived	No. Observations:	891
Model:	Logit		Df Residuals:	888
Method:	MLE		Df Model:	2
Date:	Thu, 14 Dec 2017		Pseudo R-squ.:	0.2973
Time:	18:46:23		Log-Likelihood:	-416.94
converged:	True		LL-Null:	-593.33
			LLR p-value:	2.490e-77

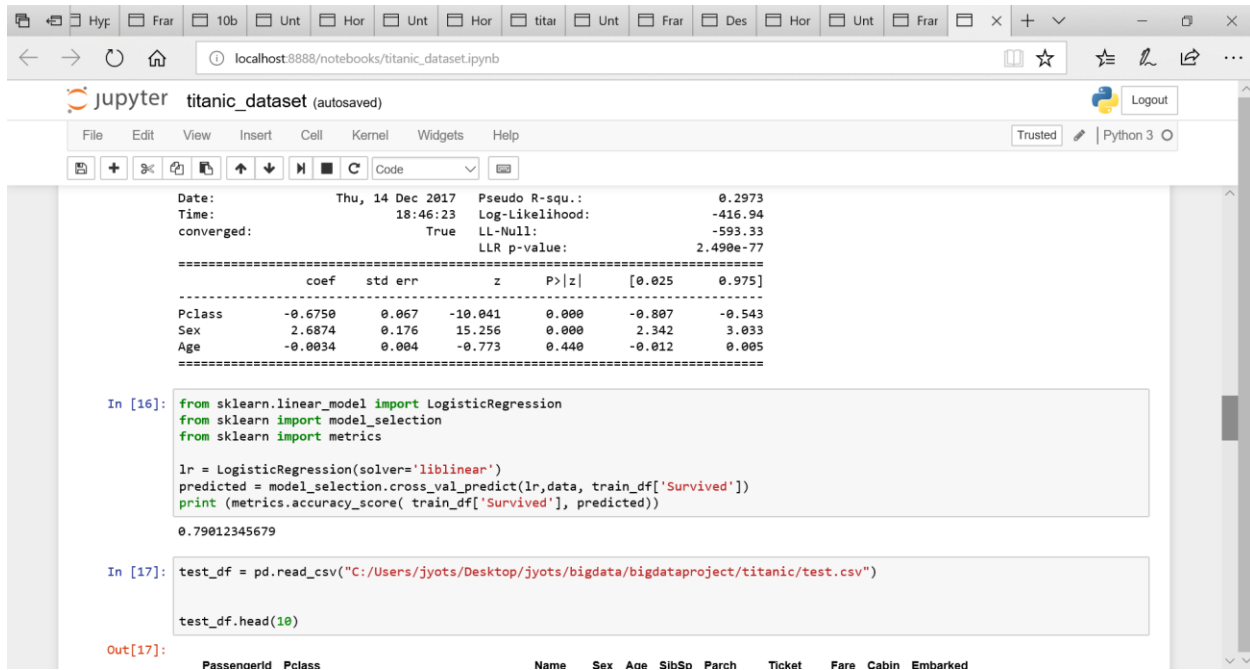
	coef	std err	z	P> z	[0.025	0.975]
Pclass	-0.6750	0.067	-10.041	0.000	-0.807	-0.543
Sex	2.6874	0.176	15.256	0.000	2.342	3.033
Age	-0.0034	0.004	-0.773	0.440	-0.012	0.005


```
In [16]: from sklearn.linear_model import LogisticRegression
from sklearn import model_selection
from sklearn import metrics

lr = LogisticRegression(solver='liblinear')
```

Prediction accuracy

With this model the prediction accuracy came out was 79 % .



The screenshot shows a Jupyter Notebook interface with the following content:

Model Training Results:

Date:	Thu, 14 Dec 2017	Pseudo R-squ.:	0.2973
Time:	18:46:23	Log-Likelihood:	-416.94
converged:	True	LL-Null:	-593.33
		LLR p-value:	2.490e-77

Model Coefficients:

	coef	std err	z	P> z	[0.025	0.975]
Pclass	-0.6750	0.067	-10.041	0.000	-0.807	-0.543
Sex	2.6874	0.176	15.256	0.000	2.342	3.033
Age	-0.0034	0.004	-0.773	0.440	-0.012	0.005

Code Snippets:

```
In [16]: from sklearn.linear_model import LogisticRegression
from sklearn import model_selection
from sklearn import metrics

lr = LogisticRegression(solver='liblinear')
predicted = model_selection.cross_val_predict(lr,data, train_df['Survived'])
print (metrics.accuracy_score( train_df['Survived'], predicted))

0.79012345679

In [17]: test_df = pd.read_csv("C:/Users/jyots/Desktop/jyots/bigdata/bigdataproject/titanic/test.csv")

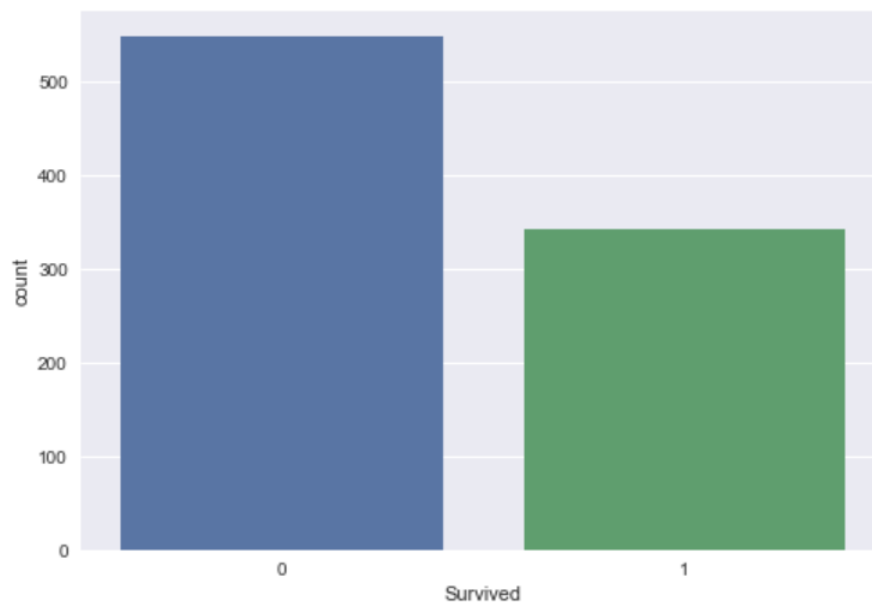
test_df.head(10)
```

Output:

PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
-------------	--------	------	-----	-----	-------	-------	--------	------	-------	----------

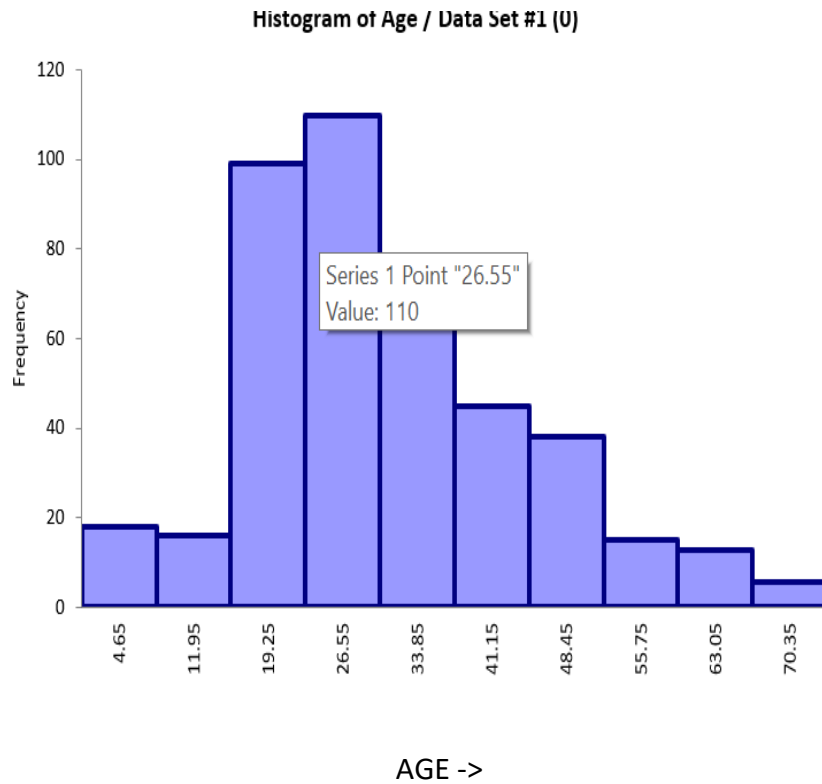
Plot for survival count

Here we can see that the passengers survived are less in number.

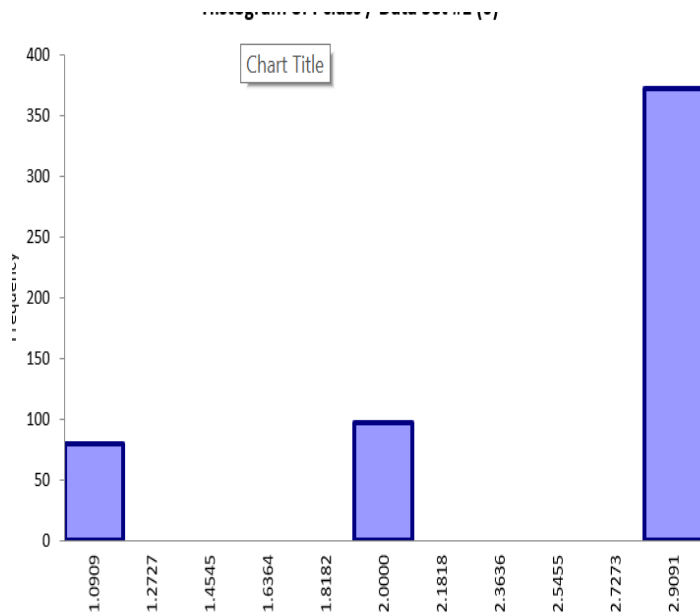


Plot for survival vs age

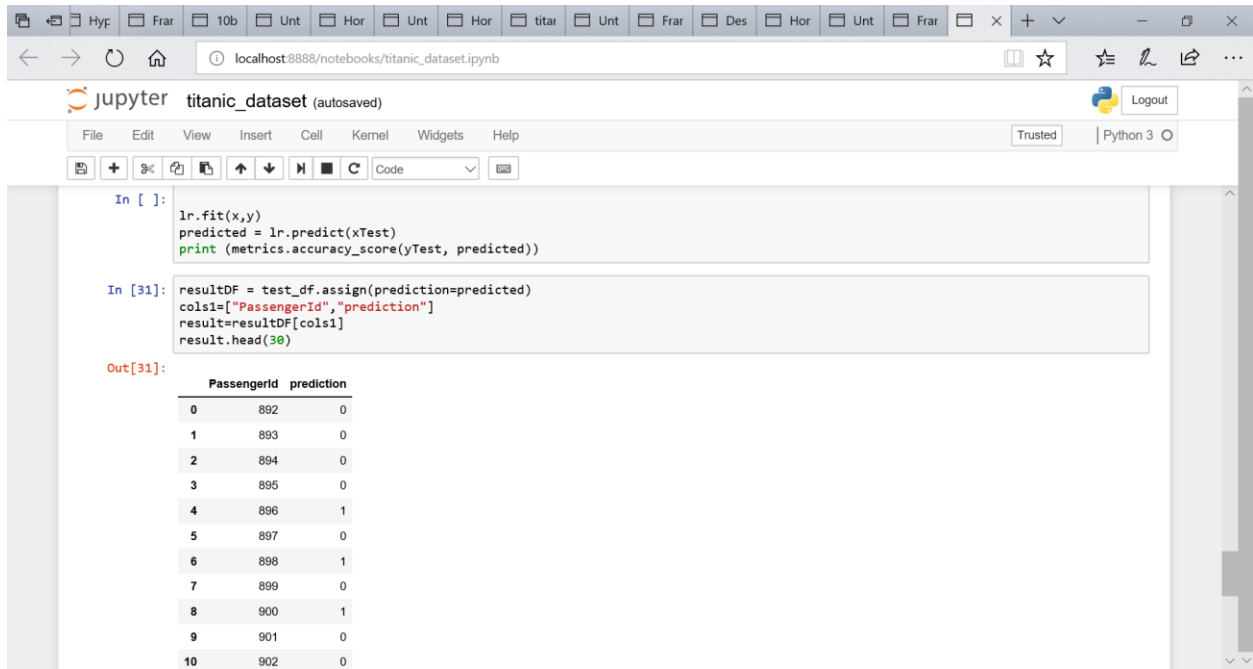
People aged in the range of 20 to 35 had more survival rate.



Plot for survival vs Pclass



Output is shown as below:



The image shows a Jupyter Notebook interface with the following components:

- Browser Address Bar:** localhost:8888/notebooks/titanic_dataset.ipynb
- Jupyter Header:** titanic_dataset (autosaved) | Trusted | Python 3
- Code Cells:**
 - In []:**

```
lr.fit(x,y)
predicted = lr.predict(xTest)
print (metrics.accuracy_score(yTest, predicted))
```
 - In [31]:**

```
resultDF = test_df.assign(prediction=predicted)
cols=["PassengerId","prediction"]
result=resultDF[cols]
result.head(30)
```
- Output [31]:**

	PassengerId	prediction
0	892	0
1	893	0
2	894	0
3	895	0
4	896	1
5	897	0
6	898	1
7	899	0
8	900	1
9	901	0
10	902	0