**MINI PROJECT REPORT**

**At**

**Satyabhama Institute of Science and Technology**

**(Deemed to be University)**

Submitted in partial fulfillment of the requirements for the

Award of Bachelor of Engineering Degree in

Computer Science and Engineering

By

**SUMITH POKKULURI**

**(Reg. No.38110573)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**SCHOOL OF COMPUTING**

**SATHYABAMA**

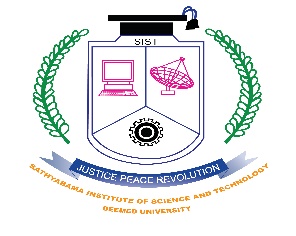
**INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade “A” by NAAC**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600119 TAMILNADU.**

**AUGUST2020**

**SATHYABAMA**

**INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade “A” by NAAC**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119**

**www.sathyabama.ac.in**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**BONAFIDE CERTIFICATE**

This is to certify that this Project Report is the bonafide work of **SUMITH POKKULURI (Reg.no.38110573)**who carried out the project entitled **“SURVIVAL PREDICTION IN TITANIC”** under my supervision from April 2020 to June 2020.

**Internal Guide**

DR.A.Viji Amutha Mary, M.TECH., Ph.D.

**Head of the Department**

Dr. S. VIGNESHWARI, M.E., (Ph.D.)

**Submitted for Viva voce Examination held on\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Internal Examiner External Examiner**

**DECLARATION**

I **SUMITH POKKULURI** (**Reg.no.37110396**) hereby declare that the Project Report Entitled **“SURVIVAL PREDICTION IN TITANIC”** done by me under the guidance of **Dr. A. Viji Amutha Mary,** Department of Computer Science and Engineering at Sathyabama Institute of Science and Technologyis submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

**DATE:**

**PLACE: CHENNAI**  **SIGNATURE OF THE CANDIDATE**

**ACKNOWLEDGEMENT**

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA**for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. SASIKALA, M.E., (Ph.D.), Dean, School of Computing,Dr.S. Vigneswari M.E., Ph.D., and Dr.L. Lakshmanan M.E., Ph.D.,** Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.A. Viji Amutha Mary, M.TECH., Ph.D**for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of **COMPUTER SCIENCE AND ENGINEERING** who were helpful in many ways for the completion of the project.

**TRAINING CERTIFICATE**

****

**ABSTRACT**

The sinking of the RMS Titanic caused the death of thousands of passengers and crew is one of the deadliest maritime disasters in history. One of the reasons that the shipwreck led to such loss of life was that there were not enough lifeboats for the passengers and crew. The interesting observation which comes out from the sinking is that some people were more likely to survive than others, like women, children were the one who got the priority to rescue. The objective is to first explore hidden or previously unknown information by applying exploratory data analytics on available dataset and then apply different machine learning models to complete the analysis of what sorts of people were likely to survive. After this the results of applying machine learning models are compared and analyzed on the basis of accuracy.

**

The sinking of the RMS Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers and crew. This sensational tragedy shocked the international community and led to better safety regulations for ships.

|  |  |  |
| --- | --- | --- |
|  | **TABLE OF CONTENTS** |  |
|  |  |  |
| **CHAPTER NO.** | **TITLE** | **PAGE NO** |
|  | **ABSTRACT** | **vi** |
|  | **LIST OF FIGURES** | **viii** |
| **1.** | **INTRODUCTION** | **01** |
|  | 1.1 PROBLEM STATEMENT | 01 |
|  | 1.2OVERVIEW OF PROJECT | 01 |
| **2.** | **AIM AND SCOPE OF THE PROJECT** | **03** |
|  | 2.1 AIM OF THE PROJECT | 03 |
|  | 2.2 SCOPE OF THE PROJECT | 03 |
|  | 2.3 OBJECTIVE | 05 |
| **3.** | **WORK AND METHODOLOGY** | **06** |
|  | 3.1 PROCESS-FLOW | 06 |
|  | 3.2 DESCRIPTION OF DATA | 09 |
|  | 3.3DATA-CLEANING | 11 |
|  | 3.4 EXPLORATORY DATA ANALYSIS  3.5 METHODOLOGY  3.5.1 FEATURE ENGINEERING  3.5.2 MACHINE LEARNING MODELS  3.5.3 MODEL EVALUTION  3.5.4 PREDICTION | 13  14  14  14  15  16 |
|  |  |  |
|  | 3.6 ENVIRONMENT |  |
|  | 3.6.1 JUPYTER NOTEBOOK | 17 |
|  | 3.6.2 PYTHON  3.7 WORK 19  **4.RESULTS AND DISCUSSIONS**  4.1 RESULTS  **5.SUMMARY AND CONCLUSION**  5.1 CONCLUSION  5.2 FUTURE ENHANCEMENT  **REFRENCES**  **APPENDIX** | 18 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**CHAPTER 1:**

**INTRODUCTION**

* 1. **PROBLEM STATEMENT**

The Project revolves about the sinking of the Titanic, The 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 project the challengeis to build a predictive model that answers the question: “what sorts of people were more likely to survive?” using passenger data (i.e. name, age, gender, socio-economic class, etc.).

* 1. **OVERVIEW OF THE PROJECT**

**STEP-1:**

First, import the required Python dependencies.

**importmatplotlib.pyplotasplt**

%matplotlib inline

**importrandom**

**importnumpyasnp**

**importpandasaspd**

**fromsklearnimport** datasets, svm, cross\_validation, tree, preprocessing, metrics

**importsklearn.ensembleasske**

**importtensorflowastf**

**fromtensorflow.contribimport** skflow

***STEP-2:***

Once we have read the spreadsheet file into a Pandas data frame (imagine a hyper powered Excel table), we can peek at the first five rows of data using the head () command.

titanic\_df=pd.read\_excel('titanic3.xls','titanic3',index\_col=**None**,na\_values=['NA'])

titanic\_df.head()

The column heading variables have the following meanings:

* **survival:** Survival (0 = no; 1 = yes)
* **class:** Passenger class (1 = first; 2 = second; 3 = third)
* **name:** Name
* **sex:** Sex
* **age:** Age
* **sibsp:** Number of siblings/spouses aboard
* **parch:** Number of parents/children aboard
* **ticket:** Ticket number
* **fare:** Passenger fare
* **cabin:** Cabin
* **embarked:** Port of embarkation (C = Cherbourg; Q = Queenstown; S = Southampton)
* **boat:** Lifeboat (if survived)
* **body:** Body number (if did not survive and body was recovered)

**STEP-3:**

Now that we have the data in a dataframe, we can begin an advanced analysis of the data using powerful single-line Pandas functions. First, let’s examine the overall chance of survival for a Titanic passenger.

In [3]:

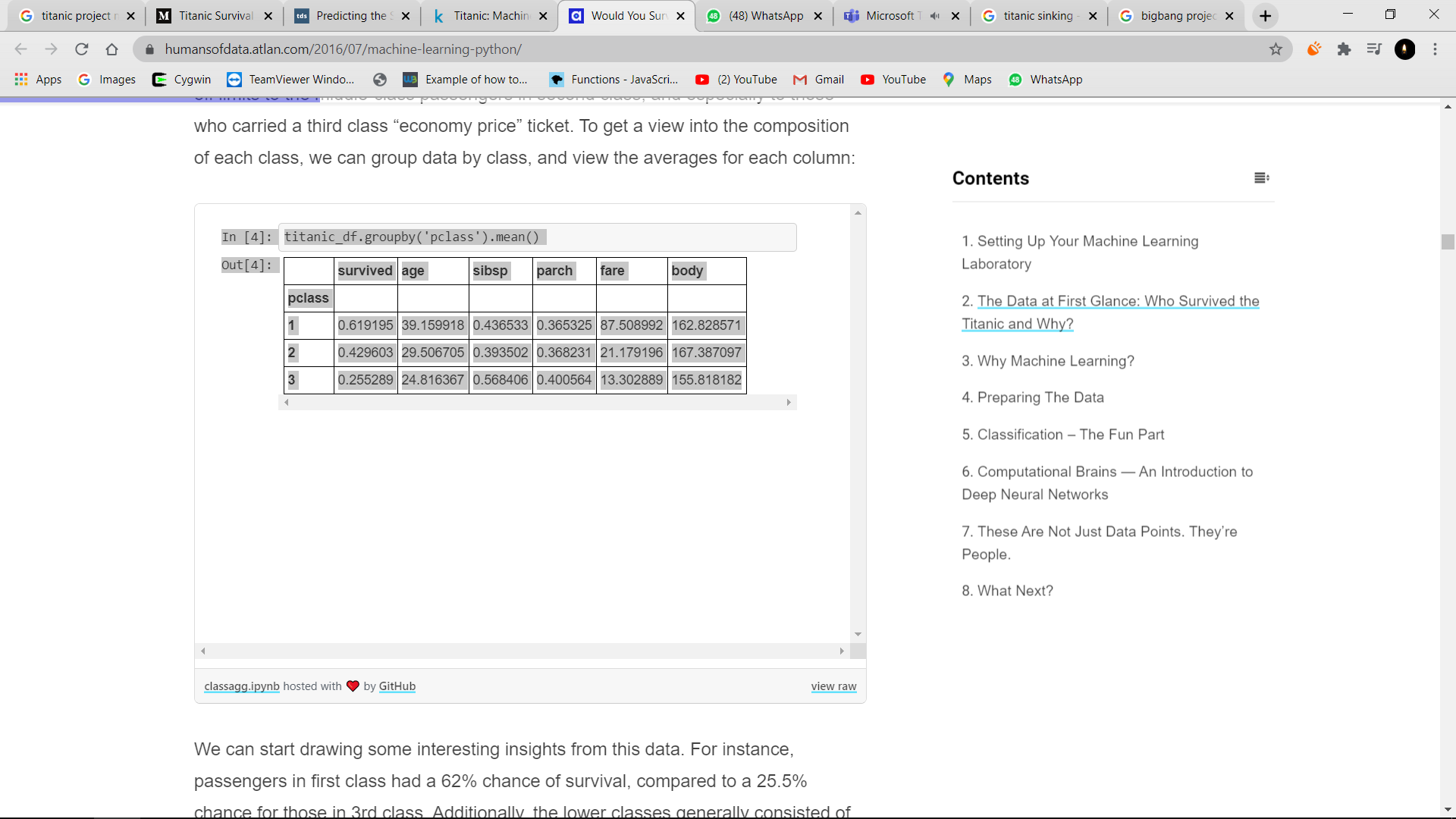
titanic\_df['survived'].mean()

Out[3]:

0.3819709702062643

The calculation shows that only 38% of the passengers survived. Not the best odds. The reason for this massive loss of life is that the Titanic was only carrying 20 lifeboats, which was not nearly enough for the 1,317 passengers and 885 crew members aboard. It seems unlikely that all of the passengers would have had equal chances at survival, so we will continue breaking down the data to examine the social dynamics that determined who got a place on a lifeboat and who did not.

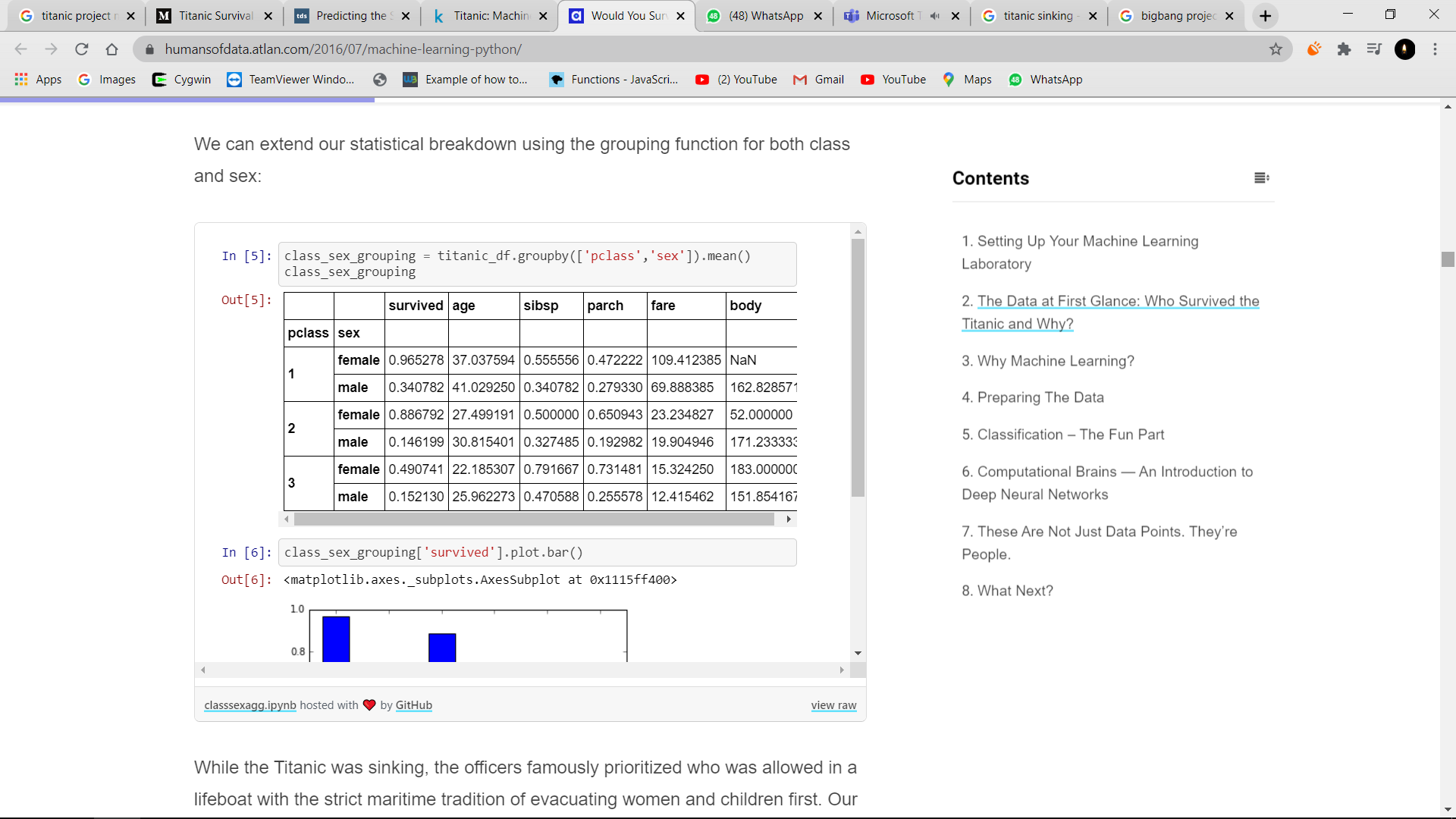
Social classes were heavily stratified in the early twentieth century. This was especially true on the Titanic, where the luxurious first-class areas were completely off limits to the middle-class passengers in second class, and especially to those who carried a third class “economy price” ticket. To get a view into the composition of each class, we can group data by class, and view the averages for each column:

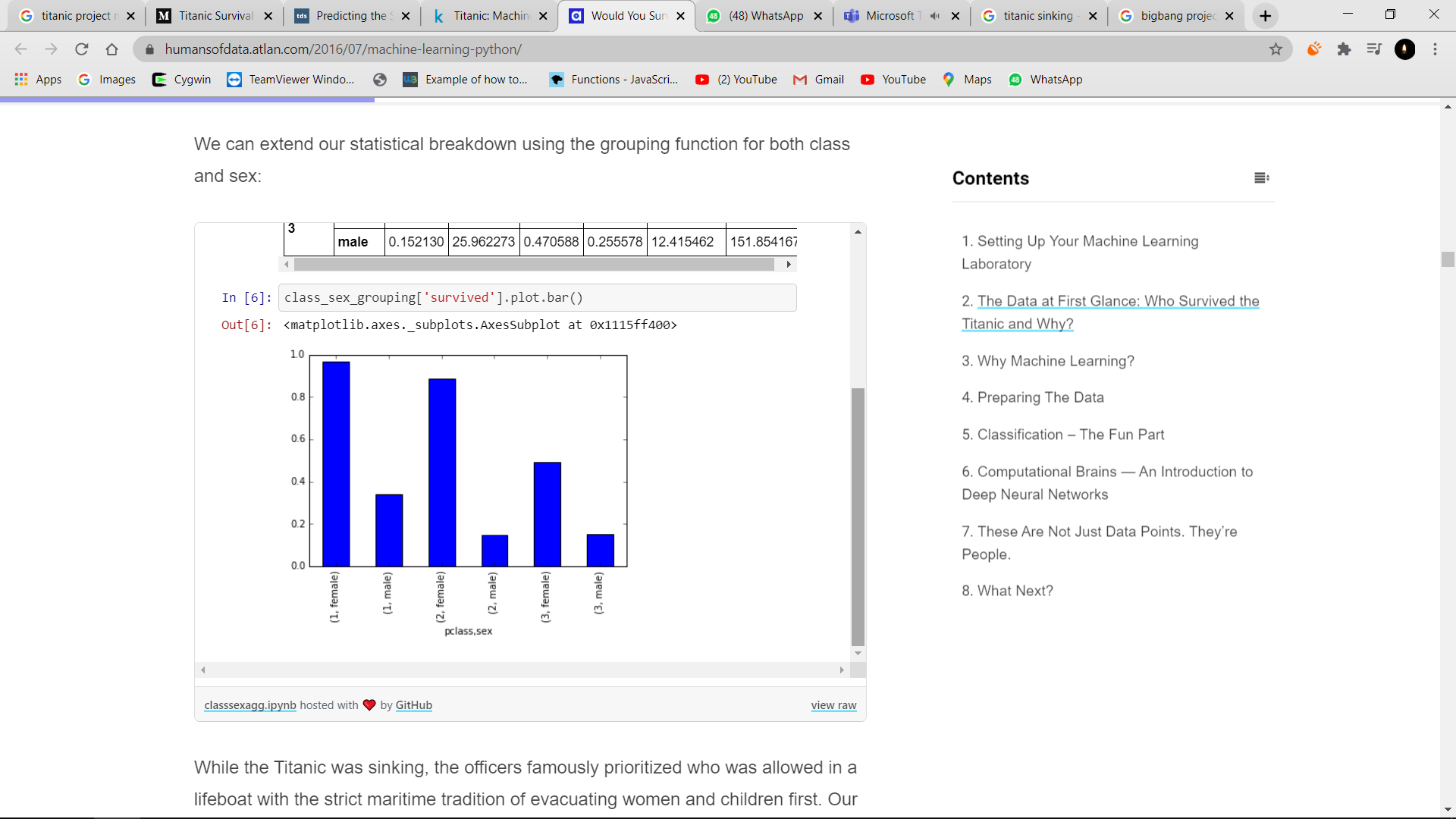


**STEP-4:**

We can start drawing some interesting insights from this data. For instance, passengers in first class had a 62% chance of survival, compared to a 25.5% chance for those in 3rd class. Additionally, the lower classes generally consisted of younger people, and the ticket prices for first class were predictably much higher than those for second and third class. The average ticket price for first class (£87.5) is equivalent to $13,487 in 2016.

We can extend our statistical breakdown using the grouping function for both class and sex:

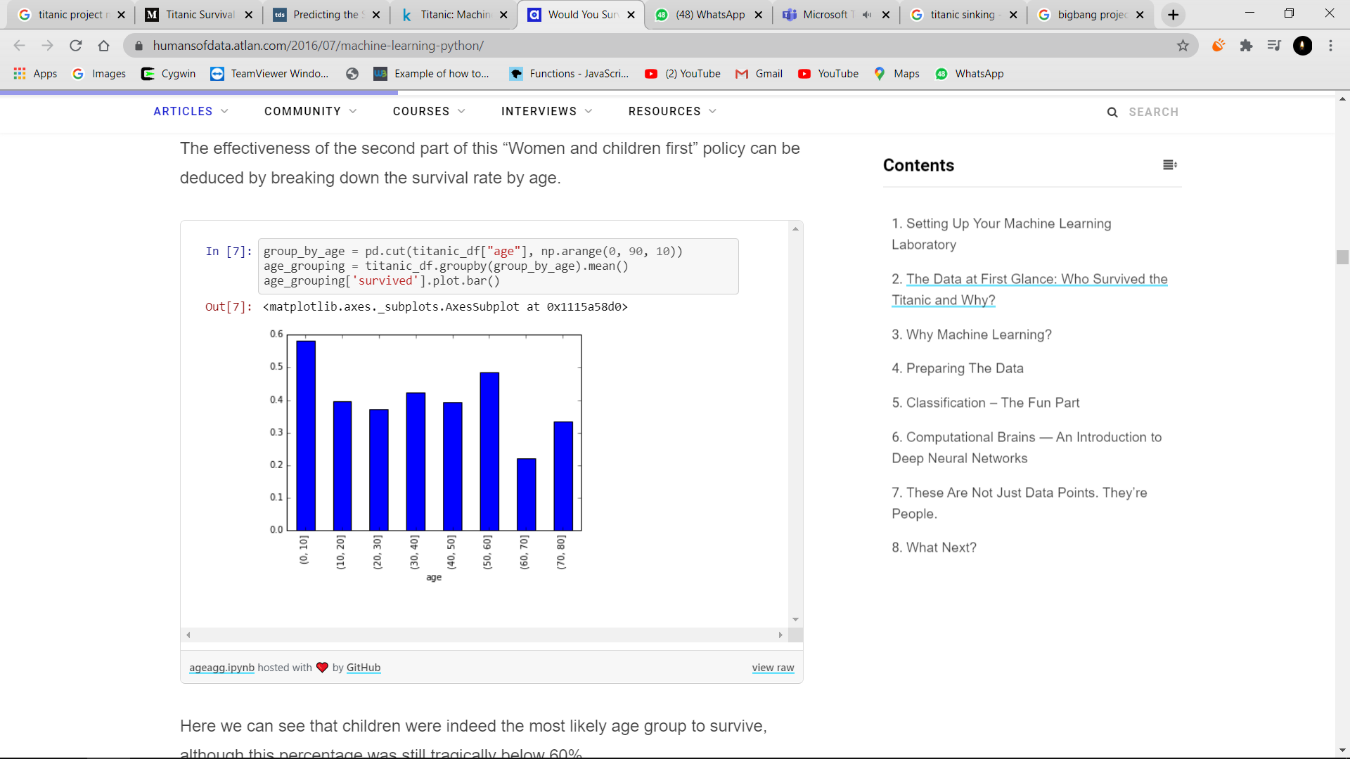




**STEP-5:**

While the Titanic was sinking, the officers famously prioritized who was allowed in a lifeboat with the strict maritime tradition of evacuating women and children first. Our statistical results clearly reflect the first part of this policy as, across all classes, women were much more likely to survive than the men. We can also see that the women were younger than the men on average, were more likely to be traveling with family, and paid slightly more for their tickets.

The effectiveness of the second part of this “Women and children first” policy can be deduced by breaking down the survival rate by age.

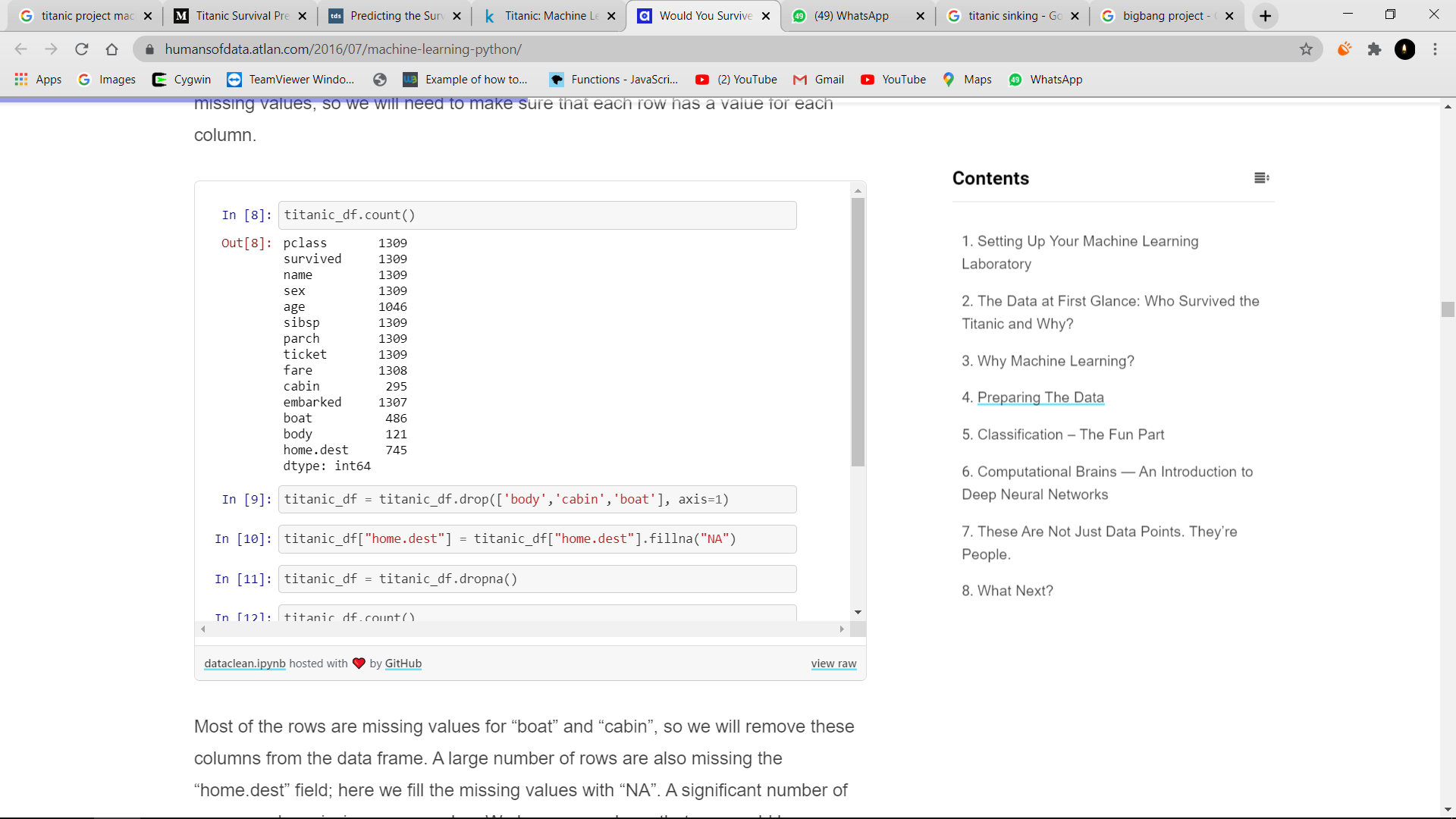


**STEP-6:**

## Preparing the Data

Before we can feed our data set into a machine learning algorithm, we have to remove missing values and split it into training and test sets.

If we perform a count of each column, we will see that much of the data on certain fields is missing. Most machine learning algorithms will have a difficult time handling missing values, so we will need to make sure that each row has a value for each column.

****

**CHAPTER 2**

**AIM AND SCOPE OF THE PROJECT**

**2.1 AIM OF THE PROJECT**

The goal of the project was to predict the survival of passengers based off a set of data. We retrieve necessary data and evaluate accuracy of our predictions. The historical data has been split into two groups, a 'training set' and a 'test set'. For the training set, we are provided with the outcome (whether or not a passenger survived). We used this set to build our model to generate predictions for the test set. For each passenger in the test set, we had to predict whether or not they survived the sinking. Our score was the percentage of correctly predictions

**2.2SCOPE OF THE PROJECT**

The requirements:

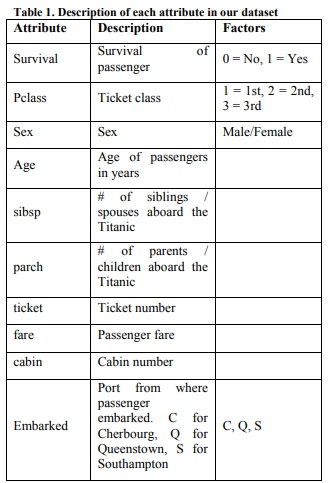
* Programming language Python and its libraries NumPy (to perform matrix operations) and SciKit-Learn (to apply machine learning algorithms)
* Several machine learning algorithms (decision tree, random forests, extra trees, linear regression)
* Feature Engineering techniques

Sources used:

* Open source web Application Jupyter Notebook(https://jupyter.org)
* Python 3.7 with the libraries numpy, sklearn, and matplotlib
* MicrosoftExcel

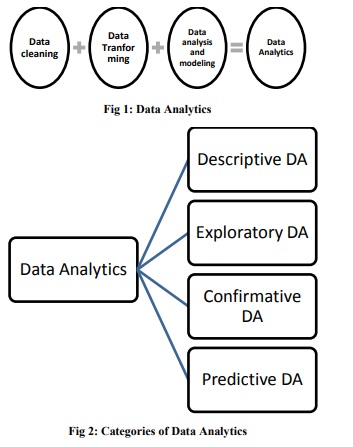
**2.3 OBJECTIVE**

The objective of this project is then to build a predictive model to predict which of the passengers survived the ship wreck. In particular, the response variable Survived will be modeled given ten possible predictors. The remainder of this report includes background on the methods used to build the predictive model, specifically classification and regression trees, cost complexity pruning, bagging and random forests. A case study based on the RMS Titanic data implementing the methods will be conducted.



2.3.1 **PURPOSE:**

* The Purpose is to perform exploratory data analytics to mine various information in the dataset available and to know effect of each field on survival of passengers by applying analytics between every field of dataset with “Survival” field.
* The predictions are done for newer data sets by applying machine learning algorithm. The data analysis will be done on applied algorithms and accuracy will be checked. Different algorithms are compared on the basis of accuracy and the best performing model is suggested for predictions

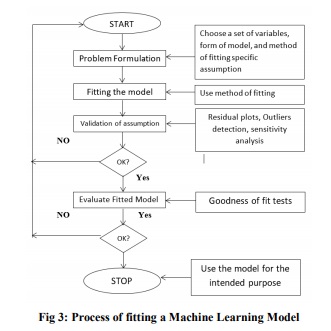


**CHAPTER 3**

**WORK AND METHODOLOGY**

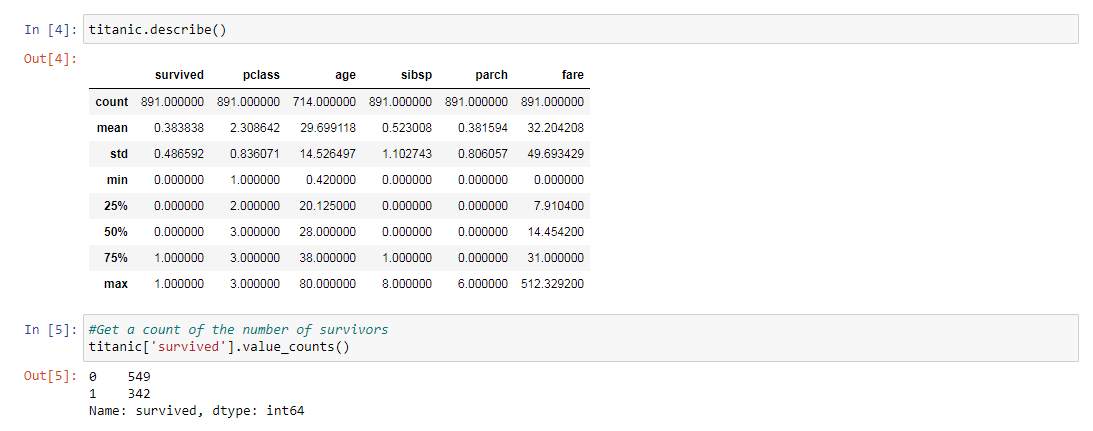
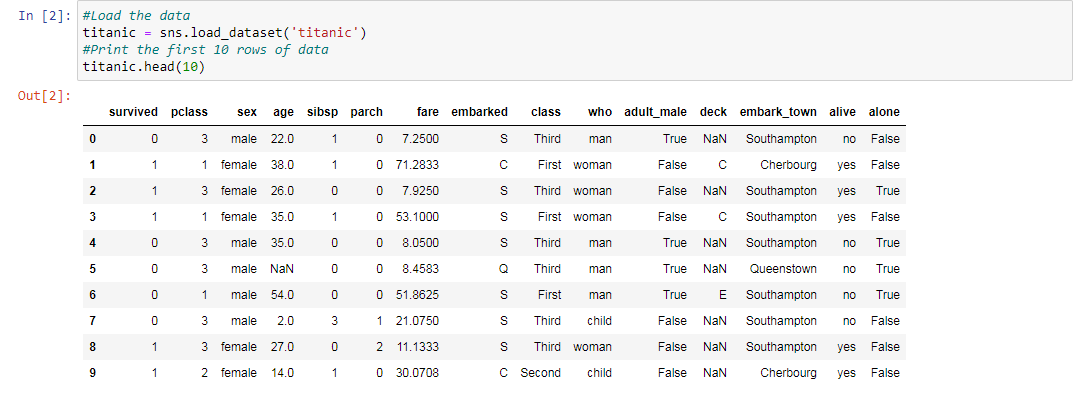
**3.1. PROCESS FLOW**

There is a step by step approach to choose a particular model for the current problem. We need to decide whether a particular machine learning model is suitable for our problem or not. Here we can see process flow being followed

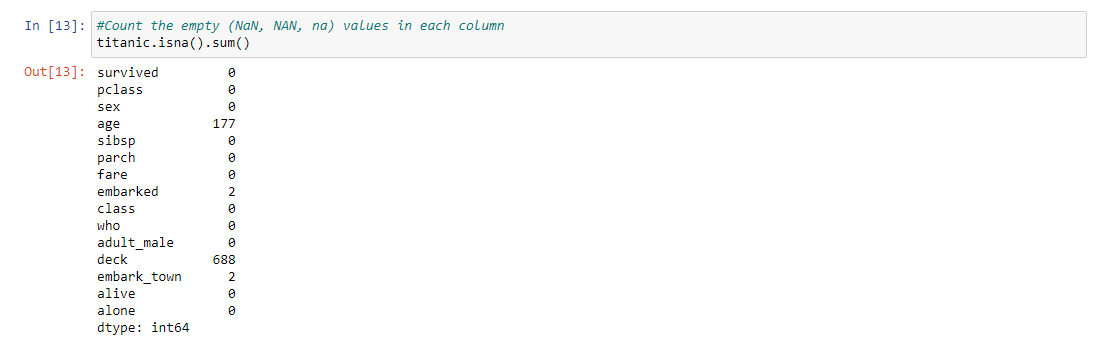


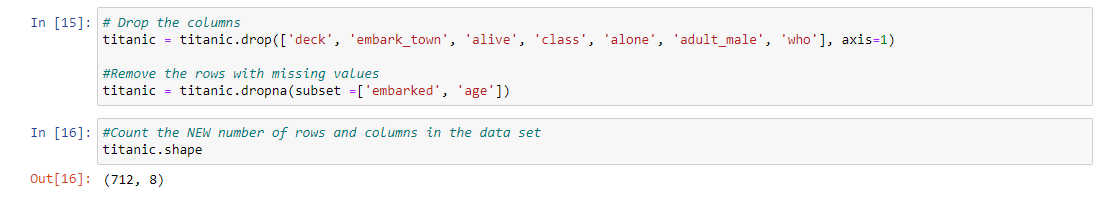
**3.2. Description of Data**

After importing the required libraries, we will load the titanic csv data set using sns\_load.dataset() function. After reading the file we can check the top 10 entries by using the head() function. We can describe the given set by using describe() function.



**3.3. DATA CLEANING**

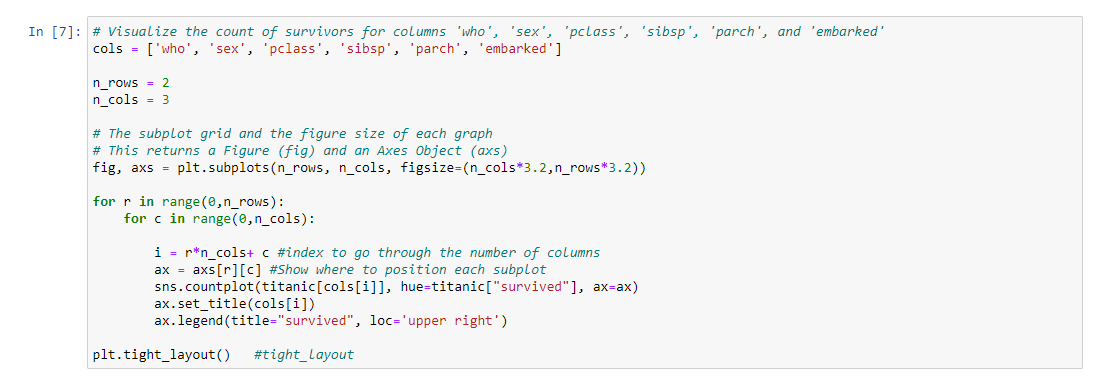
Before applying any type of data analytics on the dataset, the data is first cleaned. There are some missing values in the dataset which needs to be handled. In attributes like Age, Cabin and Embarked, missing values are replaced with random sample from existing age. Since the data can have missing fields, incomplete fields, or fields containing hidden information, a crucial step in building any prediction system is Feature Engineering. For instance, the fields Age, Fare, and Embarked in the training and test data, had missing values that had to be filled in.,

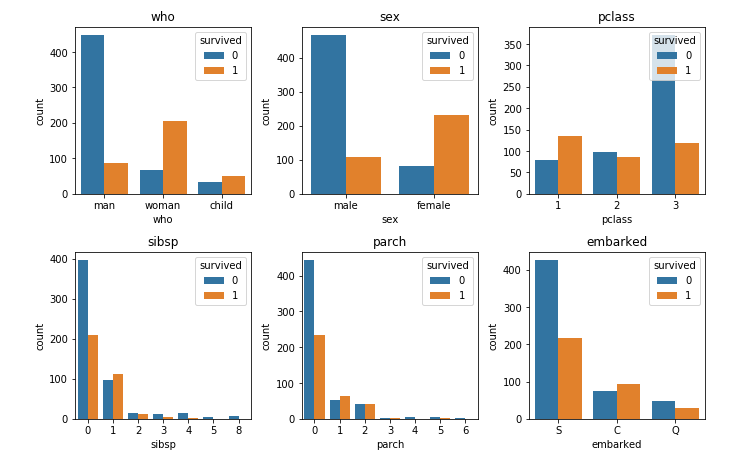


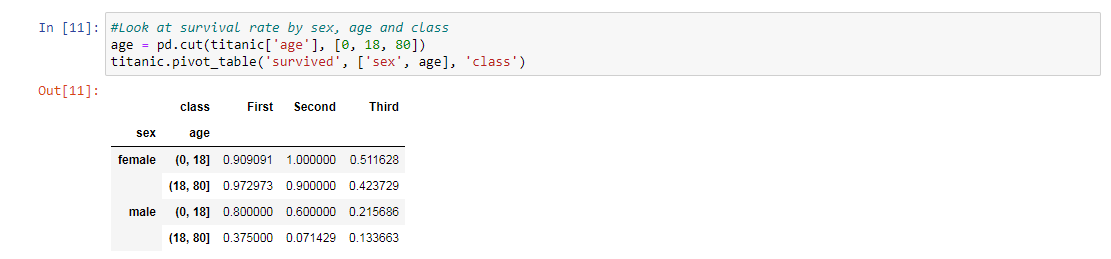
**3.4. EXPLORATORY DATA ANALYSIS**

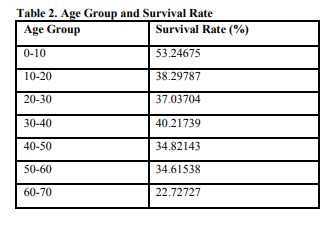
We are going to perform exploratory data analysis for our problem in the first stage. In exploratory data analysis dataset is explored to figure out the features which would influence the survival rate. The data is deeply analysed by finding a relationship between each attribute and survival

In the below figs we created some plots for better understandment of the data using Count plots of various variables vs Survived count.



In the same way there are some more facts we found. There is a table showing age group and survival rate of that age group

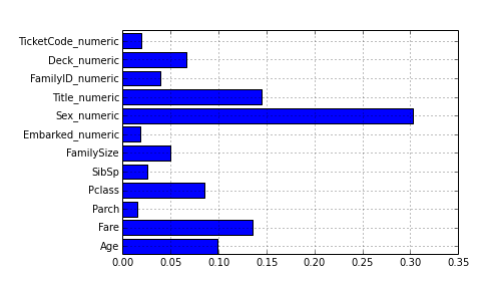




**3.5. METHODOLOGY**

**3.5.1. Feature Engineering**

Feature engineering is the most important part of data analytics process. It deals with, selecting the features that are used in training and making predictions. In feature engineering the domain knowledge is used to find features in the dataset which are helpful in building machine learning model. It helps in understanding the dataset in terms of modeling. A bad feature selection may lead to less accurate or poor predictive model. The accuracy and the predictive power depend on the choice of correct features. It filters out all the unused or redundant features. Based on the exploratory analysis above, following features are used age, sex, cabin, title, Pclass, family size (parch plus sibsp columns), fare, embarked. Survival column is chosen as response column. These features are selected because their values have an impact on the rate of survival. These features will be the value of “x” in the bar-plots. If wrong features where selected then even the good algorithm may produce the bad predictions. Therefore, feature engineering acts like a backbone in building an accurate predictive model.



**3.5.2 Machine Learning Models**

Various machine learning models are implemented to validate and predict survival.

**3.5.2.1 Logistic Regression**

Logistic regression is the technique which works best when dependent variable is dichotomous (binary or categorical). [23] The data description and explaining the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables is done with the help of logistic regression. It is used to solve binary classification problem, some of the real life examples are spam detection- predicting if an email is spam or not, health-Predicting if a given mass of tissue is benign or malignant, marketing- predicting if a given user will buy an insurance product or not.

**3.5.2.2 Decision Tree**

Decision tree is a supervised learning algorithm. This is generally used in problems based on classification. It is suitable for both categorical and continuous input and output variables. Each root node represents a single input variable (x) and a split point on that variable. The dependent variable (y) is present at leaf nodes. For example: Suppose there are two independent variables, i.e. input variables (x) which are height in centimeter and weight in kilograms and the task to find gender of person based on the given data. (Hypothetical example, for demonstration purpose only).

**3.5.2.3 Random Forest**

Random forest algorithm is supervised classification algorithm. The algorithm basically makes forest with large number of trees. The higher the number of trees in the forest gives the higher accuracy results. Random forest algorithm can be used for both classification and regression problems. For instance, it will take random samples of 100 observation and 5 randomly chosen initial variables to build a model. The same process is repeated a number of times, then the final prediction is made according to the observations. Final prediction is a function (mean) of each prediction

**3.5.3. MODEL EVALUATION**

The accuracy of the model is evaluated using “confusion matrix”. A confusion matrix is a table layout that allows to visualize the correctness and the performance of an algorithm. **3.5.3.1 Confusion Matrix**

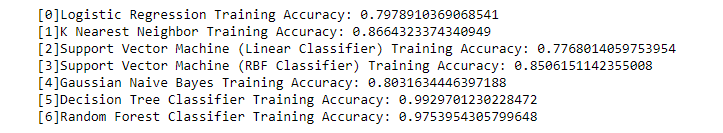
A confusion matrix is a method to verify how accurately the classification model works. It gives the actual number of predictions which were correct or incorrect when compared to the actual result of the data. The matrix is of the order N\*N, here N is the number of values. Performance of such models is commonly evaluated using the data in the matrix. **Sensitivity:** It defines the percentage of actual positive which are correctly identified, and is complementary to the false negative rate. Sensitivity= true positive/(true negative + false positive). The ideal value for sensitivity is “1.0” and minimum value is “0.0”

**Specificity:** It measures the proportion of negatives which are correctly identified, and is complementary to the false positive rate. Specificity= true negatives/(true negatives + false positives). The ideal value for specificity is “1.0” and least value is “0.0”.

**Positive Predictive Value:** It gives the performance measure of the statistical test. It is a ratio true positive (event that makes true prediction and subject result is also true) and the sum of true positive and false positive (event that makes false prediction and subject result is also false).

**Accuracy:**It gives the measure of percentage of correct prediction done by the model/algorithm. The best value is “1.0” and the worst value is “0.0”

The fig. given below shows the accuracy of the respective models corresponding to the data set given.

**3.5.4. PREDICTION**

Here we can choose any of the models to predict survival of test sample. Since we have evaluated all models by using confusion matrix we will predict by using model which has highest accuracy. We performed prediction on data dataset by using logistic model and SVM.

**3.6 ENVIRONMENT**

**3.6.1JUPYTER NOTEBOOK**

### 

Jupyter Notebook (formerly IPython Notebooks) is a web-based interactive computational environment for creating Jupyter notebook documents. The "notebook" term can colloquially make reference to many different entities, mainly the Jupyter web application, Jupyter Python web server, or Jupyter document format depending on context. A Jupyter Notebook document is a JSON document, following a versioned schema, and containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media, usually ending with the ".ipynb" extension.

A Jupyter Notebook can be converted to a number of open standard output formats (HTML, presentation slides, LaTeX, PDF, ReStructuredText, Markdown, Python) through "Download As" in the web interface, To simplify visualisation of Jupyter notebook documents on the web, the nbconvert library is provided as a service through NbViewer which can take a URL to any publicly available notebook document, convert it to HTML on the fly and display it to the user.

Jupyter Notebook provides a browser-based REPL built upon a number of popular open-source libraries:

* IPython
* ØMQ
* Tornado (web server)
* jQuery
* Bootstrap (front-end framework)
* MathJax

Jupyter Notebook can connect to many kernels to allow programming in many languages. By default Jupyter Notebook ships with the IPython kernel. As of the 2.3 release[8][9] (October 2014), there are currently 49 Jupyter-compatible kernels for many programming languages, including Python, R, Julia and Haskell.



**3.6.2 PYTHON 3.7.4**

**Python** is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.[28]

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library

Those new to programming can benefit from Python's high level of abstraction. It is highly interactive and known for its "strong opinions" around specific syntax (including whitespace). Python, like other high-level languages, has a garbage collection process to manage memory or delete unused resources. A user can receive instant feedback from the interpreter by typing **python** on the command line or by using projects like JupyterLab if they want a browser-based development experience. Many users also appreciate that Python has a strict syntax enforced by the compiler, making it easy to have a single "right way" to write a program



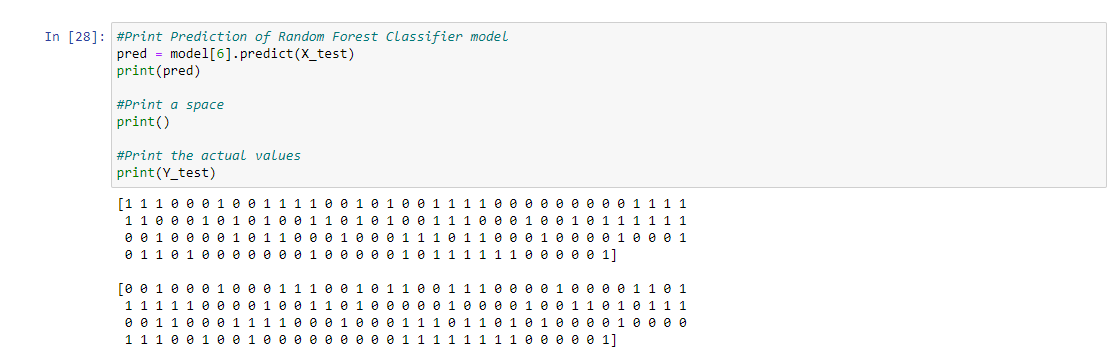
**3.7 WORK**

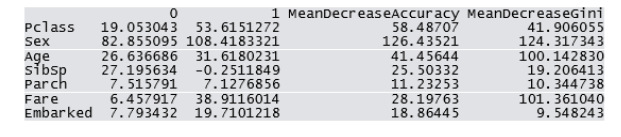
I designed an predictivemodel using python and with help of libraries like pandas, numpy, seaborn, mat.lib.I used an open source application “Jupyter notebook” to compile the python code.Using seaborn package, titanic data set has been loaded and the whole file and data was analysed by using count plots and graphswhich contains data set with many variables like age , gender, embark , p-class, etc.. Then we compare the variables for better understanding. . my model predicts if we would survive the titanic with actual data provided so that we can enhance the prediction accuracy for individual input which led to accuracy of 80% using model “random forest classifier”. After analysing data, I have created few different models like logistic regression, k nearest neighbour, decision tree classifier, decision tree classifier, random tree classifier, etc.. to check the prediction accuracy of given data. After thorough analysis, Random forest classifier has optimal prediction accuracy. So we used Random forest classifier as our source model.

**CHAPTER 4**

**RESULTS AND DISCUSSION**

**4.1 RESULTS**

****

****

The following sections provide an overview of two analyses performed in effort to achieve the goal of predicting survival of the passengers on the RMS Titanic. The first analysis consists of very few changes in the data and incorporates methods such as regression and classification trees, cost complexity pruning, bagging, and random forests. The second analysis replicates the methods used in the first analysis while incorporating feature engineering. In these analyses, we have identified the important elements in the prediction, i.e. predictor variables, split points, etcetera.

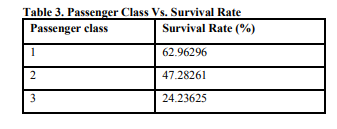
Alongside implementing the methods used and assessing these methods using prediction accuracy, we are able to summarize the important components of our solution when predicting survival of passengers on the RMS Titanic. All of the methods in the first analysis make use of the predictor variables, Sex, Fare, Age, Pclass, and SibSp. All of the methods in the second analysis make use of the predictor variables, Title, Pclass, FamilySize, and Deck. Of the predictor variables, we consider the most influential to be Sex or Title, Fare, Age, and Pclass. This is important because these findings support the claim that women and children, as well as the upper class, were given a priority of being rescued

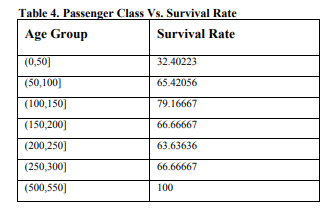
**CHAPTER 5**

**SUMMARY AND CONCLUSION**

**5.1 CONCLUSION**

Data cleaning is the first step while performing data analysis. Exploratory data analytics helps one to understand the dataset and the dependency among the attributes. EDA is used to figure out the relationship between the features of the dataset. This is done by using various graphical techniques. The one used above is ggplot and histograms. By applying EDA some conclusions are drawn and facts are found. There is high influence of age on survival. We can see from table-2 that as age increases survival decreases. It can be seen that survival rate of female is very high (approx. 74%) and survival rate of male is very low. This fact can also be verified by extracting titles (Mr, Mrs, Ms etc) from name column. Survival rate with title Mr. is approximately 16% while survival rate for Mrs. is 79%

We found that Passengers who were travelling in first class is more likely to survive.We combined parch and sibsp column to know family size of a particular passenger. We found that survival rate increases when family size lies from 0 to 3. But when family size becomes greater than 3 survival rate decrease. Similarly it is found that passengers who has more cabins has higher survival rate

With these figure we can say that higher the fare higher will be survival rate. In feature engineering the actual parameters to be used while designing the training model and prediction model is found out on the basis of exploratory data analytics process. Machine Learning models predict the values of passengers who survived. Logistic regression technique is used in making predictions in classification problem.

The confusion matrix gives the accuracy of all the models, the logistic regression is proves to be best among all with an accuracy of 0.837261504. This means the predictive power of logistic regression in this dataset with the chosen features is very high. It is clearly stated that the accuracy of the models may vary when the choice of feature modelling is different. Ideally logistic regression and support vector machine are the models which give a good level of accuracy when it comes to classification problem.

As a result of our work, we gained valuable experience of building prediction systems and achieved our best score on Kaggle: 80.383% of correct predictions

We performed featured engineering techniques

• Changed alphabetic values to numeric

• Used linear regression algorithm to fill in missing ages

• We used several prediction algorithms in python

• Decision tree

• Random forests

• Extra trees

• We achieved our best score 80.383% correct predictions

**5.2 FUTURE ENHANCEMENT**

This project involves implementation of data analytics and machine learning. This project work can be used as reference to learn implementation of EDA and machine learning from very basic level. In future, the idea can be extended by making more advanced graphical user interface with the aid of newer libraries like shiny in R. An interactive page can be created, i.e. if the value of an attribute is varied on the scale then the values corresponding to its graph (plot or histogram) will also change. It will be helpful to draw much focused conclusions by combining results we obtained

**REFERENCES**

[1] Analyzing Titanic disaster using machine learning algorithms-Computing, Communication and Automation (ICCCA), 2017 International Conference on 21 December 2017, IEEE.

[2] Eric Lam, Chongxuan Tang, "Titanic Machine Learning From Disaster", LamTang-Titanic Machine Learning From Disaster, 2012.

[3] S. Cicoria, J. Sherlock, M. Muniswamaiah, L. Clarke, "Classification of Titanic Passenger Data and Chances of Surviving the Disaster", Proceedings of Student-Faculty Research Day CSIS, pp. 1-6, May 2014.

[4]MICHAEL AARON WHITLEY, Using statistical learning to predict survival of passengers on the RMS Titanic by Michael Aaron Whitley, 2015

**WEBSITES**

<https://medium.com/better-programming/titanic-survival-prediction-using-machine-learning-4c5ff1e3fa16>

<https://github.com/Sumith20/Titanic-survival-Prediction>

**APPENDIX**

**SOURCE CODE**

*#Import Libraries*

**import numpy as np**

**import pandas as pd**

**import seaborn as sns**

**import matplotlib.pyplot as plt**

*#Load the data*

**titanic = sns.load\_dataset('titanic')**

**#Print the first 10 rows of data**

**titanic.head(10)**

*#Count the number of rows and columns in the data set*

**titanic.shape**

**titanic.describe()**

*#Get a count of the number of survivors*

**titanic['survived'].value\_counts()**

*#Visualize the count of number of survivors*

**sns.countplot(titanic['survived'],label="Count")**

*# Visualize the count of survivors for columns 'who', 'sex', 'pclass', 'sibsp', 'parch', and 'embarked'*

**cols = ['who', 'sex', 'pclass', 'sibsp', 'parch', 'embarked']**

**n\_rows = 2**

**n\_cols = 3**

*# The subplot grid and the figure size of each graph*

*# This returns a Figure (fig) and an Axes Object (axs)*

**fig, axs = plt.subplots(n\_rows, n\_cols, figsize=(n\_cols\*3.2,n\_rows\*3.2))**

**for r in range(0,n\_rows):**

**for c in range(0,n\_cols):**

**i = r\*n\_cols+ c #index to go through the number of columns**

**ax = axs[r][c] #Show where to position each subplot**

**sns.countplot(titanic[cols[i]], hue=titanic["survived"], ax=ax)**

**ax.set\_title(cols[i])**

**ax.legend(title="survived", loc='upper right')**

**plt.tight\_layout() #tight\_layout**

*#Look at survival rate by sex*

**titanic.groupby('sex')[['survived']].mean()**

*#Look at survival rate by sex and class*

**titanic.pivot\_table('survived', index='sex', columns='class')**

*#Look at survival rate by sex and class visually*

**titanic.pivot\_table('survived', index='sex', columns='class').plot()**

*#Look at survival rate by sex, age and class*

**age = pd.cut(titanic['age'], [0, 18, 80])**

**titanic.pivot\_table('survived', ['sex', age], 'class')**

*#Plot the Prices Paid Of Each Class*

**plt.scatter(titanic['fare'], titanic['class'], color = 'red', label='Passenger Paid')**

**plt.ylabel('Class')**

**plt.xlabel('Price / Fare')**

**plt.title('Price Of Each Class')**

**plt.legend()**

**plt.show()**

*#Count the empty (NaN, NAN, na) values in each column*

**titanic.isna().sum()**

*#Look at all of the values in each column & get a count*

**for val in titanic:**

**print(titanic[val].value\_counts())**

**print()**

*# Drop the columns*

**titanic = titanic.drop(['deck', 'embark\_town', 'alive', 'class', 'alone', 'adult\_male', 'who'], axis=1)**

*#Remove the rows with missing values*

**titanic = titanic.dropna(subset =['embarked', 'age'])**

*#Count the NEW number of rows and columns in the data set*

**titanic.shape**

**titanic.dtypes**

*#Print the unique values in the columns*

**print(titanic['sex'].unique())**

**print(titanic['embarked'].unique())**

*#Encoding categorical data values (Transforming object data types to**integers)*

**from sklearn.preprocessing import LabelEncoder**

**labelencoder = LabelEncoder()**

*#Encode sex column*

**titanic.iloc[:,2]= labelencoder.fit\_transform(titanic.iloc[:,2].values)**

*#print(labelencoder.fit\_transform(titanic.iloc[:,2].values))*

*#Encode embarked*

**titanic.iloc[:,7]= labelencoder.fit\_transform(titanic.iloc[:,7].values)**

*#print(labelencoder.fit\_transform(titanic.iloc[:,7].values))*

*#Print the NEW unique values in the columns*

**print(titanic['sex'].unique())**

**print(titanic['embarked'].unique())**

*#Split the data into independent 'X' and dependent 'Y' variables*

**X = titanic.iloc[:, 1:8].values**

**Y = titanic.iloc[:, 0].values**

*# Split the dataset into 80% Training set and 20% Testing set*

**from sklearn.model\_selection import train\_test\_split**

**X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size = 0.2, random\_state = 0)**

*#Feature Scaling*

**from sklearn.preprocessing import StandardScaler**

**sc = StandardScaler()**

**X\_train = sc.fit\_transform(X\_train)**

**X\_test = sc.transform(X\_test)**

*#Create a function within many Machine Learning Models*

**def models(X\_train,Y\_train):**

*#Using Logistic Regression Algorithm to the Training Set*

**from sklearn.linear\_model import LogisticRegression**

**log = LogisticRegression(random\_state = 0)**

**log.fit(X\_train, Y\_train)**

*#Using KNeighborsClassifier Method of neighbors class to use Nearest**Neighbor algorithm*

**from sklearn.neighbors import KNeighborsClassifier**

**knn = KNeighborsClassifier(n\_neighbors = 5, metric = 'minkowski', p = 2)**

**knn.fit(X\_train, Y\_train)**

*#Using SVC method of svm class to use Support Vector Machine**Algorithm*

**from sklearn.svm import SVC**

**svc\_lin = SVC(kernel = 'linear', random\_state = 0)**

**svc\_lin.fit(X\_train, Y\_train)**

*#Using SVC method of svm class to use Kernel SVM Algorithm*

**from sklearn.svm import SVC**

**svc\_rbf = SVC(kernel = 'rbf', random\_state = 0)**

**svc\_rbf.fit(X\_train, Y\_train)**

*#Using GaussianNB method of naïve\_bayes class to use Naïve Bayes**Algorithm*

**from sklearn.naive\_bayes import GaussianNB**

**gauss = GaussianNB()**

**gauss.fit(X\_train, Y\_train)**

*#Using DecisionTreeClassifier of tree class to use Decision Tree**Algorithm*

**from sklearn.tree import DecisionTreeClassifier**

**tree = DecisionTreeClassifier(criterion = 'entropy', random\_state = 0)**

**tree.fit(X\_train, Y\_train)**

*#Using RandomForestClassifier method of ensemble class to use**Random Forest Classification algorithm*

**from sklearn.ensemble import RandomForestClassifier**

**forest = RandomForestClassifier(n\_estimators = 10, criterion = 'entropy', random\_state = 0)**

**forest.fit(X\_train, Y\_train)**

*#print model accuracy on the training data.*

**print('[0]Logistic Regression Training Accuracy:', log.score(X\_train, Y\_train))**

**print('[1]K Nearest Neighbor Training Accuracy:', knn.score(X\_train, Y\_train))**

**print('[2]Support Vector Machine (Linear Classifier) Training Accuracy:', svc\_lin.score(X\_train, Y\_train))**

**print('[3]Support Vector Machine (RBF Classifier) Training Accuracy:', svc\_rbf.score(X\_train, Y\_train))**

**print('[4]Gaussian Naive Bayes Training Accuracy:', gauss.score(X\_train, Y\_train))**

**print('[5]Decision Tree Classifier Training Accuracy:', tree.score(X\_train, Y\_train))**

**print('[6]Random Forest Classifier Training Accuracy:', forest.score(X\_train, Y\_train))**

**return log, knn, svc\_lin, svc\_rbf, gauss, tree, forest**

*#Get and train all of the models*

**model = models(X\_train,Y\_train)**

**from sklearn.metrics import confusion\_matrix**

**for i in range(len(model)):**

**cm = confusion\_matrix(Y\_test, model[i].predict(X\_test))**

*#extracting TN, FP, FN, TP*

**TN, FP, FN, TP = confusion\_matrix(Y\_test, model[i].predict(X\_test)).ravel()**

**print(cm)**

**print('Model[{}] Testing Accuracy = "{} !"'.format(i, (TP + TN) / (TP + TN + FN + FP)))**

**print()** *# Print a new line*

*#Get the importance of the features*

**forest = model[6]**

**importances = pd.DataFrame({'feature':titanic.iloc[:, 1:8].columns,'importance':np.round(forest.feature\_importances\_,3)})**

**importances = importances.sort\_values('importance',ascending=False).set\_index('feature')**

**importances**

*#Visualize the importance*

**importances.plot.bar()**

*#Print Prediction of Random Forest Classifier model*

**pred = model[6].predict(X\_test)**

**print(pred)**

*#Print a space*

**print()**

*#Print the actual values*

**print(Y\_test)**

**my\_survival = [[1,1,20,0, 0, 5, 1]]**

**name=”Sumith”**

*#Print Prediction of Random Forest Classifier model*

**pred = model[6].predict(my\_survival)**

**print(pred)**

**if pred == 0:**

**print('Oh no!’,name,’ You Died')**

**else:**

**print('Nice!’,name,’ You survived')**

**Other source code ref:**

[**https://github.com/Sumith20/Titanic-survival-Prediction**](https://github.com/Sumith20/Titanic-survival-Prediction)

**SCREENSHOTS**

**6.1 Final Output code**

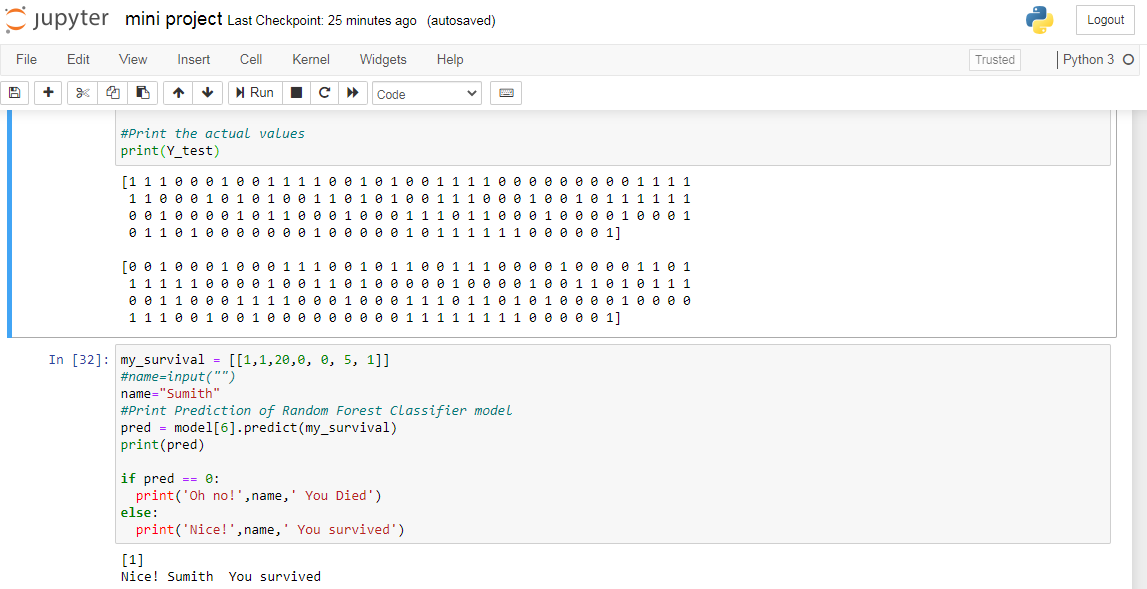
****

FIG.NO:6.1.1

**6.2 Excel file Screenshot**

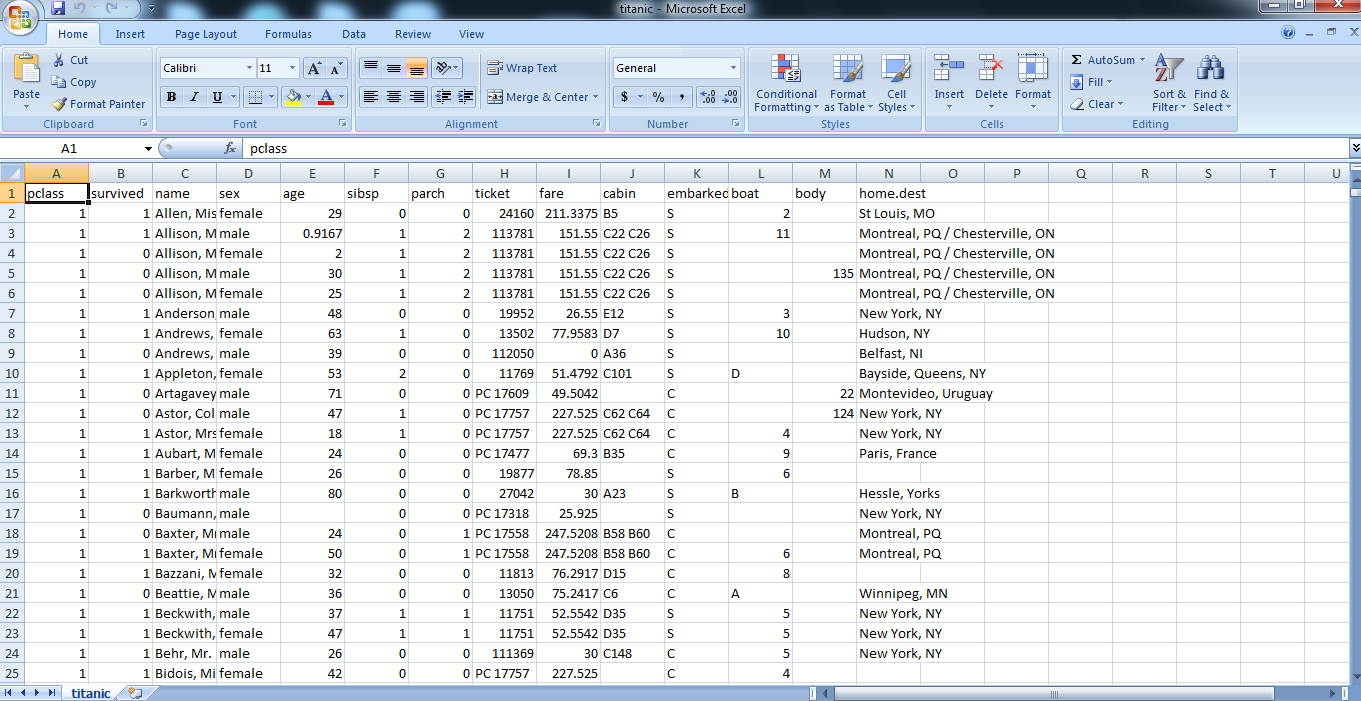
****

FIG.NO:6.2.1