**1: Introduction**

**Titanic survival**  
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.  
  
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. Although there was some  
element of luck involved in surviving the sinking, some groups of people were more  
likely to survive than others, such as women, children, and the upper-class.  
  
In this challenge, we ask you to complete the analysis of what sorts of people were likely to survive.  
In particular, we ask you to apply the tools of machine learning to predict which passengers survived the tragedy.

### 2: PYTHON

### 2.1 History-

The beta version was released in July 2010, with the 1.0 arriving 3 months later. Version 2.0 was released on 13 December 2011, version 3.0 on 24 September 2013, and version 4.0 on November 19, 2014.

PyCharm Community Edition, the open source version of PyCharm, became available on 22 October 2013.

### 2.2 Advantages/Benefits of Python-

The diverse application of the Python language is a result of the combination of features which give this language an edge over others. Some of the benefits of programming in Python include:

#### 1. Presence of Third Party Modules:

The Python Package Index (PyPI) contains numerous third-party modules that make Python capable of interacting with most of the other languages and platforms.

#### 2. Extensive Support Libraries:

Python provides alarge standard library which includes areas like internet protocols, string operations, web services tools and operating system interfaces. Many high use programming tasks have already been scripted into the standard library which reduces length of code to be written significantly.

#### 3. Open Source and Community Development:

Python language is developed under an OSI-approved open source license, which makes it free to use and distribute, including for commercial purposes.

Further, its development is driven by the community which collaborates for its code through hosting conferences and mailing lists, and provides for its numerous modules.

#### 4. Learning Ease and Support Available:

Python offers excellent readability and uncluttered simple-to-learn syntax which helps beginners to utilize this programming language. The code style guidelines, PEP 8, provide a set of rules to facilitate the formatting of code. Additionally, the wide base of users and active developers has resulted in a rich internet resource bank to encourage development and the continued adoption of the language.

#### 5. User-friendly Data Structures:

Python has built-in list and dictionary data structures which can be used to construct fast runtime data structures. Further, Python also provides the option of dynamic high-level data typing which reduces the length of support code that is needed.

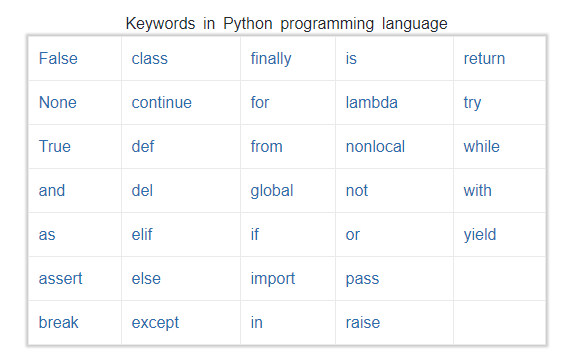
#### 6. Productivity and Speed:

Python has clean object-oriented design, provides enhanced process control capabilities, and possesses strong integration and text processing capabilities and its own unit testing framework, all of which contribute to the increase in its speed and productivity. Python is considered a viable option for building complex multi-protocol network applications.

**2.3 Keywords-**

Keywords are the reserved words in Python. We cannot use a keyword as variable name, function name or any other identifier.

Here's a list of all keywords in Python Programming



**3: Libraries**

As Python has gained a lot of traction in the recent years in Data Science industry, I wanted to outline some of its most useful libraries for data scientists and engineers, based on recent experience.And, since all of the libraries are open sourced, we have added commits, contributors count and other metrics from Github, which could be served as a proxy metrics for library popularity.

**3.1 NumPy-**

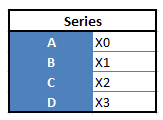
When starting to deal with the scientific task in Python, one inevitably comes for help to Python’s SciPy Stack, which is a collection of software specifically designed for scientific computing in Python (do not confuse with SciPy library, which is part of this stack, and the community around this stack). This way we want to start with a look at it. However, the stack is pretty vast, there is more than a dozen of libraries in it, and we want to put a focal point on the core packages (particularly the most essential ones).The most fundamental package, around which the scientific computation stack is built, is NumPy (stands for Numerical Python). It provides an abundance of useful features for operations on n-arrays and matrices in Python. The library provides vectorization of mathematical operations on the NumPy array type, which ameliorates performance and accordingly speeds up the execution.

**3.2 Pandas-**

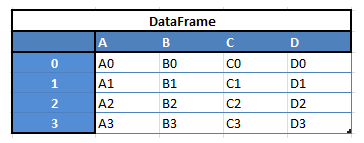
Pandas is a Python package designed to do work with “labeled” and “relational” data simple and intuitive. Pandas is a perfect tool for data wrangling. It designed for quick and easy data manipulation, aggregation, and visualization.

There are two main data structures in the library:

**“Series”** — one-dimensional



**“Data Frames”**, two-dimensional



Here is just a small list of things that you can do with Pandas:

* Easily delete and add columns from DataFrame
* Convert data structures to DataFrame objects
* Handle missing data, represents as NaNs
* Powerful grouping by functionality

**3.3 Scipy-**

SciPy is a library of software for engineering and science. Again you need to understand the difference between SciPy Stack and SciPy Library. SciPy contains modules for linear algebra, optimization, integration, and statistics. The main functionality of SciPy library is built upon NumPy, and its arrays thus make substantial use of NumPy. It provides efficient numerical routines as numerical integration, optimization, and many others via its specific submodules. The functions in all submodules of SciPy are well documented — another coin in its pot.

**3.4 SciKit-Learn 0.19.1-**

Scikits are additional packages of SciPy Stack designed for specific functionalities like image processing and machine learning facilitation. In the regard of the latter, one of the most prominent of these packages is scikit-learn. The package is built on the top of SciPy and makes heavy use of its math operations.Thescikit-learn exposes a concise and consistent interface to the common machine learning algorithms, making it simple to bring ML into production systems. The library combines quality code and good documentation, ease of use and high performance and is de-facto industry standard for machine learning with Python.

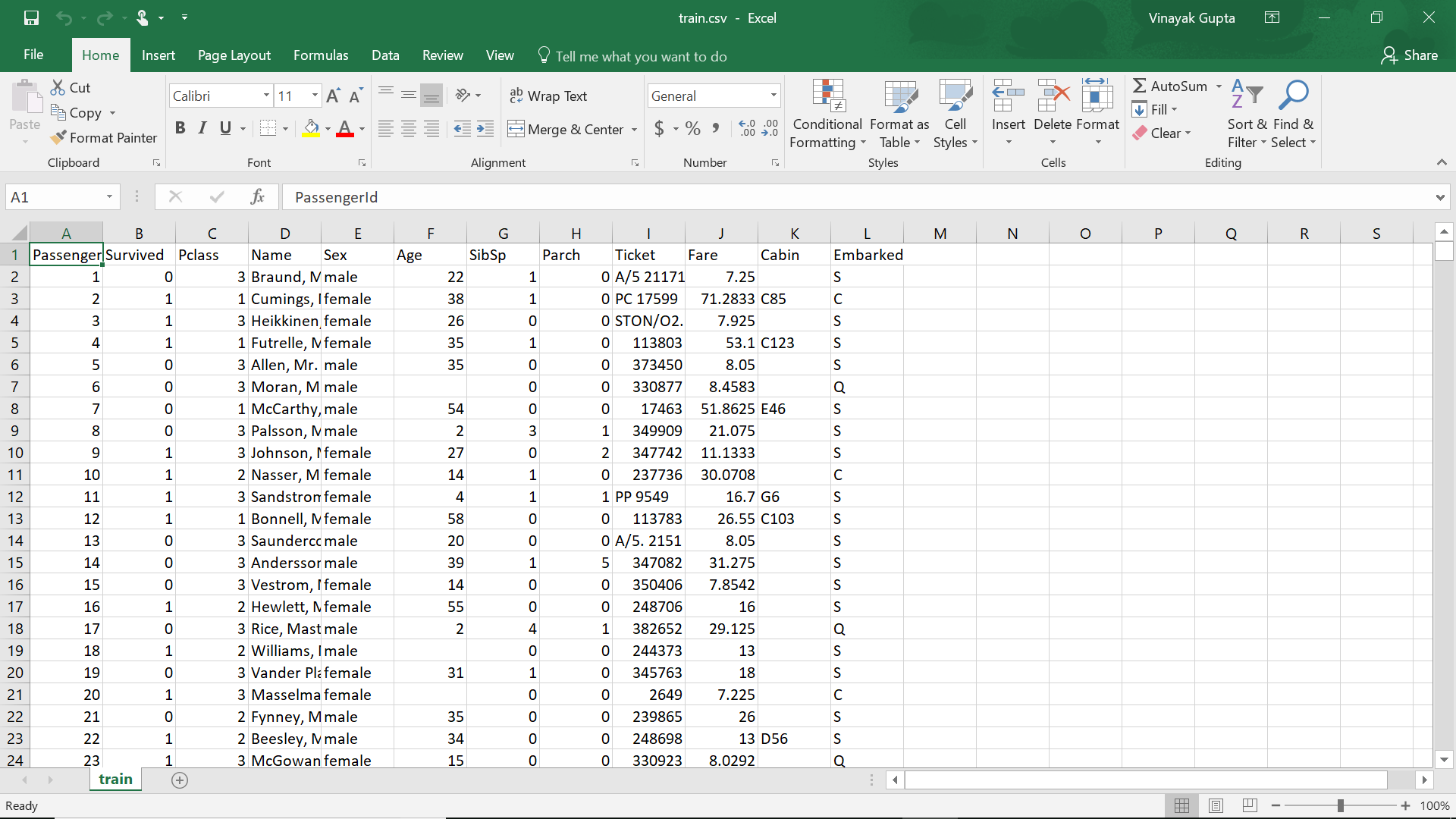
**3.5 Matplotlib-**

Another SciPy Stack core package and another Python Library that is tailored for the generation of simple and powerful visualizations with ease is Matplotlib. It is a top-notch piece of software which is making Python (with some help of NumPy, SciPy, and Pandas) a cognizant competitor to such scientific tools as MatLab or Mathematica.However, the library is pretty low-level, meaning that you will need to write more code to reach the advanced levels of visualizations and you will generally put more effort, than if using more high-level tools, but the overall effort is worth a shot.With a bit of effort you can make just about any visualizations:

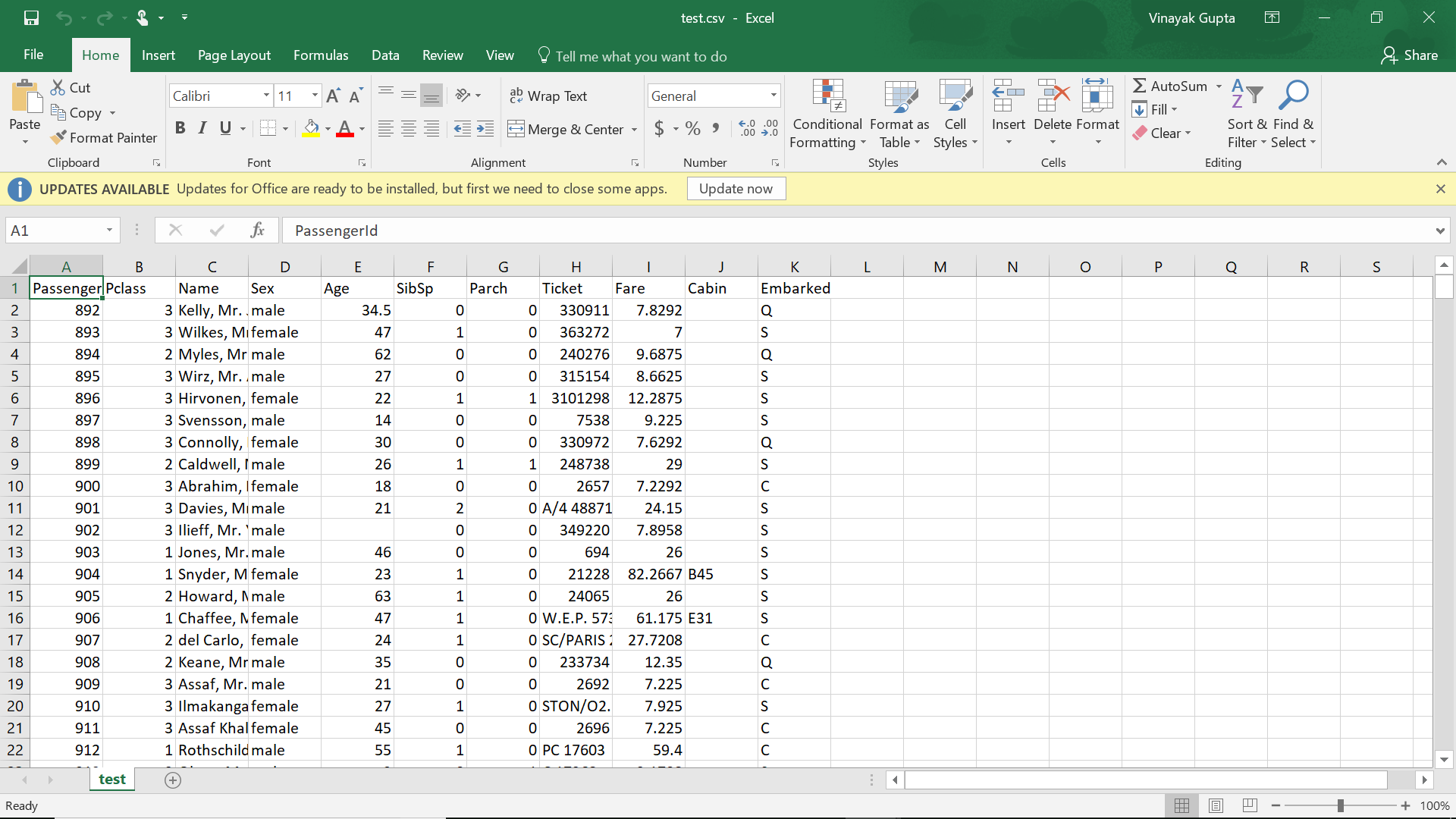
* Line plots
* Scatter plots
* Bar charts and Histograms

**4: DATA SETS**

* In a dataset a training set is implemented to build up a model, while a test (or validation) set is to validate the model built. Data points in the training set are excluded from the test (validation) set. Usually a dataset is divided into a training set, a validation set (some people use 'test set' instead) in each iteration, or divided into a training set, a validation set and a test set in each iteration.
* **test.csv : 891 entries, 12 attributes (68% of dataset)**



* **train.csv : 419 entries, 11 attributes (38% of dataset)**



**5: DESCRIPTION**

**5.1 Spliting data into training and test**

The data has been split into two groups:  
  
1) training set (train.csv)  
2) test set (test.csv)  
The training set should be used to build your machine learning models.  
For the training set, we provide the outcome (also known as the "ground truth") for each passenger.  
Your model will be based on "features"like passengers' gender and class.  
You can also use feature engineering to create new features.  
  
The test set should be used to see how well your model performs on unseen data.  
For the test set, we do not provide the ground truth for each passenger.  
It is your job to predict these outcomes.  
For each passenger in the test set, use the model you trained to predict whether or not they  
 survived the sinking of the Titanic.  
  
  
**5.2 Description about data:-**

Below is a brief information about each columns of the dataset:

|  |  |  |
| --- | --- | --- |
| 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 |

**6: IMPLEMENTATION**

*The following steps are followed:*

*1)Import Necessary Libraries  
2)Read In and Explore the Historic Data  
3)Data Analysis  
4)Data Visualization  
5)Cleaning Data  
6)Choosing the Best Model  
7)Creating Submission File*

*Step-1) Import Necessary Libraries  
First off, we need to import several Python libraries such as numpy, pandas,matplotlib and seaborn.*

*STEP-2) Read in and Explore the Data  
It's time to read in our training and testing data using pd.read\_csv, and take a first look at the training data using the describe() function.  
import train and test CSV files*

*STEP-3) Data Analysis  
We're going to consider the features in the dataset and how complete they are.  
get a list of the features within the dataset*

*1)There are a total of 891 passengers in our training set.  
  
2)The Age feature is missing approximately 19.8% of its values.Hence Age feature is pretty important to survival,so we should probably attempt to fill these gaps.  
  
3)The Cabin feature is missing approximately 77.1% of its values.Since so much of the feature is missing, it would be hard to fill in the missing values.We'll probably drop these values from our dataset.  
  
4)The Embarked feature is missing 0.22% of its values, which should be relatively harmless.check for any other unusable values*

*Relationship between Features and Survival  
In this section, we analyze relationship between different featureswith respect to Survival. We see how different feature values show different survival chance. We also plot different kinds of diagrams to visualize our data and findings.  
  
  
STEP-4) Data Visualization  
It's time to visualize our data so we can estimate few predictions.*

*1)PassengerId: An unique index for passenger rows. It starts from 1 for first row and increments by 1 for every new rows.  
  
2)Survived: Shows if the passenger survived or not. 1 stands for survived and 0 stands for not survived.  
  
3)Pclass: Ticket class. 1 stands for First class ticket. 2 stands for Second class ticket. 3 stands for Third class ticket.  
  
4)Name: Passenger's name. Name also contain title. "Mr" for man. "Mrs" for woman. "Miss" for girl. "Master" for boy.  
  
5)Sex: Passenger's sex. It's either Male or Female.  
  
6)Age: Passenger's age. "NaN" values in this column indicates that the age of that particular passenger has not been recorded.  
  
7)SibSp: Number of siblings or spouses travelling with each passenger.  
  
8)Parch: Number of parents of children travelling with each passenger.  
  
9)Ticket: Ticket number.  
  
10)Fare: How much money the passenger has paid for the travel journey.  
  
11)Cabin: Cabin number of the passenger. "NaN" values in this column indicates that the cabin number of that particular passenger has not been recorded.  
  
12)Embarked: Port from where the particular passenger was embarked/boarded.*

*STEP-5) Cleaning Data  
Time to clean our data to account for missing values and unnecessary information!  
Looking at the Test Data  
Let's see how our test data looks!*

*Cabin Feature  
we'll start off by dropping the Cabin feature since not a lot more useful information can be extracted from it.*

*Ticket Feature  
we can also drop the Ticket feature since it's unlikely to yield any useful information.*

*Embarked Feature  
now we need to fill in the missing values in the Embarked feature.*

*Age Feature  
Next we'll fill in the missing values in the Age feature. Since a higher percentage of values are missing, it would be illogical to fill all of them with the same value (as we did with Embarked). Instead, let's try to find a way to predict the missing ages. create a combined group of both datasets*

*STEP-6) Choosing the Best Model*

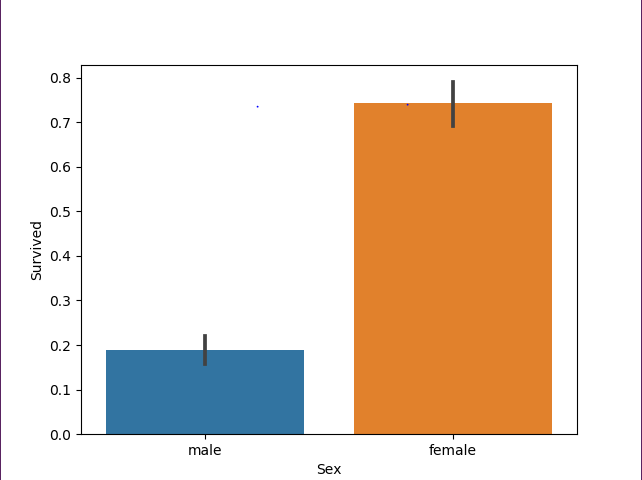
*Testing Different Models  
I will be testing some of the following models with my training data (got the list from here):  
  
1) Logistic Regression  
2) Gaussian Naive Bayes  
3) Support Vector Machines  
4) Linear SVC  
5) Perceptron  
6) Decision Tree Classifier  
7) Random Forest Classifier  
8) KNN or k-Nearest Neighbors  
9) Stochastic Gradient Descent  
10) Gradient Boosting Classifier*

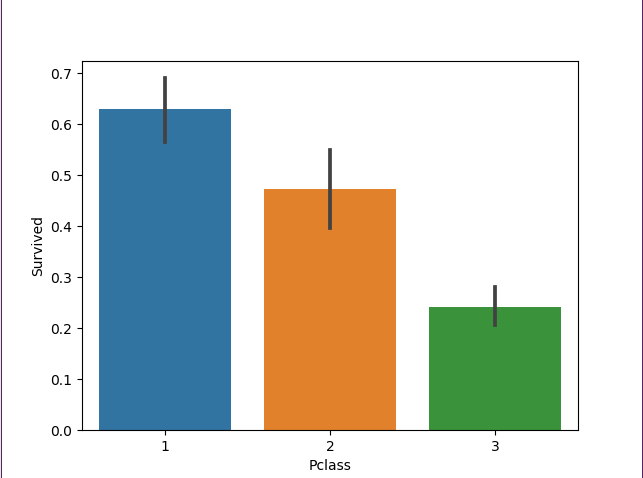
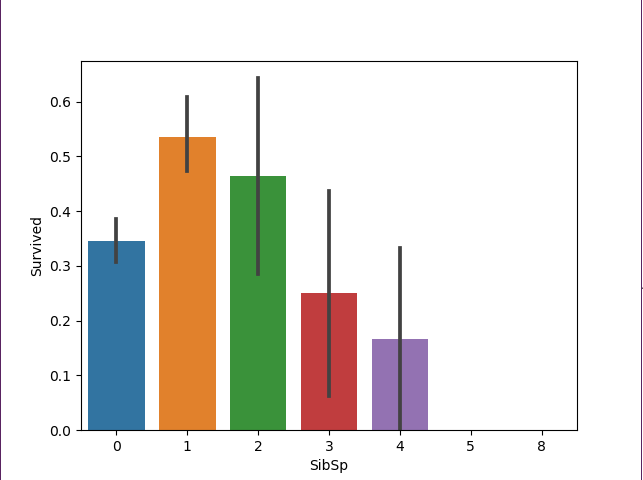
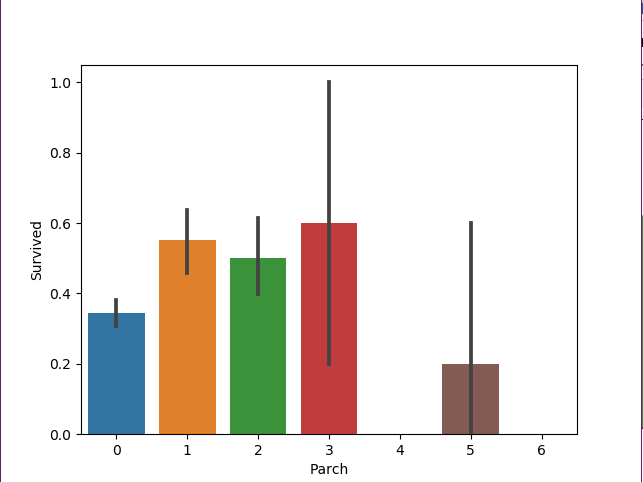
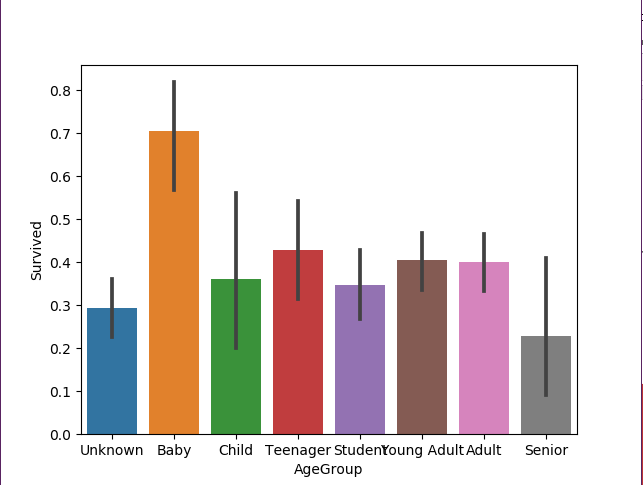
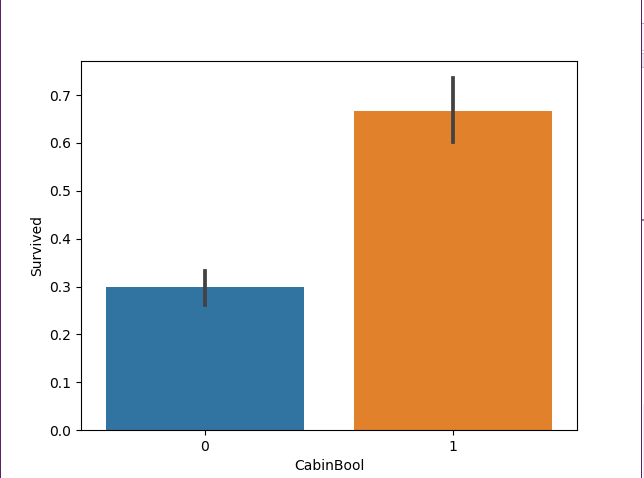
*STEP-7) Creating Submission Result File  
It is time to create a submission.csv file which includes our predictions for test data set ids as PassengerId and predict survival*

**7: GRAPHS AND OUTPUT**

**7.1 Graphs and Relationships:-**

* **Sex vs Survival**



* **Class vs survival**
* **Sibblings vs survival** 
* **Parent-child relationship vs survival** 
* **Age vs survival** 
* **Cabin vs survival** 

**7.2Output:-**  
Relation between different models:  
 Model Score  
#6 Random Forest 86.29  
#9 Gradient Boosting Classifier 84.77  
#2 Support Vector Machines 82.74  
#8 Stochastic Gradient Descent 80.71  
#5 Decision Tree 80.20  
#0 Logistic Regression 79.19  
#4 Perceptron 79.19  
#1 Gaussian Naive Bayes 78.68  
#3 Linear SVC 78.17  
#7 KNN 77.66

**References**

* https://rcplindia.in/about-us.php
* https://medium.com/activewizards-machine-learning-company/top-15-python-libraries-for-data-science-in-in-2017-ab61b4f9b4a7
* https://www.python.org/downloads/