

Jordan University of Science and Technology

College of Computer Sciences & Information Technology

Student Name: Ihab k. m. shhadat ID: 128016 Lujain h. m. Smadi ID: 121868

The RMS Titanic was a British passenger liner that sank in the North Atlantic Ocean in the early morning hours of 15 April 1912, after it collided with an iceberg during its maiden voyage from Southampton to New York City. There were an estimated 2,224 passengers and crew aboard the ship, and more than 1,500 died, making it one of the deadliest commercial peacetime maritime disasters in modern history. The RMS Titanic was the largest ship afloat at the time it entered service and was the second of three Olympic-class ocean liners operated by the White Star Line. The Titanic was built by the Harland and Wolff shipyard in Belfast.

 Kaggle has created a number of competitions designed for beginners. The most popular of these competitions and the one we’ll be looking at is about predicting which passengers survived the sinking of the Titanic. In this competition, we have a data set of different information about passengers aboard the Titanic, and we see if we can use that information to predict whether those people survived or not.  We have two files: Training data set and Testing Data set.

The training set has a number of feature columns a column of the target values we are trying to predict: in this case, Survival. The testing set contains all of the same feature columns which contain various descriptive data, as well as but is missing the target value column.

So, firstly, we have read the training and testing data and stored them in train and test using these equations respectively:

train=pd.read\_csv('train.csv')

test=pd.read\_csv('test.csv')

we listed the training data file and we got the next result:

**891** entries available, ranged from 0 to 890.

Data columns (total of 12 columns):

* PassengerID— A column to identify each row and make submissions easier.
* Survived— Whether the passenger survived or not and the value we are predicting (0=No, 1=Yes).
* Pclass— The class of the ticket the passenger purchased *(1=1st, 2=2nd, 3=3rd)*
* Sex— The passenger’s gender
* Age— The passenger’s age in years
* SibSp— The number of siblings or spouses the passenger had aboard the Titanic
* Parch— The number of parents or children the passenger had aboard the Titanic
* Ticket— The passenger’s ticket number
* Fare— The fare the passenger paid
* Cabin— The passenger’s cabin number
* Embarked— The port where the passenger embarked *(C=Cherbourg, Q=Queenstown, S=Southampton)*

in addition to the survived column.

dtypes: float64(2), int64(5), object(5)

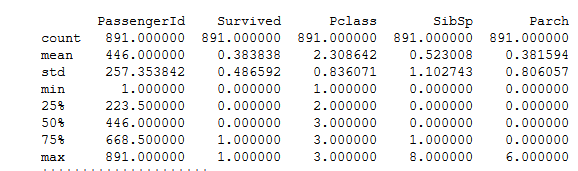
memory usage: 83.6+ KB

and the test data file have listed and the result was the same as the training data file **except for** the survived column.

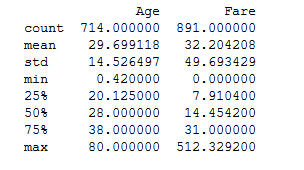
dtypes: float64(2), int64(4), object(5)

memory usage: 36.0+ KB.

Then we printed a Summary Statistic of the Numeric (int64) and  (float64) columns as we can see in the next figures :

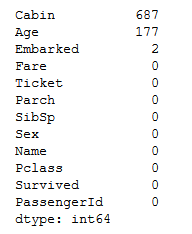


*fig(1): a Summary Statistic of the Numeric (*int64*)columns*



*fig(2): a Summary Statistic of the Numeric (*float64*)columns*

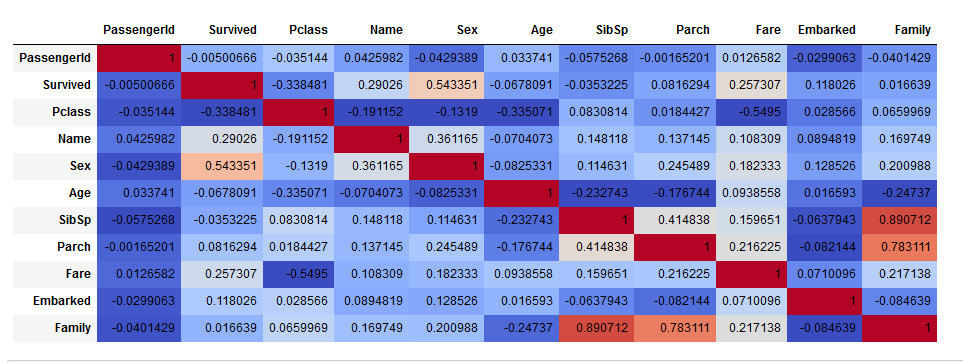
Then  we  listed  the  **null** cell count for each column,  and  we  have  got  the  next  result:



*fig(2):* Null cell *columns*

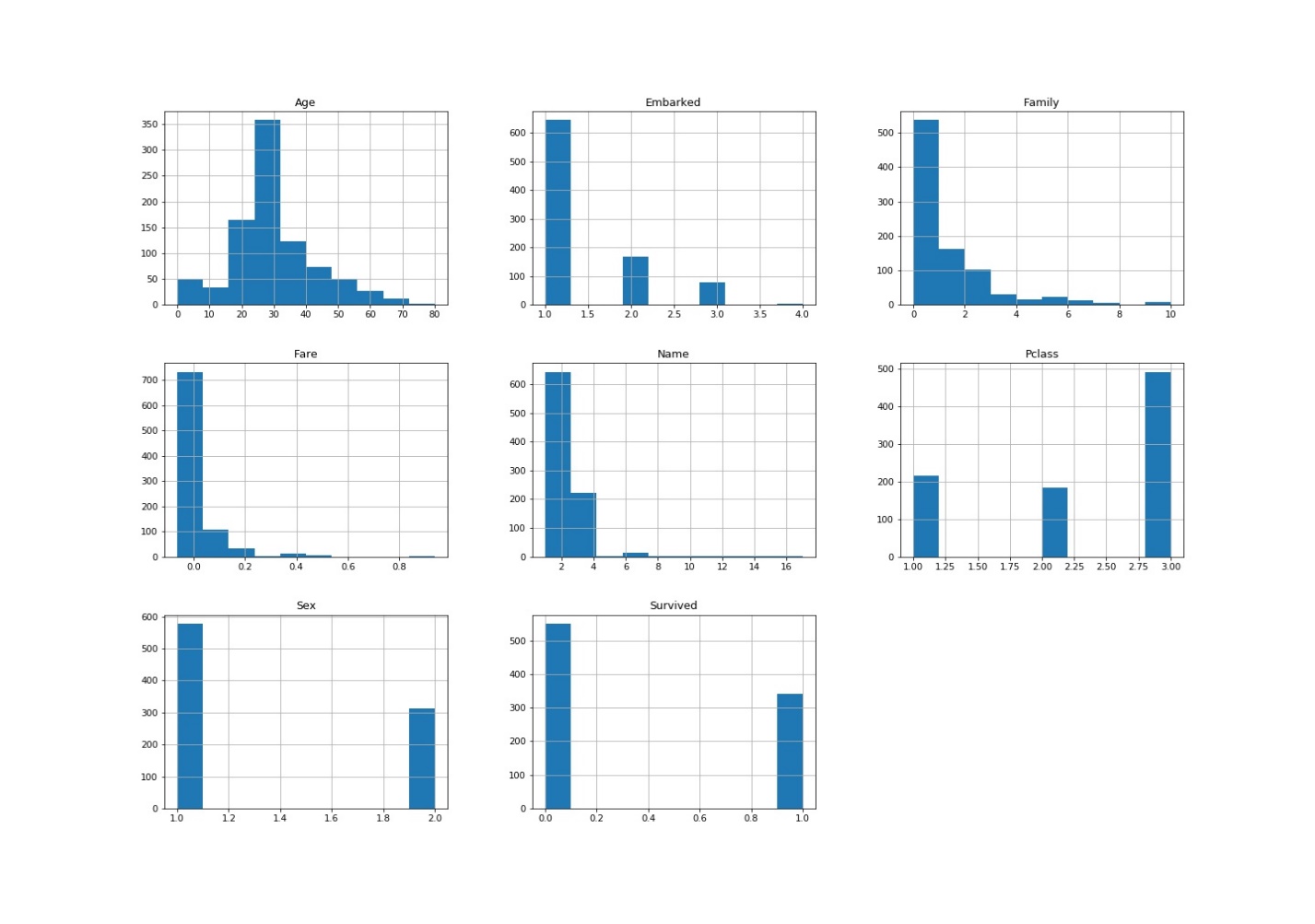
After we listed all the important information with its details, we started by write functions for mapping sex which gives male "1" and females "2",  then we write a *fillAge* function which fills all null values of Age column by the average of ages. Also, we have written a *Name* function which receives the data set from each of the two files( train, test), this function represents the first name of passengers (surname) in each file.

*fillEmbarkedAndMap* is a function that fills the null values of an Embarked column by the most repeated embarked appears in the train data.  Also, we have merged Patch and SibSp in a new feature called "family" because we noticed that both of them have a high correlation as we see in the next figure:

*fig(3): the  correlation  between  features*

Also, we fill the null values in test data by the mean of fare column and we scale this column to be from 0 to 16.

We have drawn a histogram for each feature, this histogram shows each feature and the number of passengers related to that feature:



*fig(3): number  of  passengers  related  to  each  feature*

In our code, we have dropped the useless features such as passenger, Ticket SibSp, Parch and Cabin. The ticket has dropped because there is about 650 different value. We replace SibSp and Parch by family because of the high correlation between them as we mentioned before. According to Cabin, we dropped it because there is about 687 null value.

After this operations, the ('Pclass' 'Name' 'Sex' 'Age' 'Fare' 'Embarked' 'Family') features remained in our data set, so we applied different classifiers on it and we calculated the accuracy after each one as the next table shows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Accuracy | Precision | Recall | F1\_Score |
| linearSVM | 0.791304 | 0.747077 | 0.701876 | 0.719348 |
| rbfSVM | 0.677894 | 0.846905 | 0.198679 | 0.319351 |
| ploySVM | 0.821591 | 0.795272 | 0.725277 | 0.757017 |
| NB | 0.792434 | 0.744016 | 0.707758 | 0.722502 |
| logreg | 0.800286 | 0.759944 | 0.707758 | 0.731015 |
| DT1 | 0.776640 | 0.800509 | 0.578730 | 0.660222 |
| DT2 | 0.777795 | 0.700760 | 0.742668 | 0.718262 |
| DT3 | 0.800211 | 0.792133 | 0.675277 | 0.719625 |
| RFC | 0.836192 | 0.828551 | 0.731032 | 0.773653 |
| GBC | 0.828307 | 0.811113 | 0.728005 | 0.764411 |
| BCNB | 0.786803 | 0.739239 | 0.695993 | 0.713370 |
| BCDT | 0.837290 | 0.823712 | 0.736871 | 0.776176 |
| ABC | 0.814837 | 0.785505 | 0.719395 | 0.747484 |
| VC | 0.830561 | 0.813985 | 0.728133 | 0.767073 |
| KNN | 0.784581 | 0.748902 | 0.663768 | 0.701991 |