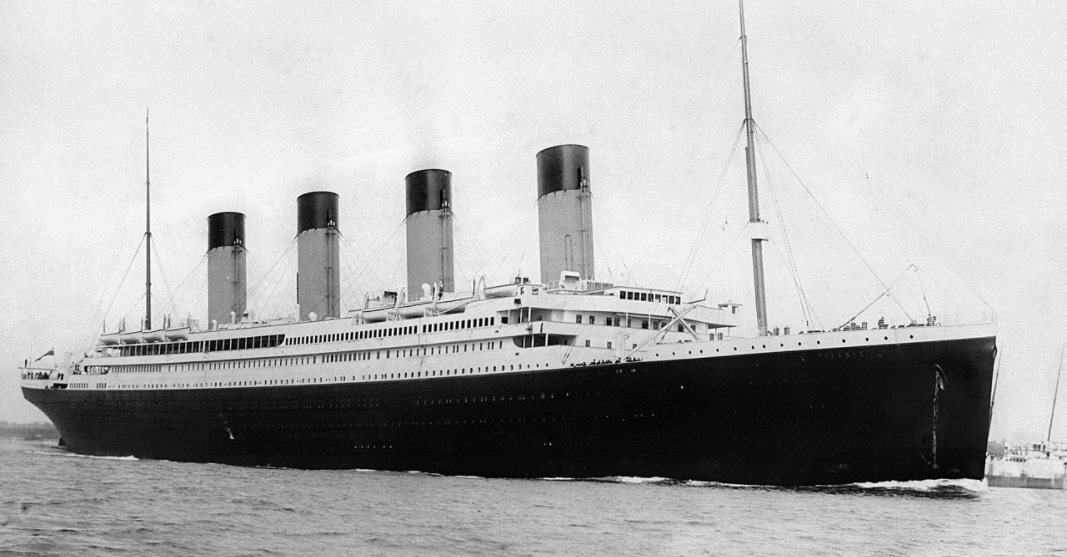
**Prediction of Survival of Titanic Passengers**





RMS Titanic was a British passenger liner operated by the White Star Line that sank in the North Atlantic Ocean on 15 April 1912, after striking an iceberg during her maiden voyage from South Hampton to New York City. Of the [estimated 2,224 passengers and crew](https://en.wikipedia.org/wiki/Sinking_of_the_RMS_Titanic#Casualties_and_survivors) aboard, more than 1,500 died.

**PROBLEM definition**

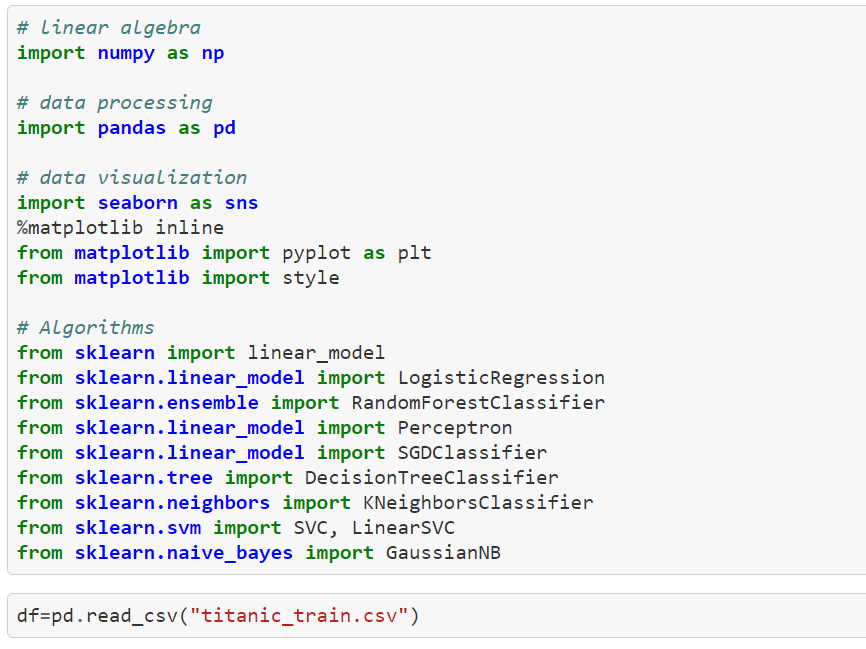
In this document, we will go through the whole process of creating a machine learning model on the famous Titanic dataset, which is used by many people all over the world. It provides information on the fate of passengers on the Titanic, summarized according to economic status (class), sex, age , sibling counts, embarkment points, and whether or not they survived the disaster. Based on these features, we needed to predict if an arbitrary passenger on Titanic would survive the sinking or not.

Unfortunately, there weren’t enough lifeboats for everyone onboard, resulting in the death of 1502 out of 2224 passengers and crew.

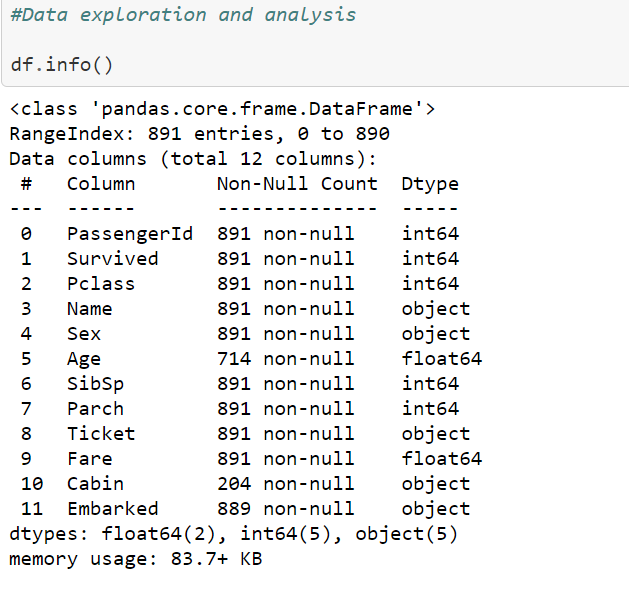
While there was some element of luck involved in surviving, it seems some groups of people were more likely to survive than others.

In this challenge, we ask you to build a predictive model that answers the question: “what sorts of people were more likely to survive?” using passenger data (ie name, age, gender, socio-economic class, etc).

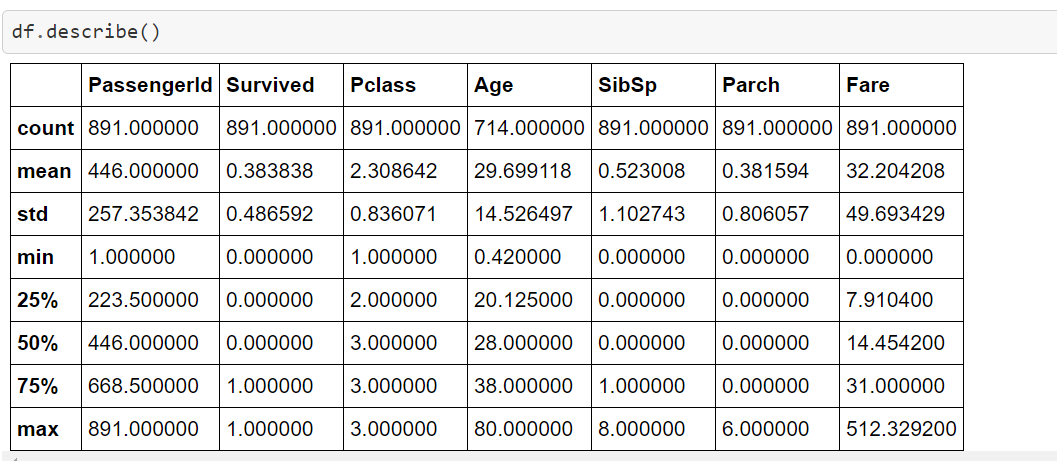
* **Importing libraries & Getting Data**



**Data Analysis**



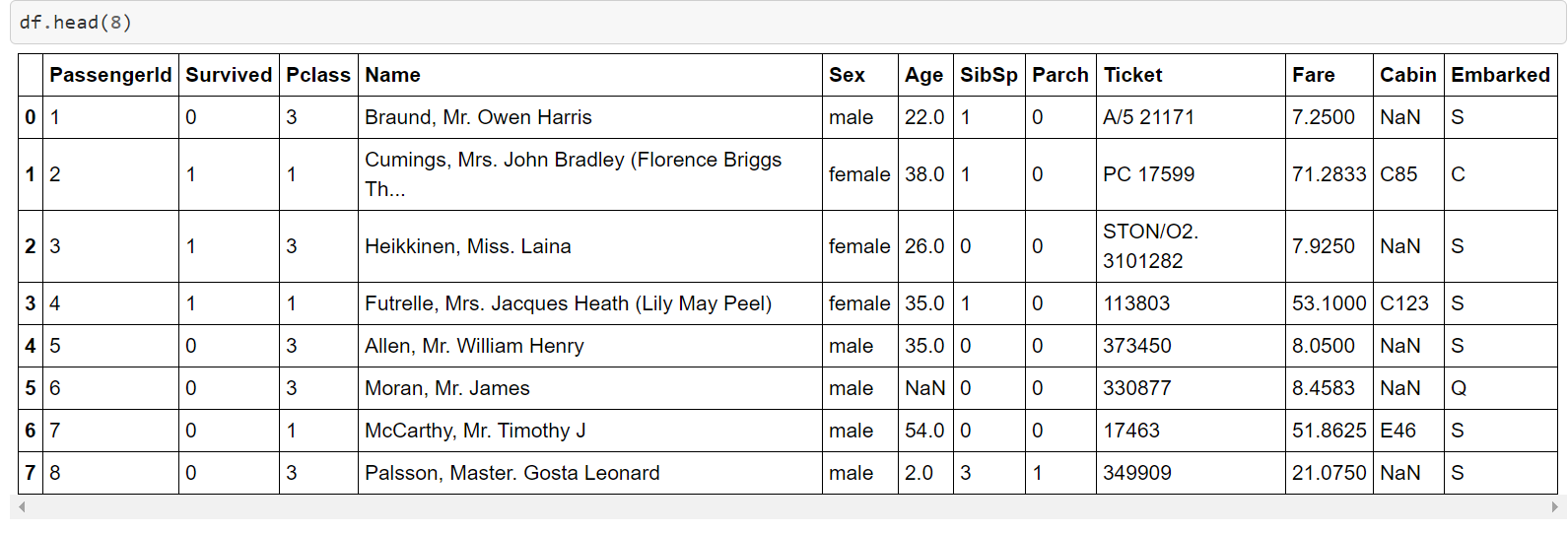
The training-set has 891 examples and 11 features plus the target variable (survived). 2 of the features are floats, 5 are integers and 5 are objects.



* Above we can see that 38% out of the training-set survived the Titanic.

We can also see that the passenger ages range from 0.4 to 80.

On top of that we can already detect some features, that contain missing values, like the ‘Age’ feature. Let’s get a detailed description on the data.



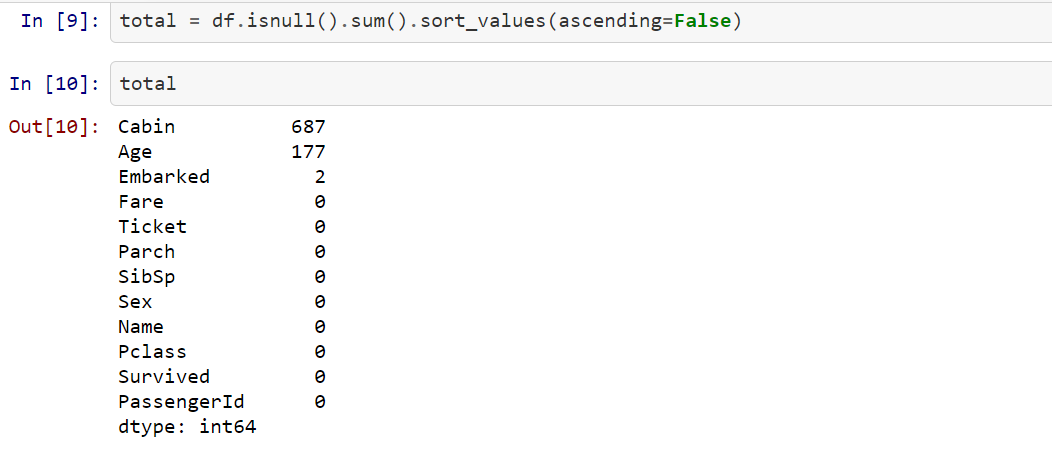
From a sample of the RMS Titanic data, we can see the various features present for each passenger on the ship:

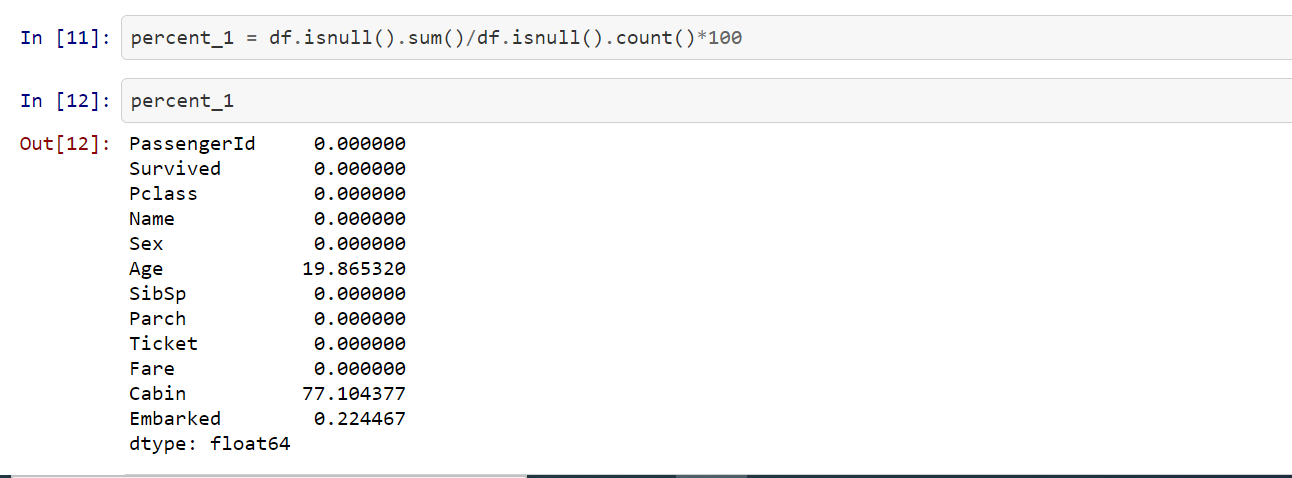
* **Survived**: Outcome of survival (0 = No; 1 = Yes)
* **Pclass**: Socio-economic class (1 = Upper class; 2 = Middle class; 3 = Lower class)
* **Name**: Name of passenger
* **Sex**: Sex of the passenger
* **Age**: Age of the passenger (Some entries contain NaN)
* **SibSp**: Number of siblings and spouses of the passenger aboard
* **Parch**: Number of parents and children of the passenger aboard
* **Ticket**: Ticket number of the passenger
* **Fare**: Fare paid by the passenger
* **Cabin** Cabin number of the passenger (Some entries contain NaN)
* **Embarked**: Port of embarkation of the passenger (C = Cherbourg; Q = Queenstown; S = Southampton)

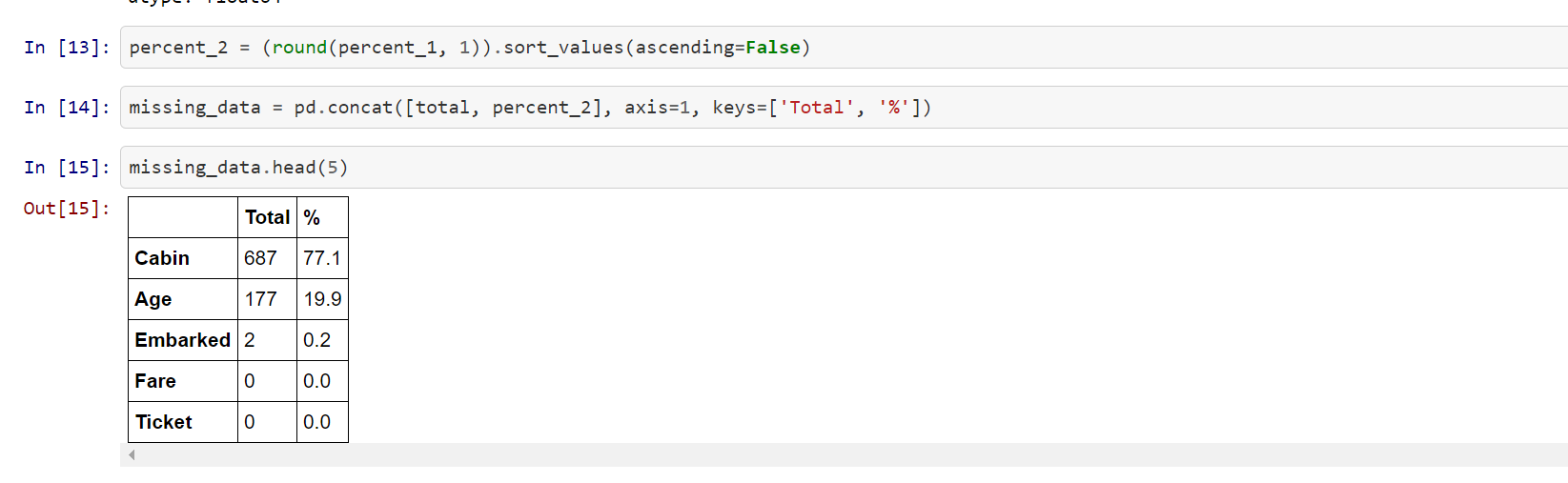
**Observations on dataset**

* First of all, that we need to convert a lot of features into numeric ones , so that the machine learning algorithms can process them.
* Furthermore, we can see that the features have widely different ranges, that we will need to convert into roughly the same scale.
* We can also spot some more features, that contain missing values (NaN = not a number), that we need to deal with.

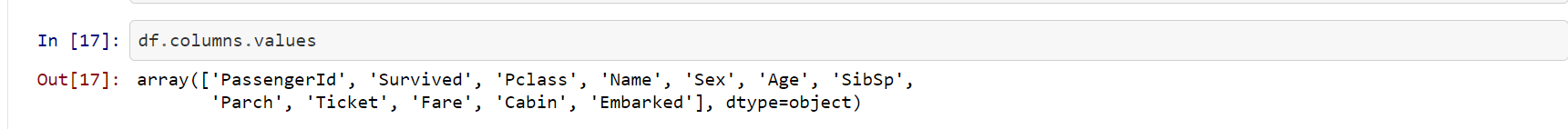
**Data Exploration and Visualization**

* **To check on the missing values** 

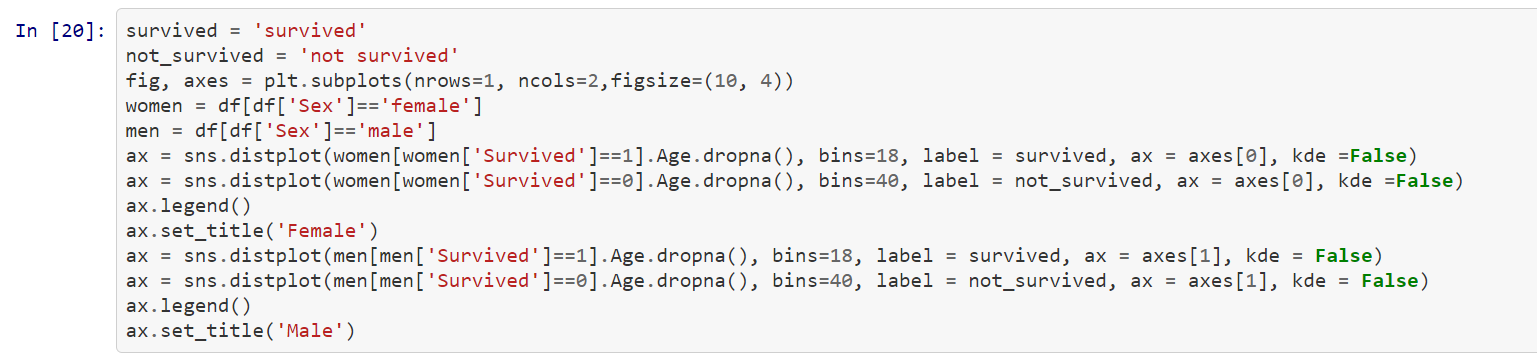
As Seen above, we have missing values in **Cabin ,Age, Embarked** .Percentage of missing data 

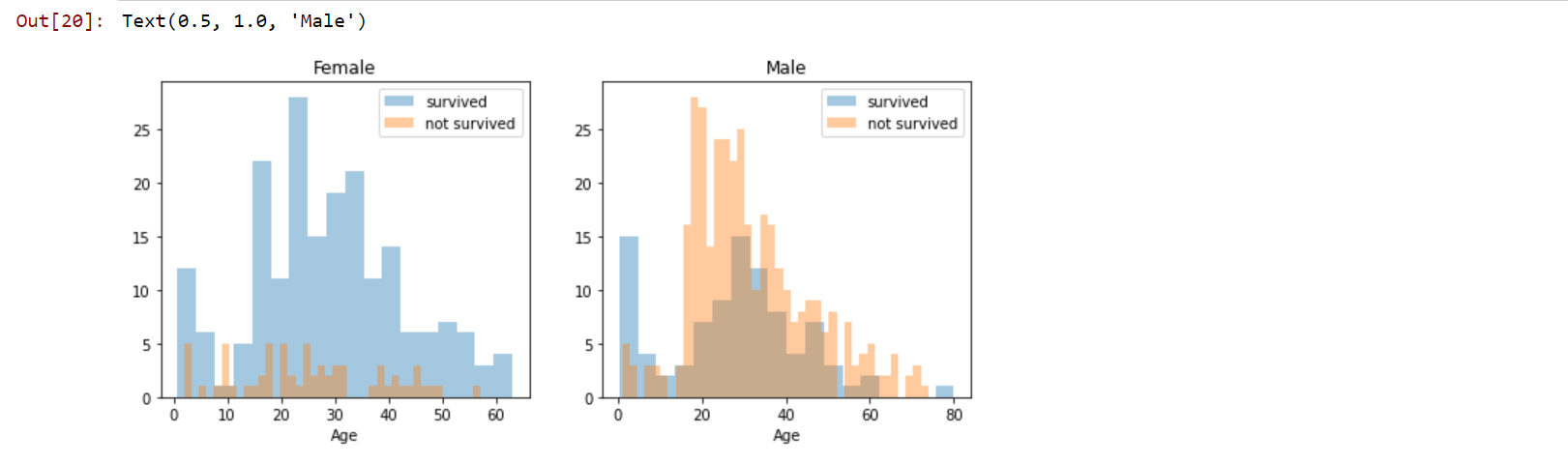


* The **Embarked** feature has only **2** missing values, which can easily be filled. It will be much more tricky, to deal with the ‘Age’ feature, which has 177 missing values. The ‘Cabin’ feature needs further investigation, but it looks like that we might want to drop it from the dataset, since 77 % of it are missing.



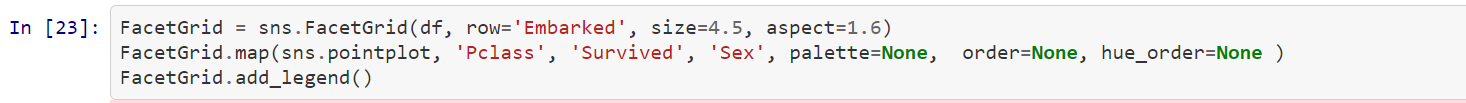
* It would make sense if everything except ‘PassengerId’, ‘Ticket’ and ‘Name’ would be correlated with a high survival rate
* Exploring variable Age and Sex

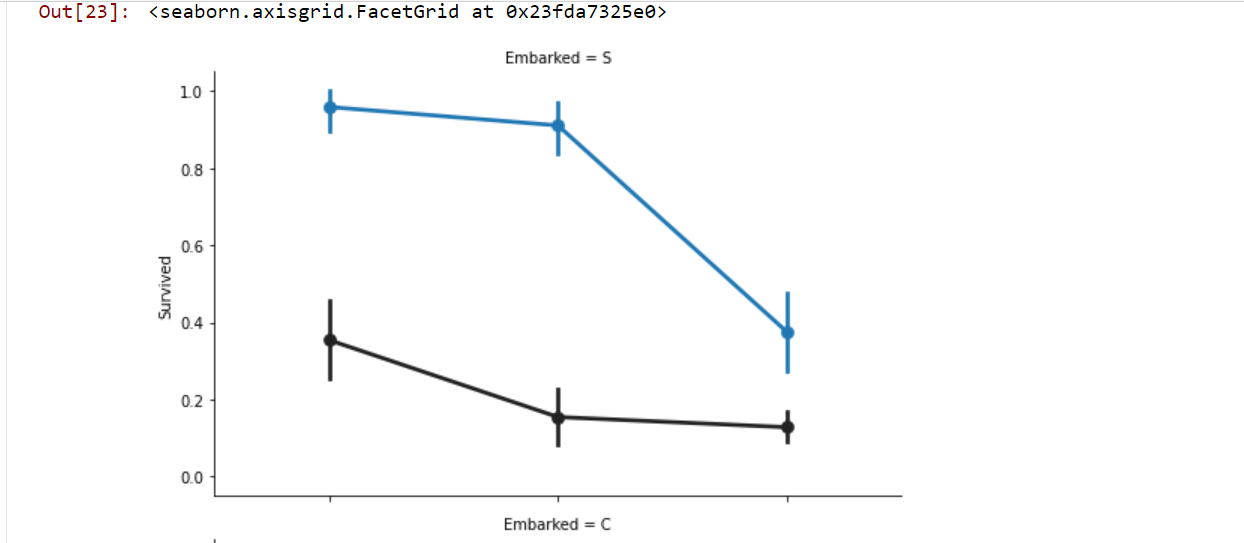


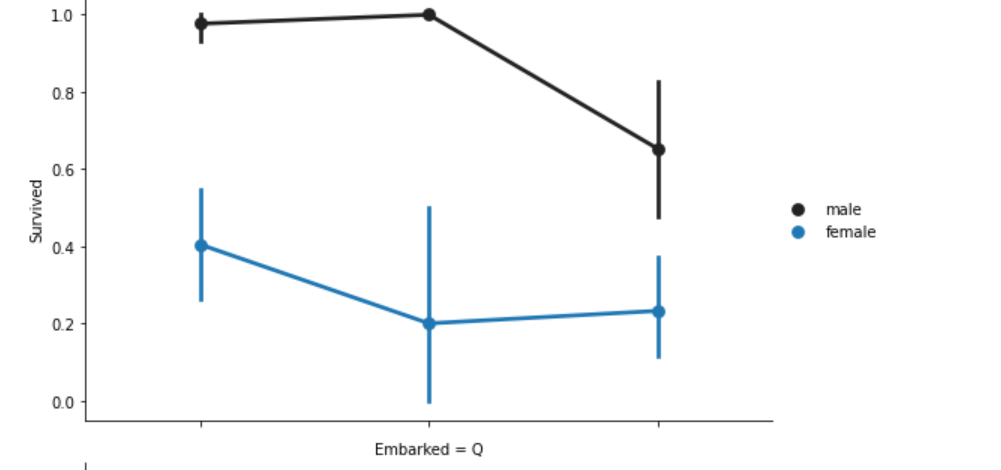


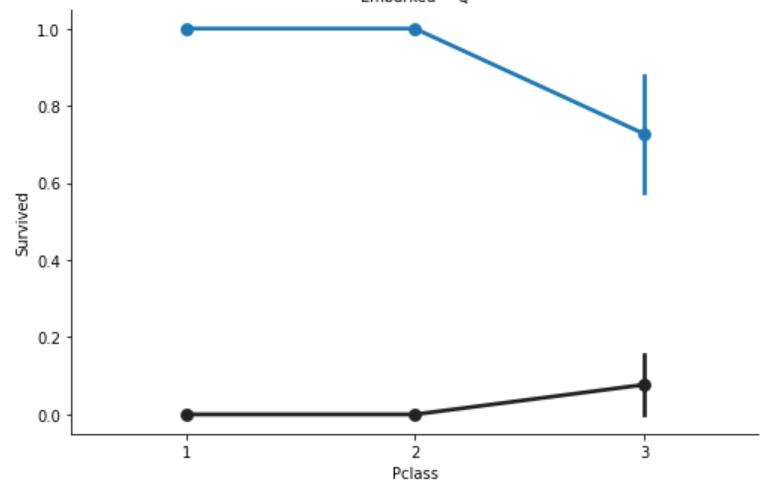
**Observation**

* You can see that **men have a high probability of survival** when they are between **18 and 30 years old**, which is also a little bit true for **women** but not fully. For women the survival chances are higher between **14 and 40**
* **Embarked pclass and sex**





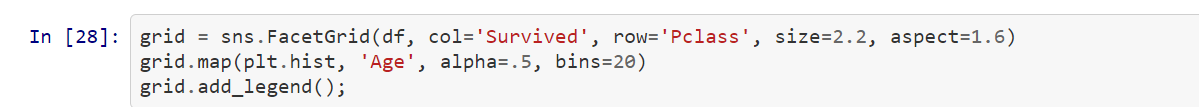


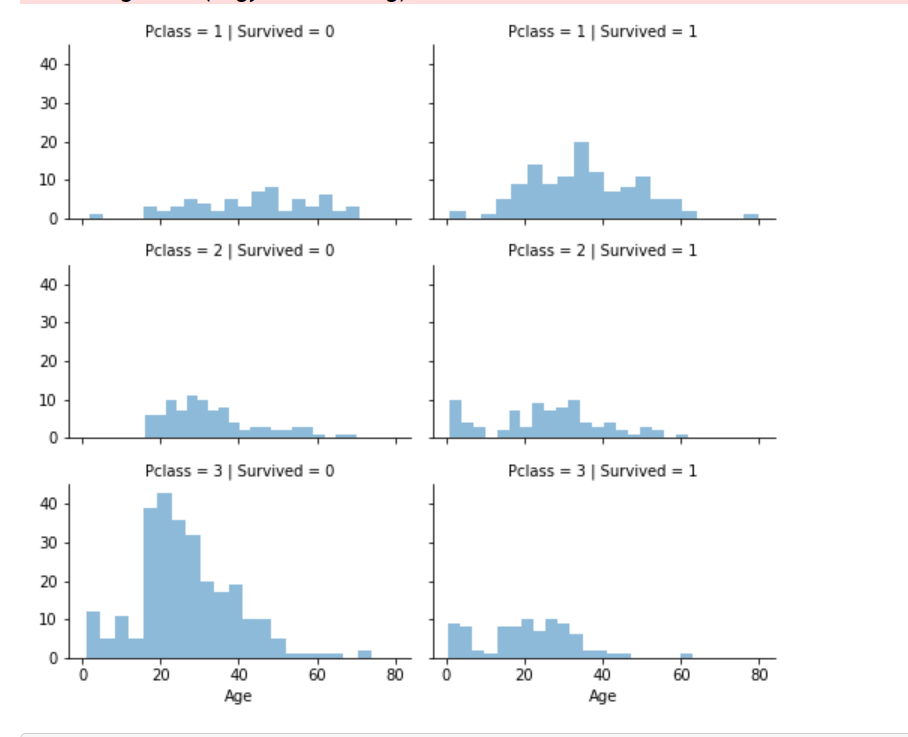


* Embarked seems to be correlated with survival, depending on the gender.
* **Women** on port **Q and on port S have a higher chance of survival**. The inverse is true, if they are **at port C**. **Men have a high survival probability if they are on port C**, but a **low probability if they are on port Q or S**.

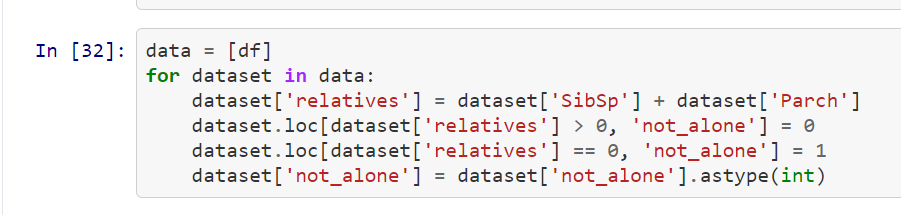


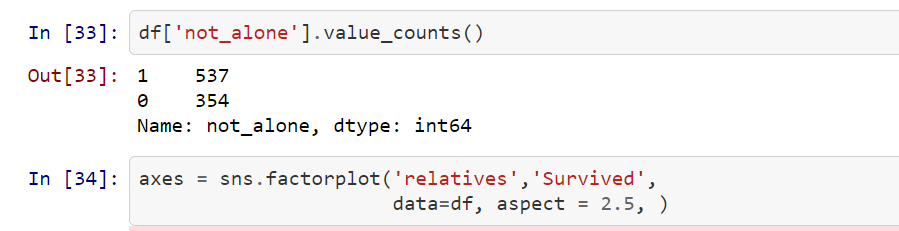
* Here we see clearly, that Pclass is contributing to a person’s chance of survival, especially if this person is in class 1. We will create another pclass plot below.

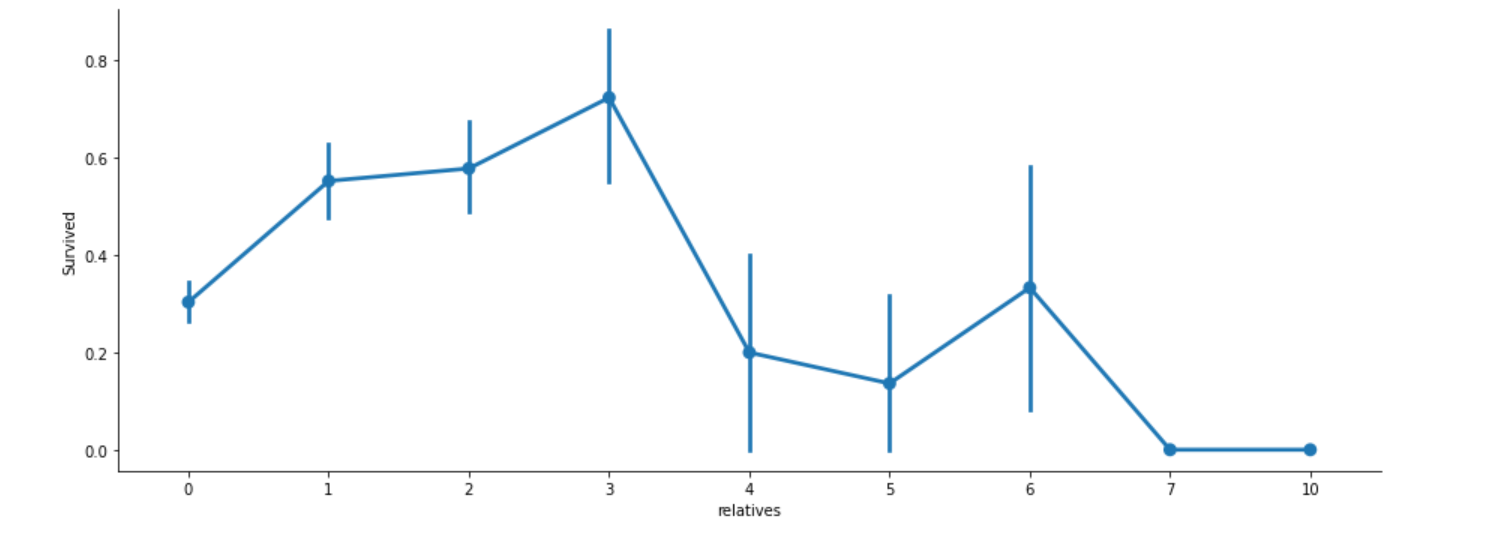




* The plot above confirms our assumption about pclass 1, but we can also spot a high probability that a person in pclass 3 will not survive.
* **Exploring SibSp and Parch Variable**
* SibSp and Parch would make more sense as a combined feature, that shows the total number of relatives, a person has on the Titanic. I will create it below and also a feature that shows if someone is not alone



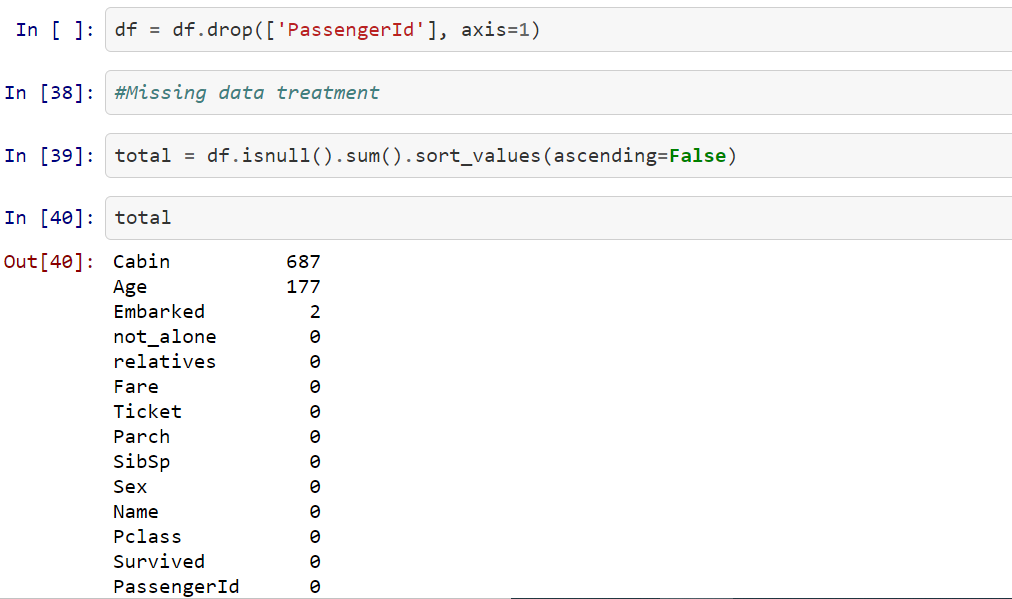




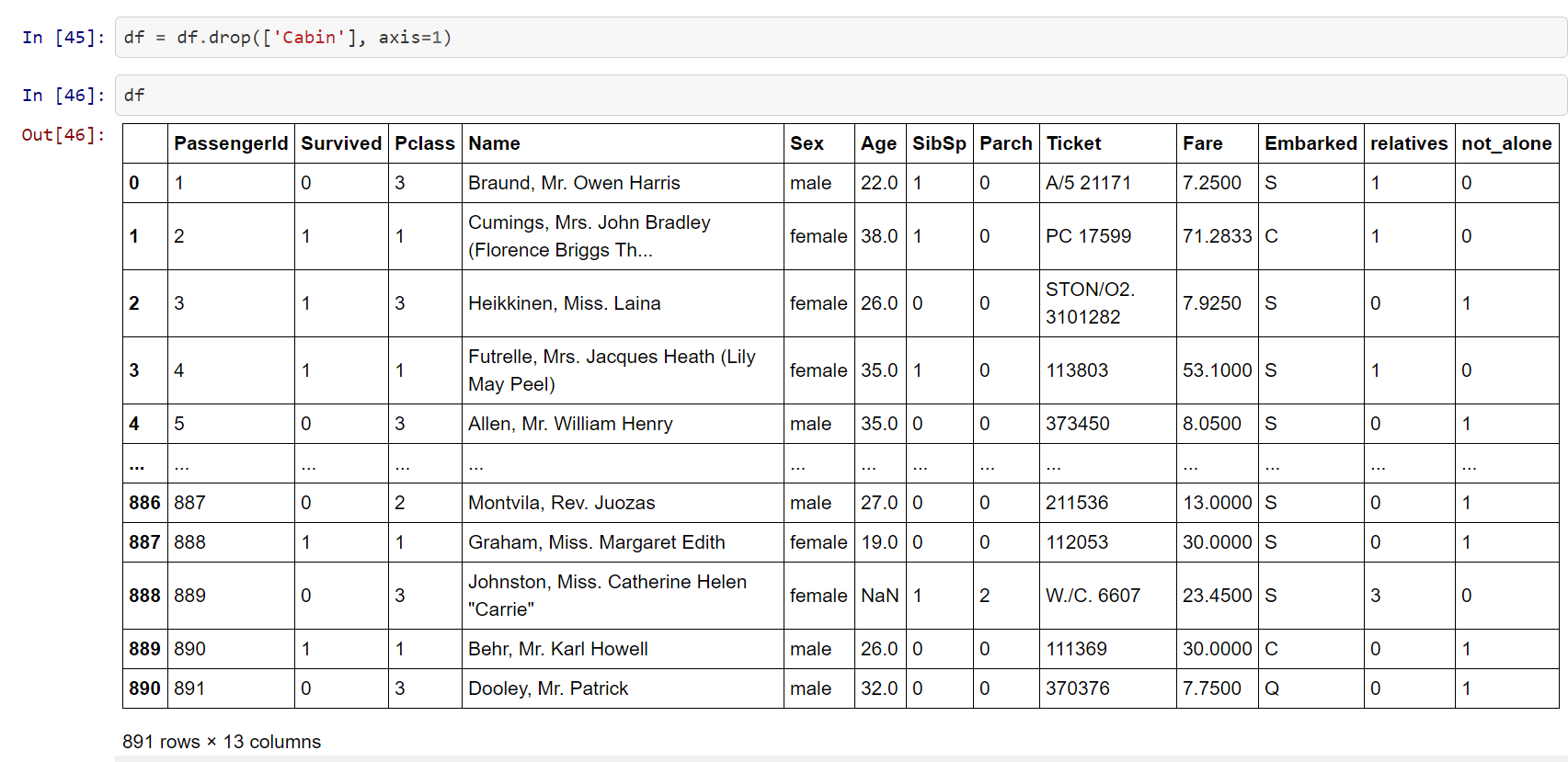
* Here we can see that you had a **high probabilty of survival with 1 to 3 realitves**, but a **lower one if you had less than 1 or more than 3 (**except for some cases with 6 relatives).

**Data Preprocessing**

* First, I will drop ‘PassengerId’ from the train set, because it does not contribute to a person’s survival probabilityso lets drop it .

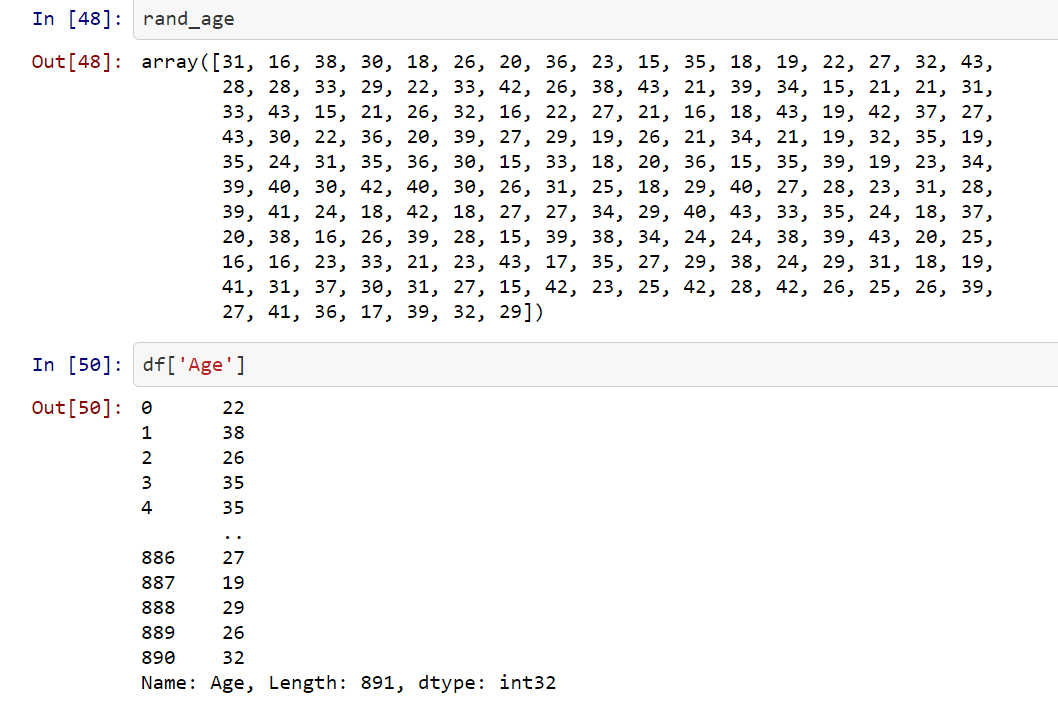


**Removing unwanted columns**



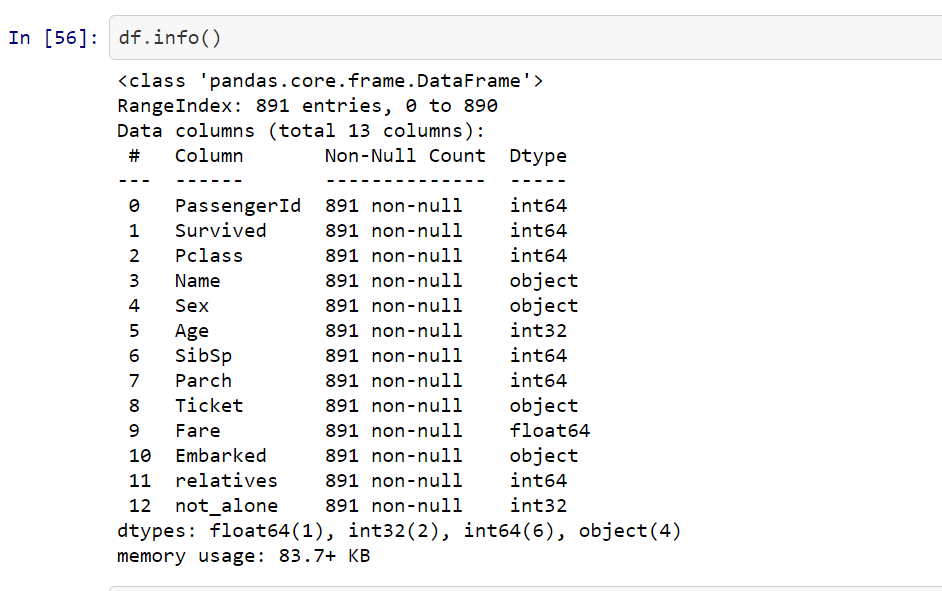
* **Imputing missing values/Data Cleaning**
* **Age**



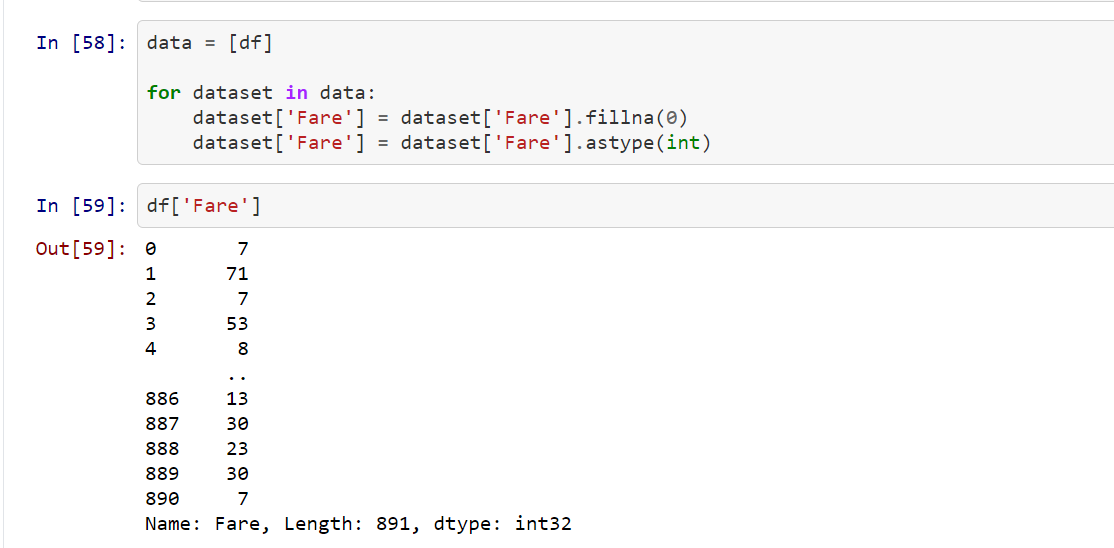


* **Embarked Variable**:
* Since the Embarked feature has only 2 missing values, we will just fill these with the most common one





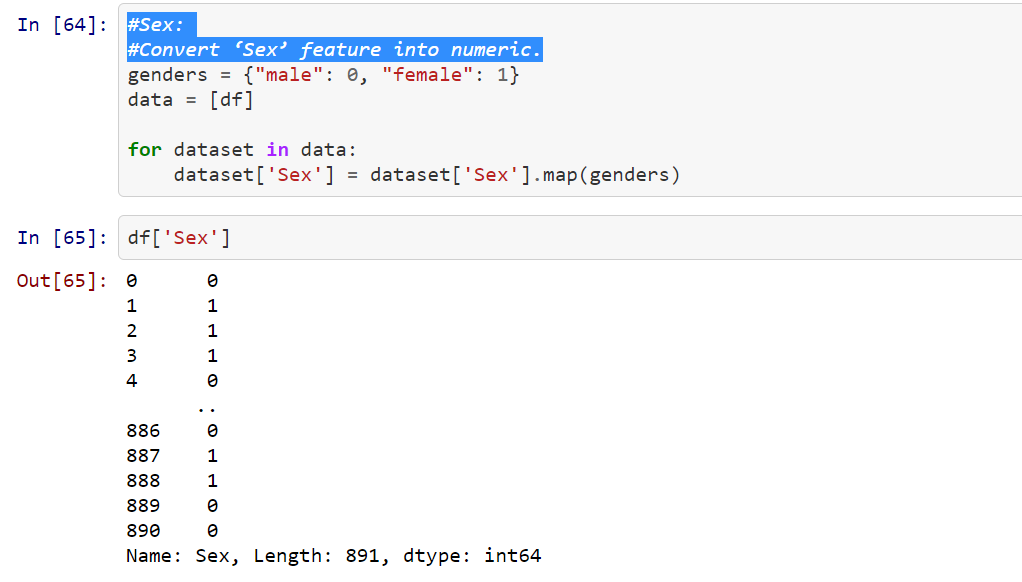
* Above you can see that ‘Fare’ is a float and we have to deal with 4 categorical features: Name, Sex, Ticket and Embarked

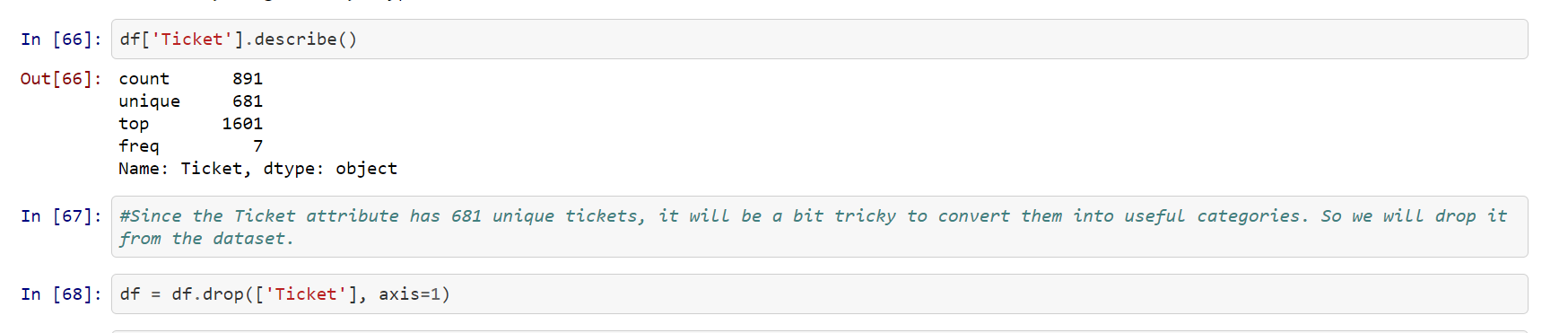


**Convert to Numeric data and then drop it**

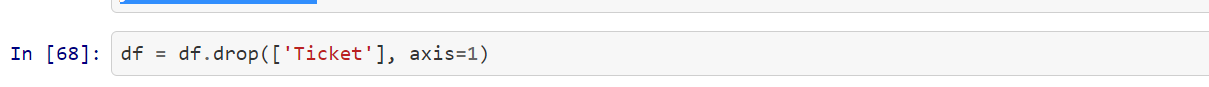


**Convert Variable ‘Sex’ feature into numeric.**

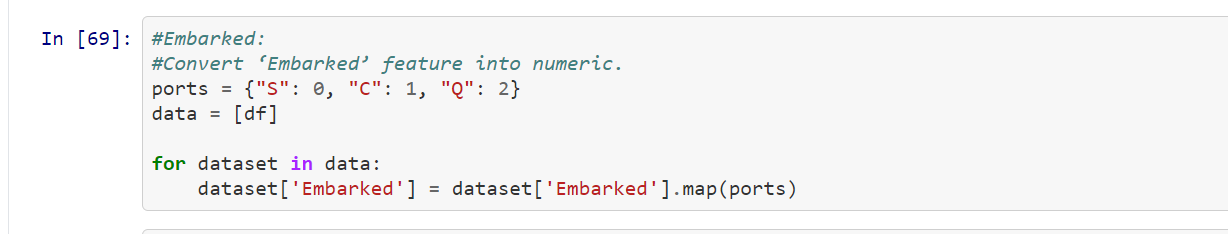




Since the Ticket attribute has 681 unique tickets, it will be a bit tricky to convert them into useful categories. So we will drop it from the dataset.



Variable Embarked:

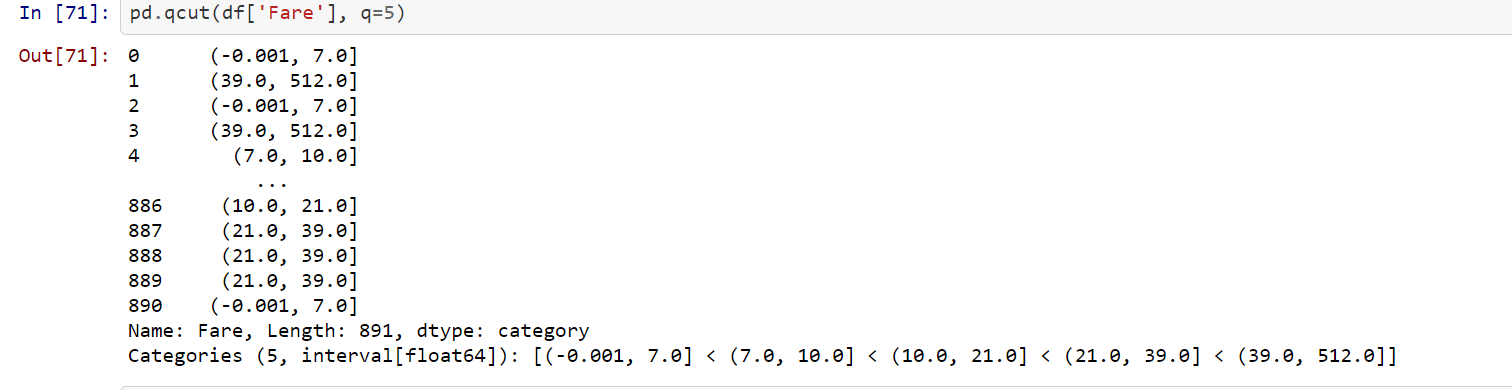


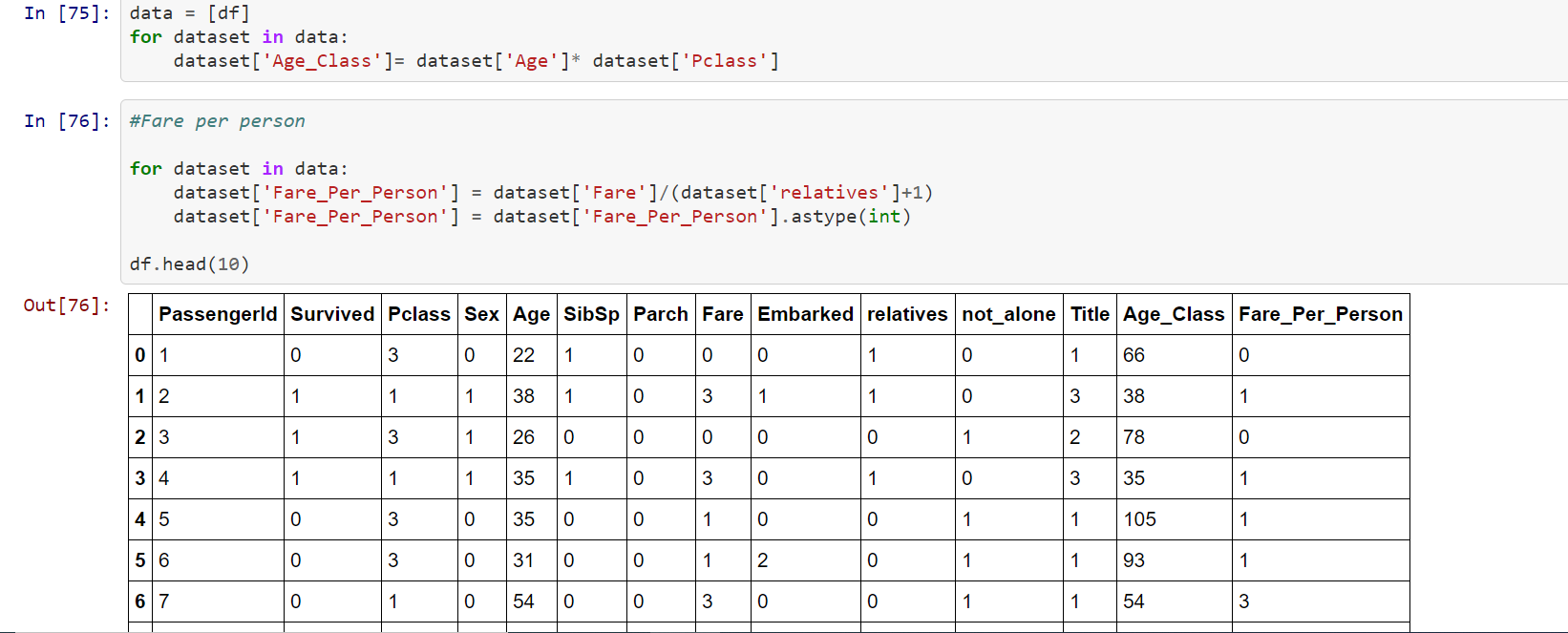
Creating new Features

I will add two new features to the dataset, that I compute out of other features.

# Age times Class

#Fare person





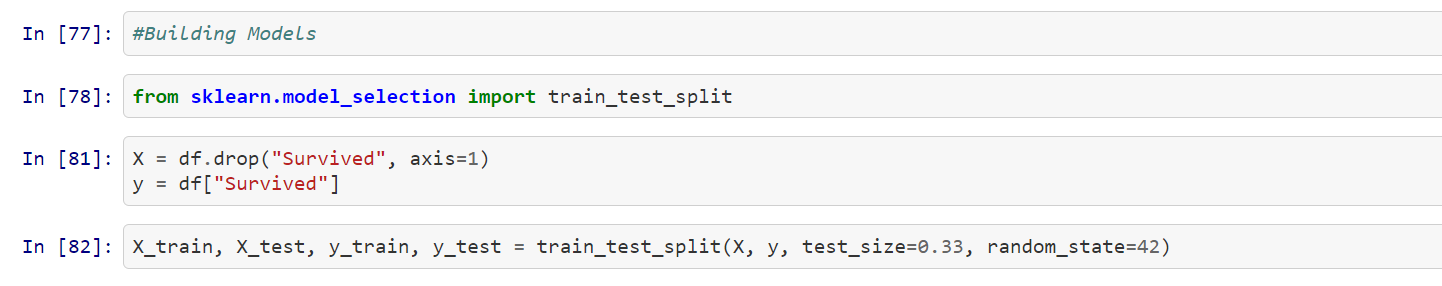
Observation:

As a part of data cleaning ,In our current data set we had few attributes/features which are not numeric. We converted those to numerical values.

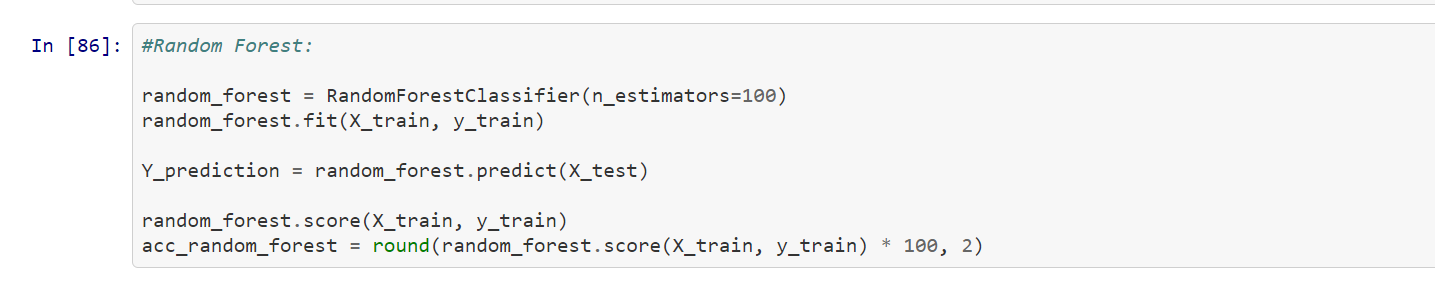
* Sex attribute is having values (female/male), we have mapped this to 0/1 respectively.
* Title a newly generated attribute is mapped to 1/2/3/4/5 values and missing value is filled with 0.
* Embarked attribute is mapped to 0/1/2 based on station onboarding.
* Four categories of fare mapped to four categorical values (0/1/2/3)
* Five age categories mapped to five categorical values (0/1/2/3/4)
* Dropped attributes which are not required for prediction, like PassengerId, Name, Ticket, Cabin, SibSp, Parch etc.

**Model Building**

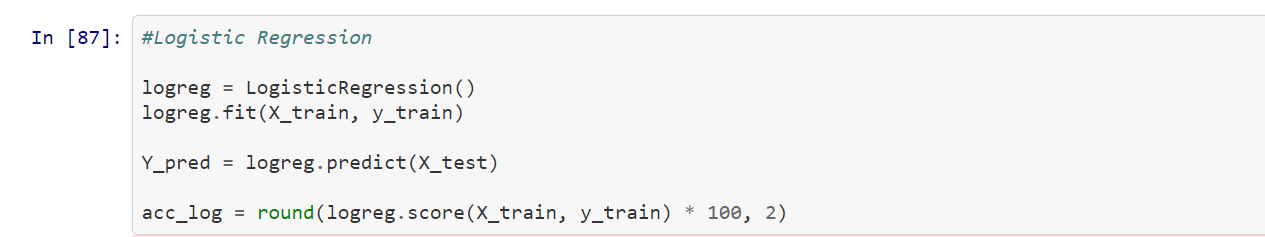
One can apply multiple classifiers on this data, in this exercise I will be using [*scikit learn*](http://scikit-learn.org/) inbuilt libraries. Below are the list of classifiers



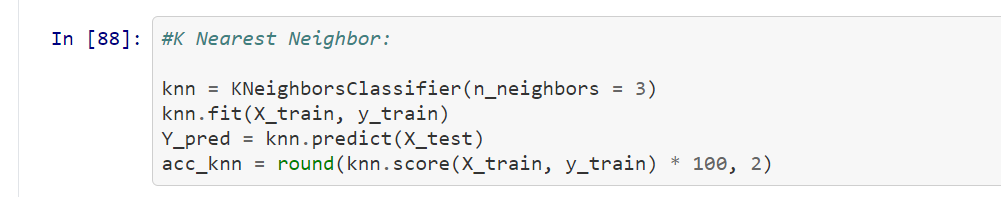
Random forest



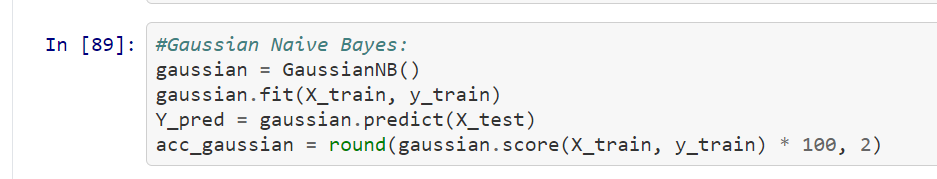
Logistic Regression



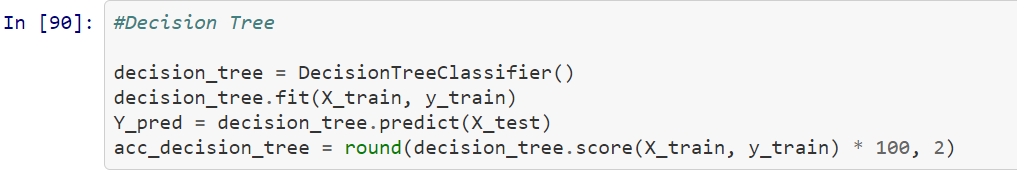
K-Nearest Neighbour



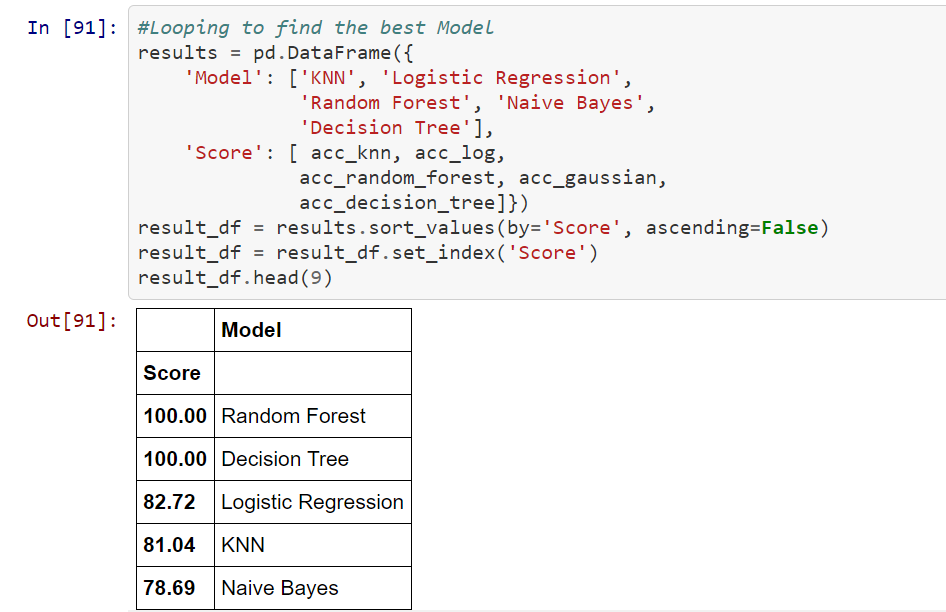
Gaussian Naïve-Bayes



Decision Tree



#**To Find the best Model**

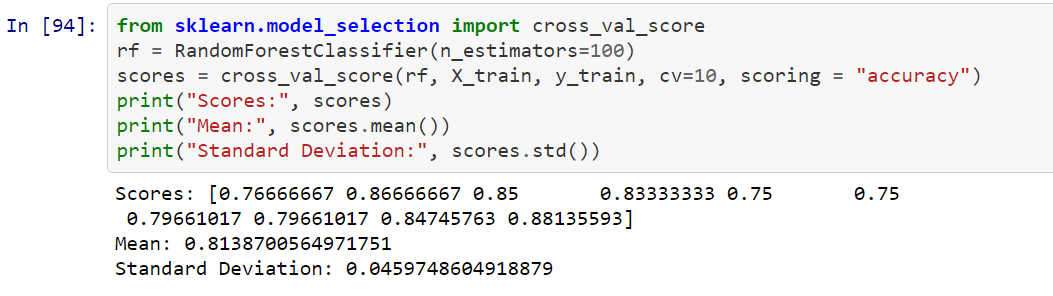


we can see, the Random Forest classifier goes on the first place.

But first, let us check, how random-forest performs, when we use cross validation

**K cross fold Validation**

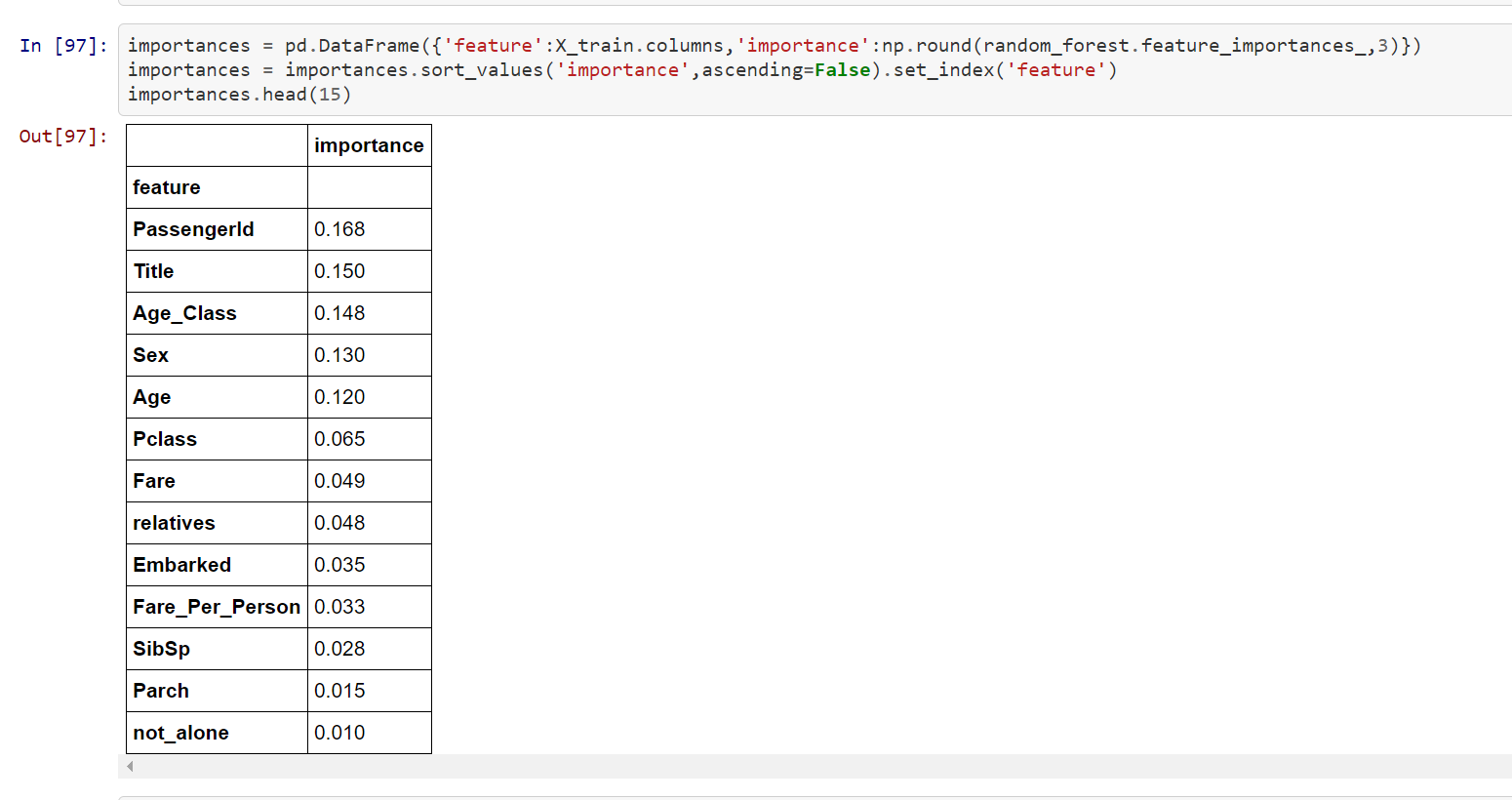
K-Fold Cross Validation randomly splits the training data into **K subsets called folds**. Let’s image we would split our data into 4 folds (K = 4). Our random forest model would be trained and evaluated 4 times, using a different fold for evaluation everytime, while it would be trained on the remaining 3 folds.

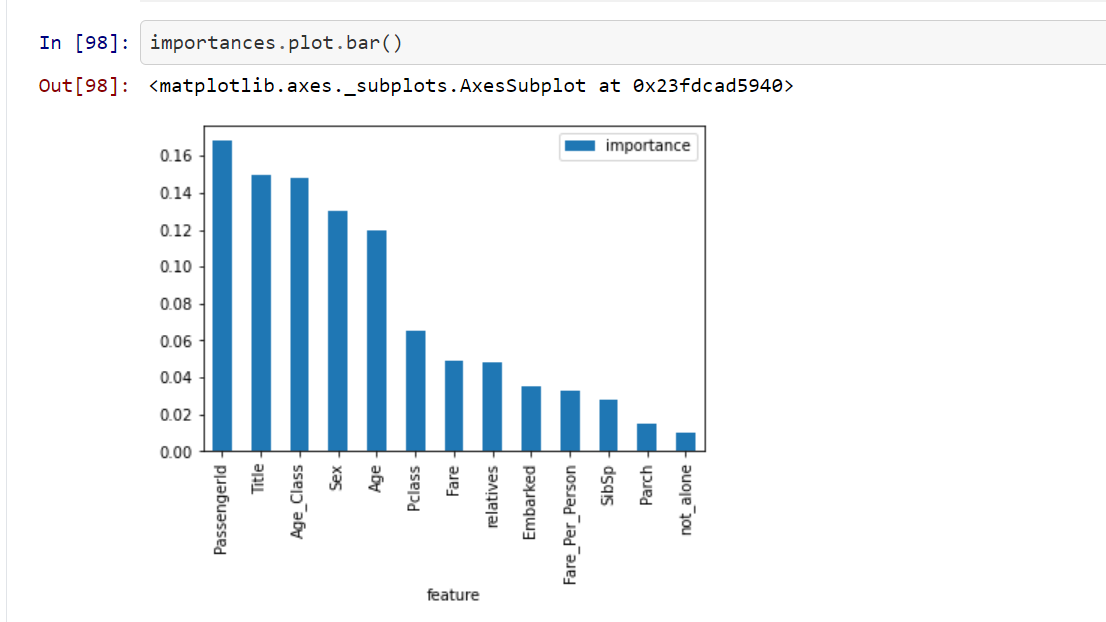
.

This looks much more realistic than before. Our model has a average accuracy of 81% with a standard deviation of 4 %

**FEATURE ENGINEERING**

Another great quality of random forest is that they make it very easy to measure the relative importance of each feature

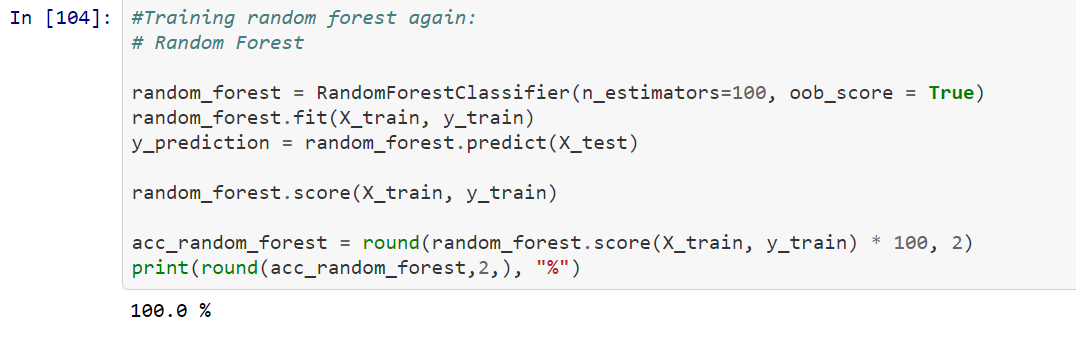




**CONCLUSION**

**not\_alone and Parch** doesn’t play a significant role in our random forest classifiers prediction process because of that I will drop them from the dataset and train the classifier again.

We could also remove more or less features, but this would need a more detailed investigation of the features effect on our model. But I think it’s just fine to remove only Alone and Parch

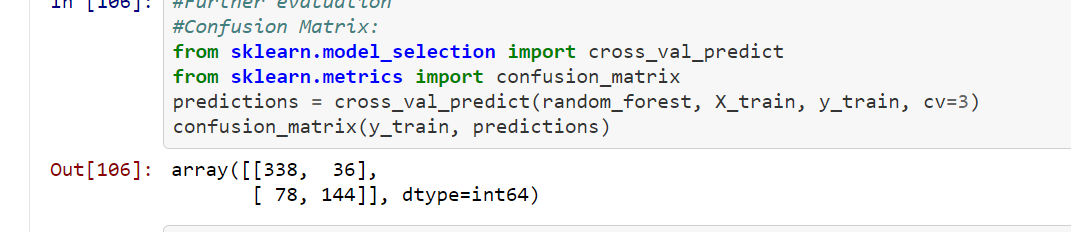


**Hyper parameter tuning**

Below you could see the code of the hyperparameter tuning for the parameters criterion, min\_samples\_leaf, min\_samples\_split and n\_estimators.

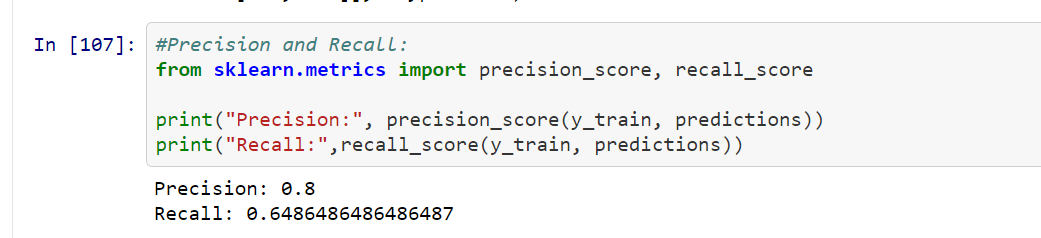


Now that we have a proper model, we can start evaluating it’s performace in a more accurate way. Previously we only used accuracy and the oob score, which is just another form of accuracy.

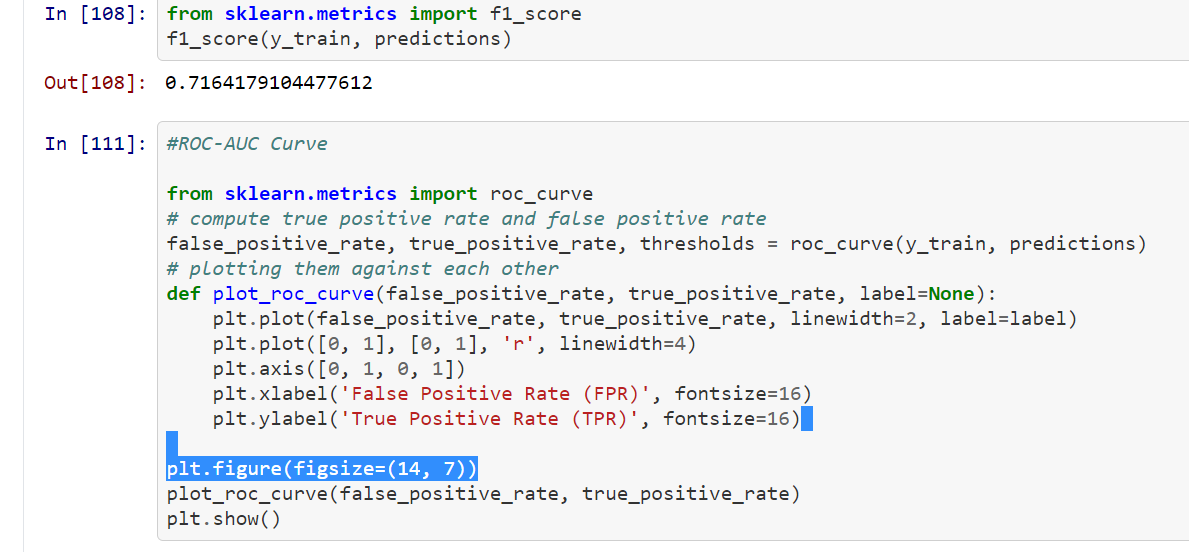
* **Confusion Matrix** A confusion matrix gives us a lot of information about how well the model does, but there’s a way to get even more, like computing the classifiers precision.
* 
* **F-Score**

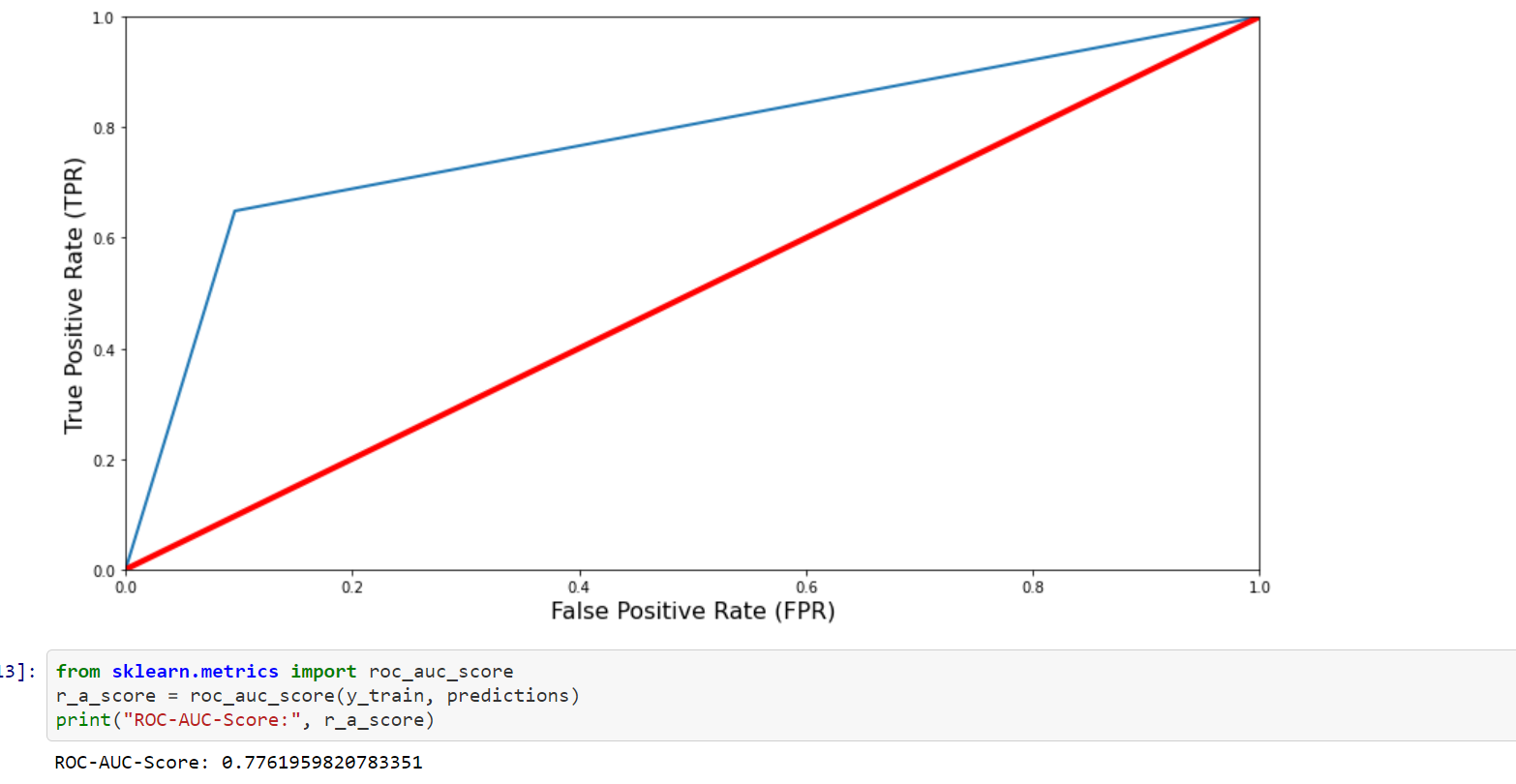
We can combine precision and recall into one score, which is called the F-score. The F-score is computed with the harmonic mean of precision and recall. Note that F-Score assigns much more weight to low values. As a result of that, the classifier will only get a high F-score, if both recall and precision are high

* **Precision & Recall**



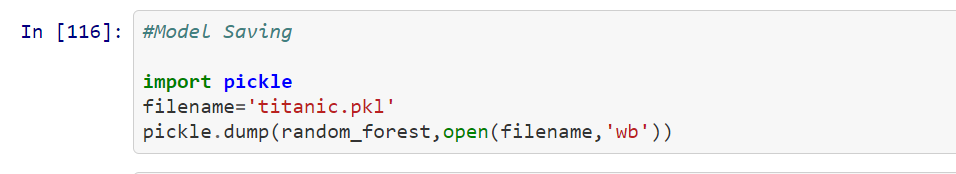
The ROC AUC Score is the corresponding score to the ROC AUC Curve. It is simply computed by measuring the area under the curve, which is called AUC.A classifiers that is 100% correct, would have a ROC AUC Score of 1 and a completely random classiffier would have a score of 0.5.

* ROC-AUC Curve



We have a ROC AUC Score 77%

Model Saving

* 

Conclusion of dataset

We started with the data exploration , checked the missing data and feature Engineering. During this process used seaborn and matplotlib to do the visualizations. During the data preprocessing part, computed missing values, converted features into numeric ones, grouped values into categories and created a few new features. Next,started training 5 different machine learning models, picked one of them (random forest) and applied cross validation on it. Then checked how random forest works, took a look at the importance it assigns to the different features and tuned it’s performace through optimizing it’s hyperparameter values. Lastly, we looked at it’s confusion matrix and computed the models precision, recall and f-score.

Lets take a look at the before and after dataset

