Capstone Project
Data Scientist Nanodegree Program

Yuhui Yin
June 22, 2020

Titanic: Machine Learning from Disaster

Definition

Project Overview

The sinking of the Titanic is 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, I did exploratory analysis of what sorts of people were likely to survive. And at the same time, I applied the tools of machine learning to predict which passengers survived the tragedy.
This project is inspired by the competition on Kaggle.

Problem Statement

The goal of this project is to find out what sorts of people were likely to survive and to predict which passengers survived the tragedy, the tasks involved are the following:

1. Data Preparation
2. Feature Selection
3. Exploratory Data Analysis and Visualization
4. Data Preprocessing
5. Parameters Selection
6. Models Selection
7. Prediction

Scores

The score for evaluating models is the 10-fold cross-validation score calculated by **cross_val_score** function from **sklearn.model_selection** package.

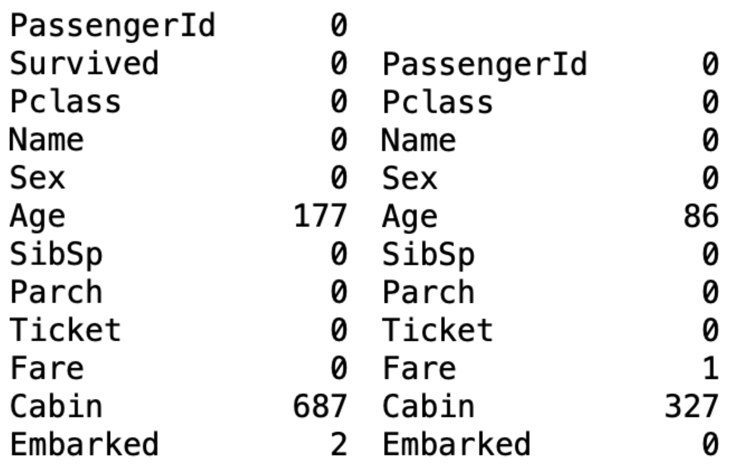
Exploratory Data Analysis

Data Preparation

There are two datasets: train.csv and test.csv. The first one contains 891 passengers' information as well as their survival condition. The second one contains 418 passengers' information without their survival condition. I imported them as df_train and df_test.
The passengers' information including:

VARIABLE	DEFINITION	KEY	TYPE
SURVIVAL	Survival	0 = No, 1 = Yes	Integrate number
PCLASS	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd	Integrate number
SEX	Sex		Categorical
AGE	Age in years		Continuous number
SIBSP	# of siblings / spouses aboard the Titanic		Integrate number
PARCH	# of parents / children aboard the Titanic		Integrate number
TICKET	Ticket number		Object
FARE	Passenger fare		Continuous number
CABIN	Cabin number		Object
EMBARKED	Port of embarkation	C = Cherbourg, Q = Queenstown, S = Southampton	Categorical

The numbers of missing values in both sets are as follows:



```
PassengerId      0
Survived         0  PassengerId      0
Pclass           0  Pclass           0
Name             0  Name             0
Sex              0  Sex              0
Age            177  Age             86
SibSp            0  SibSp            0
Parch            0  Parch            0
Ticket           0  Ticket           0
Fare             0  Fare             1
Cabin          687  Cabin          327
Embarked         2  Embarked         0
```

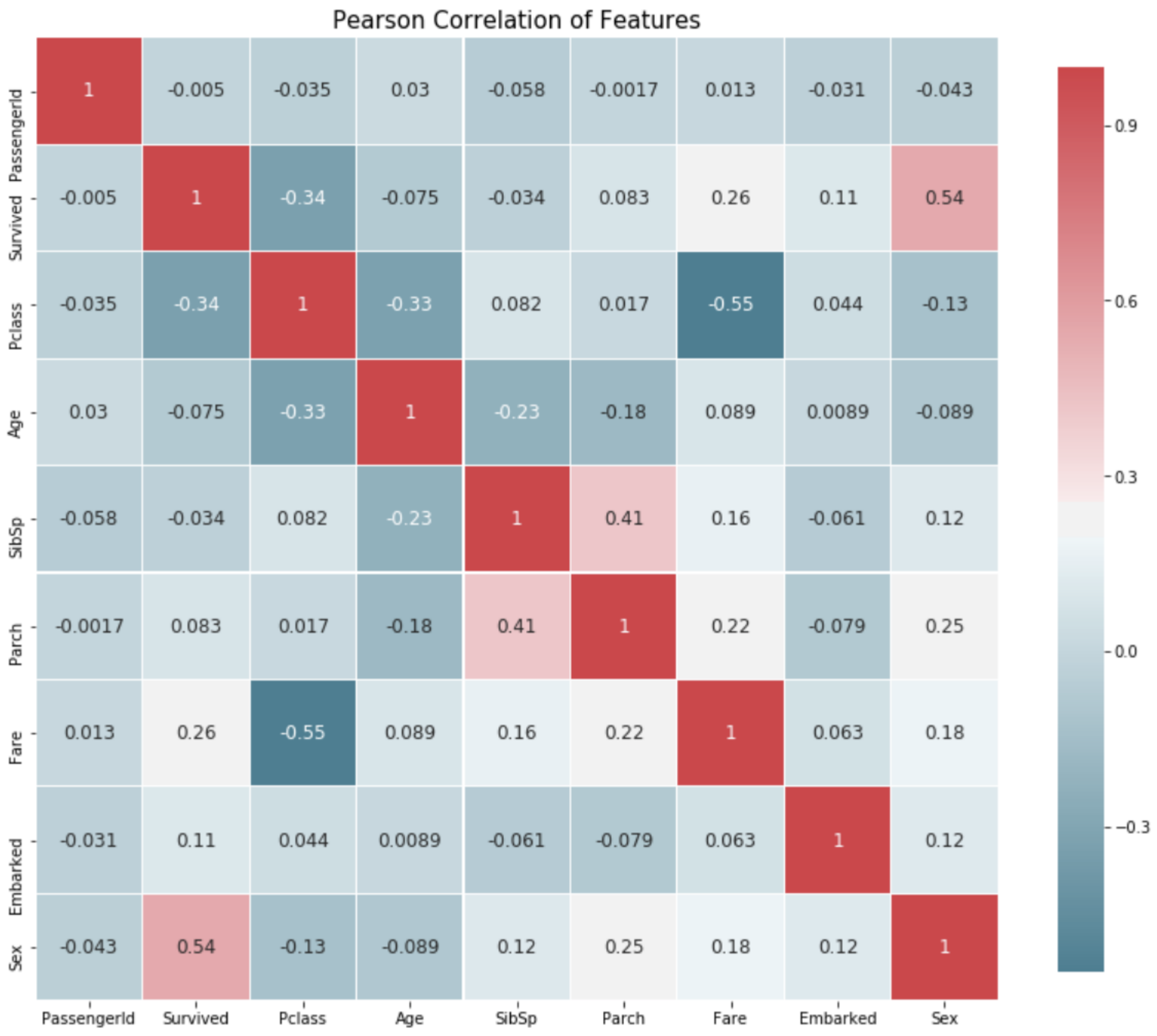
The numbers of missing values in **Cabin** are too large in both sets, so I drop the whole column;
The number of missing values in **df_train['Embarked']** is just 2, and in **df_test['Embarked']** is 0, so I drop the two rows in df_train;
For the missing value of **Age**, I fill them by the columns' mean values;
For the only missing value of **Fare** in **df_test**, I fill it by the column's mean value too.

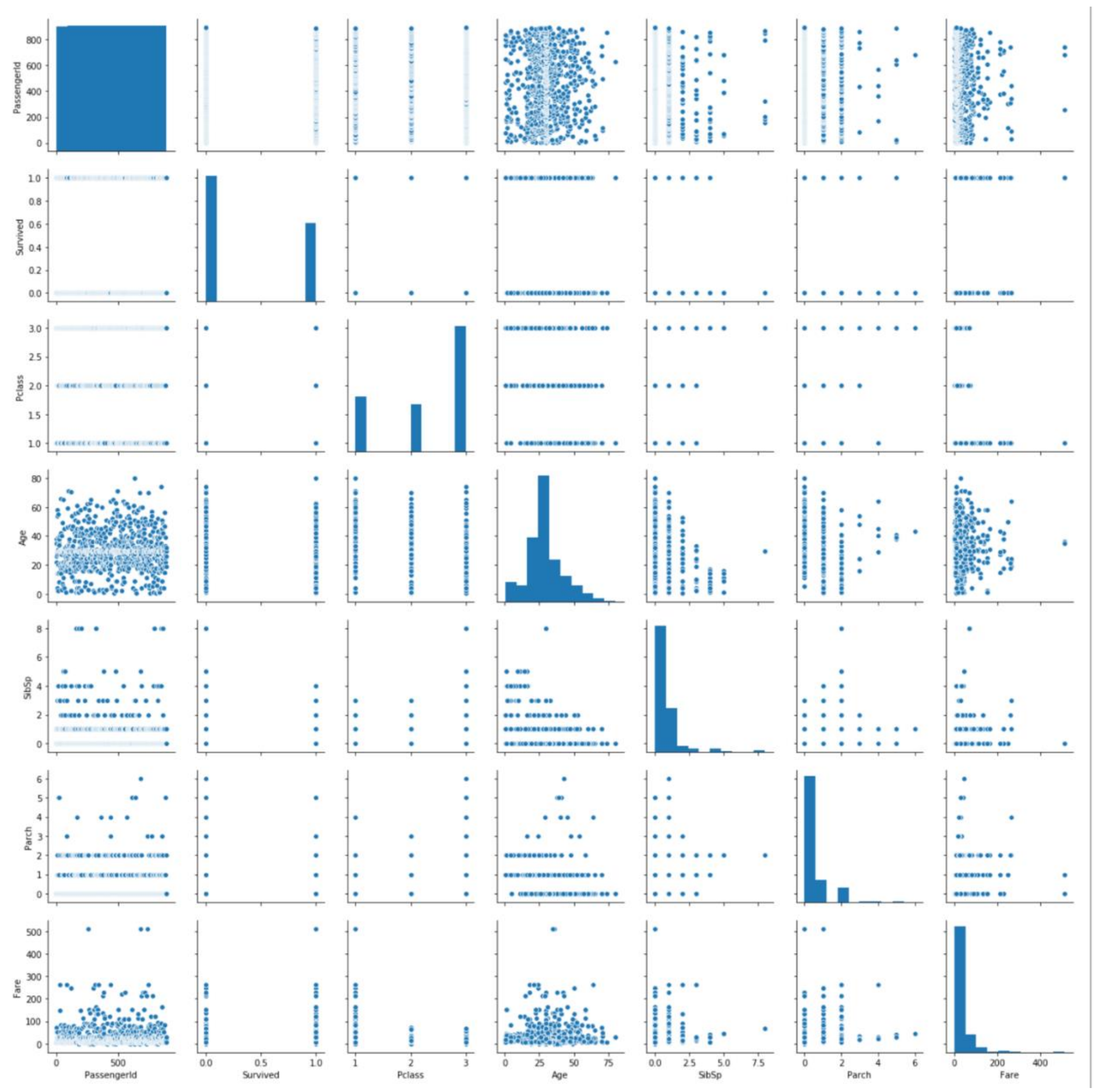
Feature Selection

There are 11 features remaining in the datasets, they are: **['PassengerId', 'Survival', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp', 'Parch', 'Ticket', 'Fare', 'Embarked'].**
For the **Ticket** column, after aggregating data, it shows that the prefix of **Ticket** numbers are highly correlated with the **Pclass**. At the same time, there are also large number of duplicated values in it, which is not make sense for the ticket numbers always come randomly and should not duplicated. So I drop the **Ticket** column.
For the **Name** column, the only useful information is the prefix for everyone, which is also revealed in the **Age** and **Sex** columns, So I drop the **Name** column.

Analysis and Visualization

By using the **pairplot** and **heatmap** functions from **seaborn** package, we can take a look at the overall relationships between every two features:





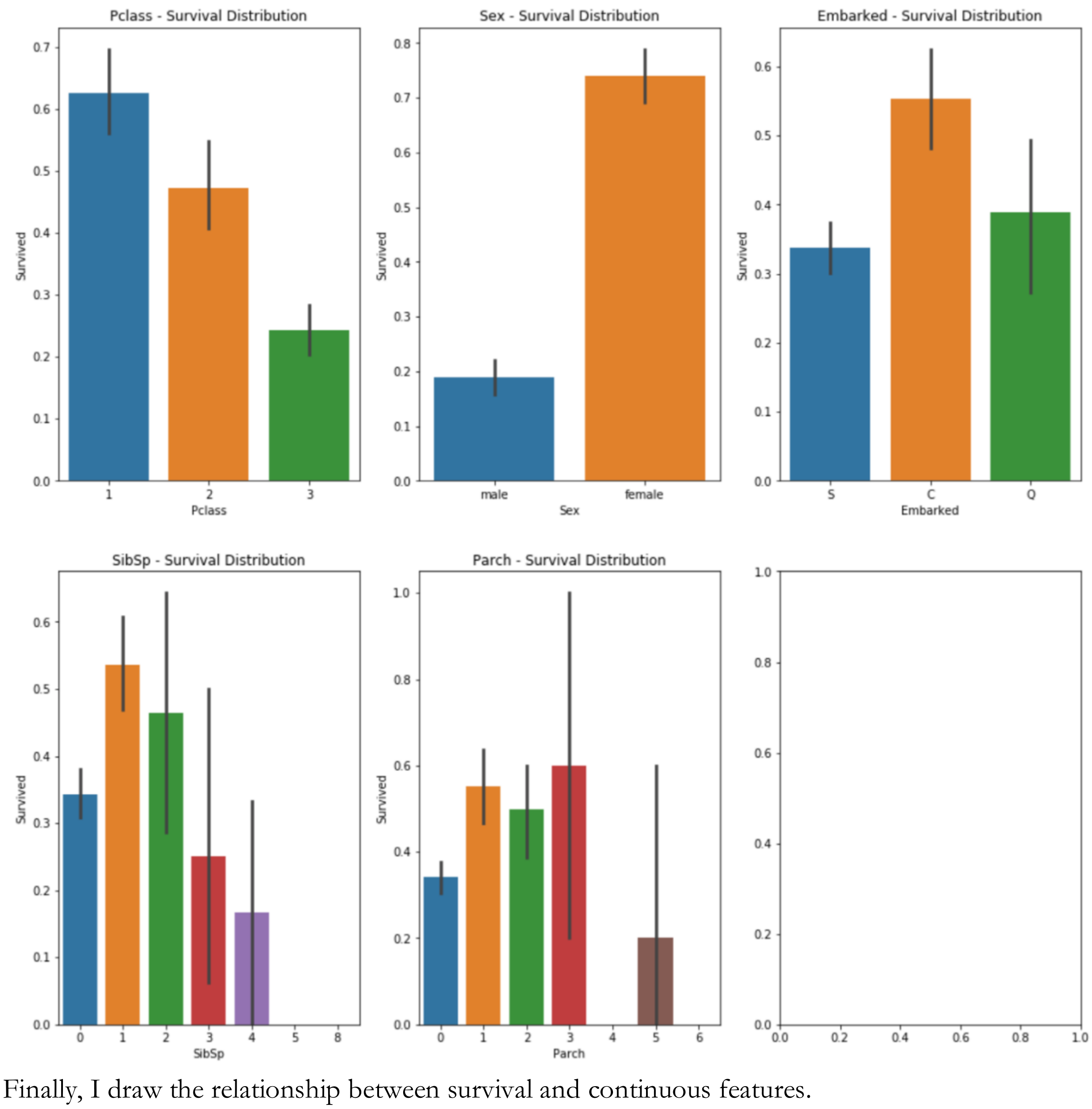
From above two figures we can get that there is no significant correlation between any two features.

Then I draw the relationships between survival rate and Categorical & Integrate features by **barplot** from **seaborn** package.

We can see from the following figures that:

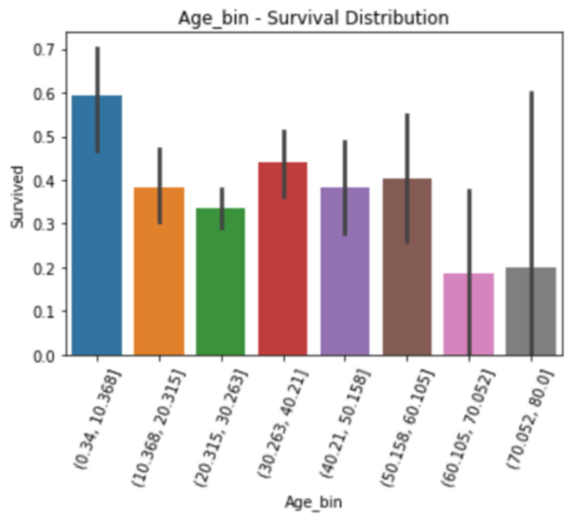
1. Among all ticket classes, the survival rate is highest for class 1 which is more than 60% and lowest for class 3 which is less than 30%;
2. Survival rate is much higher for female than for male, there are more than 70% female survive while less than 20% male survive;
3. Among all Port of embarkation, people embarked from Cherbourg have the highest survival rate which is larger than 50% while those embarked from Southampton have the lowest survival rate which is nearly 35%;

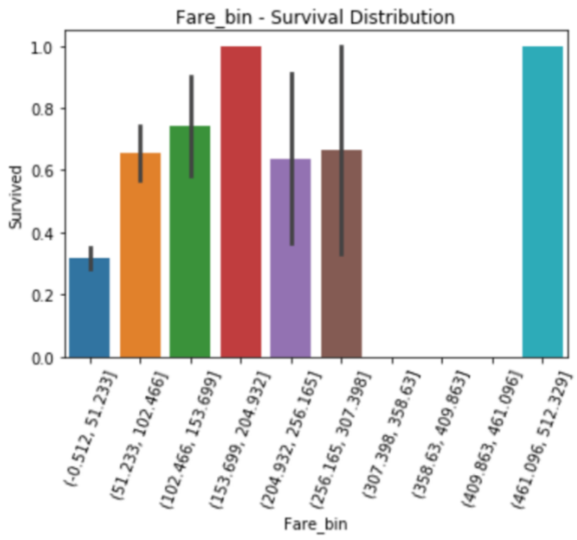
4. People who have less than 3 of siblings / spouses aboard have larger survival rate than those have over or equal to 3, those who have more than 4 siblings / spouses aboard have a survival rate of 0;
5. People who have 1 to 3 of parents / children aboard have larger survival rate than those who have other number of parents / children aboard.



Finally, I draw the relationship between survival and continuous features.
From the following figures we can get that:

1. People whose ages are less than 10.4 have the highest survival rate and those whose ages are more than 60.105 have lower survival rate;
2. For those who died, most of them paid fare less than 50. After calculation, we can know that people who paid less than 50 have a survival rate of 32% while those who paid more than 100 have a survival rate of 73.6%, people who pay more than 300 are all survived.





Methodology

Data Preprocessing

For easier prediction, I converted Categorical values into Integrate numbers and split the dataset using function **train_test_split** from **sklearn.model_selection package.**

Parameters Selection

By using **GridSearchCV** function from **sklearn.model_selection package,** I tried to find the best parameters for every model and stored the model with best parameters in a list called best_models.

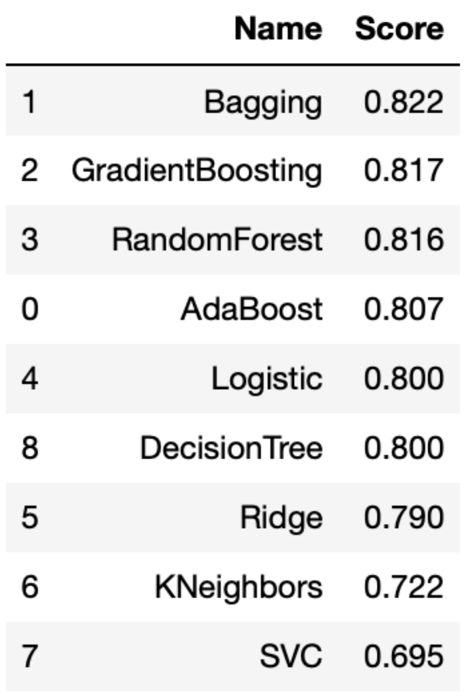
Models Selection

By comparing the 10-fold cross-validation scores calculated by **cross_val_score** function from **sklearn.model_selection** package. I chose the model with highest score.

Conclusion

Results

The scores for every model are as follows. We can find that BaggingClassifier got the highest score.



	Name	Score
1	Bagging	0.822
2	GradientBoosting	0.817
3	RandomForest	0.816
0	AdaBoost	0.807
4	Logistic	0.800
8	DecisionTree	0.800
5	Ridge	0.790
6	KNeighbors	0.722
7	SVC	0.695

Improvements

In this project, I chose the best parameters by using **GridSearchCV** and chose the best model by using 10-fold cross-validation score. To get a more accurate result, maybe I could increase the number of parameters for grid search as well as increase the number of folds for cross-validation. What's more, I only chose seven classifiers for evaluation, and there are still other classifiers I can try.