Titanic - Machine Learning from Disaster

In this challenge, I will 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)

Since the challenge is to predict if a passenger will survive, i.e. a true of false value, we are dealing with a binary classification model.

# Input Data

The data has been split into two groups:

* training set (train.csv)
* test set (test.csv)

We will load the data into pandas data frame for analysis

Graphical user interface, text, application

Description automatically generated

The columns are:

* PassengerId: A unique Id for every passenger, of type Integer. We will use this column as index moving forward.
* Survived: This column is only available in the training data set, and it is the value we should predict. The value can be 0 or 1.

A picture containing chart

Description automatically generated

* Pclass: In this column we have a value of 1,2 or 3. This value indicates the passenger passage class. 1st class, 2nd class, or 3rd class.

Graphical user interface, text, application

Description automatically generated

* Name: The name of the passenger. The format of the column is Last Name, Title. First Name
* Sex: The sex of the passenger. Possible values are male or female.
* Age: The age of the passenger in years as a float. The ages can vary between 0.17 and 80 years.

Text, letter

Description automatically generated

* SibSp: The number of Siblings/Spouses accompanying a passenger.
* Parch: The number of Parents/Children accompanying a passenger.
* Ticket: The number of the ticket
* Fare: The amount paid by the passenger to board the titanic.
* Cabin: The number of the Cabin the passenger resided in.
* Embarked: The port of Embarkation. Possible values are S(Southampton, England, the 1st stop), C(Cherbourg, France, the second stop), Q(Queenstown, Ireland, the last stop).

# Data Analysis and Preparation

Let’s reload the data frames using PassengerId as index.

Graphical user interface, text, application

Description automatically generated

## Pclass

Let’s find survival rate as a function of Pclass.

Graphical user interface, text, application

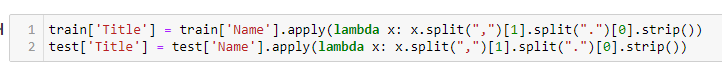
Description automatically generated

The survival rate varies significantly according to the Pclass. This means the Pclass column can have a significant importance in our prediction model.

We will use a MinMaxScaler to rescale the data in the range [0,1].

## Name

The first and last name don’t hold much information, however we can use the title of the passenger, as it’s an indicator of the social status of a person (Married, Single, Rich, …)



Possible values are:

Text

Description automatically generated

We can see that Dona is only available under test data, we will suppose that this title has a high probability of surviving.

We also have duplicate titles, “Mlle” is the same as “Miss” and “Mme” is the same as “Mrs”, so we will join these 2 categories.

Next step is to get the probability of survival per title and map these values to “Title” column.

Text

Description automatically generated

Plotting the survival rate vs. Title gives us the following:

Chart, bar chart

Description automatically generated

As we can see, there is a relation between the title and the survival rate of a passenger. (A Mrs or a Miss is more likely to survive than a Mr).

## Sex

Chart, waterfall chart

Description automatically generated

Females have a 74% survival rate, vs a 19% survival rate for males. This feature will be very important for the prediction model.

In the histogram we can see that number of males that did not survive is around 450, compared to less than 100 females.

The number of surviving females in around 250 compared to around 100 surviving males.

We will use a Label encoder, to transform male and female to 0 and 1.

Text

Description automatically generated

## SibSp and Parch

Graphical user interface, text, application, email

Description automatically generated

There appears to be a relation between the number of survival probability, and the Number of Sibilings, spouses, parents, and children. We will combine these two features into a single feature, called Family

Chart, bar chart

Description automatically generated

Since there is a relation between Family Members count, and survival rate, we will keep this column, and apply Normalization, using MinMaxScaler

A picture containing graphical user interface

Description automatically generated

## Ticket

The ticket number does not seem to give us any information, since appears to be no pattern. We will drop this column.

A picture containing text

Description automatically generated

## Fare

Chart

Description automatically generated

In the lower range of fares, the number of passengers who did not survive is greater than that of those who survived. However, once the fare is more than 50 pounds, number of survivors is higher than that of non survivors.

To do the processing, we will normalize the data using MinMaxScaler, and we will fill Nan values using most frequent value.

Text

Description automatically generated with medium confidence

## Cabin

The cabin column can give is 2 information.

Did the passenger have a cabin?

What class was the cabin?

Let’s extract the Cabin class from Cabin



We will use X to denote people without a Cabin.

Chart

Description automatically generated

We can see a link between the cabin, and the survival rate. However, since we have many types of Cabins, and the link between them isn’t clear, we will apply One Hot Encoding, using get\_dummies.

Table

Description automatically generated with low confidence

## Embarked

Chart

Description automatically generated with medium confidence

Grouping by Embarked port, we can see port C has the best survival chance, followed by port Q and S. We will replace the missing values with S and will apply get\_dummies on the data frame.



# Prediction

We are dealing with a binary classifying problem, and we will use a Linear Regression Classifier, from scikit learn, to do the prediction.

We will customize the model by changing the solver and the penalty.

The options we have for solvers are:

* liblinear
* sag
* saga

To find the best combination, we will split the train dataset, into train, and test using train\_test\_split function, and get the accuracy, precision and recall score for all models.

Text

Description automatically generated

It seems Logistic Regression with L1 penalty gives the best score overall.

Next step is to predict the results. We will retain the models using the whole train dataset and save the predictions to csv files.

Text, letter

Description automatically generated

After submitting the csv file, we get these scores:

Graphical user interface, text, application, email

Description automatically generated

The best scores are 0.78468 achieved using L1 penalty, with saga or liblinear solvers.