Sri Lanka Institute of Information Technology

Year 4 – Semester 1

**Machine Learning**

**For**

**Cyber Security**

Kaggle Submission: Titanic

|  |  |
| --- | --- |
| **Registration No** | **Name** |
| IT16170612 | M.N.Randeniya |
| IT16177406 | G.R.V.C.B.Samaranayake |

**Titanic: Machine Learning from Disaster**

**Predict survival on the Titanic**

1. Defining the problem statement
2. Collecting the data
3. Exploratory data analysis
4. Feature engineering
5. Modelling
6. Testing
7. **Defining the problem statement**

Complete the analysis of what sorts of people were likely to survive.

Apply the tools of machine learning to predict which passengers survived the Titanic tragedy.

Link to the problem: <https://www.kaggle.com/c/titanic>

1. **Collecting the data**

Training data set and testing data set are collected from Kaggle.

Link to data set: <https://www.kaggle.com/c/3136/download-all>

Load train, test dataset using Pandas.

A screenshot of a cell phone

Description automatically generated

1. **Exploratory data analysis**

First view the train data set. Printing first 5 rows of the train dataset and results should be like this.

A screenshot of a computer

Description automatically generated

In this data set you can view some value names “NaN”. We can consider those as missing data in the Data set.

**Data Dictionary**

* Survived: 0 = No, 1 = Yes
* pclass: Ticket class 1 = 1st, 2 = 2nd, 3 = 3rd
* sibsp: No of siblings / spouses aboard the Titanic
* parch: No of parents / children aboard the Titanic
* ticket: Ticket number
* cabin: Cabin number
* embarked: Port of Embarkation C = Cherbourg, Q = Queenstown, S = Southampton

**Total rows and columns**

Now view the test data set. As you can see, in this data set has 11 fields and in *Cabin* field has missing values.

A screenshot of a computer

Description automatically generated

To get the total rows and columns of test and train data set we can enter the following code lines.

A screenshot of a cell phone

Description automatically generated

As you can see, there are 891 rows and it means there were 891 passengers and 12 fields/features of the train data set. In test data set, you can see 418 rows/passengers and 11 fields. The only missing field is *Survived* field because we are going to predict the survived using the other features.

Also, we need to get the information of train and test data set. To do that try the following code.

A close up of text on a white background

Description automatically generatedA screenshot of text

Description automatically generated

We can see that *Age* value is missing for many rows.

Out of 891 rows, the *Age* value is present only in 714 rows.

Similarly, *Cabin* values are also missing in many rows. Only 204 out of 891 rows have *Cabin* values.

To get the no of missing rows in these 2 fields, we can perform a code line like this.

A close up of text on a white background

Description automatically generatedA screenshot of a cell phone

Description automatically generated

There are 177 rows with missing *Age*, 687 rows with missing *Cabin* and 2 rows with missing *Embarked* information in train data set.

There are 86 rows with missing *Age*, 327 rows with missing *Cabin* and 1 row with missing *Fare* information in test data set

**Import python lib for visualization**

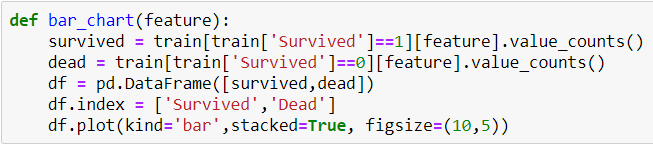
Now we are going to visualize the data set using bar charts to identify the relationship between features of survived and dead. To perform bar charts, we can use *seaborn* charts.

A screenshot of a cell phone

Description automatically generated

For the bar charts, we are taking the following features.

* Pclass
* Sex
* SibSp (# of siblings and spouse)
* Parch (# of parents and children)
* Embarked
* Cabin



In here, we have created a function of bar chart. We have gotten the *Survived* feature as our parameter and generate 2 bar charts. One is for survived and other one is for dead.

Now get the features *Sex, Pclass, SibSp, Parch* and *Embarked*.

A screenshot of a cell phone

Description automatically generated

The Chart confirms **Women** more likely survived than **Men.**

A screenshot of a cell phone

Description automatically generated

The Chart confirms **1st class** more likely survived than **other classes.**

The Chart confirms **3rd class** more likely dead than **other classes.**

**A screenshot of a cell phone

Description automatically generated**

The Chart confirms**a person aboarded with more than 2 siblings or spouse**more likely survived.  
The Chart confirms**a person aboarded without siblings or spouse**more likely dead.

**A screenshot of a cell phone

Description automatically generated**

The Chart confirms **a person aboarded with more than 2 parents or children** more likely survived.

The Chart confirms **a person aboarded alone** more likely dead.

**A screenshot of a cell phone

Description automatically generated**

The Chart confirms **a person aboarded from C** slightly more likely survived.

The Chart confirms **a person aboarded from Q** more likely dead.

The Chart confirms **a person aboarded from S** more likely dead.

1. **Feature engineering**

Feature engineering is the process of using domain knowledge of the data to create features (**feature vectors**) that make machine learning algorithms work.

In feature engineering, we are going to fill out the missing fields of train and test data set, changing the text values into numerical values because mostly, machine learning understands the numerical values, discard unwanted fields and create new fields that would be important from discarded fields.

Display the train data set again.

A screenshot of a computer

Description automatically generated

In here, we take the *Name* feature. We use this field to extract the title of the person with the combining train and test data set.

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

Now we are going to perform a title mapping using following titles. And provides numerical values to those titles.

* Mr: 0
* Miss: 1
* Mrs: 2
* Others: 3

A screenshot of a cell phone

Description automatically generated

Then the results would be like this.

A screenshot of a computer

Description automatically generated

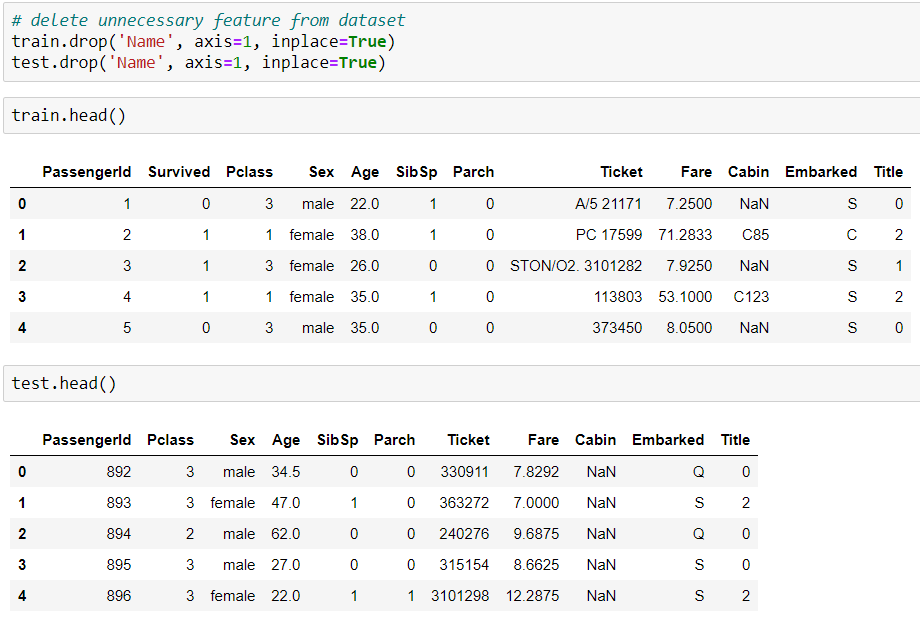
Then the bar chart would be look like this.

A screenshot of a cell phone

Description automatically generated

The bar chart confirms **Miss** & **Mrs** more likely survived than **Mr.**

Now we can drop the *Name* feature in both train and test data set because it is no longer needed.



Now we are going to perform a sex mapping. And provides numerical values to those titles.

* Male: 0
* Female: 1

A screenshot of a social media post

Description automatically generated

The bar chart confirms **Female** more likely survived than **Male.**

Now we are going to perform an *Age* mapping. But the problem is some *Age* is missing. To fill out missing ages, we can get the *Title's* median age for missing *Age.*

A picture containing screenshot

Description automatically generated

Now display the chart to identify the relationship between survived and dead based on age.

A close up of a map

Description automatically generated

To get a closer look, you can use “plt.xlim(age, age)”. This filters the survived and dead based on a range of age.

A screenshot of a social media post

Description automatically generatedA screenshot of a social media post

Description automatically generatedA screenshot of a social media post

Description automatically generatedA screenshot of a social media post

Description automatically generatedA screenshot of a cell phone

Description automatically generated

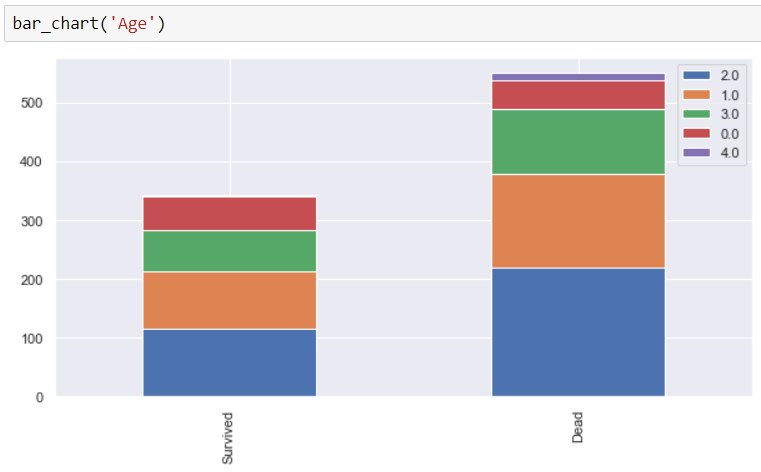
Now we must convert the numerical ages into categorical values.

* child: 0
* young: 1
* adult: 2
* mid-age: 3
* senior: 4

A screenshot of a cell phone

Description automatically generated

Now the bar chart will look like this.



Now we are going to perform an *Embarked* mapping just like *Age*. But the problem is some *Embarked* is missing. To fill out missing Embarked, we can follow this process*.*

We can categorize *Embarked* according to the ticket class.

A screenshot of a cell phone

Description automatically generated

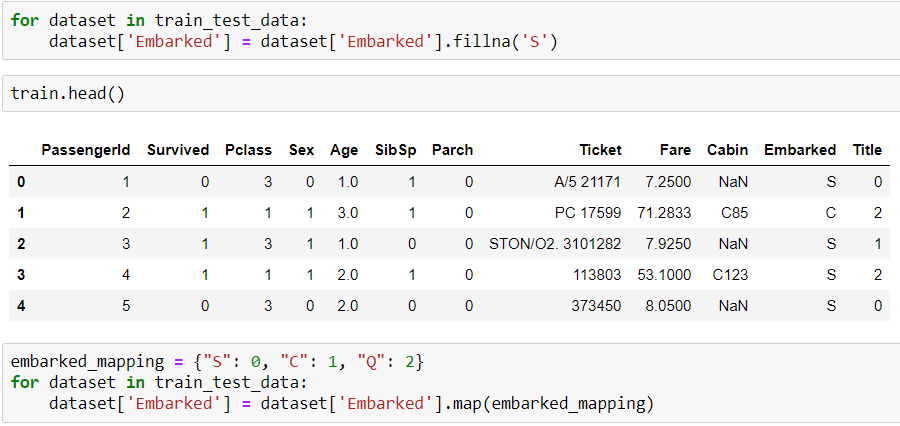
With the bar chart result we can identify,

* More than 50% of 1st class are from S embark.
* More than 50% of 2nd class are from S embark.
* More than 50% of 3rd class are from S embark.

A screenshot of a cell phone

Description automatically generated

Because of this we can fill out missing embark with S embark. With that we need to change the numerical values into categorical values.



Now we need to fill out missing *Fare* with median fare for each *Pclass*.

A screenshot of a cell phone

Description automatically generated

Then view the chart to analyze the fare of survived and dead.

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

The chart confirms that **people who got the cheap tickets died more than the people who got expensive tickets**.

Now we must convert the numerical ticket prices into categorical values.

A screenshot of a cell phone

Description automatically generated

Now we are going to perform a *Cabin* mapping just like above fields.

*A screenshot of a cell phone

Description automatically generated*

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

Now we must convert the *Cabin* values into categorical values.

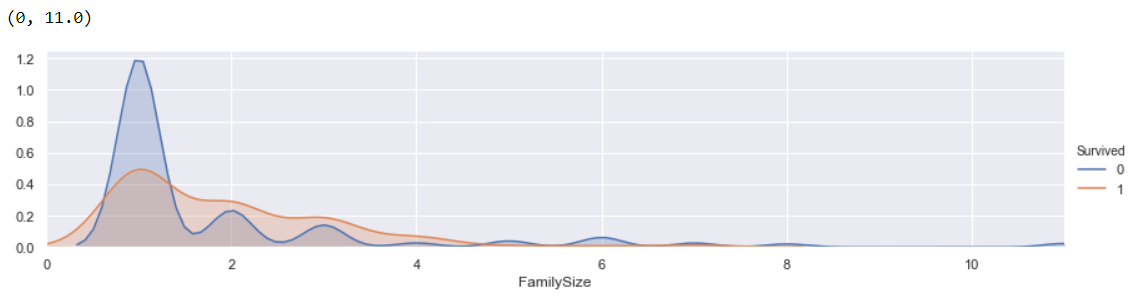
A screenshot of a cell phone

Description automatically generated

Now we are going to perform a *family* mapping just like above fields. To perform this, we must combine *SibSP,* *Parch* and add 1 to get the accurate family size.

A screenshot of a cell phone

Description automatically generated



The chart confirms that **family people have a high chance to survive more than single people**.

Then we must convert the numerical ticket prices into categorical values

A screenshot of a cell phone

Description automatically generated

With this mapping we can drop some features from the data set.

A screenshot of a cell phone

Description automatically generated

Finally, we have numerical valued feature vectors to perform the predictions of survival on the Titanic.

A screenshot of a cell phone

Description automatically generated

1. **Modelling**

Now we must import classifier modules for machine learning algorithms. We have chosen the most popular and high accuracy algorithms.

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

**Cross Validation (K-fold)**

A picture containing wall

Description automatically generated

**kNN**

A screenshot of a cell phone

Description automatically generated

**Decision Tree**

A screenshot of a cell phone

Description automatically generated

**Random Forest**

A screenshot of a cell phone

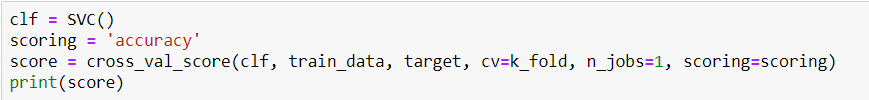
Description automatically generated

**Naive Bayes**

A screenshot of a cell phone

Description automatically generated

**SVM**

****

A screenshot of a cell phone

Description automatically generated

We must choose the SVM because it gave us the highest average accuracy of prediction.

1. **Testing**

In this process we choose SVM as our classifier and use the train data to train data and dropping the *PassengerId* because it’s unnecessary to testing. With our testing data we make predictions.

A screenshot of a cell phone

Description automatically generated

Now we must create a “submission.csv” with the column names *PassengerId* and *Survived* to get the prediction results and submit it to Kaggle.

A screenshot of a cell phone

Description automatically generated

Now we can view the predictions from the final output.

A screenshot of a cell phone

Description automatically generated

Now you must upload the “submission.csv” to Kaggle to check the accuracy score.

A screenshot of a cell phone

Description automatically generated

To check the accuracy score, go to “Leaderboard”.

A screenshot of a cell phone

Description automatically generated