Machine Learning

CED17

Submitted By:- Vrishank Gupta
2016UCO1677
COE 3

Division Of Computer Engineering
Netaji Subhas University of Technology, Dwarka

Titanic: Machine Learning from Disaster

HOSTED ON :- KAGGLE

"**Until the moment she actually sinks, the Titanic is unsinkable.**"
— Julia Hughes.

OVERVIEW

The sinking of the RMS Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers and crew. This sensational tragedy shocked the international community and led to better safety regulations for ships.

One of the reasons that the shipwreck led to such loss of life was that there were not enough lifeboats for the passengers and crew. Although there was some element of luck involved in surviving the sinking, some groups of people were more likely to survive than others, such as women, children, and the upper-class.

In this challenge, we ask you to complete the analysis of what sorts of people were likely to survive. In particular, we ask you to apply the tools of machine learning to predict which passengers survived the tragedy.

DATA

The data has been split into two groups:

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

The training set should be used to build your machine learning models. For the training set, we provide the outcome (also known as the “ground truth”) for each passenger. Your model will be based on “features” like passengers’ gender and class. You can also use feature engineering to create new features.

The test set should be used to see how well your model performs on unseen data. For the test set, we do not provide the ground truth for each passenger. It is your job to predict these outcomes. For each passenger in the test set, use the model you trained to predict whether or not they survived the sinking of the Titanic.

We also include gender_submission.csv, a set of predictions that assume all and only female passengers survive, as an example of what a submission file should look like.

Data Dictionary

Variable	Definition	Key
survival	Survival	0 = No, 1 = Yes
pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd
sex	Sex	
Age	Age in years	
sibsp	# of siblings / spouses aboard the Titanic	
parch	# of parents / children aboard the Titanic	
ticket	Ticket number	
fare	Passenger fare	
cabin	Cabin number	
embarked	Port of Embarkation	C = Cherbourg, Q = Queenstown, S = Southampton

Variable Notes

pclass: A proxy for socio-economic status (SES)
1st = Upper
2nd = Middle
3rd = Lower

age: Age is fractional if less than 1. If the age is estimated, is it in the form of xx.5

sibsp: The dataset defines family relations in this way...
Sibling = brother, sister, stepbrother, stepsister
Spouse = husband, wife (mistresses and fiancés were ignored)

A DEEPER DIVE

The file Test.csv has following structure:-

Columns

- # PassengerId 1
- # Pclass 1
- # Name the name of the passenger
- # Sex
- # Age
- # SibSp of siblings / spouses aboard the Titanic
- # Parch of parents / children aboard the Titanic
- # Ticket Ticket number
- # Fare Passenger fare
- # Cabin Cabin number
- # Embarked Port of Embarkation

```
RangeIndex: 418 entries, 0 to 417
Data columns (total 11 columns):
PassengerId    418 non-null int64
Pclass         418 non-null int64
Name           418 non-null object
Sex            418 non-null object
Age            332 non-null float64
SibSp          418 non-null int64
Parch          418 non-null int64
Ticket         418 non-null object
Fare           417 non-null float64
Cabin          91 non-null object
Embarked       418 non-null object
dtypes: float64(2), int64(4), object(5)
```

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

The file train.csv has following structure:-

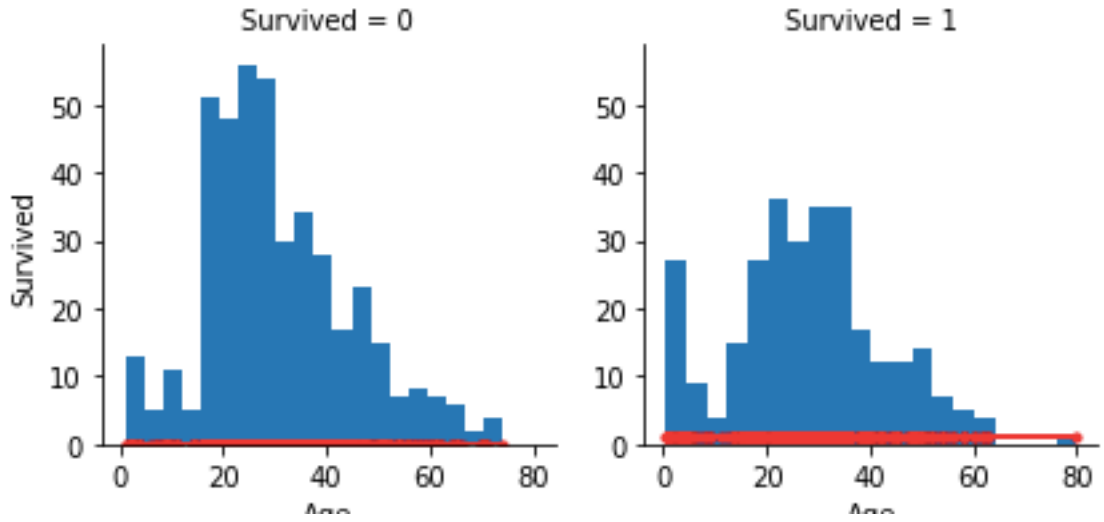
Columns

- # PassengerId type should be integers
- # Survived Survived or Not
- # Pclass Class of Travel
- A Name Name of Passenger
- A Sex Gender
- # Age Age of Passengers
- A SibSp Number of Sibling/Spouse aboard
- # Parch Number of Parent/Child aboard
- A Ticket
- # Fare
- A Cabin
- A Embarked The port in which a passenger has embarked. C - Cherbourg, S - Southampton, Q = Queenstown

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
PassengerId    891 non-null int64
Survived       891 non-null int64
Pclass         891 non-null int64
Name           891 non-null object
Sex            891 non-null object
Age            714 non-null float64
SibSp          891 non-null int64
Parch          891 non-null int64
Ticket         891 non-null object
Fare           891 non-null float64
Cabin          204 non-null object
Embarked       889 non-null object
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

SOME INSIGHTS



From the graph above, older people died and younger people survived more. But the most dense part of survived and died is basically on the same range between 20 to 50. It makes sense because the older people they have less capabilities to save themselves in disasters like these and tend to sacrifice themselves because the next generation is the future.

It is a common tradition that "Ladies first!", and the above graph proves it as more females survived as compared to males.

Moreover, as revealed from the structure of the train and test data files, some of the fields such as “Sex”, which takes values such as “Male” and “Female”, and “Embarked”, which takes values such as “S”, “C”, etc. are present, however, to apply any concerned machine learning model, we must have numeric data. Therefore , these fields must be encoded to their numeric values counterparts without the loss of any information.

Some fields like “Age” have null values in some of its instances, it must be filled with some appropriate value so as to get considerably accurate predictions.

These null values can be taken care of by filling these with the mean of rest of the available values to get approximately correct predictions.

Fields such as “Cabin”, which denotes the cabin number of the passenger and “ticket”, which describes the type of ticket the passenger has, might not contribute anything towards prediction that the concerned person survived or not, the field “Name” might also be redundant. Other fields such as “Age”, “Sex” etc. are more useful than above said fields and might be a better influencer on survival prediction, therefore the redundant fields can be dropped out from the data without the loss of any information.

The age of the individual might not be an accurate criterion for prediction, however, the age group such as age 20-40 years might give better results, therefore ages are grouped together in fewer age groups.

Therefore, it is intuitive that data must be preprocessed and cleaned before applying machine learning model to get the predictions. The following method handles all the above said concerns to produce cleaner data:-

```
def data_process(data):
    data["Fare"] = data["Fare"].fillna(data["Fare"].dropna().median())
    data["Age"] = data["Age"].fillna(data["Age"].dropna().median())

    data = data.drop(['Ticket'], axis=1)
    data = data.drop(['Cabin'], axis=1)
    freq_port = train.Embarked.dropna().mode()[0]

    data['Embarked'] = data['Embarked'].fillna(freq_port)

    data['Embarked'] = data['Embarked'].map( {'S': 0, 'C': 1, 'Q': 2} ).astype(int)
    data = data.drop(['Name'], axis=1)

    data.loc[data["Sex"] == "male", "Sex"] = 0
    data.loc[data["Sex"] == "female", "Sex"] = 1

    data.loc[ data['Age'] <= 16, 'Age'] = int(0)
    data.loc[(data['Age'] > 16) & (data['Age'] <= 32), 'Age'] = 1
    data.loc[(data['Age'] > 32) & (data['Age'] <= 48), 'Age'] = 2
    data.loc[(data['Age'] > 48) & (data['Age'] <= 64), 'Age'] = 3
    data.loc[data['Age'] > 64, 'Age']

    return data
```

Upon preprocessing the data, the situation looked like this:-

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	0	3	0	1.0	1	0	7.2500	0
1	2	1	1	1	2.0	1	0	71.2833	1
2	3	1	3	1	1.0	0	0	7.9250	0
3	4	1	1	1	2.0	1	0	53.1000	0
4	5	0	3	0	2.0	0	0	8.0500	0

MACHINE LEARNING MODEL

Here, for this problem, I have planned to use decision trees classifier.

A **decision tree** is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.
A decision tree is a flowchart-like structure in which each internal node represents a “test” on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represent classification rules.

Tree based learning algorithms are considered to be one of the best and mostly used supervised learning methods. Tree based methods empower predictive models with high accuracy, stability and ease of interpretation. Unlike linear models, they map non-linear relationships quite well. They are adaptable at solving any kind of problem at hand (classification or regression). Decision Tree algorithms are referred to as **CART (Classification and Regression Trees)**.
being popularly used in all kinds of data science problems.

"The possible solutions to a given problem emerge as the leaves of a tree, each node representing a point of deliberation and decision."

- Niklaus Wirth (1934—), Programming language designer

Methods like decision trees, random forest, gradient boosting are being popularly used in all kinds of data science problems.

Algorithm used in decision trees:

- ID3
- Gini Index
- Chi-Square
- Reduction in Variance

ID3

The core algorithm for building decision trees is called **ID3.** Developed by J. R. Quinlan, this algorithm employs a top-down, greedy search through the space of possible branches with no backtracking. ID3 uses *Entropy* and *Information Gain* to construct a decision tree. In this problem, I've used ID3 algorithm to construct decision tree.

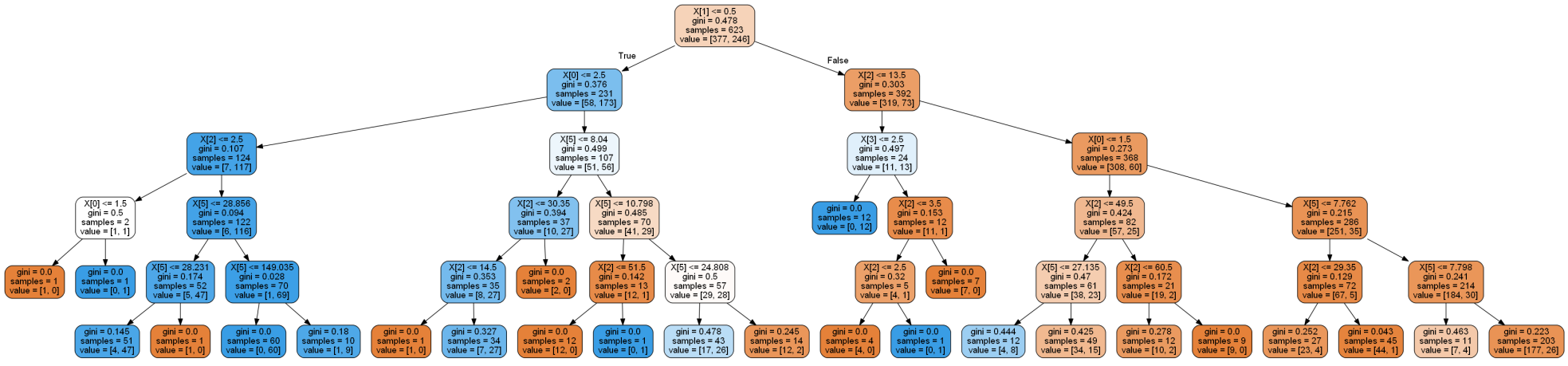
WHY DECISION TREE?

Decision trees assist managers in evaluating upcoming choices. The tree creates a visual representation of all possible outcomes, rewards and follow-up decisions in one document. Each subsequent decision resulting from the original choice is also depicted on the tree, so you can see the overall effect of any one decision. As you go through the tree and make choices, you will see a specific path from one node to another and the impact a decision made now could have down the road.

- Brainstorming Outcomes : Decision trees help you think of all possible outcomes for an upcoming choice. The consequences of each outcome must be fully explored, so no details are missed. Taking the time to brainstorm prevents overreactions to any one variable. The graphical depiction of various alternatives makes them easier to compare with each other. The decision tree also adds transparency to the process. An independent party can see exactly how a particular decision was made.

- Decision Tree Versatility : Decision trees can be customised for a variety for situations. The logical form is good for programmers and engineers. Technicians can also use decision trees to diagnose mechanical failures in equipment or troubleshoot auto repairs. Decision trees are also helpful for evaluating business or investment alternatives. Managers can recreate the math used in a particular decision tree to analysing the company's decision-making process.
- Less data cleaning required: It requires less data cleaning compared to some other modelling techniques. It is not influenced by outliers and missing values to a fair degree.
- Data type is not a constraint: It can handle both numerical and categorical variables.
- Non Parametric Method: Decision tree is considered to be a non-parametric method. This means that decision trees have no assumptions about the space distribution and the classifier structure.

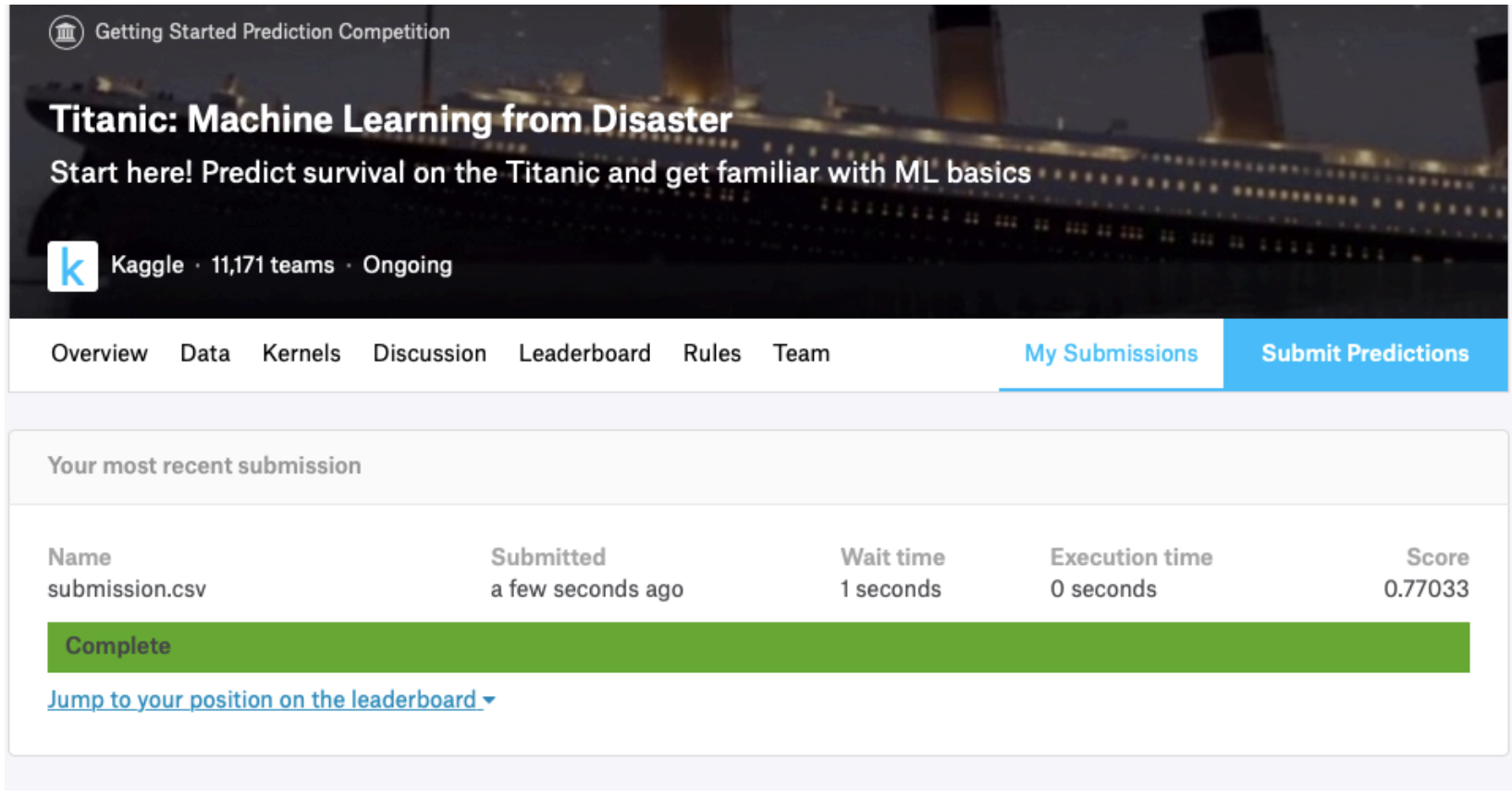
In our case, the decision tree looked like this:-



```
(418,)
[0 0 0 0 1 0 1 0 1 0 0 0 1 0 1 1 0 0 1 0 0 0 1 0 1 0 1 0 0 0 0 0 1 0 0 0 1
 1 0 0 0 0 0 0 1 1 0 0 0 1 1 0 0 1 1 0 0 0 0 0 1 0 0 0 1 0 1 1 0 0 1 1 0 1 0
 1 0 0 1 0 1 1 0 0 0 0 0 0 1 1 1 1 1 0 1 0 0 0 1 0 1 0 1 0 0 0 1 0 0 0 0 0 0
 1 1 1 1 1 0 0 1 0 1 1 0 1 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0
 0 0 1 0 0 1 0 0 1 1 0 1 1 1 1 0 0 1 0 0 1 1 0 0 0 0 0 1 1 0 1 1 0 0 1 0 1
 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0 0 1 0 0 0 0 1 0 0 0 0 1 1 0 1 0 1 0 1 0
 1 0 1 1 0 1 0 0 0 1 0 0 0 0 0 0 1 1 1 1 0 0 1 0 1 0 1 1 1 0 0 0 0 0 0 0 1
 0 0 0 1 1 0 0 0 0 1 0 0 0 1 1 0 1 0 0 0 0 1 0 1 1 1 0 0 0 0 0 0 1 0 0 0 0
 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0
 1 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 1 0 0 0 1 0 1 0 0 1 0 1 1 0 1 0 0 1 1 0
 0 1 0 0 1 1 1 0 0 0 0 0 1 1 0 1 0 0 0 1 0 1 1 0 1 1 0 1 0 0 1 0 1 0 0 0 0
 0 1 1 1 1 1 0 1 0 0 1]
(418, 11)
```

Upon running the above described model over test data, the above predictions were produced.

Upon submission on Kaggle platform, the following results were obtained:-



The model scored **0.77** which is much better than a random classifier (i.e. 0.50).

The performance could be improved upto 0.89 by using a Random Forest Classifier

To check out the code, please visit the following link:-

https://github.com/Vrishank-Gupta/ML-Titanic