Titanic
Machine Learning from Disaster

Name: Ajay Kumar Yadav | Course Title: Data Science | Duration: June - August

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.

Introduction

The goal of the project was to predict the survival of passengers based off a set of data. We used Kaggle competition "Titanic: Machine Learning from Disaster" (see *https://www.kaggle.com/c/titanic/data*) to retrieve necessary data and evaluate accuracy of our predictions. The historical data has been split into two groups, a *'training set'* and a *'test set'*. For the training set, we are provided with the outcome (whether or not a passenger survived). We used this set to build our model to generate predictions for the test set.

For each passenger in the test set, we had to predict whether or not they survived the sinking. Our score was the percentage of correctly predictions.

In our work, we learned

- Programming language Python and its libraries NumPy (to perform matrix operations) and SciKit-Learn (to apply machine learning algorithms)
- Several machine learning algorithms (decision tree, random forests, Logistic regression, support vector machine)
- Feature Engineering techniques

We used

- Online integrated development environment Jupyter Notebook
- Python 3.6.6 with the libraries numpy, sklearn, and matplotlib
- Microsoft Excel

Work plan

1. Learn programming language Python

2. Learn Shennon Entropy and write Python code to compute Shennon Entropy

3. Get familiar with Kaggle project and try using Pivot Tables in Microsoft Excel to analyze the data.

4. Learn to use SciKit-Learn library in Python, including

    a. Building decision tree

    b. Building Random Forests

    c. SVC (support Vector Machine)

    d. Using Logistic Regression algorithm

5. Performing Feature Engineering, applying machine learning algorithms, and analyzing results

Training and Test Data

Training and Test data come in CSV file and contain the following fields:

- Passenger ID
- Passenger Class
- Name
- Sex
- Age
- Number of passenger's siblings and spouses on board
- Number of passenger's parents and children on board
- Ticket
- Fare
- Cabin
- City where passenger embarked

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	Title
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	0
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	3
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	2
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	3
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	0
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q	0
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S	0
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S	3
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN	S	3
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	NaN	C	3

Null and non – Null values:

```
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
```

Feature Engineering

Since the data can have missing fields, incomplete fields, or fields containing hidden information, a crucial step in building any prediction system is *Feature Engineering*. For instance, the fields Age, Fare, and Embarked in the training and test data, had missing values that had to be filled in. The field Name while being useless itself, contained passenger's Title (Mr., Mrs., etc.), we also used passenger's surname to distinguish families on board of Titanic. Below is the list of all changes that has been made to the data.

Extraction Title from name:

The field Name in the training and test data has the form "Braund, Mr. Owen Harris". Since name is unique for each passenger, it is not useful for our prediction system. However, a passenger's title can be extracted from his or her name. We found 10 titles:

Index	Title	Number of occurrences
0	Col.	4
1	Dr.	8
2	Lady	4
3	Master	61
4	Miss	262
5	Mr.	757
6	Mrs.	198
7	Ms.	2
8	Rev.	8
9	Sir	5

We can see that title may indicate passenger's sex (Mr. vs Mrs.), class (Lady vs Mrs.), age (Master vs Mr.), profession (Col., Dr., and Rev.).

Calculating Family Size :

It seems advantageous to calculate family size as follows:

Family_Size = Parents_Children + Siblings_Spouses + 1

Extracting Deck from Cabin :

The field Cabin in the training and test data has the form "C85", "C125", where Crefers to the deck label. We found 8 deck labels: A, B, C, D, E, F, G, T. We see decklabel as a refinement of the passenger's class field since the decks A and B wereintended for passengers of the first class, etc.

Extracting Ticket_Code from Ticket :

The field Ticket in the training and test data has the form "A/5 21171". Although we couldn't understand meaning of letters in front of numbers in the field Ticket, we extracted those letters and used them in our prediction system. We found the following letters

Index	Ticket Code	Number of occurrences
0	No Code	961
1	A	42
2	C	77
3	F	13
4	L	1
5	P	98
6	S	98
7	W	19

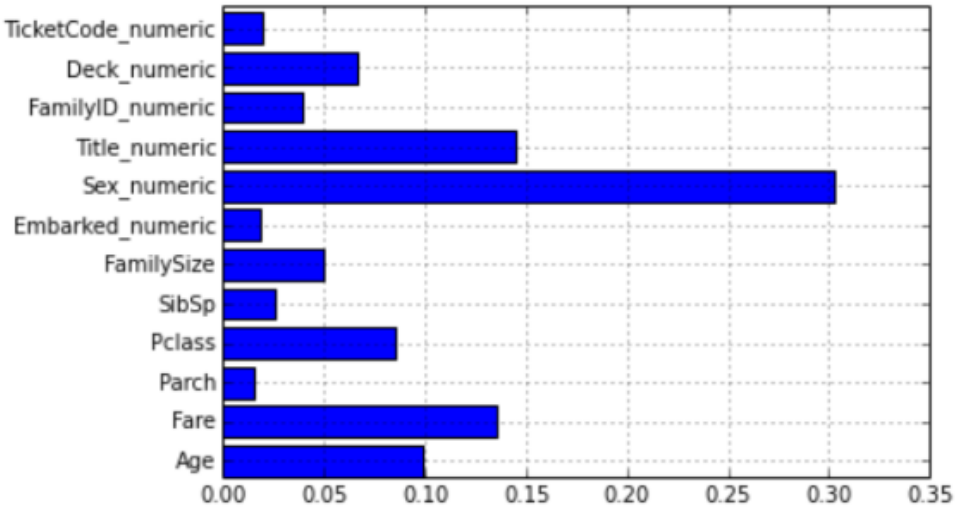
Filling in missing values in the fields Fare, Embarked, and Age:

Since the number of missing values was small, we used median of all Fare values to fill in missing Fare fields, and the letter 'S' (most frequent value) for the field Embarked.

In the training and test data, there was significant amount of missing Ages. To fill in those, we used Linear Regression algorithm to predict Ages based on all other fields except Passenger_ID and Survived.

Importance of fields :

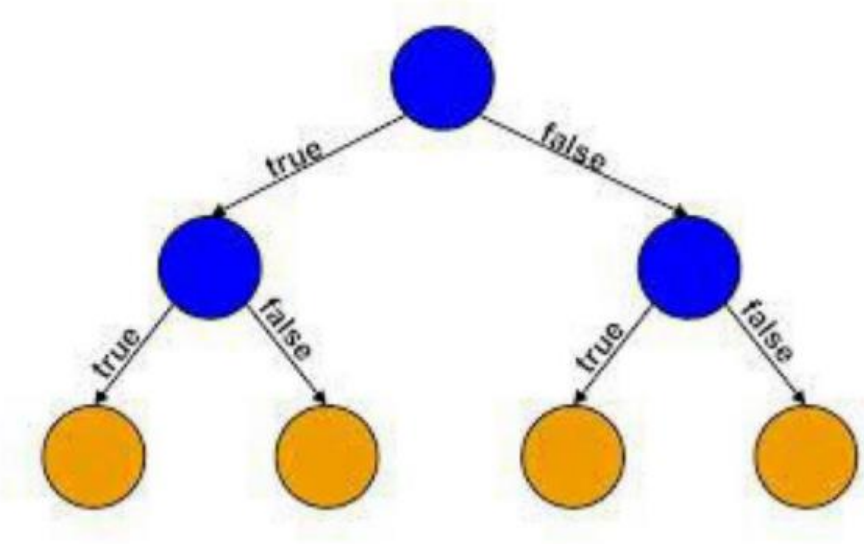
Decision Trees algorithm in the library SciKit-Learn allows to evaluate importance of each field used for prediction. Below is the chart displaying importance of each field.



We can see that the field Sex is the most important one for prediction, followed by Title, Fare, Age, Class, Deck, Family_Size, etc.

Decision Trees

Our prediction system is based on growing Decision Trees to predict the survival status. A typical Decision Tree is pictured below



The basic algorithm for growing Decision Tree:

1. Start at the root node as parent node

2. Split the parent node based on field X[i] to minimize the sum of child nodes uncertainty (maximize information gain)

3. Assign training samples to new child nodes

4. Stop if leave nodes are pure or early stopping criteria is satisfied, otherwise repeat step 1 and 2 for each new child node

Stopping Rules:

1. The leaf nodes are pure

2. A maximal node depth is reached

3. Splitting a node does not lead to an information gain

In order to measure uncertainty and information gain, we used the formula

$$IG(D_p) = I(D_p) - \frac{N_{left}}{N_p} I(D_{left}) - \frac{N_{right}}{N_p} I(D_{right})$$

where

- IG : Information Gain
- I : Impurity (Uncertainty Measure)
- N_p, N_{left}, N_{right} : number of samples in the parent, the left child, and the right child nodes
- D_p, D_{left}, D_{right} : training subset of the parent, the left child, and the right child nodes

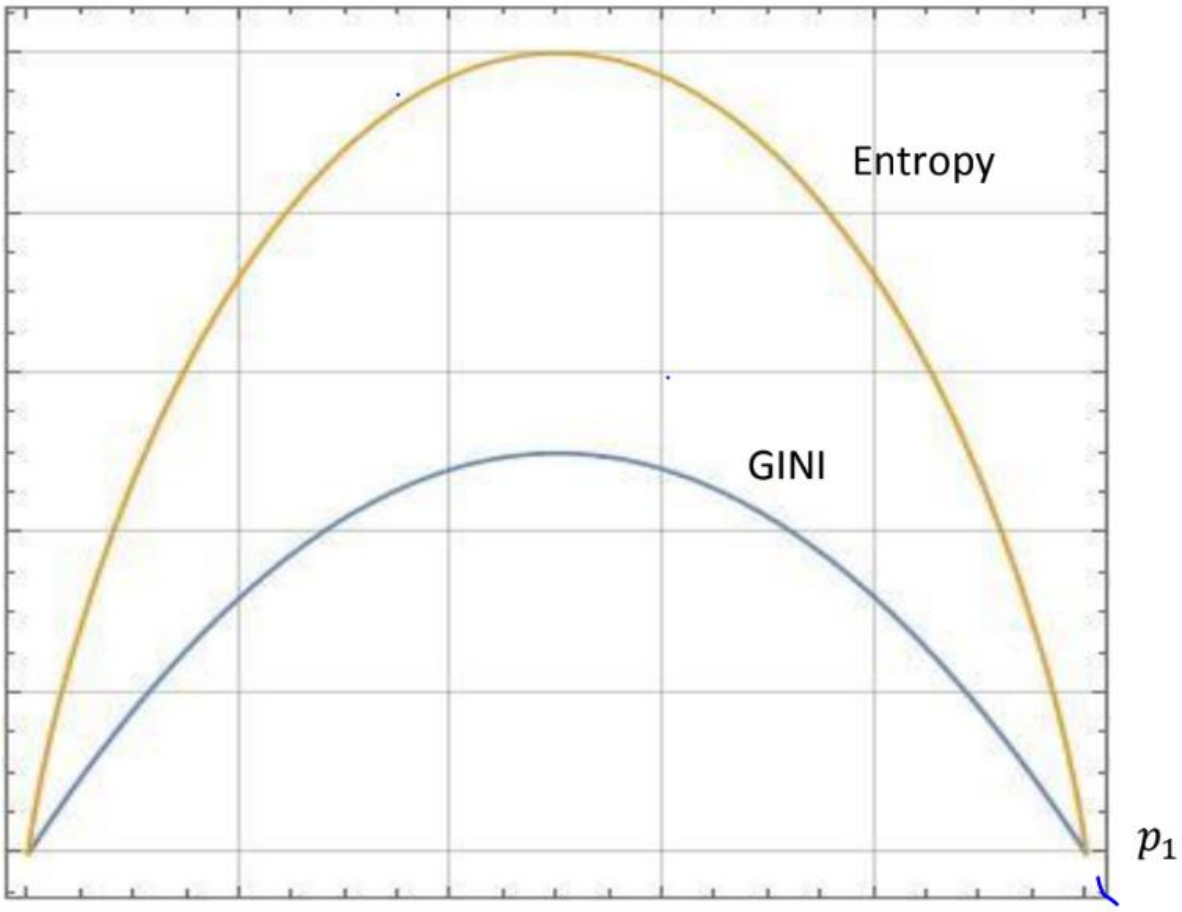
For Uncertainty Measure, we used Entropy defined by

$$I(p_1, p_2) = -p_1 \log_2 p_1 - p_2 \log_2 p_2$$

and GINI index defined by

$$I(p_1, p_2) = 2p_1 p_2$$

The graphs of both measures are given below



We can see on the graph that when probability of an event is 0 or 1, then theuncertainty measure equals to 0, while if probability of an event is close to 1⁄2, then the uncertainty measure is maximum.

Random Forest

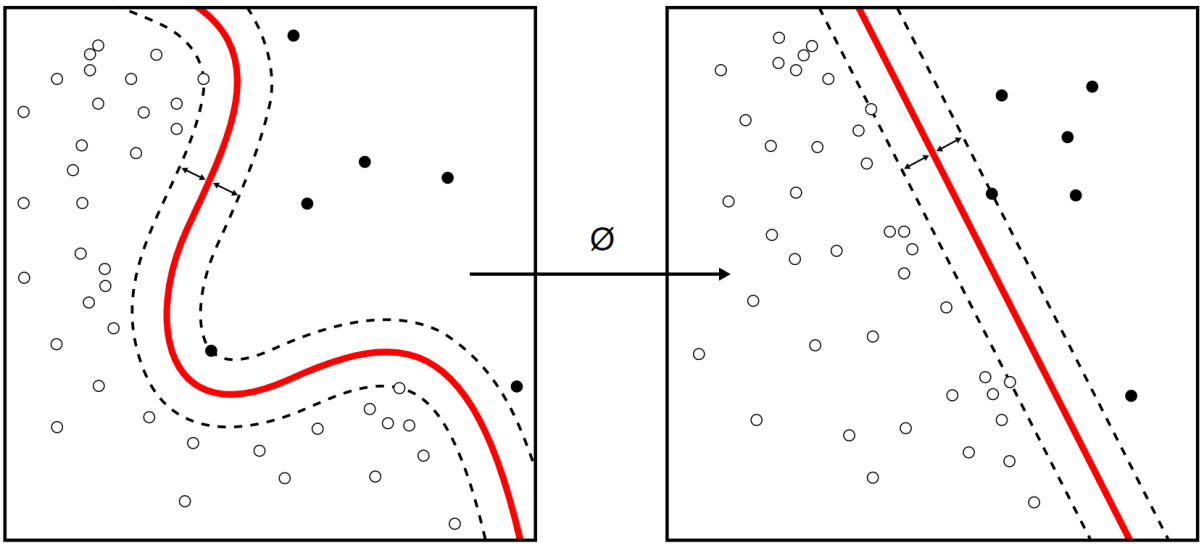
One common issue with all machine learning algorithms is Overfitting. For DecisionTree, it means growing too large tree (with strong bias, small variation) so it loses its ability to generalize the data and to predict the output. In order to deal with overfitting, we can grow several decision trees and take the average of their predictions. The library SciKit-Learn provides to such algorithm Random Forest and ExtraTrees.

In Random Forest, we grow N decision trees based on randomly selected subset ofthe data and randomly selected M fields.

In ExtraTrees, in addition to randomness of subsets of the data and of field, splits of nodes are chosen randomly.

Support Vector Machine

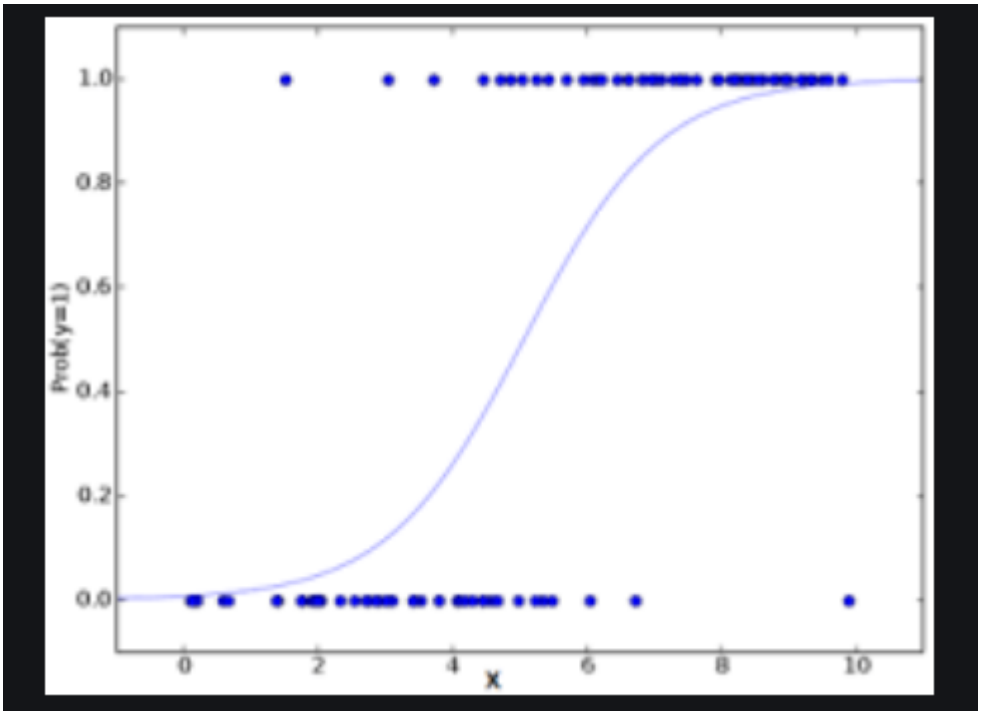
In machine learning, **support-vector machines** (**SVMs**, also **support-vector networks**) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall.



Logistic Regression

In statistics, the **logistic model** (or **logit model**) is used to model the probability of a certain class or event existing such as pass/fail, win/lose, alive/dead or healthy/sick. This can be extended to model several classes of events such as determining whether an image contains a cat, dog, lion, etc... Each object being detected in the image would be assigned a probability between 0 and 1 and the sum adding to one.

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, **logistic regression** (or **logit regression**) is estimating the parameters of a logistic model (a form of binary regression). Mathematically, a binary logistic model has a dependent variable with two possible values, such as pass/fail which is represented by an indicator variable, where the two values are labeled "0" and "1".



Conclusion

As a result of our work, we gained valuable experience of building prediction systems and achieved our best score on Kaggle: 79.425% of correct predictions (inKaggle leaderboard, it corresponds to positions 1964 out of 10924 participants).

We performed featured engineering techniques :

- Changed alphabetic values to numeric
- Calculated family size
- Extracted title from name and deck label from ticket number
- Used linear regression algorithm to fill in missing ages

We used several prediction algorithms in python:

- Logistic Regression
- Support Vector Machine
- Random forests
- Decision tree
- We achieved our best score 79.425% correct predictions