Comparitive Analysis of Support Vector Machines to Three-Layer Neural Network for classification on Titanic Dataset

Sankalp Ram Saoji
Machine learning
AITS
Dublin, Ireland

Abstract*— *The sinking of the titanic ship is one of the most terrifying disaster in the history of this world, killing 722 passengers on board including the crew member in the year 1912. There have been dozens of studies done but still researchers are attracted towards finding the impact of an individual for their survival or death. The objective of this paper is to compare two machine learning algorithms such as support vectors machines to three-layer neural network for classification problem which predicts individual's statues of survived or not survived in the disaster. The titanic dataset is publically available on kaggle with different attributes of an individual passenger to predict survived or not survived.

Keywords—component, formatting, style, styling, insert (*key words****)***

I. INTRODUCTION

Machine learning and data analysis has reached outrageous researches in this modern era. One of the researches in the field of machine learning is done using an unusual and terrifying disaster that caused many people to death was the sinking of a ship mostly famously called as the Titanic in the north Atlantic Ocean on 15th of April, 1912. The reports on Titanic disaster concludes that more than 1500 were killed out of 2224 passengers including the crew members [1] (chaterjee). The research and analysis on what basis the survival has been impacted is going on till date. Therefore, Kaggle the one of the biggest data repository for machine learning has made the titanic data publically available for individual and companies for the research of impacts of survival.

The aim of this research project is to compare prediction accuracies, if the person survived or not in the disaster using the titanic dataset for two different machine learning algorithms. The target variable in the dataset is survival which has two values one is survived and other is not survived. There are many independent variables such as age, gender, class, family size, name and title, ticket type, fare, cabin type. The target variable mentioned above has 2 classes so this is classification problem and two machine learning algorithms used for this research are support vector machine and three-layer neural network.

Research question: "Comparing the accuracies of support vector machines to three-layer neural network for classification of people survived or not survived on the titanic dataset."

II. LITERATURE REVIEW

This section focuses on how other researcher used various machine learning algorithms for classification problems.

A. Regression models

Chatterjee [1] applied two regression models such as multiple logistic regression and logistic regression to classify whether a passenger is survived in the disaster or not.

The performance metrics measured by him across various cases and concluded that, the maximum accuracy obtained from Multiple Linear Regression is 78.426%;

The maximum accuracy reached using Logistic Regression is 80.756%.

B. Decision trees and random forest

Datla [2] compared the classification results of Decision tree and Random Forests algorithms for Titanic dataset. Decision tree is resulted with the maximum accuracy of 84% correctly classified instances, while Random Forests could only reach 81% comparatively.

As concerned with feature engineering, Datla [2] created more variables such as "survived", "child", "new_fare", "title", "Familysize", "FamilyIdentity" which were not present in original Titanic dataset and also mean values were used to replace the missing values in the dataset.

C. SVM and Neural networks

The paper published by Meyer et al., [3] compared Support vector machines to neural networks and concluded the error rates of nearly 20.81% for SVM and 21.28% for neural networks. Thus, SVM worked little well compared to neural network for classification on titanic dataset.

D. Adaboost, SVM and RBF

The author Ratsch et al. [4] compared Adaboost classifiers to SVM and RBF classifiers. For classification on titanic dataset, and found out the minimum error rate was 22.4%

Li et al. [5], Obtained a mimnimun error rate of 21.8% using the SVM as a component classifier for Adaboost for classification on titanic dataset.

The paper follows a pattern such that the next section is about data pre-processing, exploratory data analysis and feature engineering followed by implementation and results chapter and after that is conclusion, future work and references.

III. DATA PREPROCESSING, EXPLORATORY DATA ANALYSIS AND FEATURE ENGINEERING

Data preprocessing and exploratory data analysis are the first step in implementing machine learning models. In this

research project we started with exploring the features in the titanic dataset.

The first feature choose for this was the Passenger class (Pclass), we first checked for null values for all the features. There were no null values in Pclass so we moved further to check how is Pclass correlated to the survival and from the figure 1, below we can clearly see that the higher the number of passenger class (lower value in data) has higher survival rate therefore Pclass can be important feature.

	Pclass	Survived
0	1	0.629630
1	2	0.472826
2	3	0.242363

Figure 1 : Pclass vs survived

The second feature in this dataset was name in this feature the names were added with their designated titles which contains information of gender and their status, so we do not need name, we just extracted titles from the data. There were total 18 different title in which 4 main with maximum number of people were 'Master', 'Mr', 'Miss', 'Mrs' and other titles were merged into one labeled as 'others'. Then we checked correlation of this feature with survival and from the figure 2, we can conclude that Miss and Mrs were having maximum survival rate and Mr were having the minimum survival rate.

	Title	Survived
0	Master	0.575000
1	Miss	0.701087
2	Mr	0.156673
3	Mrs	0.796875
4	Others	0.318182

Figure 2 : Title vs survived

This feature is in text format so we need to convert this feature using dummy encoding to binary format. We used concat function in python for converting feature labels to binary.

The next independent variable was sex in which we have the gender of passengers travelling in the ship. There were 2 labels male and female. The correlation between sex and survival was very interesting as females had a 74% chance of survival and males just had 18% chance of survival. So this was one of the most important feature for classification. The labels were the mapped to 0 as male and 1 as female for further modelling. Figure 3 shows survival rate versus sex.

	Sex	Survived
0	female	0.742038
1	male	0.188908

Figure 3 : Sex vs survived

The next 2 features sibsp and Parch were merged into one feature as family size as it contains number of siblings and number of parents on board. So we found the correlation between family size and survival rate. There we came across a fact that family size of 4 had maximum survival rate and rate was decreasing as the family size decreases, but family size more than 4 had minimum percentage of survival. Figure 4 shows the correlation of family size with survival rate and family with size 0 are mapped as family with more than 4 persons.

	Family	Survived
0	0	0.161290
1	1	0.303538
2	2	0.552795
3	3	0.578431
4	4	0.724138

Figure 4 : Family size vs survived

As per the above features ticket type and fare were also explored to find more hidden patterns in the data. Figure 5 shows box plot between Pclass and fare.

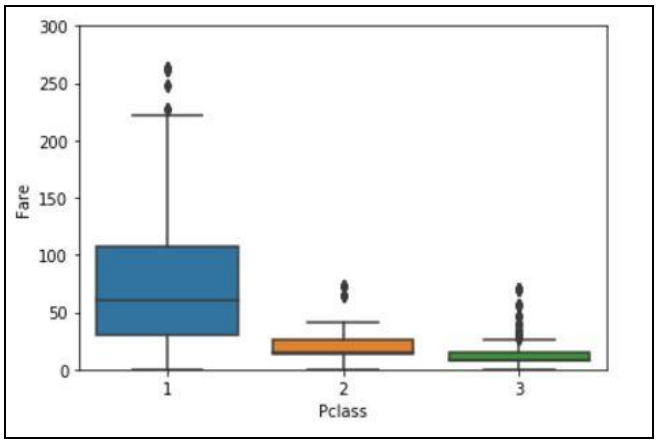

Figure 5 : Box plot of pclass vs fare

The fare was different for all the Pclass so we also decided to find the correlation between fare and the survival rate. The figure 6 below shows scatter plot between fare and survival rate. From the plot below we can conclude that people with less fare had less chance of survival compared to people with high fares.

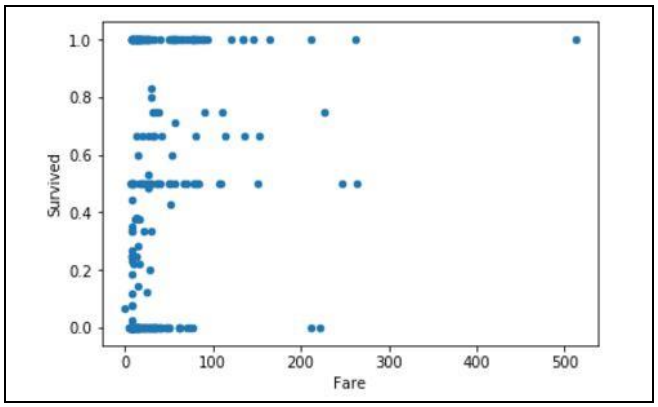

Figure 6 : Scatter plot of Fare vs Survived

Thus, we used other features such as age, ticket type and used functions such as binning and dummy encoding on them to feed perfect data for our machine learning models to get efficient results. we also used random forest regressor function to impute missing values in our age feature. Figure 7 below shows the final dataset used for modelling.

	PassengerId	Pclass	Sex	Survived	Master	Miss	Mr	Mrs	Others	Family	Fare-bin	Age-bin	Ticket_1	Ticket_2	Ticket_3	Ticket_4	Ticket_C	Ticket_P	Tick
0	1	3	0	0.0	0	0	1	0	0	2	1	4	0	0	0	1	0	0	
1	2	1	1	1.0	0	0	0	1	0	2	5	5	0	0	0	0	0	1	
2	3	3	1	1.0	0	1	0	0	0	1	2	4	0	0	0	0	0	0	
3	4	1	1	1.0	0	0	0	1	0	2	5	5	1	0	0	0	0	0	
4	5	3	0	0.0	0	0	1	0	0	1	2	5	0	0	1	0	0	0	
5	6	3	0	0.0	0	0	1	0	0	1	2	4	0	0	1	0	0	0	
6	7	1	0	0.0	0	0	1	0	0	1	5	6	1	0	0	0	0	0	
7	8	3	0	0.0	1	0	0	0	0	0	3	1	0	0	1	0	0	0	
8	9	3	1	1.0	0	0	0	1	0	3	3	4	0	0	1	0	0	0	
9	10	2	1	1.0	0	0	0	1	0	2	4	3	0	1	0	0	0	0	
10	11	3	1	1.0	0	1	0	0	0	3	3	1	0	0	0	0	0	1	

Figure 7 : Dataset

IV. IMPLEMENTATION AND RESULTS

This section describes how we implemented the machine learning models. First step is the dataset; the dataset was downloaded from Kaggle and it contained two datasets one is training and testing. For the comparison of accuracies of two algorithms we needed the target variable in both training and testing dataset so for this project we divided the training dataset into train and test. The split use was 70% for training and 30% for testing data.

The implementation of this project was carried out in python and the libraries and packages used were pandas, numpy, matplotlib, seaborn, sklearn and keras.

A. Neural Network model

The first model build was the neural networks model with training data. The neural network build was a three-layer neural network with two layer with the relu function and last layer with the sigmoid function. The figure 8 shows the summary of the neural network. The neural network was compiled using batch size of 32 and with 200 epochs.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_4 (Dense)              (None, 16)                288
_________________________________________________________________
dense_5 (Dense)              (None, 12)                204
_________________________________________________________________
dense_6 (Dense)              (None, 1)                 13
=================================================================
Total params: 505
Trainable params: 505
Non-trainable params: 0
_________________________________________________________________
```

Figure 8 : Neural network model summary

We also tried to this neural network model with different hyper parameters such as batch size of 64, epochs changes to 100 and 500 as well, even we tried to change the training the testing split to 80 and 20 percent, but we according to this project the above combination was most efficient. Using the above mentioned neural network model we obtained an accuracy of 80.59%. Figure 9 shows accuracy of the model mentioned above.

```
from sklearn.metrics import accuracy_score
Accuracy = accuracy_score(y_test,y_final)
print("Accuracy:", Accuracy)
```

```
Accuracy: 0.8059701492537313
```

Figure 9 : Neural network accuracy

For the neural network model, we also plotted the classification report which shoes precision, recall, F1 score and support, figure 10 below shows the classification report for neural network model.

```
from sklearn.metrics import classification_report
C_report = classification_report(y_test,y_final)
print(C_report)
```

```
              precision    recall  f1-score   support

         0.0       0.80      0.91      0.85       160
         1.0       0.83      0.66      0.73       108

   micro avg       0.81      0.81      0.81       268
   macro avg       0.81      0.78      0.79       268
weighted avg       0.81      0.81      0.80       268
```

Figure 10 : Classification report

We have also implemented the confusion matrix of actual versus predicted values with labels survived and not. Figure 11 shows the confusion matrix for neural network model. We can see that out of 160 this model predicted 146 correct and 14 not correct for survived label. On the other hand, for not survived out 0f 108, 70 were correctly predicted and 38 were not correct.

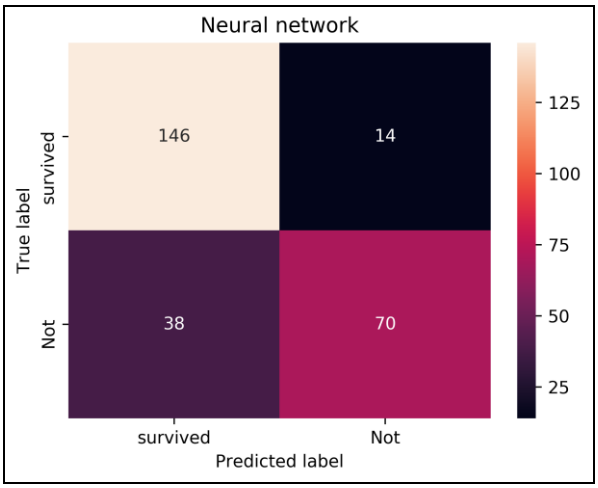

Figure 11 : Confusion matrix

B. Support vector machine

The support vector machine model was implemented for classification on the given titanic dataset and we reached an accuracy of 81.34%. It slightly outperformed neural networks. The figures 12 and 13 below shows accuracy and classification report for SVM model.

```
from sklearn.metrics import accuracy_score
Accuracy = accuracy_score(y_test,y_pred)
print("Accuracy:", Accuracy)

Accuracy: 0.8134328358208955
```

Figure 12 : SVM accuracy

```
from sklearn.metrics import classification_report
C_report = classification_report(y_test,y_pred)
print(C_report)
```

	precision	recall	f1-score	support
0.0	0.81	0.91	0.85	160
1.0	0.83	0.68	0.74	108
micro avg	0.81	0.81	0.81	268
macro avg	0.82	0.79	0.80	268
weighted avg	0.82	0.81	0.81	268

Figure 13 : Classification report

The confusion matrix for SVM model in shown as below in figure 14. We can see that out of 160 this model predicted 145 correct and 15 not correct for survived label and for not survived out 0f 108, 73 were correctly predicted and 35 were not correct.

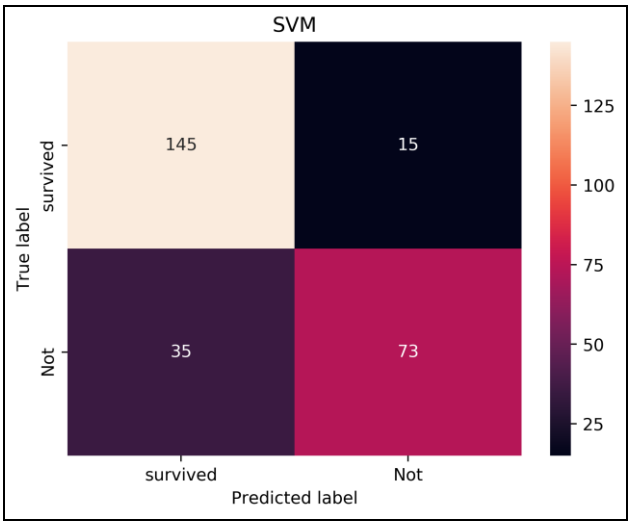

Figure 14 : Confusion matrix

V. Conclusion and Future Work

Thus, to conclude we can say that for this research project we have successfully implemented neural networks and support vector machines (SVM) models for classification using titanic dataset and would like conclude that SVM has slightly outperformed neural networks. The reason behind can be that neural networks requires large training data to perform efficiently. Therefore, SVM supported to be a better classifier on titanic data for this project.

For future work we can up sample the training dataset for more accuracy as data is very small for neural networks to train properly and we can also tune more hyper parameters, add numbers of layers in neural networks and in other case boosting can also be used to reduce error rate.

References

[1] T. Chatterjee, "Prediction of Survivors in Titanic Dataset: A Comparative Study using Machine Learning Algorithms," International Journal of Emerging Research in Management &Technology, vol. 6, pp. 1-5, June 2017.

[2] M. V. Datla, "Bench Marking of Classification Algorithms: Decision Trees and Random Forests – A Case Study using R," in Proc. I-TACT-15, 2015, pp. 1-7.

[3] D. Meyer, F. Leisch and K. Hornik, "The support vector machine under test," Neurocomputing, vol. 55, pp. 169-186, Sept. 2003.

[4] G. Rätsch, T. Onoda, and K.-R. Müller, "Soft Margins for AdaBoost," Machine Learning,vol. 42, pp. 287-320, Mar. 2001.

[5] X. Li, L. Wang, and E. Sung, "AdaBoost with SVM-based component classifiers," Engineering Applications of Artificial Intelligence, vol. 21, pp. 785-795, Aug. 2008.