COMPARE SUPPORT VECTOR MACHINES TO A 3 LAYER NEURAL NETWORKS ON THE TITANIC DATASET

PriyanshiYadav
Machine Learning Intern

at

AI Technology and Systems
Lucknow, India.
priyanshi.yadav.2328@gmail.com

***Abstract*—The RMS Titanic was a British passenger liner that sank in the North Atlantic Ocean in the early morning hours of 15 April 1912, after it collided with an iceberg during its maiden voyage from Southampton to New York City. There were an estimated 2,224 passengers and crew aboard the ship, and more than 1,500 died, making it one of the deadliest commercial peacetime maritime disasters in modern history. The RMS Titanic was the largest ship afloat at the time it entered service and was the second of three Olympic-class ocean liners operated by the White Star Line. The Titanic was built by the Harland and Wolff shipyard in Belfast. Thomas Andrews, her architect, died in the disaster.**

Keywords- Machine Learning, Support Vector Machine, Deep Learning, Keras, Neural Networks

INTRODUCTION

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.

DATASET

There are two csv files for this dataset.
We can view these csv (Comma separated files) datasets by using *.read_csv()* function from pandas.

- train.csv
  This is used for training the data because it not only contains the personal information of each passenger but also has the survival information of the person.

These are the top 5 entries of train.csv file.

	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

- test.csv

This is for predicting the survival of the passengers. All the relevant information of passenger is listed in this csv file.

The test data set looks like:

	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

1. survival - Survival (0 = No; 1 = Yes)
2. class - Passenger Class (1 = 1st; 2 = 2nd; 3 = 3rd)
3. name - Name
4. sex - Sex
5. age - Age
6. sibsp - Number of Siblings/Spouses Aboard
7. parch - Number of Parents/Children Aboard
8. ticket - Ticket Number
9. fare - Passenger Fare
10. cabin – Cabin
11. embarked – Port of Embarkation (C = Cherbourg; Q = Queenstown; S = Southampton)

ATTRIBUTES OF THE DATASET

We can see the attributes of the dataset using *data.info()* function.

```
<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
dtypes: float64(2), int64(5), object(5)
memory usage: 83.6+ KB
```

The attributes for the dataset can be divided into two parts:

1. Qualitative Attributes:

    PassengerId, Survived, Pclass, Sex, Ticket, Cabin & Embarked.

2. Quantitative Attributes:

    SlibSp, Parch & Fare.

DATA ANALYSIS

Before starting with the data analysis, we need to have a basic idea about our dataset.

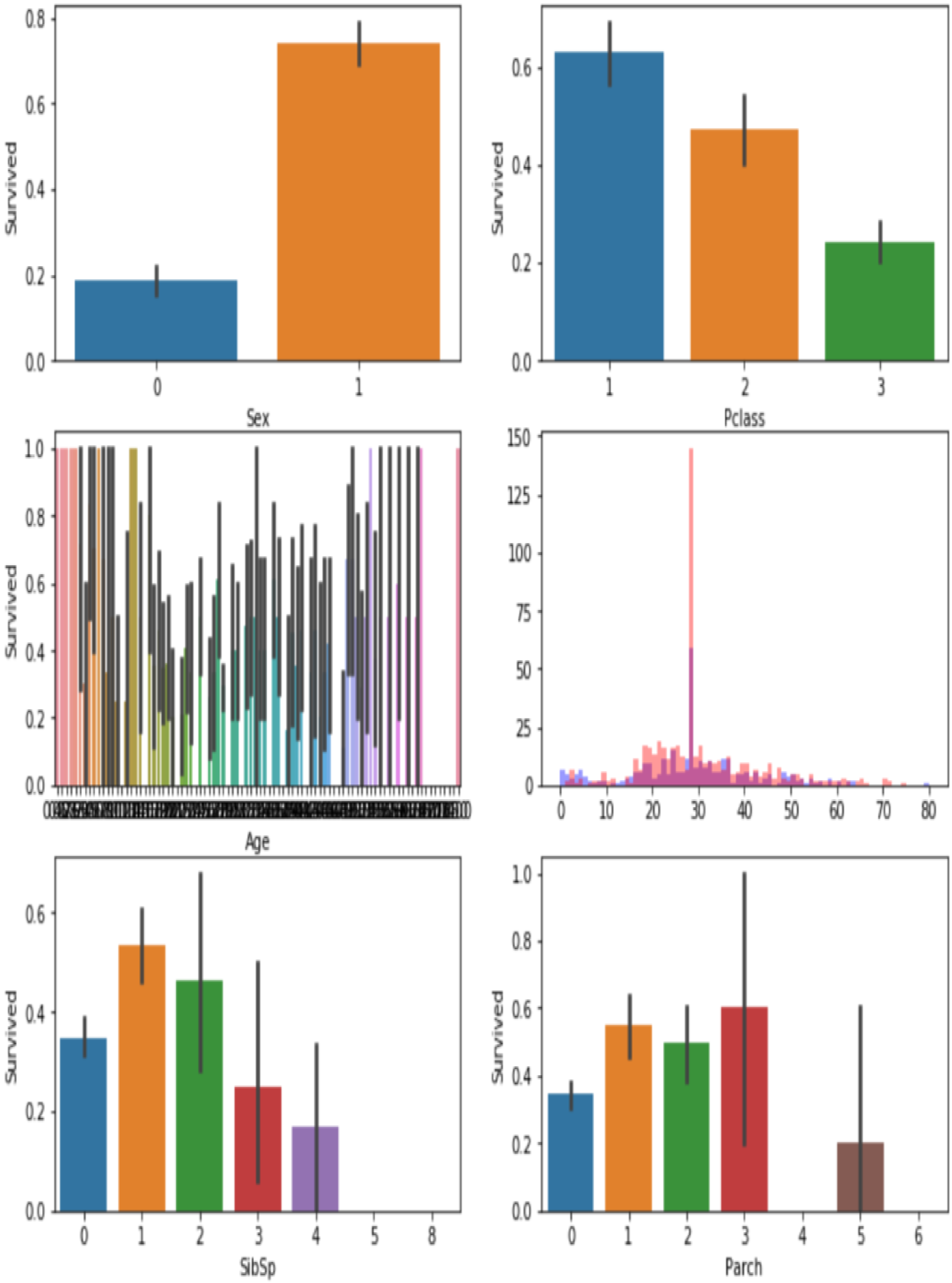
We can at very basic level use *.describe()* function to find out the mathematical relations.

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
count	891.000000	891.000000	891.000000	891.000000	891.000000	891.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	0.352413	29.361582	0.523008	0.381594	32.204208	0.361392
std	257.353842	0.486592	0.836071	0.477990	13.019697	1.102743	0.806057	49.693429	0.635673
min	1.000000	0.000000	1.000000	0.000000	0.420000	0.000000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	0.000000	22.000000	0.000000	0.000000	7.910400	0.000000
50%	446.000000	0.000000	3.000000	0.000000	28.000000	0.000000	0.000000	14.454200	0.000000
75%	668.500000	1.000000	3.000000	1.000000	35.000000	1.000000	0.000000	31.000000	1.000000
max	891.000000	1.000000	3.000000	1.000000	80.000000	8.000000	6.000000	512.329200	2.000000

.describe() function not only helps us to find the minimum and maximum values but also gives us the mean, standard deviation and quartile values.

GRAPHICAL INSIGHT OF THE DATA

We can easily estimate the survival rate by looking at the pictorial representation.



Survived
0.0
0.2
0.4
0.6
0.8
0
1
Sex
Survived
0.0
0.2
0.4
0.6
1
2
3
Pclass
Survived
0.0
0.2
0.4
0.6
0.8
1.0
Age
0
25
50
75
100
125
150
0
10
20
30
40
50
60
70
80
Survived
0.0
0.2
0.4
0.6
0
1
2
3
4
5
8
SibSp
Survived
0.0
0.2
0.4
0.6
0.8
1.0
0
1
2
3
4
5
6
Parch

DATA PREPOCESSING

In order to start our prediction, we first need to find and fill all NaN values in our dataset. Then we need to pre-process the data. We have to convert all non-numeric data into numeric for our models to work efficiently.

```
In [14]: total = data.isnull().sum().sort_values(ascending=False)
         percent_1 = data.isnull().sum()/data.isnull().count()*100
         percent_2 = (round(percent_1, 1)).sort_values(ascending=False)
         missing_data = pd.concat([total, percent_2], axis=1, keys=['Total', '%'])
         missing_data.head(5)
```

Out[14]:

	Total	%
Cabin	687	77.1
Age	177	19.9
Embarked	2	0.2
Fare	0	0.0
Ticket	0	0.0

```
In [0]: data["Fare"]=data["Fare"].fillna(data["Fare"].dropna().median())
        data["Age"]=data["Age"].fillna(data["Age"].dropna().median())


                ## converting sex data into numeric data: male=0,female=1
        data.loc[data["Sex"]=="male","Sex"]=0
        data.loc[data["Sex"]=="female","Sex"]=1


                ### converion embarking data to 0 for 'S', 1 for 'C',3 for 'Q'
        data.loc[data["Embarked"]=="S","Embarked"]=0
        data.loc[data["Embarked"]=="C","Embarked"]=1
        data.loc[data["Embarked"]=="Q","Embarked"]=2

        data['Embarked']=data['Embarked'].fillna(data['Embarked'].dropna().median())
```

SUPPORT VECTOR MACHINE

*A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labelled training data (*supervised learning*), the algorithm outputs an optimal hyperplane which categorizes new examples. In two dimensional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side.*

TUNING PARAMETERS OF SVM

I. Kernel

The function of **kernel** is to take data as input and transform it into the required form. These functions can be different types. For example linear, nonlinear, polynomial, radial basis function (RBF), and sigmoid.

I. Regularization

The Regularization parameter (often termed as C parameter in python’s sklearn library) tells the SVM optimization how much you want to avoid misclassifying each training example.

For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. Conversely, a very small value of C will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points.

II. Gamma

The gamma parameter defines how far the influence of a single training example reaches, with low values meaning ‘far’ and high values meaning ‘close’. In other words, with low gamma, points far away from plausible separation line are considered in calculation for the separation line. Whereas high gamma means the points close to plausible line are considered in calculation.

III. Margin

And finally, last but very important characteristic of SVM classifier. SVM to core tries to achieve a good margin.

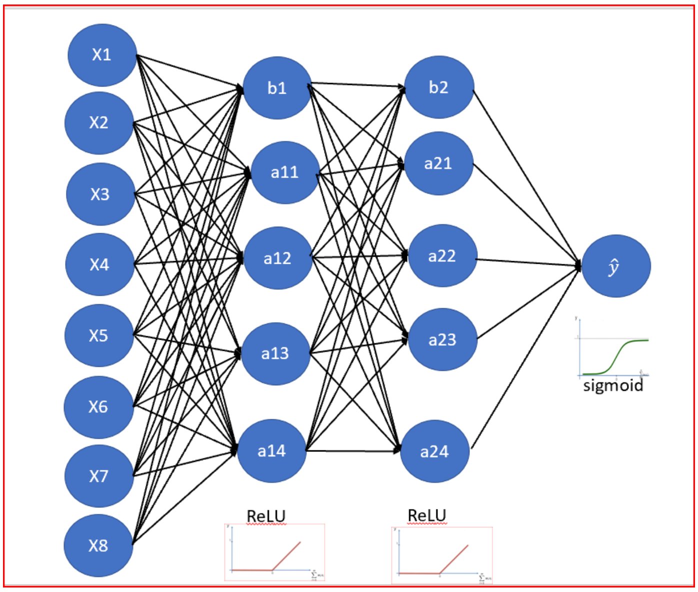
A margin is a separation of line to the closest class points.

A *good margin* is one where this separation is larger for both the classes. Images below gives to visual example of good and bad margin. A good margin allows the points to be in their respective classes without crossing to other class.

NEURAL NETWOR USING KERAS

A neural network is a series of algorithms that endeavours to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria.

I. We are using keras to build our neural network. We import the keras library to create the neural network layers.
II. There are two main types of models available in keras — Sequential and Model.
III. We will use Sequential model to build our neural network.
IV. We use Dense library to build input, hidden and output layers of a neural network.
V. We have 7 input features and one target variable. 3 Hidden layers.
VI. ReLu will be the activation function for hidden layers. As this is a binary classification problem, we will use sigmoid as the activation function.
VII. Dense layer implements
VIII. ***output = activation (dot(input, kernel) + bias)***
IX. kernel is the weight matrix. kernel initialization defines the way to set the initial random weights of Keras layers.
X. **Random normal initializer generates tensors with a normal distribution**.
XI. For uniform distribution, we can use Random uniform initializers.
XII. Keras provides multiple initializers for both kernel or weights as well as for bias units.



REFERENCES

I. https://www.investopedia.com/terms/n/neuralnetwork.asp
II. https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72
III. https://medium.com/datadriveninvestor/building-neural-network-using-keras-for-classification-3a3656c726c1
IV. https://www.kaggle.com/c/titanic
V. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
VI. https://keras.io/getting-started/sequential-model-guide/