Assignment 6: Data Lab, Support Vector Machine

Ilangg Kumaran M
EP18B007
ep18b007@smail.iitm.ac.in
Indian Institute of Technology Madras

***Abstract*—Support Vector Machine (SVM) is a supervised machine learning algorithm capable of performing classification, regression and even outlier detection. There are specific types of SVMs you can use for particular machine learning problems, like support vector regression (SVR) which is an extension of support vector classification (SVC). However, it is mostly used in classification problems. In this essay we use Support Vector classification to build a predictive model on a given problem. And evaluating the performance of the model. And make predictions on unlabelled data.**

I. Introduction

The linear SVM classifier works by drawing a straight line between two classes. All the data points that fall on one side of the line will be labeled as one class and all the points that fall on the other side will be labeled as the second.

SVMs are different from other classification algorithms because of the way they choose the decision boundary that maximizes the distance from the nearest data points of all the classes. The decision boundary created by SVMs is called the maximum margin classifier or the maximum margin hyper plane. The SVM classifier is a frontier that best segregates.

This technique is applied on the given data to automatically label pulsar candidates. Which are a rare type of Neutron star.

This essay discusses on key principles underlying Support Vector Machine classifiers, data used for the problem, the model developed and observations.

II. Support Vector Machine

In classification, the goal of the predictive model is to identify the class that generated a particular instance. Consider such an instance $x \in R_N$, a vector consisting of N features, $X = [x_1, x_2, ..., x_k]$.We need to assign it to one of the M classes $C_1, C_2, C_3....C_M$ depending on the values of the N features. In the SVM algorithm, we plot each data item as a point in N-dimensional space with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-planes that differentiates the classes very well.

A. Types of SVM

SVM can be of two types:

- Linear SVM: Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
- Non-linear SVM: Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

B. Hyperplanes

hyperplane is a line that linearly separates and classifies a set of data. Further from the hyperplane our data points lie, the more confident we are that they have been correctly classified. We therefore want our data points to be as far away from the hyperplane as possible, while still being on the correct side of it. when new testing data is added, whatever side of the hyperplane it lands will decide the class that we assign to it.

$$\theta^T X \geq 1 \quad if\ y^{(i)} = 1 \tag{1}$$

$$\theta^T X \leq -1 \quad if\ y^{(i)} = 0 \tag{2}$$

C. Margin

Nearest points from the optimal decision boundary that maximize the distance are called support vectors.The region that the closest points define around the decision boundary is known as the margin. That is why the decision boundary of a support vector machine model is known as the maximum margin classifier or the maximum margin hyperplane.

D. Support Vector Machine algorithm

In the SVM algorithm, we are looking to maximize the margin between the data points and the hyperplane. The loss function that helps maximize the margin is hinge loss.

$$min_\theta C \sum_{i=1}^{m} [y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)})] + \frac{1}{2} \sum_{j=1}^{n} \theta_j^2 \tag{3}$$

If $y = 1$, $cost_1(z) = 0$ only when $z \geq 1$ and If $y = 0$,$cost_0(z) = 0$ only when $z \leq -1$. The cost is 0 if the predicted value and the actual value are of the same sign. If they are not, we then calculate the loss value. We also add a regularization parameter the cost function. The objective of the regularization parameter is to balance the margin maximization and loss. After adding the regularization parameter, the cost functions looks like (3). Now that we have the loss function, we take partial derivatives with respect to the weights to find the gradients. Using the gradients, we can update our weights.

E. *Advantages of Support Vector Machine Algorithm*

- It works very well with limited datasets.
- Kernel SVM contains a non-linear transformation function to convert the complicated non-linearly separable data into linearly separable data.
- It is effective on datasets that have multiple features.
- It employs a subset of training points in the decision function or support vectors, making SVM memory efficient.
- Apart from common kernels, it is also possible to specify custom kernels for the decision function.

F. *Model Evaluation*

Accuracy is the basic evaluation metric that measures how correct our predictions were. In this case we simply compare predicted labels to true labels and divide by the total. Further we construct a confusion matrix which is a 2x2 matrix.

TABLE I
CONFUSION MATRIX

	Target Positive	***Target Negative***
Model positive	a	b
Model negative	c	d

$$accuracy = \frac{(a+d)}{(a+b+c+d)} \tag{4}$$

- **Precision** : the proportion of positive cases that were correctly identified.

$$precision = \frac{a}{(a+b)} \tag{5}$$

- **Recall** : the proportion of actual positive cases which are correctly identified.

$$recall = \frac{a}{(a+c)} \tag{6}$$

If we are trying to get best precision and recall at the same time for our model, we use F1 score which is the harmonic mean of precision and recall.

$$F1 = 2\frac{(precision \times recall)}{(precision + recall)} \tag{7}$$

$$\text{sensitivity} = \frac{a}{(a+c)}$$

$$\text{specificity} = \frac{d}{(b+d)}$$

III. DATA

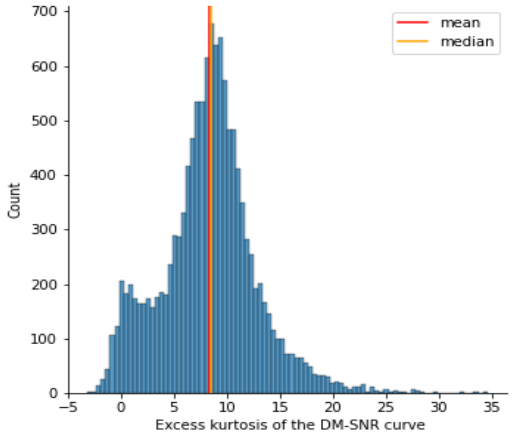
Two data sets were provided, one with data to train the model and a test dataset to make predictions. Data contains information about Neutron stars. Each candidate is described by 8 continuous variables and a single class variable. The first four are simple statistics obtained from the integrated pulse profile (folded profile). This is an array of continuous variables that describe a longitude-resolved version of the signal that has been averaged in both time and frequency. The remaining four variables are similarly obtained from the DM-SNR curve. The data contains few null values. Train data is labelled and the test data does not contain labels.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12528 entries, 0 to 12527
Data columns (total 9 columns):
 #   Column                                        Non-Null Count  Dtype
---  ------                                        --------------  -----
 0    Mean of the integrated profile               12528 non-null  float64
 1    Standard deviation of the integrated profile 12528 non-null  float64
 2    Excess kurtosis of the integrated profile    10793 non-null  float64
 3    Skewness of the integrated profile           12528 non-null  float64
 4    Mean of the DM-SNR curve                     12528 non-null  float64
 5    Standard deviation of the DM-SNR curve       11350 non-null  float64
 6    Excess kurtosis of the DM-SNR curve          12528 non-null  float64
 7    Skewness of the DM-SNR curve                 11903 non-null  float64
 8   target_class                                  12528 non-null  float64
dtypes: float64(9)
memory usage: 881.0 KB
```

Fig. 1. Features in data and data types

IV. THE PROBLEM

Pulsars are a rare type of Neutron star that produces radio emissions detectable here on Earth. The task of the model is to Predict if a star is a pulsar start or not.



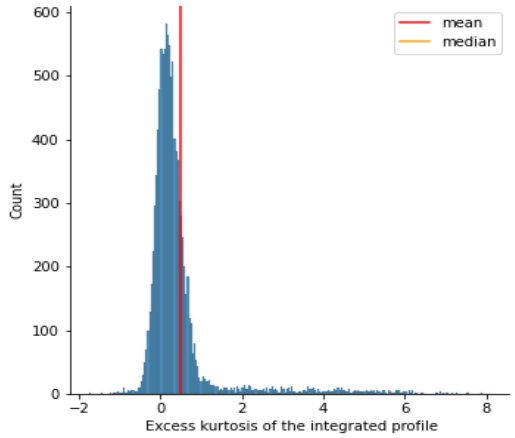
A. *Data Pre-processing*

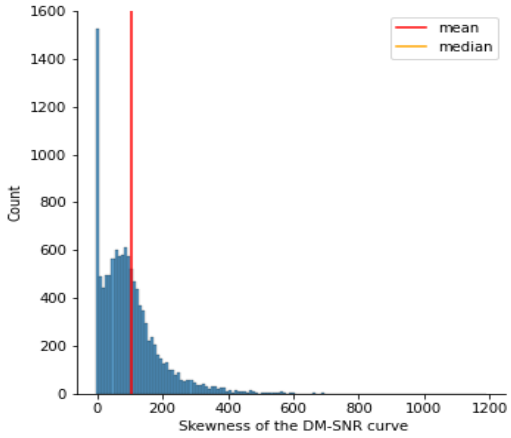
Since the data contains some missing values, these values are imputed with the mean values of the particular columns. Support Vector Machine algorithms are not scale invariant, all the values are standardize to have mean 0 and variance 1 before training.

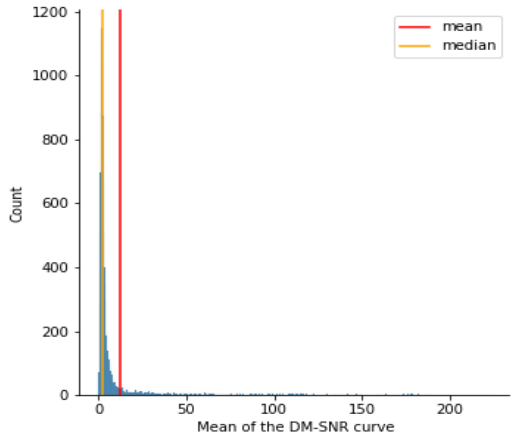
The data is split in the ration of 7:3 for training and testing, it is split in a stratified manner making sure there is same ratio of positive and negative outcomes in training data and testing data.

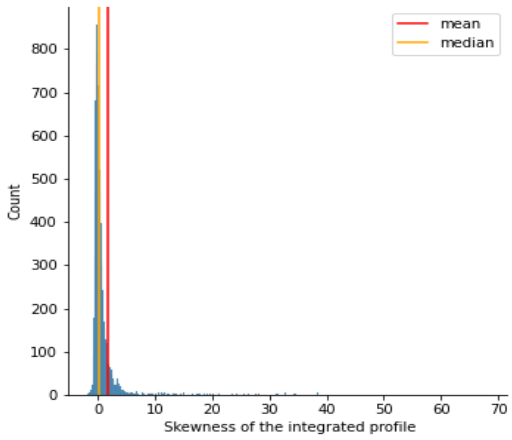
B. *Support vector machine classifier model results*

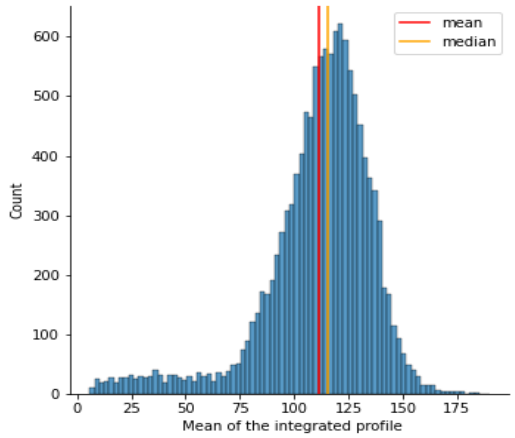
Support vector machine classifier model was trained on training data and an accuracy of 0.98 was achieved on testing

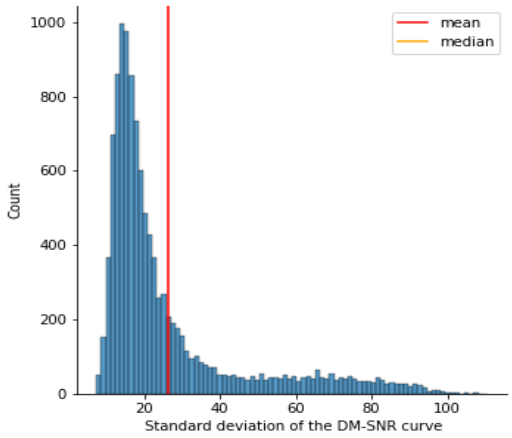

mean
median
Count
Excess kurtosis of the integrated profile

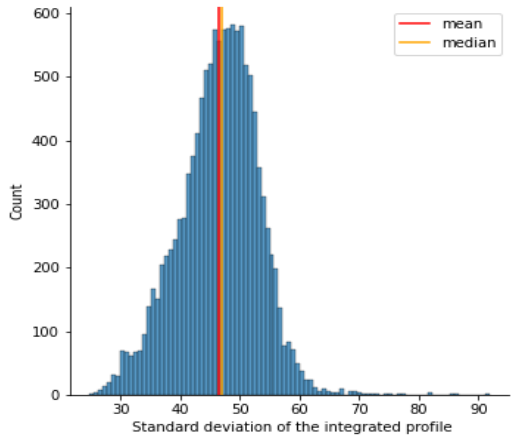

mean
median
Count
Skewness of the DM-SNR curve

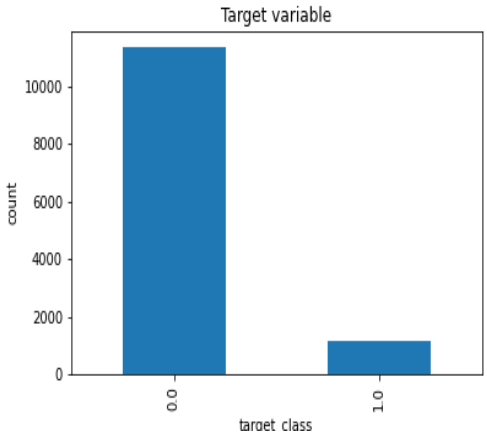

mean
median
Count
Mean of the DM-SNR curve


mean
median
Count
Skewness of the integrated profile


mean
median
Count
Mean of the integrated profile


mean
median
Count
Standard deviation of the DM-SNR curve





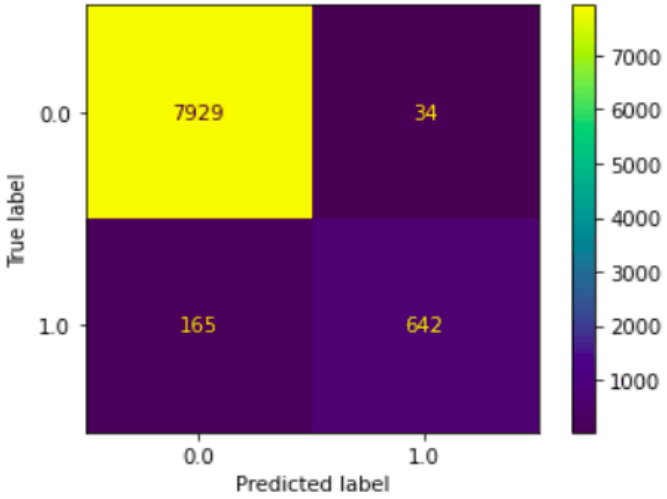


Fig. 2. confusion matrix.

	precision	recall	f1-score	support
0.0	0.98	1.00	0.99	7963
1.0	0.95	0.80	0.87	807
accuracy			0.98	8770
macro avg	0.96	0.90	0.93	8770
weighted avg	0.98	0.98	0.98	8770

Fig. 3. Model summary

data. The confusion matrix obtained from the ground truth values and predicted values is presented in Fig.

V. Conclusions

Support vector machine classifier model was trained on the given training data and the model was used to make predictions on the given test data. The number of predictions per each class can be seen in Fig. . Support vector machine classifier model was very effective for this problem and had a very good accuracy. Models with more complex decision boundary can be investigated to further increase the accuracy.

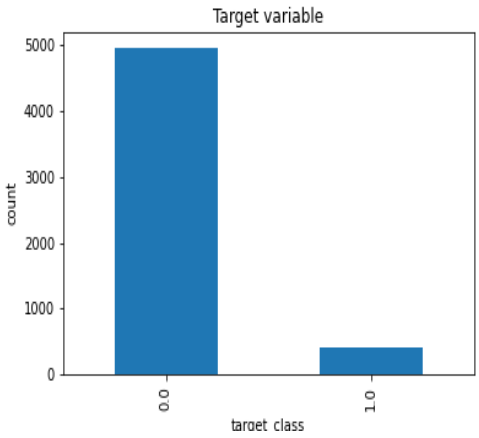


Fig. 4. Predicted classes

References

[1] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.h
[2] https://towardsdatascience.com/data-visualization-with-pandas-1571bbc541c8.
[3] https://www.ritchieng.com/machine-learning-evaluate-classification-model/.
[4] Müller, Andreas and Sarah Guido, Introduction to machine learning with Python: a guide for data scientists", 2016.
[5] https://towardsai.net/p/programming/decision-trees-explained-with-a-practical-example-fe47872d3b53.
[6] https://web.stanford.edu/ jurafsky/slp3/5.pdf.
[7] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html.