A Mathematical Essay on Support Vector Machine

Vignesh Kumar S
Department of Chemical Engineering
Indian Institute of Technology, Madras
Email address: ch18b118@smail.iitm.ac.in

***Abstract*—The purpose of this article is to understand the Support Vector Machine algorithm and the mathematics behind it. In this work, we demonstrate the application of the Support Vector classifier illustrated through Pulsar neutron star database. Due to the complex nature and the abstractness involved in identifying a pulsar star given its radio emission pattern, it is often difficult to obtain a good solution through manual approaches. Adopting a machine learning technique can help in obtaining insights and analysing different features in a large amount of data comprehensively. We try to establish a nexus to identify potential candidates for pulsar star given periodically obtained radio signal features.**

I. Introduction

In this paper, we will study Support Vector Machine, a technique used to analyze and model certain class, usually a dichotomous outcome. We will use this technique to automatically label pulsar candidates given statistics related to Integrated and DM-SNR profiles to facilitate rapid analysis. This analysis would aid in obtaining insights among a large dataset and understand the trend behind the possibility of identifying a neutron pulsar star.

The classification problem attempts to establish a relationship between a categorical target variable with one or more explanatory variable(s). Support Vector Machine follows discriminant modelling approach, which learns a function that maps each input to a class label directly from training data instead of modelling posterior probabilities like logistic regression or Naive Bayes classifier. This in general, is well suited for describing and testing hypotheses about features that could contain outliers. A Binary classifier is constructed by constructing a decision boundary and classifying inputs to classes based on the side it lies with respect to hyperplane.

The dataset used for this problem comprises mean, standard deviation, excess kurtosis and skewness of Integrated and DM-SNR profiles. We use Support Vector classifier to learn and predict pulsar candidates given these input features.

This work represents the concepts behind Support Vector Machine and evaluation metrics involved. We then use this technique to establish the relationship to predict the nature of a star it may belong to.

II. Support Vector Machine

The goal in classification problem is to take an input feature $\boldsymbol{x}$ and assign it to one of the two classes. The input space is thereby divided into decision regions whose boundaries are called decision surfaces. Support Vector Classifier without any extension is a linear classification model, which means the decision surfaces are linear functions of the input vector $\boldsymbol{x}$. However, with use of kernel tricks, a non-linear decision boundary could very well be obtained for enhanced use cases.

The support vector machine is a generalization of a simple and intuitive classifier called the maximal margin classifier, But this requires the dataset to be linearly separable which most datasets don't satisfy and cannot be applied. We then extend it to non-separable cases with this ideology and also introduce kernel tricks which enables a sophisticated way to model non-linear boundaries.

A. Maximal Margin Classifier

A hyperplane is geometry is defined as a subspace that is one dimension less than of its ambient space. Intuitively, one can think it as an extension of a line in a x-y plane or a plane in x-y-z dimensions to p dimensions.

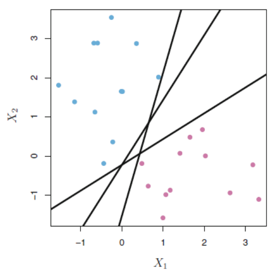
$$f(\boldsymbol{X}) = b + w_1X_1 + .. + w_pX_p = 0$$

The above represents a p-dimensional hyperplane. Suppose we have n datapoints and p features, linearly separable with classes $y_i = \{+1, -1\}$, we want:

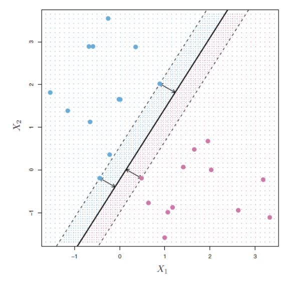
$$f(\boldsymbol{X}) > 0 \rightarrow R_1 \quad if\, y_i = +1$$
$$f(\boldsymbol{X}) < 0 \rightarrow R_2 \quad if\, y_i = -1$$

We can rewrite this into single equation as:

$$y_i f(\boldsymbol{X}) > 0$$
$$y_i(b + w_1x_1 + .. + w_px_p) > 0$$



(a) Infinite choices available to pick a decision boundary



(b) The Maximal Margin classifier

A natural choice when infinite decision boundaries are possible is to pick the one which maximizes the margin, which is the perpendicular distance between the nearest datapoint.This gives a generalization of a simple and intuitive classifier called the maximal margin classifier.

The given figure has 3 ”support vectors”. They *support* the maximal margin hyperplane in that sense that if these points are moved slightly, then the hyperplane would move as well. Moving other observations would not affect the hyperplane, provided that the observations does not cause it to cross the boundary set by the margin.

The original problem formulation could be written as:

$$\max_{w,b} \min_{i \in n} \frac{|w^T x_i + b|}{||w||}$$
$$s.t.\ sign(w^T x_i + b) = y_i$$

We could scale it such that the nearest point is at a unit distance from hyperplane and the maximization of $1/||w||$ could be re-written as minimization of $||w^2||$.

Hence,

$$\min_{w,b} \frac{1}{2}||w||^2$$
$$s.t.\ y_i(w^T x_i + b) \geq 1$$

B. Soft-Margin Classifier

As stated earlier, the maximal margin classifier is not the way to go when the dataset is inseparable or even in the case of large datapoints, the margin would be very small and wouldn’t allow any slackness. In order to obtain a greater robustness to individual observations, we aim to classify most of the training observations correctly and not every single one of them. This is the principle behind support vector classifiers a.k.a the softmargin classifier.

The margin is called soft now because it can be violated by some of the training observations. Rather than seeking the largest possible margin so that every observation is not only on the correct side of the hyperplane but also on the correct side of the margin, we instead allow some observations to be on the incorrect side of the margin or even on the incorrect side of the hyperplane.

The points that fall on the incorrect side of the hyperplane are misclassified. These points together with those that fall within and on the margin are the support vectors of the classifier and influence the decision boundary.

Let’s introduce a variable ε_i, slackness, to capture this effect. If a point i falls outside the margin, ε_i = 0 and and $\varepsilon_i > 0$ for support vectors. The soft-margin problem could be written as:

$$\min_{w,b} \frac{1}{2}||w||^2 + C\sum_{i=1}^{n} \varepsilon_i$$
$$s.t.\ y_i(w^T x_i + b) \geq 1 - \varepsilon_i$$
$$\varepsilon_i \geq 0$$

This makes solution feasible for dataset which aren’t linearly separable. Note that $\varepsilon_i > 1$ for misclassified point and $0 < \varepsilon_i \leq 1$ for points lying within the margin and on the correct side of the hyperplane and $\varepsilon_i = 0$ for the rest of the points.

C. Role of C

C determines the severity of the violations to the margin. If C$\rightarrow\ \infty$, the cost associated for misclassification is very high and there is no room for misclassification, hence results in hard-margin classifier. If C is very less, we allow more observations to violate the margin and also misclassifications.

In practice, C is treated as tuning parameter that is chosen via cross validation. When C is large, we seek narrow margins that are rarely violated. This amounts to a classifier that is highly fit to data, hence may have low bias and high variance. On the other hand, when C is small, the margin is wider, we allow more violations to it. This amounts ot fitting the data less hard and obtaining a classifier that is more potentially biased but lower variance. Hence C controls bias-variance trade-off.

D. Solution to optimization problem

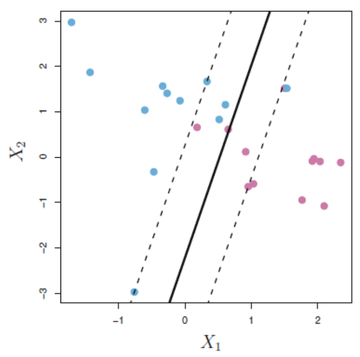
The Lagrangian of the above optimization problem in the primal space can be written as:

$$L(w, b, \varepsilon, \alpha, \beta) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{n} \varepsilon_i + \sum_{i=1}^{n} \beta_i(-\varepsilon_i)$$
$$+ \sum_{i=1}^{n} \alpha_i(1 - \varepsilon_i - y_i(w^T x_i + b))$$

where α_i and β_i are lagrange multipliers and hence are ≥ 0. It seems that the above problem is easier to solve in dual space by applying minimax theorem whose details we won’t delve into in this work. But the result of this optimization is interesting. The decision function obtained is:

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i < x, x_i >$$

where $< x, x_i >$ is dot product and $\alpha_i \neq 0$ for support vectors only. Since the decision boundary is only dependent on support vectors, SVM is quite robust to behavior of observations that are far away (outliers) from the hyperplane unlike methods like LDA. (Linear Discriminant Analysis).



(c) Example of a soft margin classifier

E. The Kernel Trick

Also note that the decision function depends on dot product of support vectors only. We can rewrite the decision function as:

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i K(x, x_i)$$

where K could represent the dot product. The intuition behind this is that we now would be able to transform X to a higher dimensional subspace and no longer need to remember that transformation but only interested in the dot product function of the resulting vector. Hence, We call this resulting function K as Kernel, which essentially is a transformation to a higher dimensional subspace but without needing to remember the transformation. This is advantageous computationally. The following are popular choices of kernel:

- $K(x_i, x_j) = x_i^T x_j$ Linear Kernel (no transformation)
- $K(x_i, x_j) = (1 + x_i^T x_j)^d$ Polynomial Kernel (Tranformation into polynomial space of degree d)
- $K(x_i, x_j) = exp(-\gamma||x_i - x_j||^2)$ RBF Kernel. (Transformation into Hilbert Space)

F. RBF Kernel

Given a test observation $x^* \equiv (x_1^*, .., x_p^*)^T$ is far from x_i, then $K(x_i, x^*) = exp(-\gamma||x_i - x^*||^2)$ will be small.

Training observations that are far from x^* will essentially play no role in the predicted class label for x^*. This means that the radial basis kernel has very local behaviour. The farther the points from support vectors, the tiny the role it would play in f(x*). Hence, as γ is large, it makes very sharp turn (a.k.a) overfits.

G. More than 2 classes

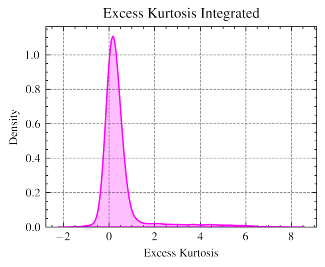
Unlike logistic regression, there is no natural extension to more than 2 classes that is possible. There are two roundabouts to deal with multiclass problem when we deal through SVM. The first is One versus One where for K classes, we construct $\binom{K}{2}$ classifiers for each pair. Then the final result is obtained through majority voting.

The second is by one versus all. We construct K SVM's where we label each class against (K-1) classes. We assign the observation to the class for which the signed distance from the hyperplane is largest as this amounts to a high level of confidence that the test observation belongs to the k^{th} class rather than to any of the other classes.

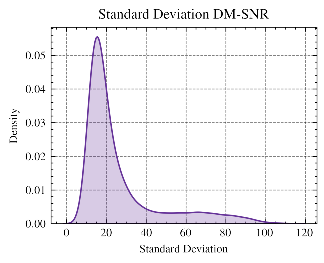
III. The Problem

In this section, We will analyse the Pulsar star dataset and try to predict candidates for pulsar star based on the information available. The support vector classification technique is used as a classification model in further analysis.

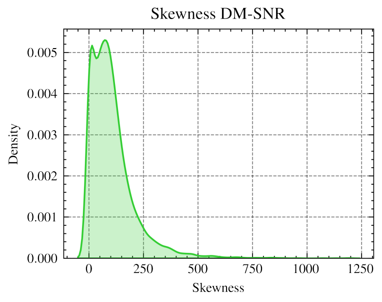
The dataset consists of statistics (Mean, standard deviation, excess kurtosis, skewness) of integrated and DM-SNR profile. Excess Kurtosis is a meaure of how fat a distribution's tail is when compared to the center of the distribution. Skewness is the degree of asymmetry observed in a probability distribution that deviates from the symmetrical normal distribution. Let's start by addressing any potential missing values.



(d)

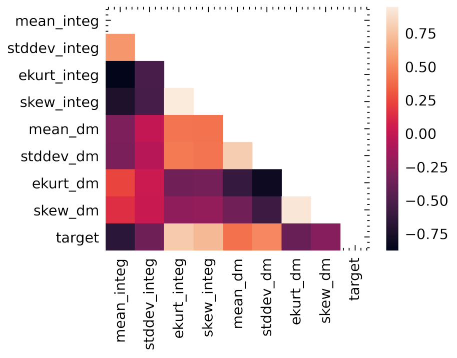


(e)



(f)

The missing values are present in only the above three features and looking at these plots, the distributions of these features possess high tails, no clear patterns between mean, median/mode.

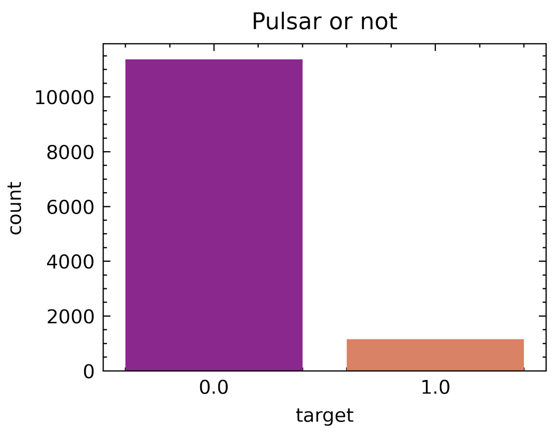


(g)

There is a fair amount of correlation between features and we could use this fact to impute the features instead of simple mean/median imputation. We hence use linear regression based imputer to fill the missing values. The descriptive statistics of both the imputed data and original data resembles (not shown due to space constraints) and hence this is a better strategy than constant imputation.

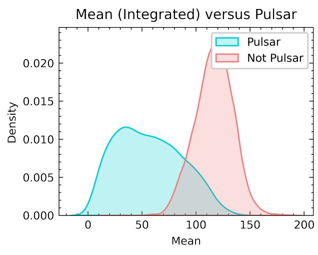
A. Exploratory Data Analysis

The given data is highly imbalanced. It only contains 9.2 % of pulsar stars and the remaining 90.8 % the non-pulsar ones.

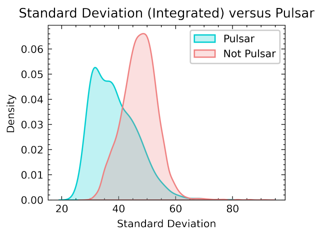


This is taken care of by setting class weight to balanced while training.

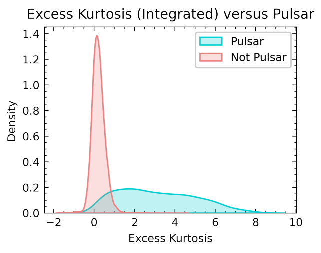
1) Integrated Profiles: We can generally say that mean and standard deviation of non pulsar inputs are high and excess kurtosis and skewness is generally lower in integrated profiles compared to pulsar stars.



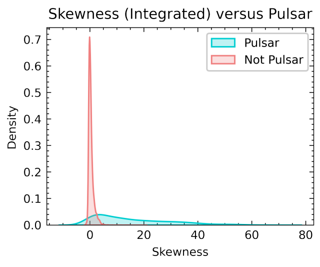
(h)



(i)

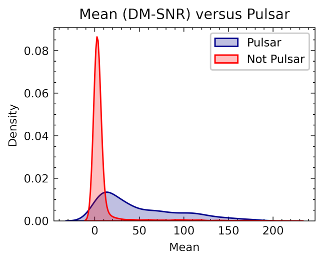


(j)

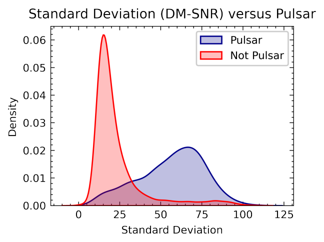


(k)

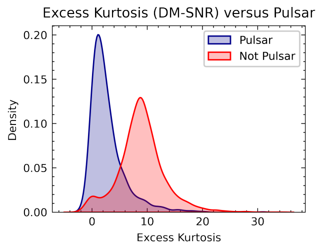
2) DM-SNR Profiles: We can generally say that mean and standard deviation of non pulsar inputs are lower and excess kurtosis and skewness is generally higher in DM-SNR profiles compared to pulsar stars.



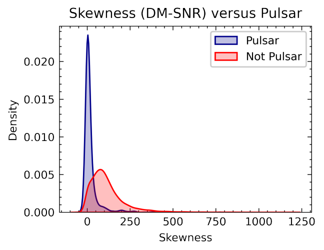
(l)



(m)



(n)



(o)

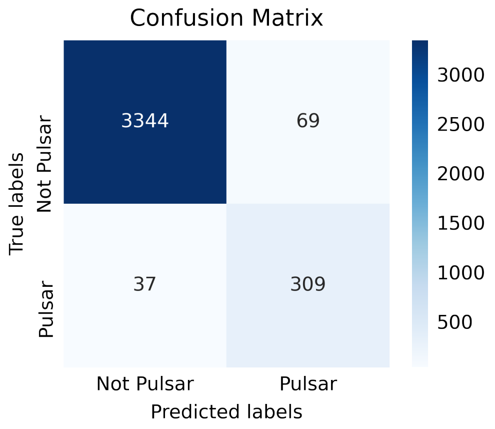
B. Data Preprocessing and Train test split

For SVM, data scaling is very vital as the distance from margin is sensitive to the scale of the feature used. We are going to do standardization here i.e, make the mean 0 and variance 1 of all the features. Also, since the dataset is imbalanced, stratified split is done to preserve the imbalancedness in train test split.

C. Model Evaluation

After data-preprocessing, a support vector machine classifier was grid searched between linear and rbf kernels on the training dataset with 5 folds of cross validation to maximise *f1-score*. This resulted in a best classifier with *rbf* kernel with C=3.703 and Gamma=0.0078. The classifier was trained on nearly 9000 training data points but had only a total of 15% (1488) support vectors implying only those many influenced the nature of decision boundary.

The following confusion matrix was obtained on the test/unseen dataset.



True \ Predicted	Not Pulsar	Pulsar
Not Pulsar	3344	69
Pulsar	37	309

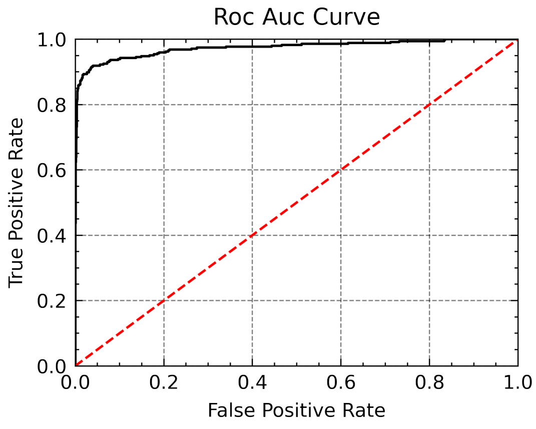
The classifier seems to perform good on identifying a pulsar dataset without many misses. We could see that it was unable to detect only around 10% of them (37) though it incorrectly classified non-pulsar as pulsar for around 69 inputs. This could be tuned depending upon the knowing the real cost associated with misclassification but as of now this seems good enough.

In a classification setting, accuracy may not be the best score to consider especially in a skewed class situation. The training data in our case is imbalanced. Precision is the ratio of correct positive predictions to the total positive predictions of our model. The recall is the ratio of positive instances that are correctly detected by our classifier. The F1 score is simply a harmonic average of these two.

The following metrics are reported for the best classifier on the test set.

Support Vector Classifier	
Metric	Score
Accuracy	0.9718
Precision	0.8174
Recall	0.8930
F1 score	0.8535

ROC curve is another common tool used with binary classifiers. The ROC curve plots the true positive rate (another name for recall) against the false positive rate. The FPR is the ratio of negative instances that are incorrectly classified as positive. It is equal to one minus the true negative rate, which is the ratio of negative instances that are correctly classified as negative. The TNR is also called specificity. Hence the ROC curve plots sensitivity (recall) versus 1 – specificity.



The dotted line represents the ROC curve of a purely random classifier; a good classifier stays as far away from that line as possible (toward the top-left corner). One way to compare classifiers is to measure the area under the curve (AUC). A perfect classifier will have a ROC AUC equal to 1, whereas a purely random classifier will have a ROC AUC equal to 0.5. The area under the ROC curve of our Support Vector classifier is 0.9738.

Note that the standard ROC Curve requires varying the probability of score threshold of a classifier and obtaining the corresponding graph of the ordered pairs (TPR,FPR) for each varied threshold. But since SVM is defined in a such a way(Discriminant approach) that it doesn't produce probability, we can approximate the type of score by computing signed distance between the hyperplane. So the above is only an estimation and isn't the true nature of ROC curve. More accurate estimate could be obtained through Relevance Vector Machines which uses Bayesian methods to infer posterior probabilities of which isn't a particular interest in this specific work.

IV. Conclusions

Based on our analysis, the key takeaways we had from this exercise are the following:

1) Support Vector Machines are robust algorithms less sensitive to outliers and hence an enhancement of the classifiers we have studied so far.
2) Unlike most classifier algorithms, only a small percentage of dataset influence the decision boundary, called the support vectors. These improve the performance and lying farther away from margin can be yielded as a confidence metric.
3) Usage of Kernels is a clever way to search in higher dimensional space without increasing computational complexity and could potentially fit the data better.

References

[1] Aurelian Geron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools and Techniques to Build Intelligent Systems, pp. 191–214
[2] Christopher M. Bishop, Pattern Recognition and Machine Learning
[3] An Introduction to Statistical Learning, Gareth James et al. pp.337–358