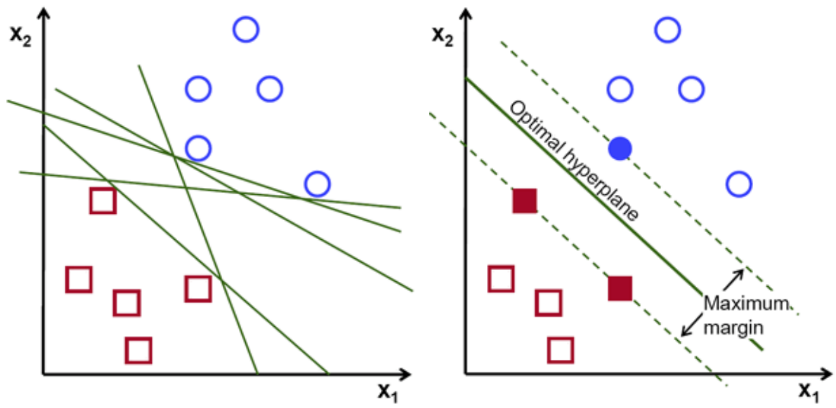
ESE 533: Final Report

Riya Patel 111512429
Siddarth Somani 114143567

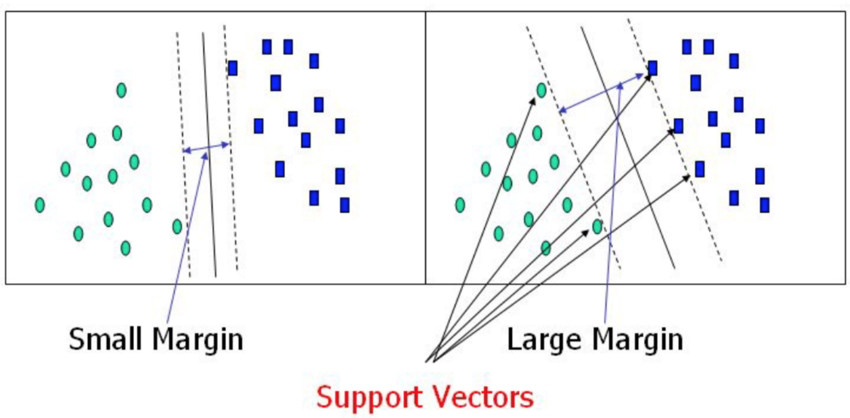
Topic: Support Vector Classification with Positive Homogenous Risk Functionals

Introduction

Support Vector Machines, or SVMs are utilized for classification and regression projects. The SVM algorithm allows one to determine the hyperplane in an N-dimensional space, (when N is the number of features) that differentiates the data points.



The goal is to determine a plane with maximum margin, or in other words the greatest distance between data points of both classes. Maximizing this allows for future data points to be classified more accurately.



The support vectors are seen in the above figure: they are the data points closer to the hyperplane. They sway the hyperplane's position and orientation. These support vectors allow us to maximize the classifier's margin. These points will eventually help build the SVM.

The objective is to maximize the margin between the data points and the hyperplane in the SVM algorithm.

Types of SVMS

We will be discussing three types of SVMs: the standard or the classical SVM, the C-SVM, and the *v*-SVM.

- The standard SVM has a distance from hyperplane φ(x)w+b= 0 to origin:b/||w||. The margin between classes is (b2-b1/||w||). The standard SVM is infeasible if the training data is not linearly separable.
- The C-SVM follows the following formula:

$$\min_{w,b,\xi} \left(\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi_i \right)$$
$$\text{s.t.} \;\; y_i(\phi(x_i)\cdot w + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \qquad i = 1, \ldots, n,$$

-

  The tradeoffs for the parameter C include separation error and value of margin (generalization ability).

- The *v*-SVM follows the following formula:

$$\min_{w,b,\rho} \left(\frac{1}{2}\|w\|^2 - \nu\rho + \frac{1}{n}\sum_{i=1}^{n} [\rho - y_i(\phi(x_i)\cdot w + b)]_+ \right)$$
$$\text{s.t.} \;\; \rho \geq 0,$$

  The problem here is that *v* is not able to take the whole range [0,1].

VaR & CVaR

CVaR, or conditional value at risk, is a derivative of the value at risk. CVaR is basically the expected shortfall. CVaR provides a more conservative approach in regards to risk exposure compared to VaR. CVaR is most often used for portfolio optimization for efficient risk management and it is used to address the defects of the VaR model. The VaR model provides a worst case loss linked to a probability and time horizon while CVaR is the expected loss in case that the worst case scenario is crossed. The CVaR formulates the expected losses past the VaR breakpoint. The CVaR values are derived from VaR's calculations:

$$CVaR = \frac{1}{1-c}\int_{-1}^{VaR} xp(x)\,dx$$

where:

$p(x)dx =$ the probability density of getting a return with value "x"

$c =$ the cut-off point on the distribution where the analyst sets the VaR breakpoint

$VaR =$ the agreed-upon VaR level

CVaR can be written as a minimization problem using ν-SVM:

$$CVaR_\alpha(X) = \min_{C\in R}\{C + \frac{1}{1-\alpha}E[X-C]_+\}$$

Substitute Expectation by Average:

$$CVaR_\alpha(X) = \min_{C\in R}\{C + \frac{1}{(1-\alpha)n}\sum_{i=1}^{n}[X_i - C]_+\}$$

Substitute: $C = -\rho,\ 1-\alpha = \nu,\ X_i = -y_i[\phi(x_i)w + b]$

Notice that we get ν-SVM:

$$\min_{w,b,\rho}\left(\frac{1}{2}\|w\|^2 + \nu\{-\rho + \frac{1}{\nu n}\sum_{i=1}^{n}[-y_i(\phi(x_i)\cdot w + b) - (-\rho)]_+\}\right)$$

Furthermore, the CVaR-representation of ν-SVM is:

$$\min_{w,b}\left(\frac{1}{2}\|w\|^2 + \nu\, CVaR_\nu\big(-y(\phi(x)\cdot w + b)\big)\right)$$

Data Collection and Portfolio Safeguard

PSG - Portfolio Safeguard (PSG) is an optimization package for solving nonlinear and mixed-integer nonlinear optimization problems in Windows and Linux. It was selected as it can implement techniques to optimize various risk functions, including CVAR, using simple lines of code.

PSG code to minimize our function

```
minimize
0.5*quadratic(matrix_quadratic)
+0.5*var_risk(0.5,matrix_data)
Box: >= -1000.0, <= 1000.0
```

matrix_data - matrix containing all the training data

The code is uploaded to PSG using .txt file and after running the code, we get our results in a .txt file
We collected data regarding the Indian diabetes dataset
(https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database) to use it for analysis.
The collected data was passed to portfolio safeguard using the above equation to generate the set of optimal weights. The results were tested using cross validation using python

Cross Validation in Machine Learning

Cross validation is an invaluable tool for data scientists that provides a way to test machine learning models. It allows scientists and engineers to evaluate their performance on an independent test data set. It involves reserving specific samples of a dataset that the model is not trained on. Next, the model is tested on this sample for evaluation. Cross validation protects a model from overfitting, especially when the data is limited.

K-Fold Cross validation has the data divided into k subsets. The handout method is repeated k times and with each time one of the k subsets is used as the test set and the other k-1 subsets are put together to form a training set. Overall, the error estimation is the mean of all k trials to obtain the total effectiveness of the model. The results will demonstrate every data point gets the chance to be in a validation set once and gets to be in a training set k-1 times. Consequently, there is less bias because most of the data is being used for the fitting. Additionally, there is less variance since a majority of the data is also being used in validation sets. We have utilized cross validation to examine our model performance.

Conclusion

We studied the theory behind support vector machines(SVM), as well as the working of various forms of support vector machines including C-SVM and nu-SVM. We understood the mathematical formulation of risk functions, including VAR(Value at Risk) and CVAR(Cumulative Value at Risk). We understood how CVAR minimization is equivalent to minimization of nu-SVM.

To apply our understanding, we tested our models using Portfolio safeguard and Python on Indian diabetes dataset. We observed an out of sample accuracy of 65%. Some of the ways to improve our results are

1) Using a larger dataset(more rows and columns)
2) Instead of optimizing CVAR, we can optimize other metrics like VAR, worst case loss or expected loss, which in turn are equivalent to other versions of SVM’s

References

Learning, UCI Machine. “Pima Indians Diabetes Database.” *Kaggle*, 6 Oct. 2016, https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database.

Tsyurmasto, Peter, et al. “Support Vector Classification with Positive Homogeneous Risk Functionals.” *Journal of Machine Learning*, 2009, pp. 183–192., https://doi.org/10.1002/9780470503065.ch11.