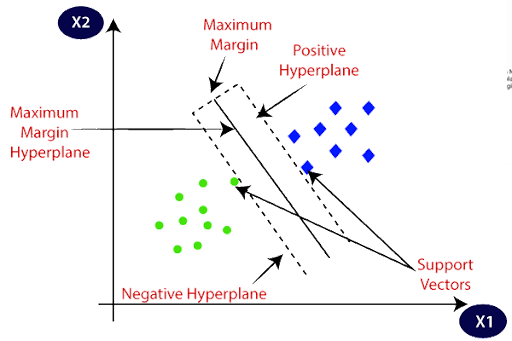
1. **What are hard margin and soft Margin SVMs?**

We maximize the distance between the line and the support vectors in both the soft and hard margins. The difference between them lies in the divisibility of the data. If our data is clearly linearly separable, we go for a hard margin. However, if this is not the case, to allow and penalize misclassification, we use soft margin. When we have the data points that make it impossible to find a linear classifier, we would have to become more lenient with our approach and let some of our data points be misclassified. In this case, we opt for the soft margin SVM. There are times when the data is linearly separable, but the margin is so small that the model is prone to overfitting or sensitive to outliers. In this case as well we can opt for a larger margin by using soft margin SVM to help the model generalize better. In a real-life scenario, data is hardly linearly separable. We update our SVM function so it may skip few outliers and be able to classify almost linearly separable points, therefore, to fix this soft SVM has a new Slack variable ( ξ ), called Xi.

1. **Explain SVM to a non-technical person.**

SVM or Support Vector Machine is a algorithm for classification and regression problems. It solves linear and non-linear problems, SVMs can be kernelized, allowing them to represent complex nonlinear decision boundaries. The algorithm creates a hyperplane or a line that separates the data into two classes. A hyperplane is a plane of dimension one less than the dimension of data space, which divides the classes of data. We use it to enhance the generalization ability of the machine. Suppose we have a dataset as shown below, we need to classify the blue rectangles from the green dots(positives from the negatives). We need to find an ideal line that separates this dataset in two classes. Then we find the points closest to the line from both the classes, these points are called support vectors. Now, we figure out the maximum distance between the line and the support vectors. This distance is called the margin. Therefore, in the SVM algorithm we make a decision boundary in such a way that the separation between the two classes is as maximum as possible.



When the data is not linearly separable, cannot draw a straight line to classify the data, we convert the data to linearly separable data in higher dimensional feature space. We add one more dimension and call it z-axis. The coordinates on z-axis are denoted by the constraint,

z = x²+y². So, basically z co-ordinate is the square of distance of the point from origin. We get the following similar looking graph.

Chart, scatter chart

Description automatically generated

This space is costly, so we use the kernel method to solve these kind of issues

1. **What is the difference between Classification and Regression when using SVM?**

support vector classification (SVC) is a classifier and support vector regression (SVR) is a regressor. SVC works on categorical data , performs classification on discrete categorical labels. SVR works around ordinal data, performs regression, predicts continuous ordered variables.

There is a difference in slack variables used in the 2 techniques. SVM for classification has one slack variable to each training data point, SVM for regression has two slack variables for each training data point.

The optimization function for SVC:

Schematic

Description automatically generated

The optimization function for SVR:

Text, schematic

Description automatically generated with medium confidence

Support Vector Regression(SVR) works on minimizing the generalization error bound to achieve generalized performance, it doesn’t focus on minimizing the observed training error. The SVR is based on the computation of a linear regression function in a high dimensional feature space where the input data points are mapped through a nonlinear function.

1. **Do we require Feature Scaling for Decision Trees? Explain.**

In feature scaling we normalize the range of features in a dataset. In the real-world datasets, features vary in various degrees. Therefore, for ML models to interpret these features on the same scale, we perform feature scaling. Each node in decision trees (CART) model, represents a single feature in a dataset. Each node is split in a homogeneous way. This split is not affected by the other features in the dataset. Therefore, we can conclude that DTs don’t require scaling and are invariant to monotonic transformations, for the scale of the features.

1. **Briefly explain the properties of Gini Impurity.**

Gini Impurity tells us what the probability is of misclassifying an observation. Gini impurity index tells us the measure of diversity inside the dataset, it is used in training of Decision trees. If the Gini index number is higher than the diversity of the dataset is also higher. A Gini Impurity of 0 is the lowest and best possible impurity that a dataset can have, it can only be achieved when everything is on the same class. When training a decision tree, by maximizing the Gini Gain we get the best split, which can be calculated if we subtract the weighted impurities of the branches from the original impurity.

1. **What are the disadvantages of Classification and Regression Trees (CART)?**

CART also knows as decision trees are easy to work with, can be used for both -classification, and regression problems, but have its own limitations, such as follows:

* For complex data, sometimes calculations can be way more complex compared to the other algorithms.
* DTs are highly immutable, a small change in the data can cause a large change in the structure of the tree thus causing instability and wrong predictions.
* It is not suitable for large datasets. With large datasets it can become very complex and lead to overfitting.
* Often DTs take higher time to train the model and problem of overfitting. With large data, It generally leads to overfitting of the data which ultimately leads to wrong predictions. To fit the data, it keeps adding new nodes and ultimately, the tree becomes too complex to interpret. This way, it loses its generalization capabilities. It is suitable for trained data, but for the unseen data, it can be problematic.
* The Decision Tree algorithm is at a times inadequate for predicting continuous values. When working on continuous numerical variables, DT loses information when it tries to categorize variables in different categories.

1. **For the following table:**
   1. **What is the Gini index for the past trend?**
   2. **What is the Gini index for the trading volume?**

|  |  |  |  |
| --- | --- | --- | --- |
| Past Trend | Open Interest | Trading Volume | Return |
| Positive | Low | High | Up |
| Negative | High | Low | Down |
| Positive | Low | High | Up |
| Positive | High | High | Up |
| Negative | Low | High | Down |
| Positive | Low | Low | Down |
| Negative | High | High | Down |
| Negative | Low | High | Down |
| Positive | Low | Low | Down |
| Positive | High | High | Up |

Gini Index for Past Trend:

P(Past Trend=Negative): 4/10

P(Past Trend=Positive): 6/10

P(Past Trend = Positive & Return = Up) = 4/6

P(Past Trend = Positive & Return = Down) = 2/6

= 1 - ((4/6)^2 + (2/6)^2) = 0.45

P(Past Trend = Negative & Return = Up), = 0

P(Past Trend = Negative & Return = Down) = 4/4

= 1 - ((0)^2 + (4/4)^2) = 0

Gini Index for the Past Trend = (6/10)0.45 + (4/10)0 = 0.27

Gini Index for Trading Volume:

P(Trading Volume=Low): 3/10

P(Trading Volume=High): 7/10

P(Trading Volume = High & Return = Up) = 4/7

P(Trading Volume = High & Return = Down) = 3/7

= 1 - ((4/7)^2 + (3/7)^2) = 0.49

P(Trading Volume = Low & Return = Up) = 0

P(Trading Volume = Low & Return = Down)= 3/3

= 1 - ((0)^2 + (1)^2) = 0

Gini Index for the Trading Volume = (7/10)0.49 + (3/10)0 = 0.34