Support Vector Machine (SVM) Algorithm

A **Support Vector Machine (SVM)** is a powerful **machine learning algorithm** widely used for both **linear and nonlinear classification**, as well as **regression** and **outlier detection** tasks. SVMs are highly adaptable, making them suitable for various applications such as **text classification**, **image classification**, **spam detection**, **handwriting identification**, **gene expression analysis**, **face detection**, and **anomaly detection**.

SVMs are particularly effective because they focus on finding the **maximum separating hyperplane** between the different classes in the target feature, making them robust for both **binary and multiclass classification**. In this outline, we will explore the **Support Vector Machine (SVM)** algorithm, its applications, and how it effectively handles both **linear and nonlinear classification**, as well as **regression** and **outlier detection** tasks.

Table of Content

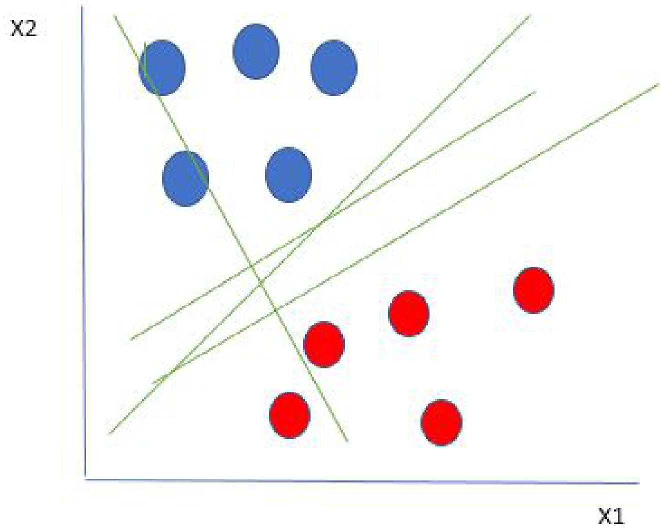
Support Vector Machine

A **Support Vector Machine (SVM)** is a supervised machine learning **algorithm** used for both **classification** and **regression** tasks. While it can be applied to regression problems, SVM is best suited for **classification** tasks. The primary objective of the **SVM algorithm** is to identify the **optimal hyperplane** in an N-dimensional space that can effectively separate data points into different classes in the feature space. The algorithm ensures that the margin between the closest points of different classes, known as **support vectors**, is maximized.

The dimension of the hyperplane depends on the number of features. For instance, if there are two input features, the hyperplane is simply a line, and if there are three input features, the hyperplane becomes a 2-D plane. As the number of features increases beyond three, the complexity of visualizing the hyperplane also increases.

Consider two independent variables, **x1** and **x2**, and one dependent variable represented as either a blue circle or a red circle.

- In this scenario, the hyperplane is a line because we are working with two features (**x1** and **x2**).
- There are multiple lines (or **hyperplanes**) that can separate the data points.
- The challenge is to determine the **best hyperplane** that maximizes the separation margin between the red and blue circles.

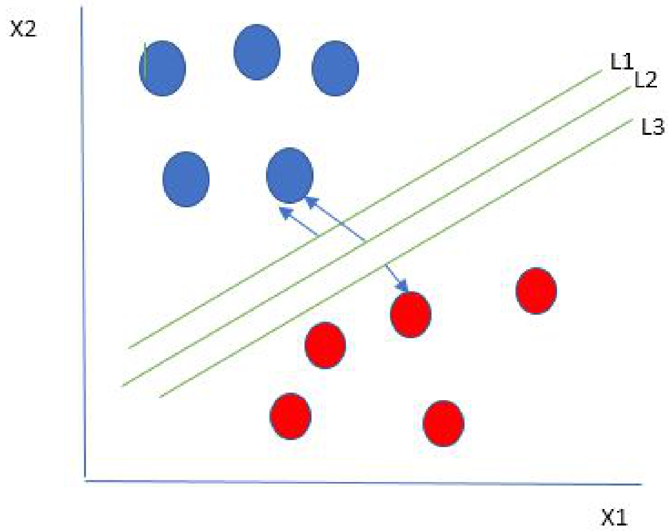


Linearly Separable Data points

From the figure above it's very clear that there are multiple lines (our hyperplane here is a line because we are considering only two input features x_1, x_2) that segregate our data points or do a classification between red and blue circles. ***So how do we choose the best line or in general the best hyperplane that segregates our data points?***

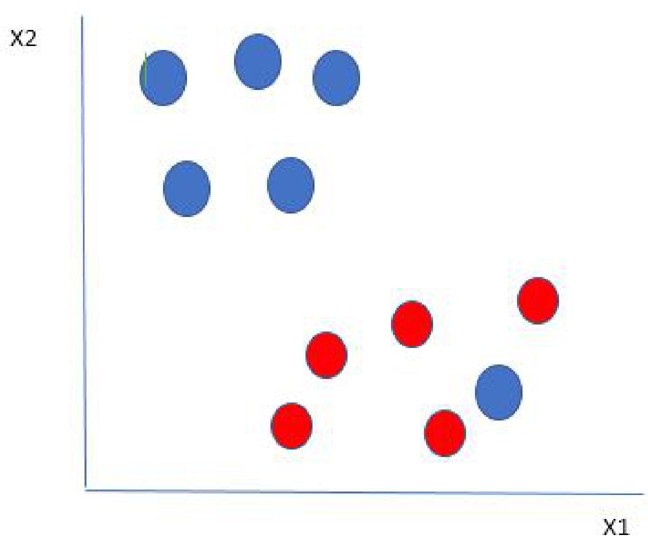
How does Support Vector Machine Algorithm Work?

One reasonable choice for the **best hyperplane** in a **Support Vector Machine (SVM)** is the one that maximizes the **separation margin** between the two classes. The **maximum-margin hyperplane**, also referred to as the **hard margin**, is selected based on maximizing the distance between the hyperplane and the nearest data point on each side.



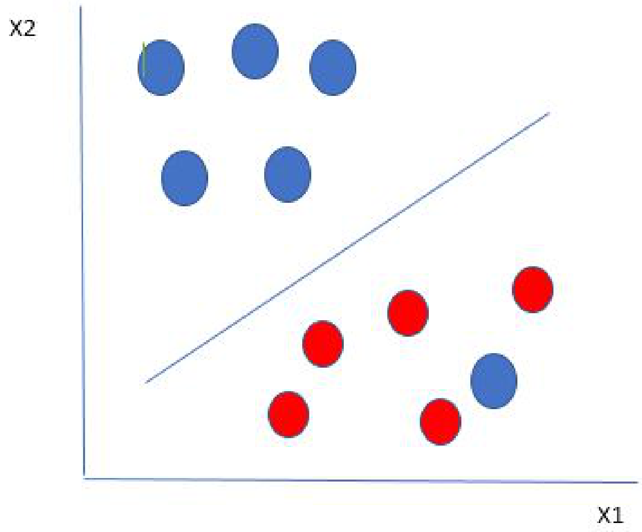
Multiple hyperplanes separate the data from two classes

So we choose the hyperplane whose distance from it to the nearest data point on each side is maximized. If such a hyperplane exists it is known as the **maximum-margin hyperplane/hard margin**. So from the above figure, we choose L2. Let's consider a scenario like shown below



Selecting hyperplane for data with outlier

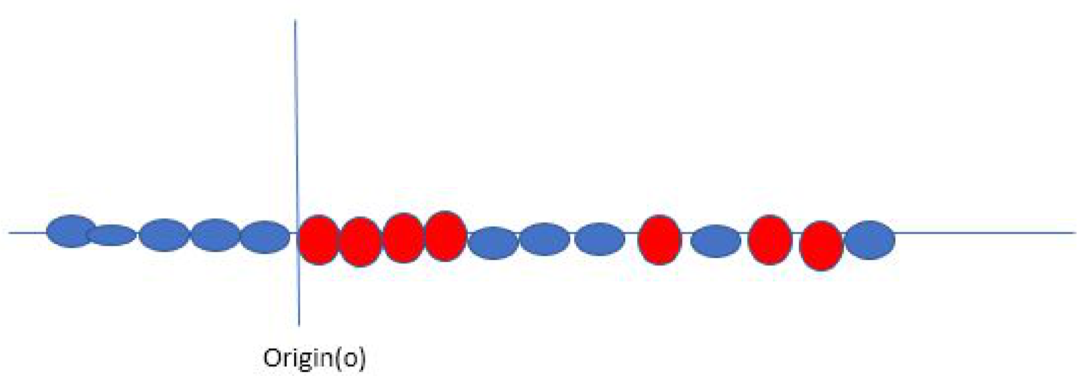
Here we have one blue ball in the boundary of the red ball. So how does SVM classify the data? It's simple! The blue ball in the boundary of red ones is an outlier of blue balls. The SVM algorithm has the characteristics



Hyperplane which is the most optimized one

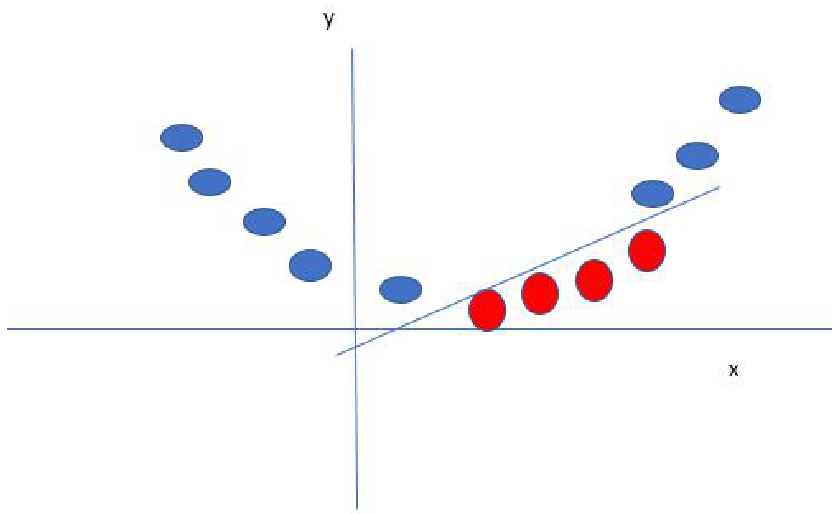
So in this type of data point what SVM does is, finds the maximum margin as done with previous data sets along with that it adds a penalty each time a point crosses the margin. So the margins in these types of cases are called **soft margins**. When there is a soft margin to the data set, the SVM tries to minimize *(1/margin+∧(∑penalty))*. Hinge loss is a commonly used penalty. If no violations no hinge loss.If violations hinge loss proportional to the distance of violation.

Till now, we were talking about linearly separable data(the group of blue balls and red balls are separable by a straight line/linear line). What to do if data are not linearly separable?



Original 1D dataset for classification

new variable y_i as a function of distance from origin o.so if we plot this we get something like as shown below



Mapping 1D data to 2D to become able to separate the two classes

In this case, the new variable y is created as a function of distance from the origin. A non-linear function that creates a new variable is referred to as a kernel.

Support Vector Machine Terminology

- **Hyperplane**: The **hyperplane** is the decision boundary used to separate data points of different classes in a feature space. For **linear classification**, this is a linear equation represented as wx+b=0.
- **Support Vectors**: **Support vectors** are the closest data points to the hyperplane. These points are critical in determining the hyperplane and the margin in **Support Vector Machine (SVM)**.
- **Margin**: The **margin** refers to the distance between the **support vector** and the hyperplane. The primary goal of the SVM algorithm is to maximize this margin, as a wider margin typically results in better classification performance.
- **Kernel**: The **kernel** is a mathematical function used in SVM to map input data into a higher-dimensional feature space. This allows the

- **Hard Margin**: A **hard margin** refers to the maximum-margin hyperplane that perfectly separates the data points of different classes without any misclassifications.
- **Soft Margin**: When data contains **outliers** or is not perfectly separable, SVM uses the **soft margin** technique. This method introduces a **slack variable** for each data point to allow some misclassifications while balancing between maximizing the margin and minimizing violations.
- **C**: The **C parameter** in SVM is a regularization term that balances margin maximization and the penalty for misclassifications. A higher **C** value imposes a stricter penalty for margin violations, leading to a smaller margin but fewer misclassifications.
- **Hinge Loss**: The **hinge loss** is a common loss function in SVMs. It penalizes misclassified points or margin violations and is often combined with a regularization term in the objective function.
- **Dual Problem**: The **dual problem** in SVM involves solving for the **Lagrange multipliers** associated with the support vectors. This formulation allows for the use of the **kernel trick** and facilitates more efficient computation.

Mathematical Computation: SVM

Consider a binary classification problem with two classes, labeled as +1 and -1. We have a training dataset consisting of input feature vectors X and their corresponding class labels Y.

The equation for the linear hyperplane can be written as:

$$w^T x + b = 0$$

The vector W represents the normal vector to the hyperplane. i.e the direction perpendicular to the hyperplane. The parameter **b** in the equation represents the offset or distance of the hyperplane from the origin along the normal vector **w**.

$$d_i = \frac{w^T x_i + b}{||w||}$$

where ||w|| represents the Euclidean norm of the weight vector w. Euclidean norm of the normal vector W

For Linear SVM classifier :

$$\hat{y} = \begin{cases} 1 & : \ w^T x + b \geq 0 \\ 0 & : \ w^T x + b < 0 \end{cases}$$

Optimization:

- **For Hard margin linear SVM classifier:**

$$\underset{w,b}{\text{minimize}} \frac{1}{2} w^T w = \underset{W,b}{\text{minimize}} \frac{1}{2} \left\| w \right\|^2$$
$$\text{subject to } y_i(w^T x_i + b) \geq 1 \ for \ i = 1, 2, 3, \cdots, m$$

The target variable or label for the i^{th} training instance is denoted by the symbol t_i in this statement. And t_i=-1 for negative occurrences (when y_i= 0) and t_i=1positive instances (when y_i = 1) respectively. Because we require the decision boundary that satisfy the constraint: $t_i(w^T x_i + b) \geq 1$

- **For Soft margin linear SVM classifier:**

$$\underset{w,b}{\text{minimize}} \ \frac{1}{2} w^T w + C \sum_{i=1}^{m} \zeta_i$$
$$\text{subject to } y_i(w^T x_i + b) \geq 1 - \zeta_i \ \ and \ \zeta_i \geq 0 \ \ for \ i = 1, 2, 3, \cdots, m$$

- **Dual Problem:** A dual Problem of the optimisation problem that requires locating the Lagrange multipliers related to the support vectors can be used to solve SVM. The optimal Lagrange multipliers α(i) that maximize the following dual objective function

$$\underset{\alpha}{\text{maximize}} : \frac{1}{2} \sum_{i \to m} \sum_{j \to m} \alpha_i \alpha_j t_i t_j K(x_i, x_j) - \sum_{i \to m} \alpha_i$$

where,

- α is the Lagrange multiplier associated with the ith training sample.
- $K(x_i, x_j)$ is the kernel function that computes the similarity between two samples x and x . It allows SVM to handle nonlinear classification

- The term $\sum \alpha_i$ represents the sum of all Lagrange multipliers.

The SVM decision boundary can be described in terms of these optimal Lagrange multipliers and the support vectors once the dual issue has been solved and the optimal Lagrange multipliers have been discovered. The training samples that have i > 0 are the support vectors, while the decision boundary is supplied by:

$$w = \sum_{i \to m} \alpha_i t_i K(x_i, x) + b$$
$$t_i(w^T x_i - b) = 1 \Longleftrightarrow b = w^T x_i - t_i$$

Types of Support Vector Machine

Based on the nature of the decision boundary, Support Vector Machines (SVM) can be divided into two main parts:

- **Linear SVM:** Linear SVMs use a linear decision boundary to separate the data points of different classes. When the data can be precisely linearly separated, linear SVMs are very suitable. This means that a single straight line (in 2D) or a hyperplane (in higher dimensions) can entirely divide the data points into their respective classes. A hyperplane that maximizes the margin between the classes is the decision boundary.
- **Non-Linear SVM:** Non-Linear SVM can be used to classify data when it cannot be separated into two classes by a straight line (in the case of 2D). By using kernel functions, nonlinear SVMs can handle nonlinearly separable data. The original input data is transformed by these kernel functions into a higher-dimensional feature space, where the data points can be linearly separated. A linear SVM is used to locate a nonlinear decision boundary in this modified space.

Popular kernel functions in SVM

The SVM kernel is a function that takes low-dimensional input space and transforms it into higher-dimensional space, ie it converts nonseparable data transformations and then finds out the process to separate the data based on the labels or outputs defined.

$$\begin{aligned}\text{Linear}: \; K(w,b) &= w^T x + b \\ \text{Polynomial}: \; K(w,x) &= (\gamma w^T x + b)^N \\ \text{Gaussian RBF}: \; K(w,x) &= \exp(-\gamma ||x_i - x_j||^n \\ \text{Sigmoid}: K(x_i,x_j) &= \tanh(\alpha x_i^T x_j + b)\end{aligned}$$

Implementing SVM Algorithm in Python

Predict if cancer is Benign or malignant. Using historical data about patients diagnosed with cancer enables doctors to differentiate malignant cases and benign ones are given independent attributes.

Steps

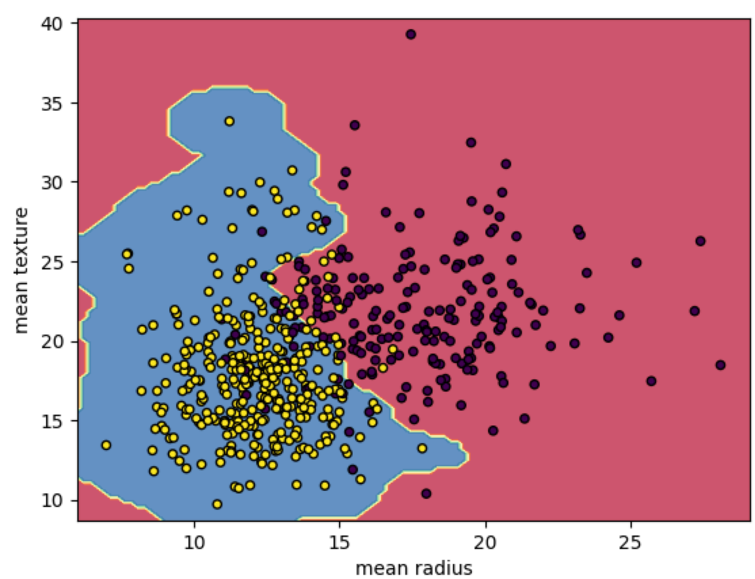
- Load the breast cancer dataset from sklearn.datasets
- Separate input features and target variables.
- Build and train the SVM classifiers using RBF kernel.
- Plot the scatter plot of the input features.
- Plot the decision boundary.
- Plot the decision boundary

Python

```python
# Load the important packages
from sklearn.datasets import load_breast_cancer
import matplotlib.pyplot as plt
from sklearn.inspection import DecisionBoundaryDisplay
from sklearn.svm import SVC

# Load the datasets
cancer = load_breast_cancer()
X = cancer.data[:, :2]
y = cancer.target

#Build the model
```

```

# Plot Decision Boundary
DecisionBoundaryDisplay.from_estimator(
        svm,
        X,
        response_method="predict",
        cmap=plt.cm.Spectral,
        alpha=0.8,
        xlabel=cancer.feature_names[0],
        ylabel=cancer.feature_names[1],
    )

# Scatter plot
plt.scatter(X[:, 0], X[:, 1],
            c=y,
            s=20, edgecolors="k")
plt.show()
```

Output:



Breast Cancer Classifications with SVM RBF kernel

Advantages and Disadvantages of Support Vector Machine (SVM)

1. **High-Dimensional Performance**: SVM excels in high-dimensional spaces, making it suitable for **image classification** and **gene expression analysis**.
2. **Nonlinear Capability**: Utilizing **kernel functions** like **RBF** and **polynomial**, SVM effectively handles **nonlinear relationships**.
3. **Outlier Resilience**: The **soft margin** feature allows SVM to ignore outliers, enhancing robustness in **spam detection** and **anomaly detection**.
4. **Binary and Multiclass Support**: SVM is effective for both **binary classification** and **multiclass classification**, suitable for applications in **text classification**.
5. **Memory Efficiency**: SVM focuses on **support vectors**, making it memory efficient compared to other algorithms.

Disadvantages of Support Vector Machine (SVM)

1. **Slow Training**: SVM can be slow for large datasets, affecting performance in **SVM in data mining** tasks.
2. **Parameter Tuning Difficulty**: Selecting the right **kernel** and adjusting parameters like **C** requires careful tuning, impacting **SVM algorithms**.
3. **Noise Sensitivity**: SVM struggles with noisy datasets and overlapping classes, limiting effectiveness in real-world scenarios.
4. **Limited Interpretability**: The complexity of the **hyperplane** in higher dimensions makes SVM less interpretable than other models.
5. **Feature Scaling Sensitivity**: Proper **feature scaling** is essential; otherwise, SVM models may perform poorly.

Conclusion

In conclusion, **Support Vector Machines (SVM)** are powerful algorithms in **machine learning**, ideal for both **classification** and **regression** tasks. They excel at finding the optimal hyperplane for separating data, making them

SVM's adaptability through **kernel functions** allows it to handle both linear and nonlinear data effectively. However, challenges like **parameter tuning** and potential slow training times on large datasets must be considered.

Understanding **SVM** is crucial for data scientists, as it enhances predictive accuracy and decision-making across various domains, including **data mining** and **artificial intelligence**.

Support Vector Machine (SVM) Algorithm- FAQs

How does SVM work in machine learning?

> *SVM works by finding the maximum-margin hyperplane that best separates the data points of different classes. It uses support vectors, which are the closest data points to the hyperplane, to define this boundary.*

What are the key advantages of using SVM in machine learning?

> *SVMs are effective for high-dimensional data, robust to outliers, and versatile due to kernel functions, allowing them to handle both linear and nonlinear relationships.*

What is the difference between hard margin and soft margin SVM?

> *A hard margin SVM perfectly separates classes without misclassification, while a soft margin SVM allows some misclassifications to better accommodate outliers, balancing the* margin and penalties.