Machine Learning

Dr.Hajialiasgari

Tehran University
Of
Medical Science

February 1, 2025

TEHRAN UNIVERSITY
OF
MEDICAL SCIENCES

1. SVM
2. Naïve Bayes

Overview of SVM

- Support Vector Machine (SVM) is a supervised machine learning algorithm used for classification and regression tasks.
- SVM aims to find the optimal hyperplane that separates data points of different classes with the maximum margin.
- It can handle both linearly separable and non-linearly separable data using the kernel trick.

Advantages and Disadvantages of SVM

Advantages:

- Effective in high-dimensional spaces.
- Works well with clear margin separation.
- Versatile with different kernel functions.

Disadvantages:

- Not suitable for very large datasets.
- Requires careful selection of the kernel and regularization parameter.
- Sensitive to noise and overlapping classes.

Applications of SVM

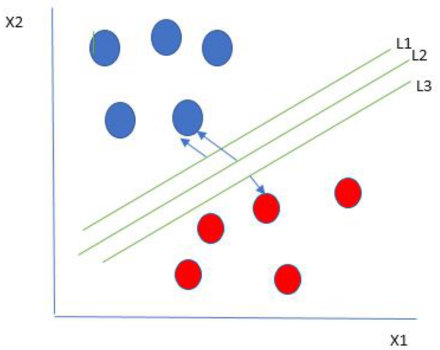
- Text and hypertext categorization.
- Image classification and object detection.
- Bioinformatics, e.g., cancer classification and protein structure prediction.
- Handwriting recognition.
- Intrusion detection in cybersecurity.

How SVM Works: Linear SVM

- SVM separates data into two classes by finding the best hyperplane.
- **Linear SVM:** For linearly separable data, SVM finds a straight hyperplane or a flat plane in higher dimensions to separate classes.
- The **support vectors** are the closest data points to the hyperplane. They define the margin and influence the hyperplane's orientation and position.
- SVM maximizes the margin between classes for better generalization.

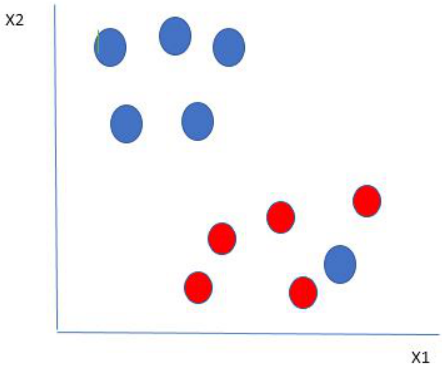
How SVM Works : Linear SVM (Cont.)

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.

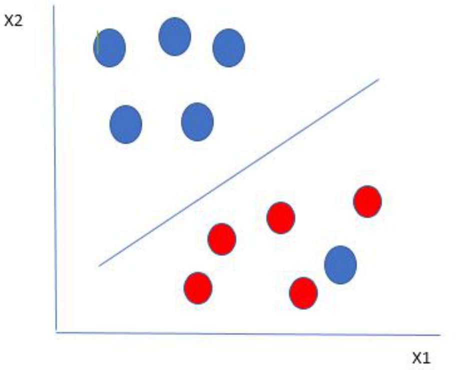
How SVM Works : Linear SVM (Cont.)



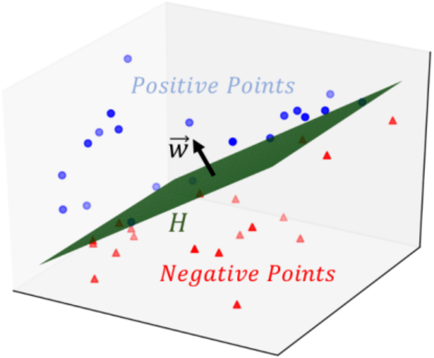
What About Now ??

How SVM Works : Linear SVM (Cont.)

Its simple! The blue ball in the boundary of red ones is an outlier of blue balls. The SVM algorithm has the characteristics to ignore the outlier and finds the best hyperplane that maximizes the margin. SVM is robust to outliers.



Hyperplane SVM

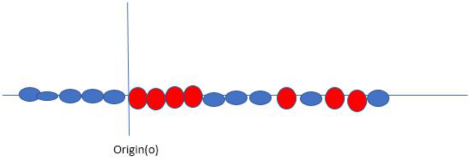


How SVM Works: Non-Linear SVM

- **Non-Linearly Separable Data:** A linear hyperplane cannot separate the classes.
- **Kernel Trick:** SVM maps data to a higher-dimensional space where a linear hyperplane can separate the classes.
- Common kernels:
  - Linear Kernel: $K(x,z) = x^T z$
  - Polynomial Kernel: $K(x,z) = (x^T z + c)^d$
  - RBF Kernel: $K(x,z) = \exp(-\gamma||x-z||^2)$

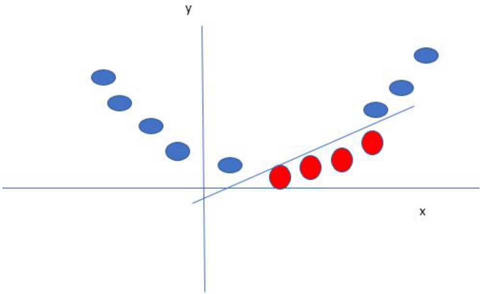
How SVM Works : Non-Linear SVM (Cont.)

How about this? Can we separate red and blue dot with a linear line ?



How SVM Works : Non-Linear SVM (Cont.)

SVM solves this by creating a new variable using a kernel. We call a point xi on the line and we create a new variable yi as a function of distance from origin o.



How SVM Works: Margin Maximization

- The **margin** is the distance between the hyperplane and the nearest data points from each class.
- SVM maximizes this margin while ensuring correct classification.
- **Regularization:** For non-linearly separable data, SVM introduces a parameter C to balance margin size and classification errors.

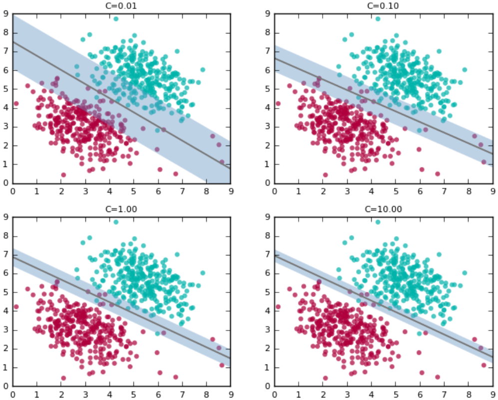
How SVM Works: Decision Rule

- After training, SVM predicts the class label y for a new sample x using:

$$y = \text{sign}(w^T x + b)$$

- w and b are learned parameters that define the hyperplane.
- The decision rule determines on which side of the hyperplane a data point lies.

Soft VS Hard Margin



1. SVM
2. Naïve Bayes

Conditional Probability

Conditional probability is the probability of an event occurring given that another event has already occurred.

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)} \tag{1}$$

Example: Conditional Probability

Suppose 60% of students in a school like math, and 30% of students both like math and play chess. The probability that a student plays chess given that they like math is:

$$P(C \mid M) = \frac{P(C \cap M)}{P(M)} = \frac{0.3}{0.6} = 0.5 \tag{2}$$

Bayes' Theorem

Bayes' theorem describes how to update probabilities based on new evidence:

$$P(A \mid B) = \frac{P(B \mid A)\,P(A)}{P(B)} \tag{3}$$

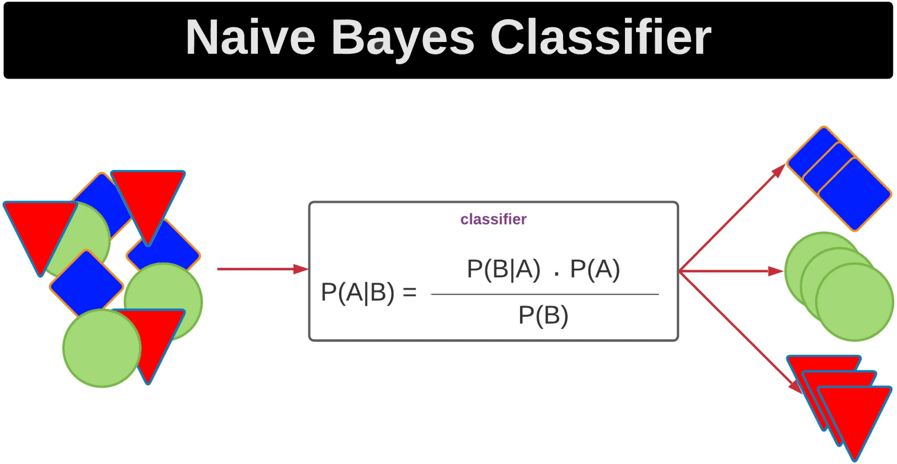
Example: Bayes' Theorem

Suppose a test for a disease is 95% accurate when a person has the disease and 90% accurate when they do not. If 1% of the population has the disease, the probability that a randomly selected person who tested positive actually has the disease is computed as follows:

$$P(D \mid T) = \frac{P(T \mid D)P(D)}{P(T)} \tag{4}$$

$$= \frac{0.95 \times 0.01}{(0.95 \times 0.01) + (0.1 \times 0.99)} \approx 0.087 \tag{5}$$

Naïve Bayes in Classification



Naïve Bayes in Machine Learning

Naïve Bayes is a classification algorithm based on Bayes' theorem with an assumption of independence among features. It is commonly used for:

- Spam filtering
- Sentiment analysis
- Medical diagnosis

How It Works in Medical Diagnosis

- The goal is to predict the presence of a disease given symptoms.
- For example, a patient may exhibit symptoms such as fever, cough, and fatigue, and we aim to predict the disease.

Bayes' Theorem

Bayes' Theorem calculates the probability of a disease given symptoms:

$$P(Disease|Symptom_1, Symptom_2, \ldots, Symptom_n) = \frac{P(Disease)\prod_{i=1}^{n} P(Symptom_i|Disease)}{P(Symptom_1, Symptom_2, \ldots, Symptom_n)}$$

- $P(Disease)$: Prior probability of the disease.
- $P(Symptom_i|Disease)$: Likelihood of observing symptom $Symptom_i$ given the disease.
- $P(Symptom_1, \ldots, Symptom_n)$: Evidence, constant across diseases.

Conditional Independence Assumption

Naive Bayes assumes that symptoms (features) are conditionally independent given the disease, simplifying calculations and allowing for efficient handling of many symptoms.

Practical Example

For a patient with symptoms such as fever, cough, and fatigue, Naive Bayes calculates the probability of different diseases (e.g., flu, pneumonia, COVID-19) based on:

- Prior probability of each disease.
- Likelihood of each symptom given the disease.

$$P(Disease|Symptoms) = \frac{P(Disease)\prod_{i=1}^{n} P(Symptom_i|Disease)}{P(Symptom_1, \ldots, Symptom_n)}$$

Why Its Effective

- **Simplicity**: Despite assuming conditional independence, Naive Bayes often provides accurate results.
- **Efficiency**: It handles large datasets well, typical in medical data.
- **Interpretability**: The model is easy to understand, making it suitable for healthcare applications.

Conclusion

Naive Bayes is an efficient and effective tool for medical diagnosis, predicting diseases based on symptoms, and is widely used in healthcare due to its simplicity and accuracy.

For more information and code check the related notebook

End of Classification