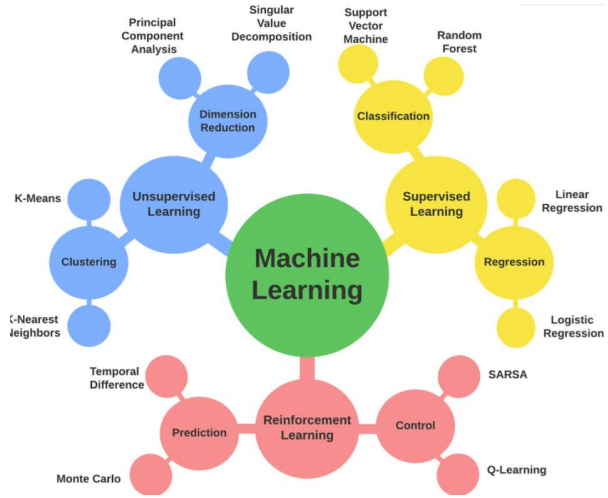
Các phương pháp học máy
Machine learning methods

4 TC: 2 LT – 2 TH

Giảng viên: **Tạ Hoàng Thắng**

tahoangthang@gmail.com

0975399307



Naive Bayes

Definitions

Wikipedia: In statistics, naive Bayes classifiers are a family of linear "probabilistic classifiers" which assumes that the features are conditionally independent, given the target class.

- The strength (naivety) of this assumption is what gives the classifier its name.
- These classifiers are among the simplest Bayesian network models.
- **One of the most simple and effective classification algorithms.**
  - Naïve Bayes classifier aids in the rapid development of machine learning models with **rapid prediction capabilities.**

https://en.wikipedia.org/wiki/Naive_Bayes_classifier

https://www.geeksforgeeks.org/naive-bayes-classifiers/

Naive Bayes

Definitions

Naïve Bayes algorithm is used for classification problems. It is highly used in **text classification.**

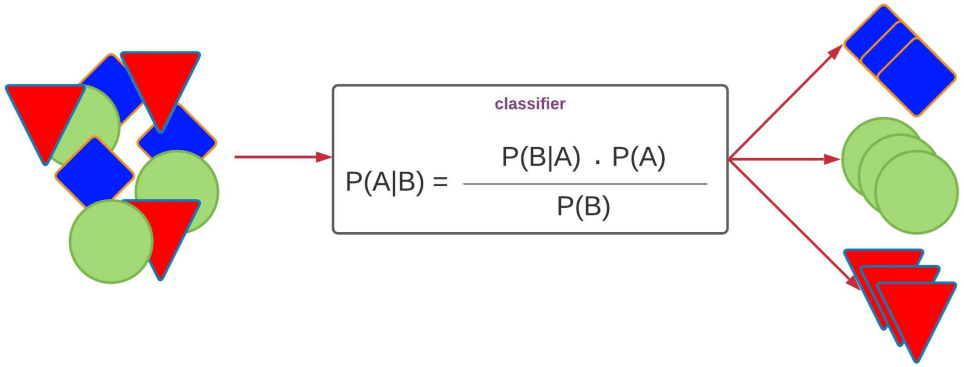
- In text classification tasks, **data contains high dimension (as each word represent one feature in the data).**
- It is used in spam filtering, sentiment detection, rating classification etc. The advantage of using naïve Bayes is its speed.
- It is fast and making prediction is easy with high dimension of data.

Naive Bayes

The name of Naive Bayes

The "**Naive**" part of the name indicates the simplifying assumption made by the Naïve Bayes classifier.

- **The classifier assumes that the features used to describe an observation are conditionally independent, given the class label.**
- The "**Bayes**" part of the name refers to Reverend Thomas Bayes, an 18th-century statistician and theologian who formulated Bayes' theorem.



Naive Bayes

Example

Consider a fictional dataset that describes the weather conditions for playing a game of golf. Given the weather conditions, each tuple classifies the conditions as **fit("Yes")** or **unfit("No")** for playing golf.

	Outlook	Temperature	Humidity	Windy	Play Golf
0	Rainy	Hot	High	False	No
1	Rainy	Hot	High	True	No
2	Overcast	Hot	High	False	Yes
3	Sunny	Mild	High	False	Yes
4	Sunny	Cool	Normal	False	Yes
5	Sunny	Cool	Normal	True	No

Naive Bayes

Example

The dataset is divided into two parts, namely, **feature matrix and the response vector**.

- **Feature matrix** contains all the vectors(rows) of dataset in which each vector consists of the value of dependent features.
  - In above dataset, features are 'Outlook', 'Temperature', 'Humidity' and 'Windy'.
- **Response vector** contains the value of class variable(prediction or output) for each row of feature matrix.
  - In above dataset, the class variable name is 'Play golf'.

Naive Bayes

Assumption of Naive Bayes

The fundamental Naive Bayes assumption is that each feature makes an:

- **Feature independence:** The features of the data are conditionally independent of each other, given the class label.
- **Continuous features are normally distributed:** If a feature is continuous, then it is assumed to be normally distributed within each class.
- **Discrete features have multinomial distributions:** If a feature is discrete, then it is assumed to have a multinomial distribution within each class.
  - If you have n independent trials and each trial can result in one of k categories, the multinomial distribution describes the probability of observing a specific count of outcomes in each category.
- **Features are equally important:** All features are assumed to contribute equally to the prediction of the class label.
- **No missing data:** The data should not contain any missing values.

Naive Bayes

Bayes' Theorem

Bayes' Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes' theorem is stated mathematically as the following equation:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

where A and B are events and P(B) ≠ 0

- Basically, we are trying to find probability of event A, given the event B is true. Event B is also termed as **evidence**.
- P(A) is the **priori** of A (the prior probability, i.e. Probability of event before evidence is seen). The evidence is an attribute value of an unknown instance(here, it is event B).
- P(B) is Marginal Probability: Probability of Evidence.
- P(A|B) is a posteriori probability of B, i.e. probability of event after evidence is seen.
- P(B|A) is Likelihood probability i.e the likelihood that a hypothesis will come true based on the evidence.

Naive Bayes

Bayes' Theorem

Bayes' Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes' theorem is stated mathematically as the following equation:

Now, with regards to our dataset, we can apply Bayes' theorem in following way:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}$$

where, y is class variable and X is a dependent feature vector (of size *n*) where:

$$X = (x_1, x_2, x_3, ..., x_n)$$

Naive Bayes

Bayes' Theorem

First row:

```
X = (Rainy, Hot, High, False)
y = No
```

So basically, $P(y|X)$here means, the probability of "Not playing golf" given that the weather conditions are "Rainy outlook", "Temperature is hot", "high humidity" and "no wind".

	Outlook	Temperature	Humidity	Windy	Play Golf
0	Rainy	Hot	High	False	No
1	Rainy	Hot	High	True	No
2	Overcast	Hot	High	False	Yes
3	Sunny	Mild	High	False	Yes
4	Sunny	Cool	Normal	False	Yes
5	Sunny	Cool	Normal	True	No

Naive Bayes

Bayes' Theorem

First row:

```
X = (Rainy, Hot, High, False)

y = No
```

Now, if any two events A and B are independent, then,

```
P(A,B) = P(A)P(B)
```

Hence, we reach to the result:

$$P(y|x_1,\ldots,x_n) = \frac{P(x_1|y)P(x_2|y)\ldots P(x_n|y)P(y)}{P(x_1)P(x_2)\ldots P(x_n)}$$

which can be expressed as:

$$P(y|x_1,\ldots,x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1)P(x_2)\ldots P(x_n)}$$

Now, as the denominator remains constant for a given input, we can remove that term:

$$P(y|x_1,\ldots,x_n) \propto P(y)\prod_{i=1}^{n} P(x_i|y)$$

Naive Bayes

Bayes' Theorem

Now, we need to create a classifier model. For this, we find the probability of given set of inputs for all possible values of the class variable *y* and pick up the output with maximum probability. This can be expressed mathematically as:

$y = argmax_y P(y) \prod_{i=1}^{n} P(x_i|y)$

So, finally, we are left with the task of calculating $P(y)$and $P(x_i|y)$.

Please note that $P(y)$ is also called class probability and $P(x_i|y)$ is called conditional probability.

The different naive Bayes classifiers differ mainly by the assumptions they make regarding the distribution of $P(x_i|y)$.

Naive Bayes

Bayes' Theorem

	Outlook	Temperature	Humidity	Windy	Play Golf
0	Rainy	Hot	High	False	No
1	Rainy	Hot	High	True	No
2	Overcast	Hot	High	False	Yes
3	Sunny	Mild	High	False	Yes
4	Sunny	Cool	Normal	False	Yes
5	Sunny	Cool	Normal	True	No
6	Overcast	Cool	Normal	True	Yes
7	Rainy	Mild	High	False	No
8	Rainy	Cool	Normal	False	Yes
9	Sunny	Mild	Normal	False	Yes
10	Rainy	Mild	Normal	True	Yes
11	Overcast	Mild	High	True	Yes
12	Overcast	Hot	Normal	False	Yes
13	Sunny	Mild	High	True	No

Outlook

	Yes	No	P(yes)	P(no)
Sunny	2	3	2/9	3/5
Overcast	4	0	4/9	0/5
Rainy	3	2	3/9	2/5
Total	9	5	100%	100%

Temperature

	Yes	No	P(yes)	P(no)
Hot	2	2	2/9	2/5
Mild	4	2	4/9	2/5
Cool	3	1	3/9	1/5
Total	9	5	100%	100%

Humidity

	Yes	No	P(yes)	P(no)
High	3	4	3/9	4/5
Normal	6	1	6/9	1/5
Total	9	5	100%	100%

Wind

	Yes	No	P(yes)	P(no)
False	6	2	6/9	2/5
True	3	3	3/9	3/5
Total	9	5	100%	100%

Play		P(Yes)/P(No)
Yes	9	9/14
No	5	5/14
Total	14	100%

For example, probability of playing golf given that the temperature is cool, i.e:

- **P(temp. = cool | play golf = Yes) = 3/9**
- **P(play golf = Yes) = 9/14.**

Naive Bayes

Bayes' Theorem

Test:

```
today = (Sunny, Hot, Normal, False)
```

$$P(Yes|today) = \frac{P(SunnyOutlook|Yes)P(HotTemperature|Yes)P(NormalHumidity|Yes)P(NoWind|Yes)P(Yes)}{P(today)}$$

and probability to not play golf is given by:

$$P(No|today) = \frac{P(SunnyOutlook|No)P(HotTemperature|No)P(NormalHumidity|No)P(NoWind|No)P(No)}{P(today)}$$

Since, P(today) is common in both probabilities, we can ignore P(today) and find proportional probabilities as:

$$P(Yes|today) \propto \frac{3}{9}.\frac{2}{9}.\frac{6}{9}.\frac{6}{9}.\frac{9}{14} \approx 0.02116$$

and

$$P(No|today) \propto \frac{3}{5}.\frac{2}{5}.\frac{1}{5}.\frac{2}{5}.\frac{5}{14} \approx 0.0068$$

Naive Bayes

Bayes' Theorem

Test:

$$P(Yes|today) + P(No|today) = 1$$

These numbers can be converted into a probability by making the sum equal to 1 (normalization):

$$P(Yes|today) = \frac{0.02116}{0.02116+0.0068} \approx 0.0237$$

and

$$P(No|today) = \frac{0.0068}{0.0141+0.0068} \approx 0.33$$

Since

$$P(Yes|today) > P(No|today)$$

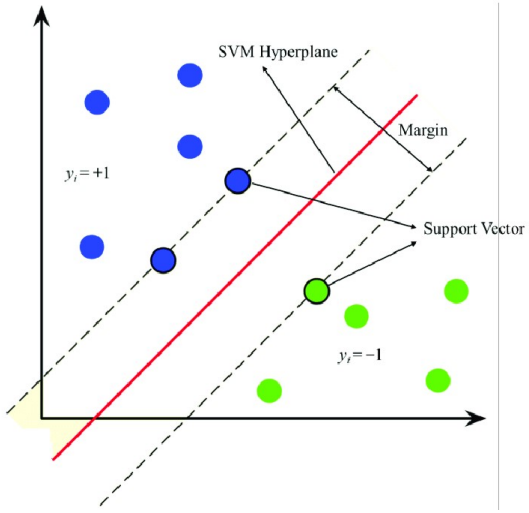
So, prediction that golf would be played is 'Yes'.

https://www.youtube.com/watch?v=XzSlEA4ck2I

SVM

Definitions

Wikipedia: In machine learning, support vector machines (SVMs, also **support vector networks**) are supervised max-margin models with associated learning algorithms that analyze data for classification and regression analysis.



SVM

Definitions

Geeksforgeeks: 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.

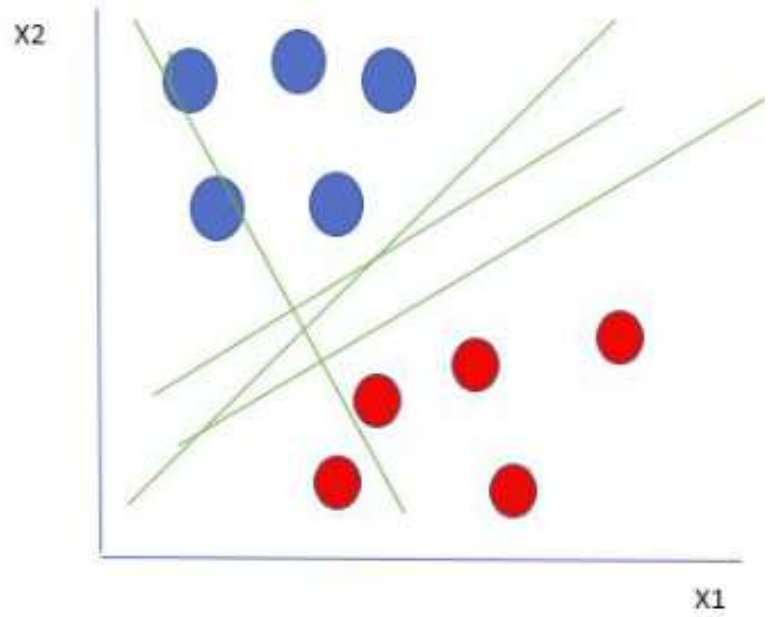
- making them robust for both binary and multiclass classification.

SVM

Definitions

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

- 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.**

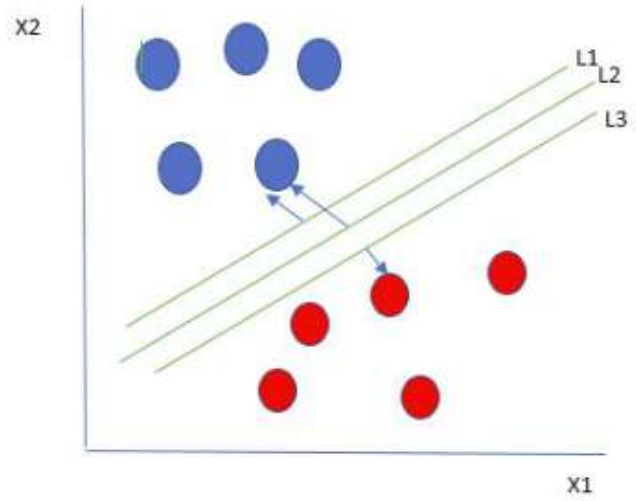


SVM

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.

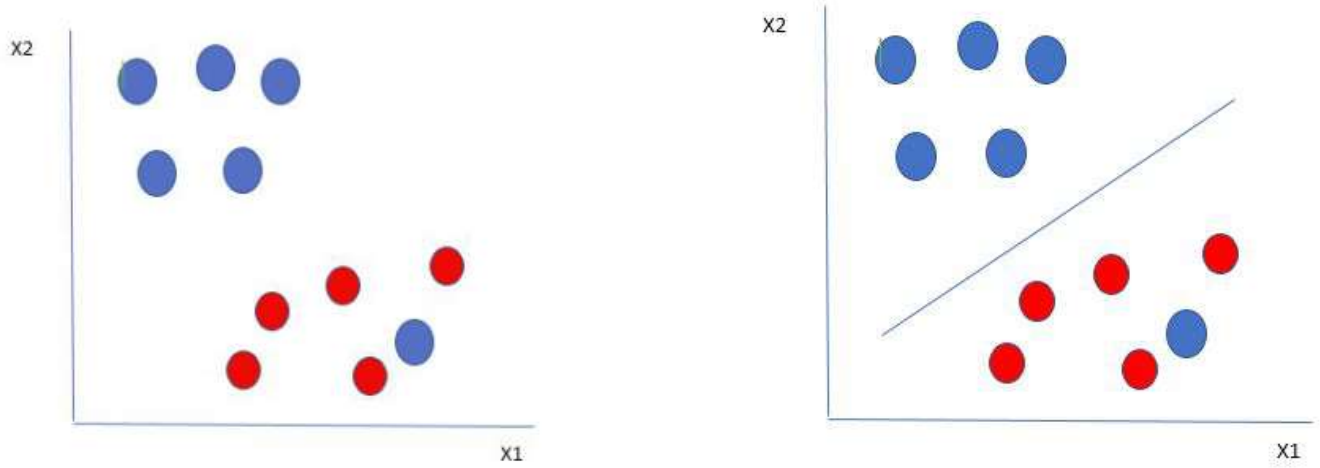


SVM

How does Support Vector Machine Algorithm Work?

Here we have one blue ball in the boundary of the red ball. So how does SVM classify the data? The blue ball in the boundary of red ones is an outlier of blue balls.

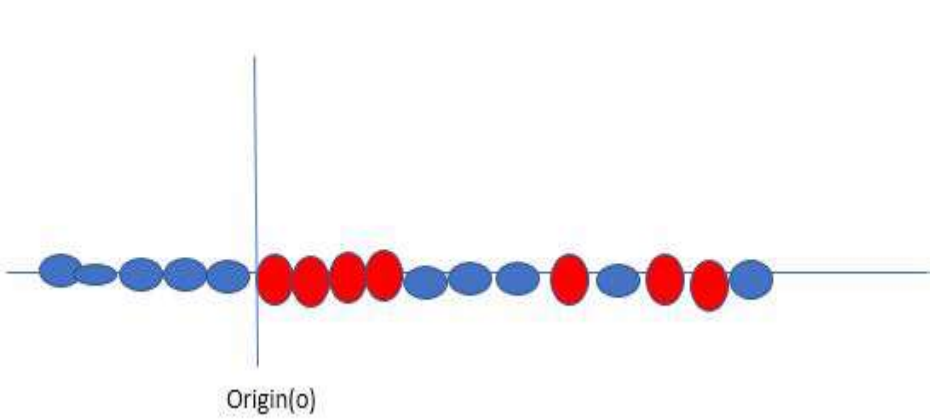
- The SVM algorithm has the characteristics to **ignore the outlier** and finds the best hyperplane that maximizes the margin. **SVM is robust to outliers.**

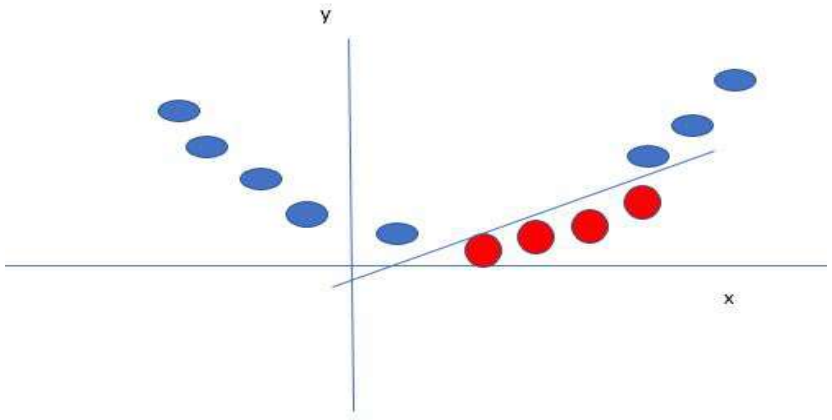


SVM

How does Support Vector Machine Algorithm Work?

SVM solves this by creating a new variable using a **kernel**. We call a point $\mathbf{x_i}$ on the line and we create a new variable $\mathbf{y_i}$ as a function of distance from origin.





SVM

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.

SVM

Support Vector Machine Terminology

Kernel: The kernel is a mathematical function used in SVM **to map input data into a higher-dimensional feature space**. This allows the SVM to find a hyperplane in cases where data points are not linearly separable in the original space.

- Common kernel functions include linear, polynomial, radial basis function (RBF), and sigmoid.

Hard Margin: A hard margin refers to the maximum-margin hyperplane that **perfectly separates the data points** of different classes without any misclassifications.

SVM

Support Vector Machine Terminology

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.

SVM

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**.

The distance between a data point x_i and the decision boundary can be calculated as:

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

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

For Linear SVM classifier :

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

SVM

Mathematical Computation: SVM

Optimization:

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

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

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:**

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

SVM

Mathematical Computation: SVM

https://www.youtube.com/watch?v=rhz7AGlO5fw