Department of Mathematics, University of Udine

Support Vector Machines and their use in IR

Lorenzo Zanolin.
lorenzo.zanolin@spes.uniud.it

May 12, 2023

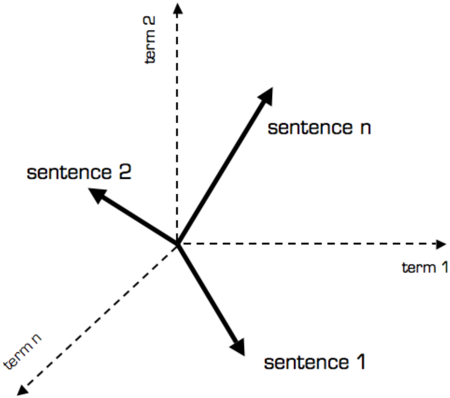
Outline

1. SVM over linearly separable data
2. SVM over non linearly separable data
3. SVM use in IR

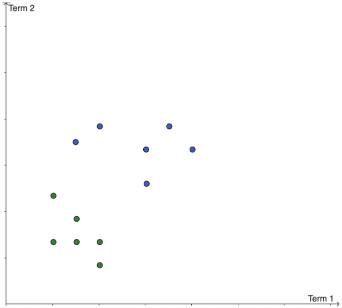
Vector Space

Suppose you have to classify whether a document is *relevant* or not. We can think to use *terms* as features to divide properly the data. We will use a *t-dimensional* vector space to represent our documents.

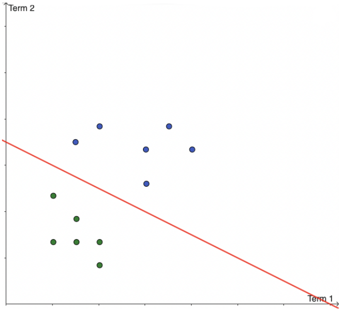
Suppose you have to classify whether a document is *relevant* or not. We can think to use *terms* as features to divide properly the data. We will use a *t-dimensional* vector space to represent our documents.



For simplicity, we can reason using a 2D Space.
Data can be separated using a *decision boundary*, which is an *hyperplane*.



For simplicity, we can reason using a 2D Space.
Data can be separated using a *decision boundary*, which is an *hyperplane*.

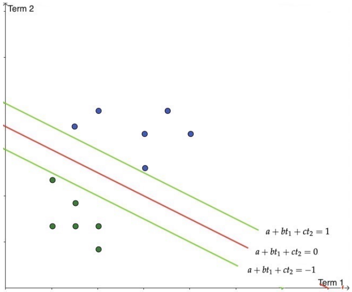


In the previous example the *decision boundary* is a line, represented by the equation $a + bt_1 + ct_2 = 0$.
We can introduce two *parallels* hyperplanes (lines) to the decision boundary, called *support vectors* whose equations are

$$\begin{cases} a + bt_1 + ct_2 = 1 \\ a + bt_1 + ct_2 = -1 \end{cases} \tag{1}$$

In the previous example the *decision boundary* is a line, represented by the equation $a + bt_1 + ct_2 = 0$.
We can introduce two *parallels* hyperplanes (lines) to the decision boundary, called *support vectors* whose equations are

$$\begin{cases} a + bt_1 + ct_2 = 1 \\ a + bt_1 + ct_2 = -1 \end{cases} \tag{1}$$



For convenience, we can rewrite the three equations using *vectorization* notation.

$$\begin{cases} b^T t + a = 0 \\ b^T t + a = \pm 1 \end{cases} \tag{2}$$

Margin

For convenience, we can rewrite the three equations using *vectorization* notation.

$$\begin{cases} b^T t + a = 0 \\ b^T t + a = \pm 1 \end{cases} \tag{2}$$

Intuitively, we can define the *margin* of the two support vectors as the distance between them.

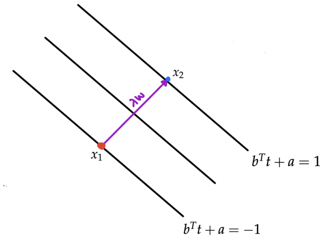
For convenience, we can rewrite the three equations using *vectorization* notation.

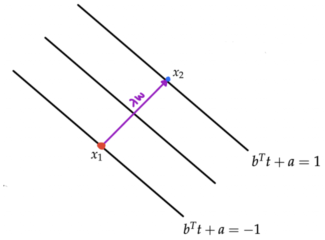
$$\begin{cases} b^T t + a = 0 \\ b^T t + a = \pm 1 \end{cases} \tag{2}$$

Intuitively, we can define the *margin* of the two support vectors as the distance between them.

Consider an arbitrary point x_1 that lies on support vector s_1. Since s_1, s_2 are parallel, consider the closest point to x_1 belonging to s_2, say x_2.

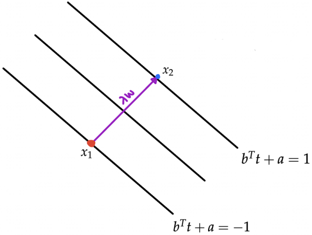
Margin





Their distance is $\lambda \|w\| = \frac{2}{\sqrt{w^T w}}$.

Vector λw represents the distance between the two margins.



Their distance is $\lambda \|w\| = \frac{2}{\sqrt{w^T w}}$.

Vector λw represents the distance between the two margins.

SVM [2, 1] are used to find the *maximum margin linear classifier*, thus we want to maximize the margin.

Remembering that we want to classify documents, our goal is to find specific b s.t. given a document x belonging to class y the decision boundary behaves as follows:

$$\begin{cases} b^T x + a \geq 1 & \textit{if } y = 1 \\ b^T x + a \leq -1 & \textit{if } y = -1 \end{cases} \tag{3}$$

Remembering that we want to classify documents, our goal is to find specific b s.t. given a document x belonging to class y the decision boundary behaves as follows:

$$\begin{cases} b^T x + a \geq 1 & \textit{if } y = 1 \\ b^T x + a \leq -1 & \textit{if } y = -1 \end{cases} \tag{3}$$

We can define the *cost function* as a system of equations:

$$\begin{cases} \min_{b,a} \frac{\sqrt{b^T b}}{2} \\ \textit{subject to} \quad y_i(b^T x_i + a) \geq 1 \quad \forall x_i \end{cases} \tag{4}$$

In real world problem it is not likely to get an exactly separate line dividing the data within the space. It would be better for the smooth boundary to ignore few data points than be curved or go in loops, around the outliers.

Soft-Margin

In real world problem it is not likely to get an exactly separate line dividing the data within the space. It would be better for the smooth boundary to ignore few data points than be curved or go in loops, around the outliers.

So, we will use *slack variables* to introduce a penalty for each misclassified point.

Soft-Margin

In real world problem it is not likely to get an exactly separate line dividing the data within the space. It would be better for the smooth boundary to ignore few data points than be curved or go in loops, around the outliers.

So, we will use *slack variables* to introduce a penalty for each misclassified point.

The new cost function will be

$$\begin{cases} \min_{b,a} \frac{\sqrt{b^T b}}{2} + C \sum_i \xi_i \\ \textit{subject to} \quad y_i(b^T x_i + a) \geq 1 - \xi_i \quad \textit{and } \xi_i > 0 \quad \forall x_i \end{cases} \tag{5}$$

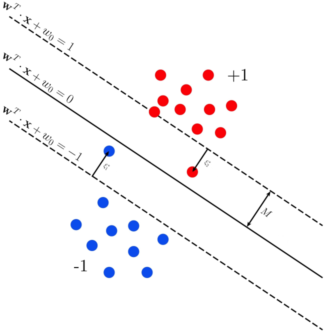
Intuitively, C represents how much the penalty counts.

The larger is C the stricter the classification is, since a larger C will give more evidence to slack variables.

$$\begin{cases} \min_{b,a} \frac{\sqrt{b^T b}}{2} + C \sum_i \xi_i \\ \textit{subject to} \quad y_i(b^T x_i + a) \geq 1 - \xi_i \quad \textit{and } \xi_i > 0 \quad \forall x_i \end{cases} \tag{6}$$

The larger is C the stricter the classification is, since a larger C will give more evidence to slack variables.

$$\begin{cases} \min_{b,a} \frac{\sqrt{b^T b}}{2} + C \sum_i \xi_i \\ \textit{subject to} \quad y_i(b^T x_i + a) \geq 1 - \xi_i \quad \textit{and } \xi_i > 0 \quad \forall x_i \end{cases} \tag{6}$$

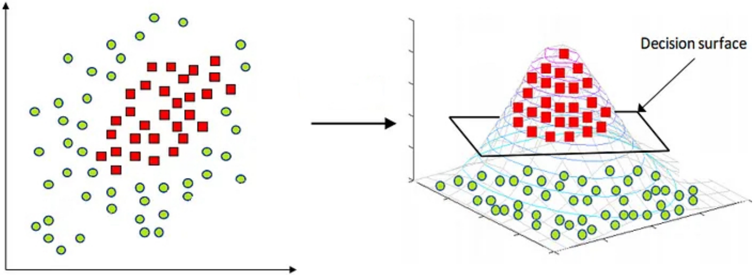


Non linearly separable data

What if our data is not linearly separable?

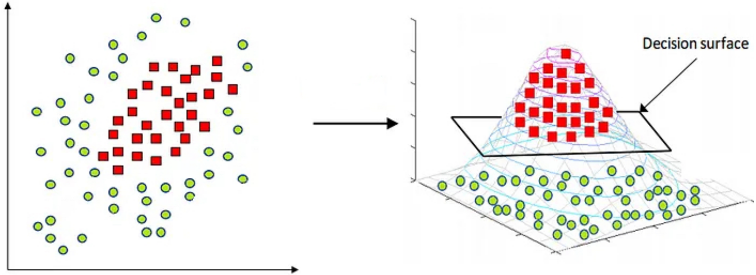
Non linearly separable data

What if our data is not linearly separable? It should be better to reason in a bigger dimensional space.



Non linearly separable data

What if our data is not linearly separable? It should be better to reason in a bigger dimensional space.



But augmenting dimensions costs a lot...

Kernel Trick

Consider the function $\phi : \mathbb{R}^3 \mapsto \mathbb{R}^{10}$ used to map points in a new vector space. Calculating the *similarity* $\phi(x_i)^T \phi(x_j)$ between each point may be intractable.

Kernel Trick

Consider the function $\phi : \mathbb{R}^3 \mapsto \mathbb{R}^{10}$ used to map points in a new vector space. Calculating the *similarity* $\phi(x_i)^T \phi(x_j)$ between each point may be intractable.

Here the *Kernel Trick* [4] comes in handy.

Kernel Trick

Consider the function $\phi : \mathbb{R}^3 \mapsto \mathbb{R}^{10}$ used to map points in a new vector space. Calculating the *similarity* $\phi(x_i)^T \phi(x_j)$ between each point may be intractable.

Here the *Kernel Trick* [4] comes in handy.
It consists of a simple linear algebra reformulation,

$$\phi(x_i)^T \phi(x_j) = K(x_i, x_j) = (1 + x_i{}^T x_j)^2. \tag{7}$$

Kernel Trick

Consider the function $\phi : \mathbb{R}^3 \mapsto \mathbb{R}^{10}$ used to map points in a new vector space. Calculating the *similarity* $\phi(x_i)^T \phi(x_j)$ between each point may be intractable.

Here the *Kernel Trick* [4] comes in handy.
It consists of a simple linear algebra reformulation,

$$\phi(x_i)^T \phi(x_j) = K(x_i, x_j) = (1 + x_i{}^T x_j)^2. \tag{7}$$

K is a *kernel function* that operates on the lower dimension vectors x_i and x_j to produce a value equivalent to the dot-product of the higher-dimensional vectors.

Instead of doing the complex computations in the 10-dimensional space, we reach the same result within the 3-dimensional space by calculating the dot product.

Gaussian Kernel

There are lots of Kernel functions, an example is the *Gaussian*.

Gaussian Kernel

There are lots of Kernel functions, an example is the *Gaussian*.

The idea behind is to use all n training points as *landmarks*.
Then we can calculate the similarities of each point and all the landmarks.

Gaussian Kernel

There are lots of Kernel functions, an example is the *Gaussian*.

The idea behind is to use all n training points as *landmarks*.
Then we can calculate the similarities of each point and all the landmarks.

$$\forall l_i \; similarity(x, l^{(i)}) = \exp(-\frac{\left\| x - l^{(i)} \right\|^2}{2\sigma^2}) \tag{8}$$

Gaussian Kernel

There are lots of Kernel functions, an example is the *Gaussian*.

The idea behind is to use all n training points as *landmarks*. Then we can calculate the similarities of each point and all the landmarks.

$$\forall l_i \; similarity(x, l^{(i)}) = \exp(-\frac{\left\| x - l^{(i)} \right\|^2}{2\sigma^2}) \tag{8}$$

The result will be a mapping for each point in a n-dimensional space. Finally, we can identify an hyperplane which can divide correctly the two classes of points.

Pratical application

A hybrid technique used in the IR field will be presented; it uses two components:

- **K-Means**: model for *unsupervised classification*;
- **SVM**: model for *supervised classification*.

A hybrid technique used in the IR field will be presented; it uses two components:

- **K-Means**: model for *unsupervised classification*;
- **SVM**: model for *supervised classification*.

K-means algorithm is one of the most used clustering algorithms. It consists of partitioning unlabeled objects into k classes, where k is fixed beforehand.
A brief explanation will follow.

K-means

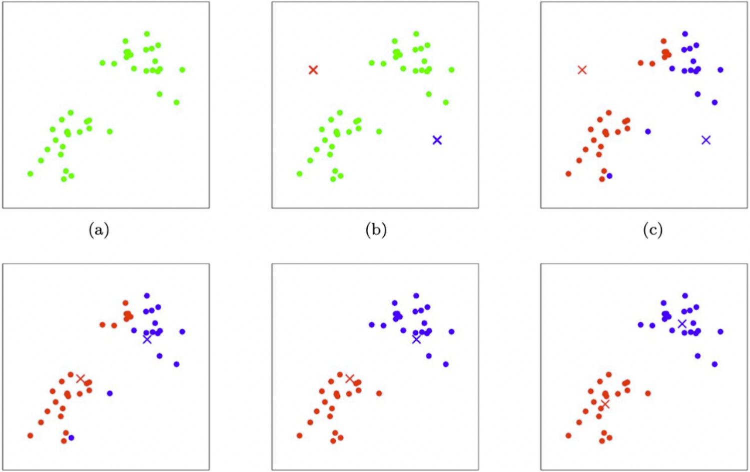
The aim is to create clusters that contain *similar documents*.
Given a training set $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$, the algorithm [5] used is the following:

K-means

The aim is to create clusters that contain *similar documents*. Given a training set $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$, the algorithm [5] used is the following:

$K \leftarrow randomValue$ ▷ multiple K will be tried
procedure K-MEANS(K)
    `initialize cluster centroids` $\mu_1, \ldots, \mu_k$ `randomly.`
    repeat until convergence{
        for $i \leftarrow 1$ to m **do**
            $c^{(i)} = arg\ min_j \|x^{(i)} - \mu_j\|$ ▷ Closest centroid to $x^{(i)}$
        end for
        for $k \leftarrow 1$ to K **do**
            $\mu_j = \frac{\sum_{i=1} m 1\{c^{(i)}=j\} x^{(i)}}{\sum_{i=1} m 1\{c^{(i)}=j\}}$ ▷ New centroid of cluster k
        end for
    }
end procedure

Visual explanation

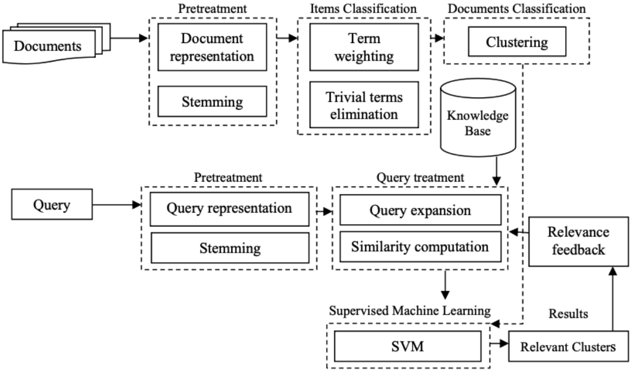


(a) (b) (c)

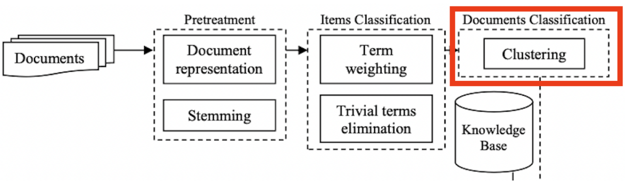
(d) (e) (f)

The proposed system [3] can be summarized by the following figure:

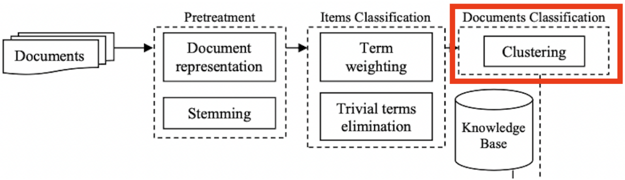
The proposed system [3] can be summarized by the following figure:



Documents clustering

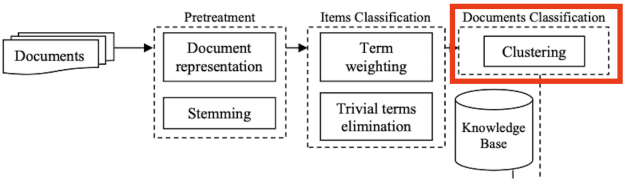


Documents clustering



First part consists of text pre-processing, followed by a vector representation of each document using stemmed terms. Next, there is the clustering phase.

Documents clustering

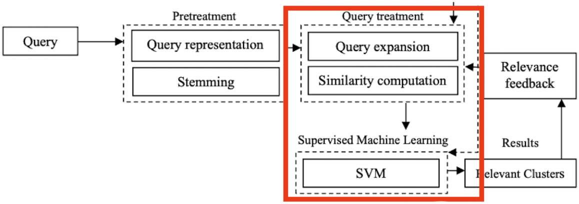


First part consists of text pre-processing, followed by a vector representation of each document using stemmed terms.
Next, there is the clustering phase.

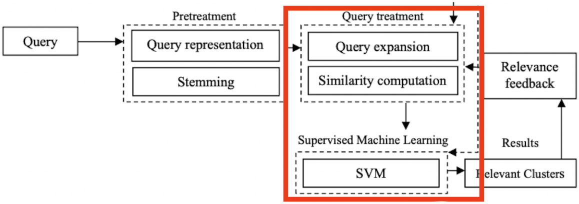
Using *k-means*, documents are classified into classes of similar vectors according to the selected terms (dimensions).

Finally, terms are classified as *Trivial*, *Decisive* or *Standard*. Only the last two are preserved and used as indicator of the class.

Queries classification

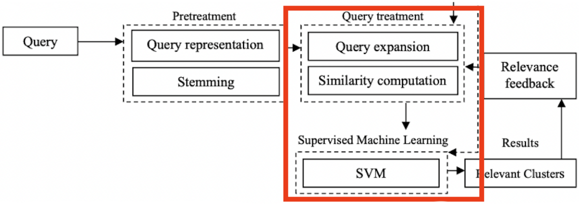


Queries classification



A given user query is first stemmed and represented by a vector (as for documents).

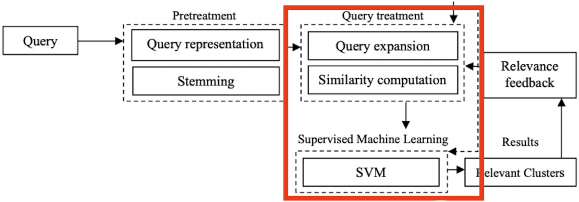
Queries classification



A given user query is first stemmed and represented by a vector (as for documents).

Then, SVMs are used to classify queries; for each cluster C_i there will be an SVM_i responsible of determining the membership of each query q_j to it (giving a probability p_i).

Queries classification



A given user query is first stemmed and represented by a vector (as for documents).

Then, SVMs are used to classify queries; for each cluster C_i there will be an SVM_i responsible of determining the membership of each query q_j to it (giving a probability p_i).

So, this process allows the selection of documents from the correct cluster (the one with the highest p_i).

Conclusions

We have seen the combination of use of SVMs and Clustering in the Information Retrieval field, with great results [3].

To conclude, SVMs are a great tool utilized for classification and regression.

Since they are less expensive than Neural Networks, usually researchers start experimenting using them before NN.

[1] Nello Cristianini and John Shawe-Taylor. “Support Vector Machines”. In: *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2000, pp. 93–124. DOI: 10.1017/CBO9780511801389.008.

[2] Vikramaditya Jakkula. “Tutorial on support vector machine (svm)”. In: *School of EECS, Washington State University* 37.2.5 (2006), p. 3.

[3] Hamid Khalifi, Abderrahim Elqadi, and Youssef Ghanou. “Support vector machines for a new hybrid information retrieval system”. In: *Procedia Computer Science* 127 (2018), pp. 139–145.

[4] B. Scholkopf et al. “Input space versus feature space in kernel-based methods”. In: *IEEE Transactions on Neural Networks* (1999). DOI: 10.1109/72.788641.

[5] Kristina P. Sinaga and Miin-Shen Yang. "Unsupervised K-Means Clustering Algorithm". In: *IEEE Access* 8 (2020), pp. 80716–80727. DOI: 10.1109/ACCESS.2020.2988796.