-UNITE!-
University Network for Innovation, Technology and Engineering

---

Aalto University

School of Science

unite!

Human-Centered Machine Learning Course

Support Vector Machines

Professor:
Cecilio Angulo

Authors:
Pablo De Ramon
Giang Le
Vasco Pearson
Amir Ingher
Sergi Garcia
Iván Quirante

Academic Course 2022-2023

1. Questions

Q1
For a given dataset, the SVM learning algorithm is applied leading to the optimal decision plane $H(x) = 3x_1 + 4x_2 + x_3 + 2$.

- Which is the value of the margin for this SVM?
- Find three vectors in the input space that could be support vectors of the plane.
- Demonstrate that the distance between parallel hyperplanes in SVM effectively is $\frac{2}{||\mathbf{w}||}$.

Solution
The margin of a given SVM is the distance between two parallel hyperplanes

$$H_{\pm}(x) = \{\mathbf{w}^{\mathsf{T}} \cdot \mathbf{x} + \gamma = \pm 1\}$$

that separate a set of training data into two classes, and is the value we want to maximize in order to find the optimal decision plane - or perceptron of optimal stability - $H(x)$ that lies exactly between $H_{\pm}(x)$.

Theory tells us that the margin m is equal to $\frac{2}{||\mathbf{w}||}$. We also know that the weight vector $\mathbf{w}$ is the normal vector of the decision plane. By simple inspection of $H(x)$ one can deduce the value of $\mathbf{w}$:

$$\mathbf{w} = (3\ 4\ 1)^{\mathsf{T}}$$

We measure its Euclidean norm as follows:

$$||\mathbf{w}|| := \sqrt{<\mathbf{w}, \mathbf{w}>} = \sqrt{3^2 + 4^2 + 1^2} = \sqrt{26} \approx 5.099$$

In consequence, the value of the margin is:

$$\boxed{m \equiv \frac{2}{||\mathbf{w}||} = \frac{2}{\sqrt{26}} \approx 0.392}$$

A SVM learning algorithm aims to find the solution to the following optimisation problem:

$$\max \quad \frac{2}{||\mathbf{w}||}$$
$$\text{s.t.} \quad y_i(\mathbf{w}^{\mathsf{T}}\mathbf{x_i} + \gamma) \geq 1 \text{ for } i = 1, \ldots, n$$

The $\mathbf{w}$ and γ that solve this problem determine the classifier: $\mathbf{x} \mapsto \text{sign}(\mathbf{w}^{\mathsf{T}}\mathbf{x} + \gamma)$, where sign(.) is the sign function. An important consequence of this geometric description is that those $\mathbf{x_i}$ that lie in the outer hyperplanes $H_{\pm}(x)$ - also known as support vectors - completely determine the max-margin hyperplane since they are extreme points of the domain polyhedron. Thus, any vector inside these planes could be a solution for the problem.
We search for three possible support vectors by finding data points that fulfill one of the following conditions:

$$H_+(x) = \{3x_1 + 4x_2 + x_3 + 2 = 1\} \rightarrow 3x_1 + 4x_2 + x_3 = -1$$
$$H_-(x) = \{3x_1 + 4x_2 + x_3 + 2 = -1\} \rightarrow 3x_1 + 4x_2 + x_3 = -3$$

Three possibilities are, for example:

$$\boxed{\mathbf{x_a} = (0\ 0 - 1)^{\mathsf{T}},\ \mathbf{x_b} = (0\ 0 - 3)^{\mathsf{T}},\ \mathbf{x_c} = (-1\ 0\ 0)^{\mathsf{T}}},$$

where $\mathbf{x_a} \in H_+(x)$ and $\{\mathbf{x_b}; \mathbf{x_c}\} \in H_-(x)$.

Proof. Now we're going to demonstrate that $m = \frac{2}{||\mathbf{w}||}$. First, let $H_{\pm}(x) = \{\mathbf{w}^\mathsf{T}\mathbf{x} + \gamma = \pm 1\}$ be the two outermost parallel hyperplanes of the SVM optimisation problem that maximize the margin between data points of different classes. Next, let's define $a = 1 - \gamma$ and $b = -1 - \gamma$ so that one can rewrite the previous subspaces as $H_+(x) = \{\mathbf{w}^\mathsf{T}\mathbf{x} = a\}$ and $H_-(x) = \{\mathbf{w}^\mathsf{T}\mathbf{x} = b\}$. Considering two data points $\mathbf{x}_+ \in H_+ \leftrightarrow \mathbf{w}^\mathsf{T}\mathbf{x}_+ = a$ and $\mathbf{x}_- \in H_- \leftrightarrow \mathbf{w}^\mathsf{T}\mathbf{x}_- = b$ whose distance is exactly m - that is, that both lie on the same line if it is drawn perpendicular to the decision plane - the relation between $\mathbf{x}_+, \mathbf{x}_-$ and $\mathbf{w}$ can be written as follows:

$$\mathbf{x}_+ = \mathbf{x}_- + \alpha \cdot \mathbf{w}, \quad \alpha \in \mathbb{R}^+$$

Computing the dot product with $< \mathbf{w}^\mathsf{T}, \cdot >$ on both sides:

$$\mathbf{w}^\mathsf{T}\mathbf{x}_+ = \mathbf{w}^\mathsf{T}\mathbf{x}_- + \alpha \cdot \mathbf{w}^\mathsf{T}\mathbf{w}$$

$$a = b + \alpha \cdot ||\mathbf{w}||^2$$

Since $a = 1 - \gamma$ and $b = -1 - \gamma$, one can easily obtain the value of α:

$$\alpha = \frac{a - b}{||\mathbf{w}||^2} = \frac{1 - \gamma - (-1 - \gamma)}{||\mathbf{w}||^2} = \frac{2}{||\mathbf{w}||^2}$$

Finally, the value the margin m can be obtained as follows:

$$\boxed{m = ||\mathbf{x}_+ - \mathbf{x}_-|| = ||\alpha \cdot \mathbf{w}|| = |\alpha| \cdot ||\mathbf{w}|| = \frac{2}{||\mathbf{w}||^2} \cdot ||\mathbf{w}|| = \frac{2}{||\mathbf{w}||}}$$

□

Q2
For the 1-dimensional training set $\mathcal{T} = \{(3;-1);(2;-1);(4;-1);(6;1);(7;1)\}$, find the kernel machine with polynomial kernel of lower degree able to separate the data set. Draw the resulting separation function.

Solution
It often happens that the sets to discriminate are not linearly separable in the feature space, as it was assumed in Q1. In order to solve this problem, a widely-known solution is to apply the **kernel trick**, which consists of finding a function $\varphi : \mathcal{X} \to \mathcal{Y}$ that maps a feature $\mathbf{x}$ into a high-dimensional space $\mathcal{Y}$ in which one can define a dot product as $K(x,y) =< \varphi(x), \varphi(y) >$. This latter bilinear mapping is known as **kernel**. In particular, the expression of the **polynomial kernel** can be written as follows:

$$K(\mathbf{x},\mathbf{y}) = (\mathbf{x}^{\mathsf{T}}\mathbf{y} + r)^d \,[1],$$

where d$\in \mathbb{R}^+$ is the degree and $r \in \mathbb{R}$ the kernel's coefficient. As can be seen, there exist infinite combinations of d and r values, that is why the team has opted to implement a parameter-search in `Python` (1) in order to find the combination that gets the best accuracy.

```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn import svm, datasets
from sklearn.model_selection import train_test_split, GridSearchCV, cross_val_score
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score

#Replicate datapoints to avoid errors in the fold split
Xp=np.array([3,3,3,2,2,4,4,4,6,6,6,6,7,7])
yp=np.array([-1,-1,-1,-1,-1,-1,-1,-1,1,1,1,1,1,1])

#Data pre-processing
X_train,X_val,y_train, y_val = train_test_split(Xp.reshape(-1,1),yp,test_size=0.5,
    random_state=0)

#Grid search
param_grid = {'kernel':['poly'], 'degree':[0,1,2,3,4], 'C':[1,10,100,1000], 'coef0'
    :[0,1,10,100,1000]}
gridSearchCV =GridSearchCV(estimator = svm.SVC(),param_grid = param_grid, cv=2,scoring="
    accuracy",refit=True,return_train_score=True)
gridSearchCV.fit(X_train,y_train)

print(gridSearchCV.best_estimator_)
print(gridSearchCV.best_score_)
#Output:
#SVC(C=1, coef0=0, degree=1, kernel='poly')
#1.0

#Plot initial and transformed data + separator hyperplane
def y(x):
      return -x+10
xplot=np.linspace(2,7,100)
yplot=y(xplot)
plt.plot([2,3,4],[0,0,0],'bo')
plt.plot([2,3,4],[2,3,4],'b*')
plt.plot([6,7],[6,7],'r*')
plt.plot([6,7],[0,0],'ro')
plt.plot(xplot,yplot,'g')

plt.grid()
plt.show()
```

Code 1: Python code

[1] Here $\mathbf{y}$ refers to a data point and not to a label.

As can be seen, the best polynomial kernel obtained is:

$$K(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y} = \varphi(\mathbf{x}) \cdot \varphi(\mathbf{y}),$$

which is a linear and homogeneous kernel. Finally, lines 27-38 of 1 plot the current (o) and transformed (*) data as well as the classification hyperplane (in green) that best separates the two classes (blue for +1 and red for -1).

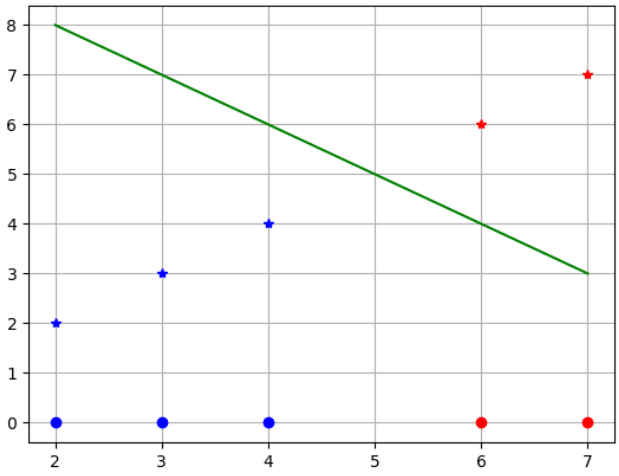

Fig. 1: Plot of (non-)transformed data and the linear classifier