LDATS2470 - Project 2023

Support vector machine on Parkinsons dataset

Teacher : **Prof. Christian Hafner**

Aliasghar Rostami Charati, 000544634
Rousseau Mathieu, 67001800
Gaetan Doucet, ? ? ?

UCLouvain
Belgium
12/05/2023

1 Introduction

Support Vector Machines (SVMs) are a popular and powerful classes of supervised learning models that are widely used in both classification and regression tasks. SVMs are based on the idea of finding the best possible decision boundary that can separate the different classes in the data. The boundary is selected such that the margin, which is the distance between the boundary and the nearest data points of each class, is maximized. SVMs are particularly useful when the number of features is much larger than the number of data points, as they can efficiently deal with high-dimensional data.

In addition to the linear SVMs, there are also non-linear SVMs, which use a kernel function to implicitly map the data to a high-dimensional space where it can be linearly separated. Some popular kernel functions include the radial basis function (RBF) kernel, polynomial kernel, and sigmoid kernel.

SVMs have several advantages over other classification methods. They are effective in high-dimensional spaces, have a regularizing effect, and can handle non-linear decision boundaries. SVMs also work well on both small and large datasets. However, SVMs can be sensitive to the choice of kernel function and the setting of its parameters, which can have a significant impact on the performance of the model.

In this project, we want to do SVM on Parkinsons dataset. Parkinson's disease is a neurodegenerative disorder that affects movement. It is caused by the loss of dopamine-producing cells in the brain. The Parkinson's Disease Classification dataset contains 195 observations and 24 variables, including the name of the subject, the status of the subject (healthy or with Parkinson's disease), and 22 biomedical voice measurements.

The goal of doing SVM on the Parkinson's dataset is to develop a machine learning model that can accurately classify patients with Parkinson's disease from healthy individuals based on the features extracted from their voice recordings. The SVM model can be trained using the labeled data (training data) in the dataset and used to predict the presence of Parkinson's disease in new, unlabeled data (testing data).

In the following, we divided our task into linear SVM, using linear kernel, and non-linear SVM that uses its own kernel which is RBF kernel. For each of them, we calculated the number of support vectors, the value of the objective function, and the proportion of correct classfications. Moreover, we visualized the results by projecting the data into the plane spanned by the first two principal components, along with its own interpretation for each part. In the end, we provided the conclusion part from whatever we reached out.

The aim of this project is to analyse a range of biomedical voice measurements from 31 people where 23 of parkinson disease. There are around 6 voice measurements per patient so that in total we have a collection of 195 observations. Each one contains severall voice measures that are detailled below. The 'status' column indicate is the patient has the parkinson disease or not.

2 Exploratory data analysis

3 SVM

3.1 Preparing the datas

One of the first thing we noticed in the exploratory data analysis was that the target classes are unbalanced (**??**). Indeed, among the 195 observations, we had 48 healthy people and 147 people having the parkinson disease. This is roughly a $3:1$ ratio. Having unbalanced classes can be misleading as the algorithm could have "good score" even if it only predicts the majority class.

In order to fix the imbalance, we upsampled the datas. The idea is to sample the datas with replacement by making multiple copy of observations belonging to the minority target class. The consequence is

having a perfectly balanced dataset with $1:1$ target class ratio.

Next, we split the dataset into a training and testing test. We kept 30% of the datas for testing purpose. We also standardized the data shifting the values to have a mean of zero and a standard deviation of 1. That is the following transformation,

$$S : x_{ij} \to z_{ij} = \frac{x_{ij} - \mu}{\sigma} \tag{1}$$

where μ and σ are respectively the mean and standard deviation of the jth-feature. This last step is important for algorithms such as SVMs that consider the distance between data. It prevent a given feature to dominate over the other so that all features have the same influence on the distance metric.

Plotting the standardized datas for the training and testing sets on the two first principal components, we notice there is no trivial separation of the datas. The two first principal components accounts for 73% of the variation in the datas (respectively 60.2% and 12.8% for the first and second principal component) which is correct but not completely representative of the reality. However, we can already assume that a linear kernel will not be effective. Let's try it first though.

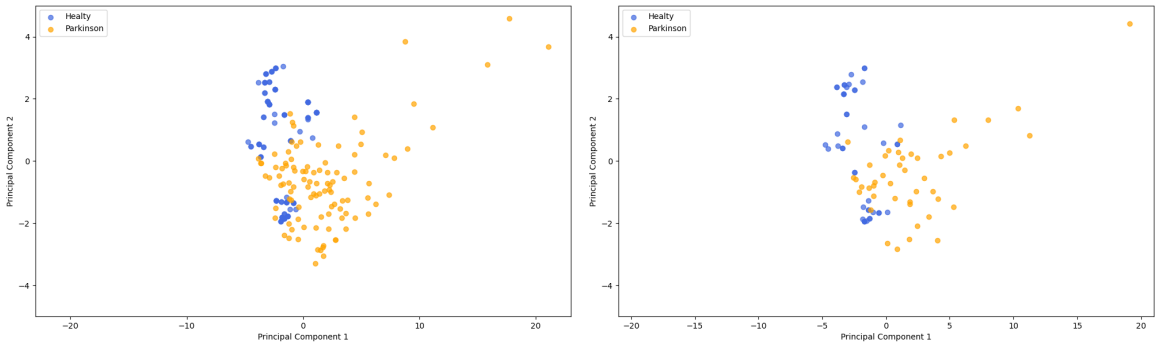

Figure 1 – *Plot of the training set (at left) and testing set (at right) on the two first principal components.*

3.2 Hard margin SVM

We first try to find a separating hyperplane using a linear kernel.

Let $D = \{ (\vec{x}_i, \vec{y}_i) \}_{i=1}^{n}$ be our set of data points with $x_i \in \mathbb{R}^d$ (where d is the dimension of the feature space) and $y_i \in \{ +1, -1 \}$.

Let $h : \mathbb{R}^d \to \mathbb{R}$, the equation of a plane defined $\forall x \in \mathbb{R}^d$ by,

$$h(\vec{x}) = \vec{w}^T \cdot \vec{x} + b \tag{2}$$

where $\vec{w} \in \mathbb{R}^d$ is a vector of weights and b is the bias.

A separating hyperplane is the set of points $\vec{x} \in \mathbb{R}^d$ that satisfy,

$$h(\vec{x}) = 0 \tag{3}$$

such that we have $\forall \vec{x}_i \in D$ the following inequality,

$$y_i \cdot h(\vec{x}_i) \geq 1 \tag{4}$$

The hyperplane should yields the maximum margin among all possible separating hyperplanes, that is the parameters $\vec{w}$ and b are the one that maximize $1/||\vec{w}||$.

An equivalent formulation is to say we want to minimize the following objective function,

$$\min_{\vec{w},b} \frac{||\vec{w}||^2}{2} \tag{5}$$

subject to the linear constraint **??**.

The found hyperplane gets us an accuracy of **0.842** on the *testing set*. However, a better way to assess the quality of a classifier is to look at the confusion matrix.

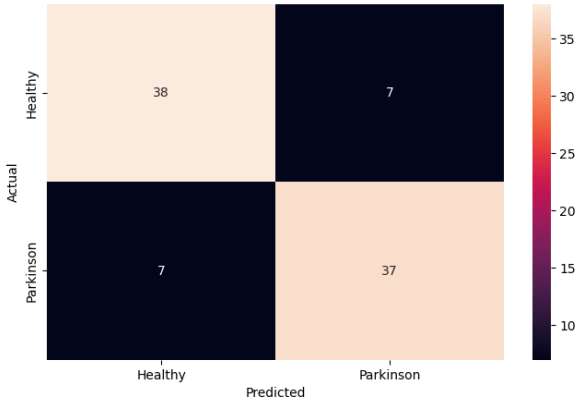

Figure 2 – *Confusion matrix on the testing set for a linear SVM*

We notice that we have effectifely $\approx 15\%$ of missclassification with 7 false positive and 7 false negative. Therefore, we can conclude that a linear kernel is not ideal. On the following plot, we project the datas points of the training set on the two first principal components along with the support vectors. We also show the separating hyperplane and the margins. We have a confirmation of our first intuition, a linear kernel is not effective in separating these datas.

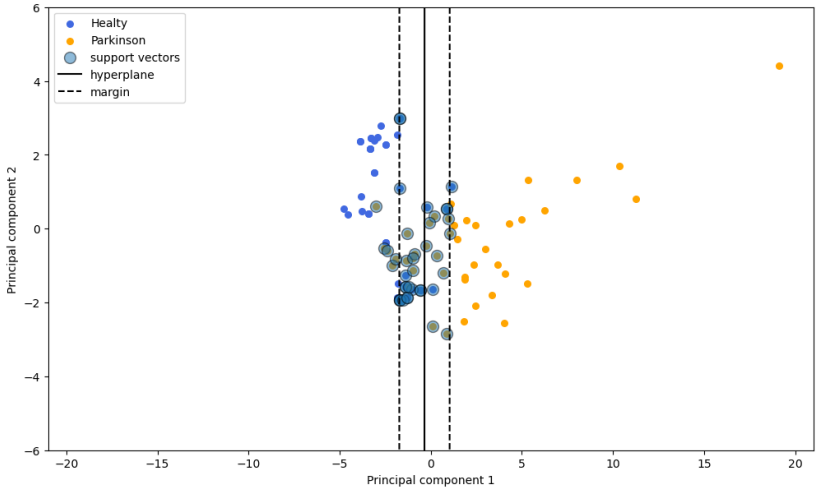

Figure 3 – *Margins and separating hyperplane for a linear kernel SVM*

3.3 Hard margin SVM

Despite the poor performance of the linear kernel. We can still try to improve it by introducing a *slack variable* ξ_i, $i = 1, \ldots, n$ that for each data point x_i indicates how much it violates the separability

condition [1].

$\forall \vec{x}_i \in D$ the inequality becomes,

$$y_i \cdot h(\vec{x}_i) \geq 1 - \xi_i \tag{6}$$

The goal is then to minimize the same objective function as before minus a penalty term,

$$\min_{\vec{w},b,\xi_i} \frac{||\vec{w}||^2}{2} - C\sum_{i=1}^{n}(x_i)^k \tag{7}$$

where $C \in \mathbb{R}$ is a regularization constant and $k \in \mathbb{R}$.

This objective function is subject to the constraint above (**??**) as well as $x_i \geq 0 \ \forall \vec{x}_i \in D$.

Performing a grid search with 5-fold cross-validation, we found an optimal regularization constant $C = 2.2$ giving a mean accuracy of **0.854** on the *validations sets* but an accuracy of **0.831** on the *testing set*. That is a slighlty worse result than the hard margin case. Let's look at the confusion matrix,

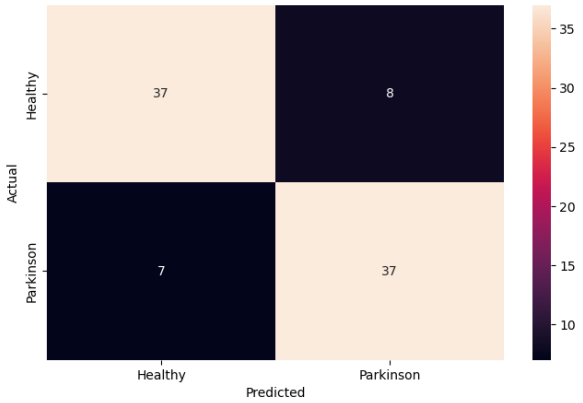

Figure 4 – *Confusion matrix on the testing set for the soft margin case*

We notice we have now 8 false positive instead of 7 so the linear classifier is definitely not helpful in separating these datas.

3.4 Kernel trick

A way to fix this problem is to map the data points in a high-dimension space by performing a non-linear transformation,

$$\phi : \mathbb{R}^d \to \mathbb{R}^l, \quad \vec{x}_i \to \phi(\vec{x}_i) \tag{8}$$

By this way, there is far more probability for the data to be linearly separable and because we perform a non-linear transformation, a linear separation in the feature space correspond to non linear decision region in the original data space.

We used a Gaussian radial basis function for the kernel,

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j) \tag{9}$$

$$= \exp\left(-\gamma(\vec{x}_i - \vec{x}_j)^2\right) \tag{10}$$

1. The separability condition ensures that the point is at least $\frac{1}{||\vec{w}||}$ from the hyperplane.

The γ parameter has to be found by cross-validation. Performing a grid search on the C and γ parameters with a 5-fold cross-validation, we found the optimal parameters to be $C = 0.455$ and $\gamma = 0.357$. With these, we get a mean accuracy score of **0.976** on the *validation sets* and an accuracy score of **0.966** on the *testing set*. That's way better than using a linear kernel. We proove it by showing the confusion matrix,

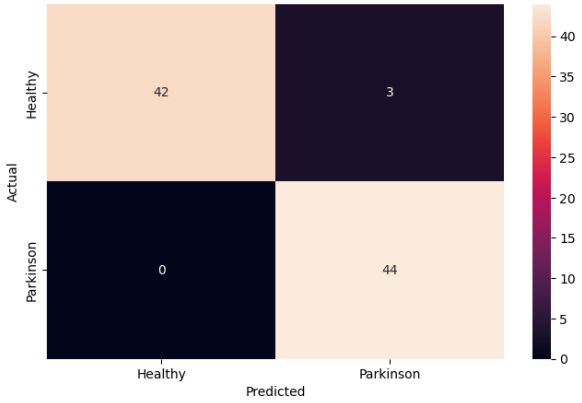

Figure 5 – *Confusion matrix on the testing set for a Gaussian kernel SVM*

Among the 44 people with parkinson disease in the training set, all are correctly classified. On the other hand, among the 45 healthy people, we still have 3 missclassified as healthy people. This classifier perfectly predict sickness but has some difficulties to predict healthyness. In a real world case, this situation is better than the converse.

Below, we can observe the decision region for the classifier.

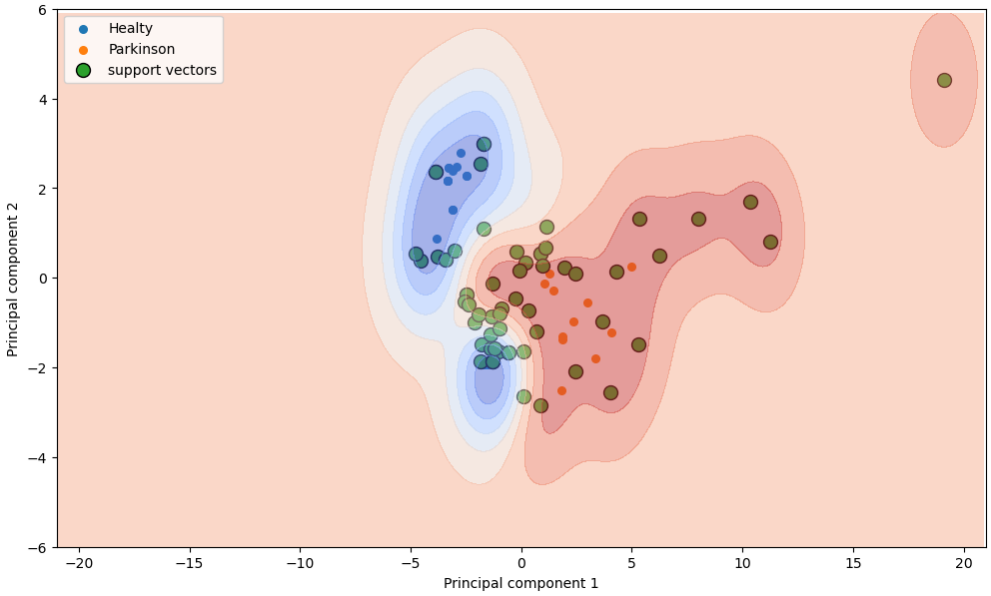

Figure 6 – *Decision region for a Gaussian RBF kernel SVM (sk-learn implementation)*

We also tried to implement our own SVM algorithm with the Gaussian RBF kernel by performing a

stochastic gradient ascent. Surprisingly, we get even better score than the *sk-learn*[2] implementation of SVM classifier. Indeed, using the same parameters for C and γ, we get a score of **0.988** on the *testing set* and the following confusion matrix. Notice we only have a single one missclassification as one healthy people is classified as sick.

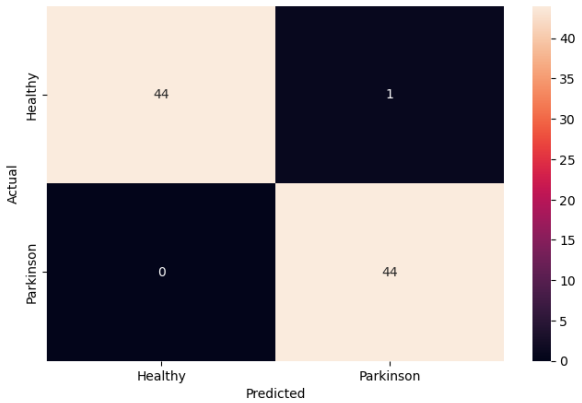

Figure 7 – *Confusion matrix on the testing set for a SVM with Gaussian RBF kernel (custom implementation)*

2. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC

Appendix

Description of the different variables

The **response variable** is *status* : 1 if the subject has the Parkinson disease and 0 if not.

The **explanatory variables** are the following :

- *name* : the suject name along the recording number.
- *mdvp.fo* : the **average** local fundamental frequency (Hz).
- *mdvp.fhi* : the **maximum** local fundamental frequency (Hz).
- *mdvp.flo* : the **minimum** local fundamental frequency (Hz).
- *mdvp.jitter_ perc (%)*, *mdvp.jitter_ abs (Abs)*, *mdvp.rap*, *mdvp.ppq*, *jitter.ddp* : these are several measures of variation in fundamental frequency.
- *mdvp.apq*, *mdvp.shimmer*, *mdvp.shimmer_ db*, *shimmer.apq3*, *shimmer.apq5*, *shimmer.dda* : these are several measures of variation in amplitude.
- *nhr*, *hnr* : 2 measures of noise to tonal components in the voice.
- *rpde*, *d2* : 2 nonlinear dynamical complexity measures.
- *dfa* : signal fractal scaling exponent.
- *spread1, spread2*, *ppe* : 3 nonlinear measures of fundamental frequency variation.