

Machine Learning in Healthcare

Homework 3



1 Clustering

In the lecture we saw the K-means algorithm for clustering. It tries to minimize the Euclidian metric between the examples and some point in space which named "centroid". Other methods try to minimize dissimilarities between the pairwise data examples. A classical algorithm which was designed to handle pairwise data is the K-medoid. This algorithm seeks to find a set of cluster-representatives (named medoid) in the dataset and assign other examples to them. The algorithm randomly picks a k-set of medoid from the data and assigns points to each medoid based on their L_1 distances to that medoid. Then, it iteratively tries to improve the assignment by swapping assigned medoid points with non-medoid points until the energy of the entire system (which is measured by the sum of distances between medoid points and their assigned data points) is minimized.

- a. Is K-medoid more robust to noise (or outliers) than the K-means algorithm?

Both algorithms are partitional, which means both of them decompose a dataset into a set of disjoint clusters. While K-means aims to minimize the total squared error, the K-medoid algorithm goal is to minimize the sum of the differences between points in the cluster to the point assigned as the center of the cluster.

To do that, the K-medoid algorithm will use a medoid point, which is the point located closest to the cluster center, as a reference point instead of using the mean of all the points in the cluster.

Since we know that the calculation of the mean value (average) is very sensitive to extreme values (outliers) and changes in the summed value (noise), we can deduce the K-means algorithm is less robust.

- b. Prove that for the 1D case ($x \in \mathbb{R}^1$) of K-means, the centroid (μ) which minimizes the term $\sum_{i=1}^m (x_i - \mu)^2$ is the mean of m examples.

If we put the summation given as function, we can find its' derivative by μ and compare it to zero in order to find the value of μ that will minimize it:

$$f(\mu) = \sum_{i=1}^m (x_i - \mu)^2$$

$$\frac{\partial f(\mu)}{\partial \mu} = -2 \sum_{i=1}^m (x_i - \mu) = 0 \rightarrow m \cdot \mu = \sum_{i=1}^m x_i \rightarrow \mu = \frac{1}{m} \sum_{i=1}^m x_i = \bar{x}$$

We can use the second derivative to make sure this value will generate a minimum:

$$\frac{\partial^2 f(\mu)}{\partial \mu^2} = 2 \Rightarrow 0 \rightarrow \text{Minimum point}$$

Bonus: Prove that the centroid (practically, the medoid) which minimizes the term $\sum_{i=1}^m |x_i - \mu|$ is the median of m examples given that μ belongs to the dataset.

If we put the summation given as a function and find its' derivative by μ and compare it to zero, we can find the μ value that will give the minimum value:

$$f(\mu) = \sum_{i=1}^m |x_i - \mu|$$
$$\frac{\partial f(\mu)}{\partial \mu} = \sum_{i=1}^m \text{sign}(x_i - \mu) = 0$$

We got that the derivative is a summary of +1s and -1s, depending on if the centroid is bigger or smaller than each points' value. In order for this equation to be equal to zero, it means that μ needs to be equal to a number that exactly half of the samples are smaller than, and that is bigger than exactly half of the samples. This is the definition of a median.

2 SVM

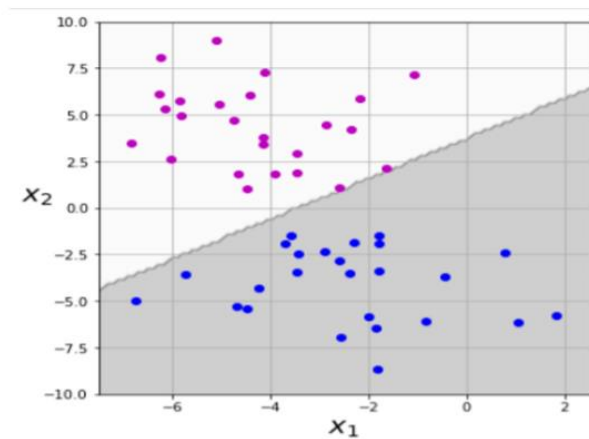
In the following figures you can see a visualization of SVM running with different settings (kernels and parameters) as follows:

The settings that were used are as follow:

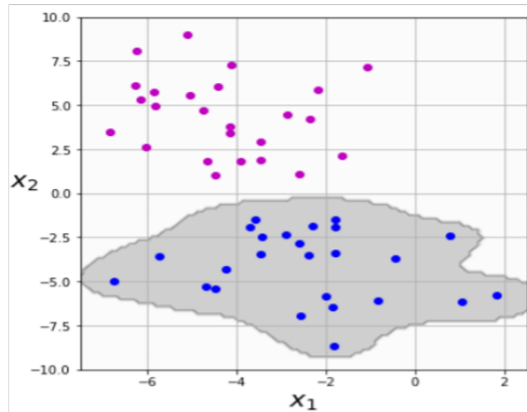
1. Linear kernel with $C = 0.01$.
2. Linear kernel with $C = 1$.
3. 2nd order polynomial kernel.
4. 10th order polynomial kernel.
5. RBF kernel with $\gamma = 0.2$.
6. RBF kernel with $\gamma = 1$.

Match every image (labeled by a capital letter) to its' setting (number).

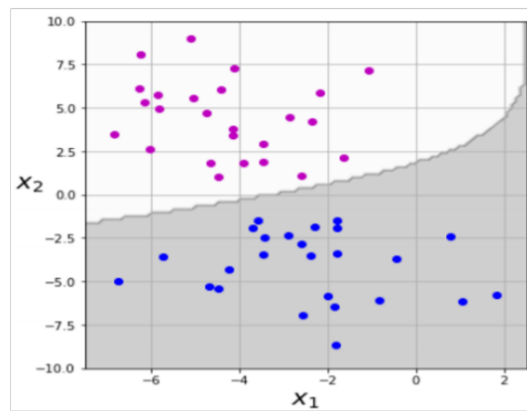
- A. We can see the separation line is approximately a straight line, which points to it being classified by a linear kernel. Since we know that the C parameter tells the SVM optimization how important it is to avoid misclassification. For larger values of C , the optimization will choose a smaller-margin hyperplane and for smaller values of C , it will choose bigger-margins. Since in this case, there are points on the decision boundary (bigger margins), it leads to the conclusions of smaller C value. → Linear kernel with $C = 0.01$ (1).



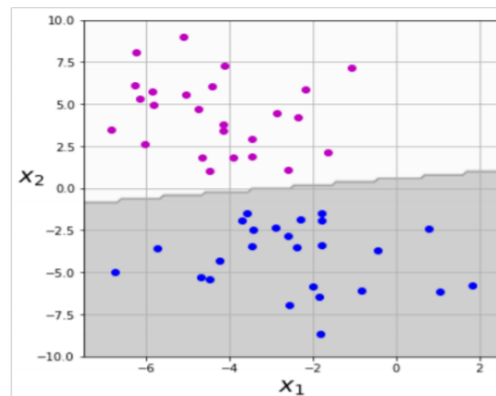
- B. Since this is a closed rounded shape, it leads to conclude this is classified by an RBF kernel. We know that the γ parameter is the inverse of the standard deviation of the kernel, or in other words, the similarity measure between two points. So, a small γ value will allow two points to be classified the same even if they are far from each other, and a larger γ will not allow that. In this case, we can see the margins of the shape are forming a smaller labeling area for one of the sections (therefore higher per precision and smaller deviation error). → RBF kernel with $\gamma = 1$ (6).



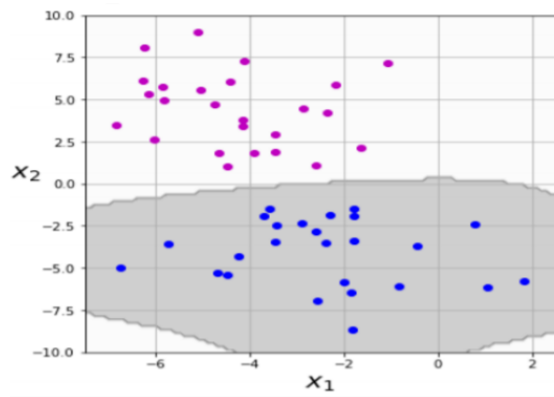
- C. As the separation is not done by a straight line nor a closed shape, it can be deduced it has been classified by polynomial kernel. The higher the polynomial order is, the more flexible the decision boundary is. In this classifier the boundary is relatively simple, so the polynomial order must be relatively low. → 2nd order polynomial kernel (3).



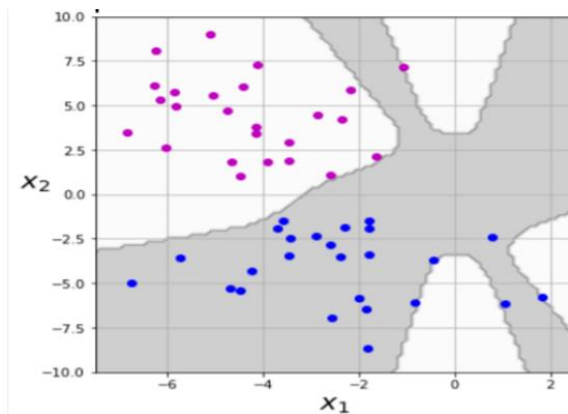
- D. The decision boundary in this case can be referred to as a straight line, it leads to the conclusion of a linear kernel. In this case, the margins are smaller so the C value must be bigger. → Linear kernel with C = 1 (2).



- E. This case presents a closed rounded shape; hence it is recognized as an RBF kernel. The shape is not closing on the samples it contains in a tight manner, behavior appropriate for lower γ values. \rightarrow RBF kernel with $\gamma = 0.2$ (5).



- F. This classifier presents a decision boundary which is not straight, so it cannot be a linear kernel. The classifier does not create a closed rounded shape, so the RBF kernel option is disqualified. The only option left is polynomial kernel. The boundary is very flexible and complex, what hints to higher polynomial order. \rightarrow 10th order polynomial kernel (4).



3 Capability of generalization


Ockham's razor states that "Non sunt multiplicanda entia sine necessitate". This is known as the law of parsimony and its' translation to English is "Entities are not to be multiplied without necessity". This concept, which is attributed mostly to the English Franciscan friar William of Ockham, basically means that the simplest explanation is usually preferred.


A more modern variation of this concept (and a much more readable one) is attributed mostly to Albert Einstein and it states that "Everything should be made as simple as possible but not simpler". This balance is a major guideline in science in general and in data science in particular.

We saw in the tutorial two methods of choosing a parsimonious model for K-means. In GMM, there is another criterion to do so and it is known as "Akaike information criterion" (AIC). It is composed of two terms and defined as follows:

$$AIC = 2p - 2 \ln \hat{L}$$


where p is the total number learned parameters and \hat{L} is the estimated likelihood given these parameter.

-  a. What is the scientific term of the balance that Einstein meant to in machine learning aspect?

The balance referred to is the trade-off between generalization  and simplification (which can be translated to the number of features).



On one hand, the goal to have a generalized model that could process new samples and data and predict accurately. On the other hand, the golden rule is to avoid overloading the model with too many features that will make it more complex, hence harder to apply on new data and potential longer processing time.

-  b. How does each of the terms in AIC affect the terms of the balance defined in a?

As AIC is a criteria that we aim to get lower values for, the $2p$ element in the equation is a way to decrease the rating of the model. Since p signifies the number of features, the more features the model will use, the worse the AIC score would be. On the other hand, the $\ln(L)$ element that represents the accuracy of the model could lower the value, hence improve the AIC score.



So, the balance between those two elements encourages the model to avoid overfitting and over complicated without harming the most important factor which is being able to provide true predictions.

- c. What are the two options that are likely to happen if this balance was violated?

If the balance is violated, the model can be over generalized or unable to generalize.

If the model has not trained on a sufficient amount of data or has too few features, it would be underfitted, the variance will be low, and the bias will be high and the model would be useless since it will not be able to make precise predictions.

If the model is trained on too many features, it would be overfitted, the variance will be

high, and the bias will be low. In this case the model would be too specific, and this will prevent it from being able to predict conclusions for new data.

- d. What are we aiming for with the AIC? Should it be high or low?

The goal is to create a model that would as precise and accurate as possible, which means bigger values for the $\ln(L)$ element, while also using the minimal number of features to avoid overfitting, which translates to smaller values for the p element. When combining those two considerations together, the result is that we should aim for a lower AIC score.