HW3

Arseny Belousov 961152006

Q1

a.

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

The K-medoid could be more robust to noise and outliers than the k-means since it minimizes the sum of common paired dissimilarities instead of the sum of the squared Euclidean distances. The possible choice of the dissimilarity function is very rich.

b.

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

We can solve for the m[th] centroid which minimizes the term $\sum_{i=1}^{m}(x_i - \mu)^2$ by differentiating and setting it equal to 0:

$$\frac{\partial}{\partial \mu}\sum_{i=1}^{m}(x_i - \mu)^2 = \sum_{i=1}^{m}\frac{\partial}{\partial \mu}(x_i - \mu)^2 = 2(x_m - \mu) = 0 <=> \ \mu = \frac{\sum_{i=1}^{m} x_i}{m}$$

Thus, we can see that the best centroid for minimizing the term is the mean of m examples.

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

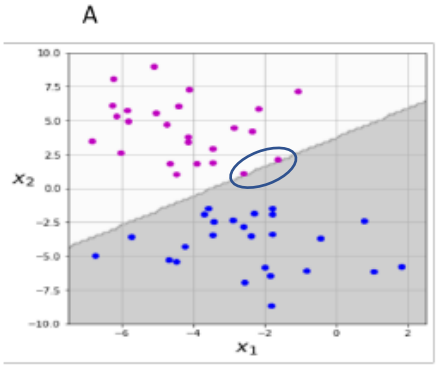
In the same way if we differentiate and set it equal to 0, we can get:

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

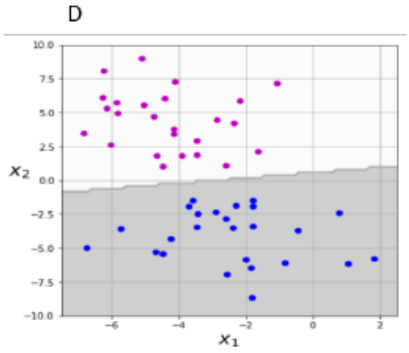
If we solve for x_m we find that x_m = median, the median of the m examples. The median of a group of points is straightforward to compute and less susceptible to distortion by outliers.

Q2

1. Here we have a linear kernel with C=0.01. As we see on figures, we have only two linear options: **A** and **D**. But which one? As we know from lectures, tutorials and see from Google, the bigger C ⇔ the more penalty SVM gets when it makes classification and thus more degree of tolerance. In other words, we see that points places closer to the line at the plot A than at the plot D => it means that for ***Linear kernel with C=0.01*** the relevant plot is the ***plot A***.



2. As the conclusion from the previous exploration, for ***Linear kernel with C=1*** the relevant plot is the ***plot D***.



3. SVM with a polynomial kernel can generate a non-linear decision boundary using polynomial features. We can see that plots with solid nonlinear separations are in Figures C, F and, possibly, E, but the last option is not suitable because the polynomials do not behave as shown in plot E. Thus, there are two options left: C and F. The second order polynomial should be flatter than the 10th order polynomial for obvious reasons: if we consider f (x) = x^2 and f (x) = x^10, for x = (- 2, -1, 0, 1, 2) we will see that an increase in the order leads to a scatter of the values of the function: $f(x)=x^2$: [4,1,0,1,4] and $f(x)=x^{10}$: [1024,1,0,1,1024]. From that example we can understand that flatter plot is the plot C and thus the ***plot C*** is the relevant for the ***2nd order polynomial kernel***.
4. As we see from the exploration of the previous part, the ***plot F*** must be the one relevant for the ***10th order polynomial kernel***.
5. At this point, we have only two unallocated figures left: figure B and figure E. To understand which figure will correspond to this example, we need to know that the RBF kernel is characterized by a squared Euclidean distance between the two feature vectors. Thus, we can understand that the larger the gamma value, the closer the area border is to the target set of values. Based on the above, we conclude that for the ***RBF kernel with*** $\gamma = 0.2$ the relevant plot is the ***plot E***.
6. As the conclusion from the exploration above and as a logical conclusion we can tell that for the ***RBF kernel with*** $\gamma = 1$ the relevant plot is the ***plot B***.

Q3

a. As I understand it, we want to get a model that is both accurate enough and not infinitely long in processing time. In addition, as we saw in the presentation of lecture C06, slide 21, simpler models allow the model to avoid overfitting.
b. ***Log likelihood*** is a measure of how likely it is that someone will see their observed data given a model. The model with the highest likelihood is the one that best "fits" the data. The number of parameters ***P*** characterizes the probability of overfitting the model (direct proportionality)
c. We can get ***overfit*** and ***underfit***: in the first case, this will mean that the model will be too accurate, and this will lead, accordingly, to overfitting, which ultimately will affect the work time and the lack of specifics we need; in the second case (with underfit), we can, on the contrary, get a model, which then will not give us sufficient accuracy, but will work with errors accordingly.
d. **AIC** is *low* for **models** with *high log-likelihoods* (the model fits the data better, which is what we want), but adds a penalty term for models with higher parameter complexity, since more parameters means a model is more likely to overfit to the training data.