# HW3

## Sagi Ben Itzhak

## Q1 Clustering

Part a

K-means is an iterative clustering algorithm that aims to divide a set of examples into a number of clusters (set by the hyper parameter K). At each iteration, the algorithm tries to minimize the Euclidian metric between the examples affiliated with a certain cluster (in the current iteration) and some representative point in space called the “centroid” that is associated with the relevant cluster. To do so, the algorithm updates each centroid by calculating the mean of all the examples affiliated with its’ cluster at the end of each iteration. This algorithm is sensitive to outliers (and noise) because its use of means, which are significantly influenced by outliers and extreme values. In the K-medoid algorithm the representative points in space, called “medoid”, associated with each cluster are assigned from the dataset itself. The algorithm tires to minimize the sum of the distances between each medoid point and the examples affiliated with its cluster by swapping assigned medoid points with non-medoid points. Similar to medians, because this algorithm only assigns medoid points from the examples in the dataset, it is more robust to outliers (and noise).

Part b

To find the centroid ($\mu$) that minimizes the term $\sum_{i=1}^{m}(x_i-\mu)^2$ for the 1D case: $x \in R^1$, we will find the derivative of this term with respect to $\mu$ and compare to 0:

$$\frac{d}{d\mu}\sum_{i=1}^{m}(x_i-\mu)^2 = -2\sum_{i=1}^{m}(x_i-\mu) = 0 \quad\rightarrow\quad \sum_{i=1}^{m}(x_i-\mu) = 0 \quad\rightarrow\quad \sum_{i=1}^{m}x_i - \mu m = 0 \quad\rightarrow$$

$$\mu = \frac{1}{m}\cdot\sum_{i=1}^{m}x_i \rightarrow mean\ of\ examples$$

To make sure that this is a minimum we will verify that the second derivative is positive:

$$\frac{d^2}{d\mu^2}\sum_{i=1}^{m}(x_i-\mu)^2 = \frac{d}{d\mu}\left(-2\sum_{i=1}^{m}(x_i-\mu)\right) = 2m > 0$$

Bonus:

In this case we want to prove that the medoid $\mu$ $(where\ \mu \in \{x_1, \dots x_m\})$ that minimizes the term $\sum_{i=1}^{m}|x_i-\mu|$ is the median of the m examples.

First I will prove for the cases of odd number of examples:

For a certain $\mu$ from the set of m examples lets define $n_1$ as the number of examples that are smaller (in value) than $\mu$, and $n_2$ as the number of examples that are greater than $\mu$. If we sort the examples in ascending order (this does not change the sum) we will get the following:

$$\sum_{i=1}^{m}|x_i-\mu| = \sum_{i=1}^{n_1}|x_i-\mu| + \mu - \mu + \sum_{i=m-n_2}^{m}|x_i-\mu| = \sum_{i=1}^{n_1}-(x_i-\mu) + \sum_{i=m-n_2}^{m}(x_i-\mu) =$$

$$\sum_{i=1}^{n_1}-x_i + \sum_{i=m-n_2}^{m}x_i + n_1\mu - n_2\mu$$

Now lets define a new continuous parameter $\mu_c$ (where $\mu_c$ is continuous, therefore does not necessarily belong to the dataset) and find the $\mu_c$ that minimizes the term $\sum_{i=1}^{m}|x_i-\mu|$, we will find the derivative of the expression above with respect to $\mu_c$ and compare to 0:

$$\frac{d}{d\mu_c}\left(\sum_{i=1}^{n_1}-x_i + \sum_{i=m-n_2}^{m}x_i + n_1\mu_c - n_2\mu_c\right) = n_1 - n_2 = 0 \rightarrow n_1 = n_2$$

From this we can conclude that the $\mu_c$ that brings the term$\sum_{i=1}^{m}|x_i-\mu|$ to a minimum satisfies the condition $n_1 = n_2$. Therefore, in the cases of odd number of examples, $\mu_c$ and $\mu$ ($\mu \in \{x_1, \dots x_m\}$) that bring the term $\sum_{i=1}^{m}|x_i-\mu|$ to a minimum are the same - $\mu_c = \mu$ . In the case where $n_1 = n_2$ the $\mu$ is the median of the dataset since the example chosen to be $\mu$ has an equal number of examples with higher and lower values than it, therefore the medoid $\mu$ which minimizes the term $\sum_{i=1}^{m}|x_i-\mu|$ is the median of the set of examples (dataset).

For even number of examples we can show that we will get the same outcome but with two candidates since the $\mu_c$ that will bring the term $\sum_{i=1}^{m}|x_i-\mu|$ will not belong to the dataset, and will be located somewhere between the two examples in the middle. In this case we will calculate the term $\sum_{i=1}^{m}|x_i-\mu|$ for both candidates for $\mu$ and choose the example that brings the term to a lower value.

## Q2 SVM

For linear SVM classifiers the hyper parameter C is used to define the magnitude of the regularization. Increasing the value of C will result in wider margins and a higher penalty for misclassifications (and examples correctly classified but are inside the margins). Decreasing C will "allow" more examples to fall inside the margins or to be misclassified. By looking at the figures we can identify two linear classifiers (decision boundary is a linear line), A and D. In figure A we can see that there are 2 example (in purple) that are right on the border line (or very close to it), while the closest blue examples are not as close to the border line compared the 2 points mentioned above. Because the margins set by the support vectors in linear SVM are equally distanced from the decision boundary (meaning that the decision boundary is exactly in the middle between the two margins) this indicates that in this case there are examples that are located inside (between) the margins set by the support vectors and are therefore penalized for it. This outcome is more typical for classifiers with lower regularizations (especially when the data is completely linearly separable) since the penalty is lower. In figure D this is not the case and the regularization seem to be more strict. From these conclusions we can safely assume that **figure A** is more suitable for **Linear kernel with C=0.01**, and **figure D** is more suitable for **Linear kernel with C=1**.

Non-linear RBF SVM classifiers are commonly used for classifications with closed boundary regions (in 2D: circle, oval, etc.). This type of classifier is suitable for the classifiers depicted in figures E and B. To understand the influence of $\gamma$ on our classifier we will need to look at the Kernel used in RBF:

$$K(x_i, x_j) = e^{-\gamma||x_i - x_j||^2} = e^{-\frac{||x_i - x_j||^2}{2\cdot\sigma^2}}$$

From the equation above we can see the relation between $\gamma$ and the variance of the gaussian: $\gamma = \frac{1}{2\cdot\sigma^2}$ . Increasing $\gamma$ will decrease the variance, and vice versa. Low gamma (high variance) means that far away points ($||x_i - x_j||^2$) will also influence the boundary (classification) resulting in larger region (area) and smoother boundary lines. By looking at the classifiers depicted in figures E and B we can see that in figure E the region of the classifier is larger, and the borders are smoother compared to the figure B, therefore the $\gamma$ in figure E is smaller. From this we can conclude that **figure B** is more suitable for **RBF kernel with $\boldsymbol{\gamma = 1}$**, and **figure E** is more suitable for **RBF kernel with $\boldsymbol{\gamma = 0.2}$**.

In the two remaining figures (C and F) it is visible that the decision boundary is not linear, but it does not form a closed classification region (at least not in the scope of the figure). This indicates that the kernels used in these classifiers are probably polynomial. In general, for polynomial kernels, there is a correlation between the order of the polynomial and the complexity of the decision boundary. Increasing the order of the polynomial will probably result in a more complex decision boundary. The decision boundary in figure F seems to be more complex compared to the decision boundary in figure C. From this we can assume that **figure C** is more suitable for **$\boldsymbol{2^{nd}}$ order polynomial kernel** and **figure F** is more suitable for **$\boldsymbol{10^{th}}$ order polynomial kernel**.

Conclusions:

| Figure | Kernel |
|---|---|
| Figure A | Linear kernel with C=0.01 |
| Figure B | RBF kernel with $\gamma = 1$ |
| Figure C | $2^{nd}$ order polynomial kernel |
| Figure D | Linear kernel with C=1 |
| Figure E | RBF kernel with $\gamma = 0.2$ |
| Figure F | $10^{th}$ order polynomial kernel |

# Q3 Capability of generalization

Part a

The term of balance that suits the saying said by Einstein in machine learning is **Generalization**. A good generalization in a model can be achieved by a reaching a good balance between the bias and the variance in our model.

### Part b

Since p is the total number of learned parameters, the term 2p represents the complexity of the model. In general, as we increase the number of parameters (p) the model will usually perform better on the training set (low bias), but if the complexity is too high the variance will be high and the generalization attribute of the model will be inferior to models with less complexity, meaning that it might perform better on the training examples, but when applied on other examples (outside the training) the performance of the less complex models will be better. As for the second term, $2\ln{(\hat{L})}$, the likelihood $\hat{L}$ measures the “goodness of fit” (how good our models fit our examples) in terms of probability. The expression $2\ln{(\hat{L})}$ will increase (become less negative since $\hat{L}$ is between 0 and 1) with the increase of $\hat{L}$, which will lead to the decrease of AIC.

### Part c

- Underfitting – the model is too simple resulting in poor performance in both the training and the test sets.
- Overfitting – the model is too complex resulting in poor performance in examples outside the training set (poor generalization), while achieving very good performance on the training examples.

### Part d

To choose the optimal model we aim to find the model with the lowest AIC. The AIC score rewards “goodness of fit” – how good our models fit our examples (assessed by the likelihood $\hat{L}$) by decreasing the AIC with increase of $\hat{L}$,  but penalizes (increase of AIC) the models with a penalty that increases with the number of learned parameters in order to avoid overfitting.

References:

https://www.quora.com/What-are-C-and-gamma-with-regards-to-a-support-vector-machine

https://www.vebuso.com/2020/03/svm-hyperparameter-tuning-using-gridsearchcv/

https://en.wikipedia.org/wiki/Akaike_information_criterion