HW3-Tom Levin 307842419

Q1 Clustering:

a.

Yes, K-medoid is more robust to noise (and outliers) than the K-means algorithm. This is true due to the minimalization of the sum of pairwise dissimilarities that K-medoid performs- instead of the minimalization of the sum of squared Euclidean distances that performed by the K-means algorithm.

K-medoids provide much more interpretations to the centers of the clusters (medoids that came from our data) than the K-means algorithm (Furthermore, we can use the K-medoids algorithm with similarity measures while K-means may not converge).

For example, lets look at the 1D series: [0, 2, 4, 6, 1000000] to visualize the robustness of K-medoids algorithm. The median of this series, which is comparable to the medoid (see my proof in the bonus section) is equal to 4. They are ignoring the outlier while the mean of this series is equal to 200,002.4. This is not surprising because the mean value is known to be more sensitive to outliers than the median.

(There is another similar algorithm that minimizing the sum of L1 distances and called K-medians)

b.

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

deriviate:

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

$$\Rightarrow \boxed{\mu = \frac{1}{m}\sum_{i=1}^{m} x_i} \quad \textit{which is the mean of } m \textit{ examples}$$

deriviate again to make sure that this is indeed a min:

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

$$\Rightarrow \mu = \frac{1}{m}\sum_{i=1}^{m} x_i \quad \text{minimize the term} \quad \sum_{i=1}^{m}(x_i - \mu)^2$$

Bonus:

for **sanity check** lets start with a specific case. For the series s=(1,2,3,4,8) we know that 3 is the median. Lets see if it minimize the sum of L1 distances (between this series to any given x).

$$\sum_{i=1}^{m}|x - s_i| = |x-1| + |x-2| + |x-3| + |x-4| + |x-8|$$

From the Triangle Inequality:

$$\begin{cases} |8-x| + |x-1| \le 7 \\ |4-x| + |x-2| \le |4-2| = 2 \end{cases} \Rightarrow \sum_{i=1}^{m}|x - s_i| \le 2 + 7 + |x-3| \Rightarrow \begin{cases} \sum_{i=1}^{m}|x - s_i| = 7 & , x = 3 \\ \sum_{i=1}^{m}|x - s_i| > 7 & , x \ne 3 \end{cases}$$

The median does minimize the sum of L1 distances for our example.

Lets proceed to our proof:

At first, I will sort x_i in an increasing order when $x_1 < x_2 < x_3 < ... < x_m$.

For the case that m is **odd**, we can say that μ belong to a specific interval in the data, i.e. $\mu \in [x_i, x_{i+1}]$ (we are given that μ belongs to the dataset).
We will get the following function

$$|X-\mu|_1 = |(x_1, x_2, x_3, ..., x_m) - (\mu, \mu, \mu, ..., \mu)|_1 = \sum_{i \geq j} (\mu - x_j) + \sum_{j>i} (x_j - \mu)$$

$$\frac{d\left(\sum_{i \geq j} (\mu - x_j) + \sum_{j>i} (x_j - \mu)\right)}{d\mu} = \sum_{i \geq j} 1 - \sum_{j>i} 1$$

We can see that gradient will be equal to zero when the same number of examples are exist in both of his sides. Furthermore, the more we approach from below to the median our function will decrease, afterwards in the median it will be equal to zero and than it will increase. We got that the centroid which minimizes the term $\sum_{i=1}^{m} |x_i - \mu|$ is the median of m examples (visualization below).

When m is an **even** number, we can proof it with a more verbal explanation:

$\sum_{i=1}^{m} |x_i - \mu|$ is the sum of distances that we want to minimize and $|x_i - \mu|$ is a specific distance.

Lets say m is equal to **10** (as we said x_i is sorted in increasing order). We will imagine that x_i are points on an axis x and we will examine them from left to right. When we are looking at x_1 from the right and advancing a small step (in length of ε) to the right (towards x_1) , the sum of distances will decrease by 10ε because we are closer by ε to each one of the 10 examples. This will happen until we reach x_1. From that point our sum of distances will increase by ε (we are moving away from x_1) and decrease by 9ε (we are getting closer to the other x_i). So on and so forth with the other x_i's while we are taking ε steps to the right. This decrease will continue until we hit a tipping point at x_5. When we are between x_5 and x_6 our sum of distances decrease at each step by 5ε (for getting closer to x_{6-10}) and also increase by 5ε (for getting far away from x_{1-5}). Hence all of the points in this interval are minimizing the sum of distances (we can declare the median to be the average between x_5 and x_6). Hence this interval is represents our median- this is the "median" of the intervals and the medoid which minimizing the term is the median (while being more specific, its accepted to choose it as the average of x_5 and x_6).

For the **odd** case we can also use the same verbal proof (this is practically the same to our calculations above). For doing that we just need to replace the term "the median interval" with the "median point" in which a small ε step to the right or a small ε step to the left will always increase our sum of distances.

I will visualize my explanation above with my sanity check example (for the odd case):

$$\left|X-\mu\right|_1=\left|(x_1,\ x_2,\ x_3,\ ...,x_m)-(\mu,\mu,\mu,\ ...,\mu)\right|_1=\left|(x_1-\mu,\ x_2-\mu,\ x_3-\mu,\ ...,x_m-\mu)\right|_1$$

$$\left(s=(1,2,3,4,8),\ \mu=median=3\right)$$

$$\begin{cases}\left|x_1-\mu\right|=\left|1-\mu\right|=\left|1-3\right|=2=\mu-x_1\ \searrow\\ \left|x_2-\mu\right|=\left|2-\mu\right|=\left|2-3\right|=1=\mu-x_2\ \searrow\\ \left|x_3-\mu\right|=\left|3-\mu\right|=\left|3-3\right|=0 \quad \bullet\\ \left|x_4-\mu\right|=\left|4-\mu\right|=\left|4-3\right|=1=x_4-\mu\ \nearrow\\ \left|x_5-\mu\right|=\left|8-\mu\right|=\left|8-3\right|=5=x_5-\mu\ \nearrow\end{cases} \Rightarrow\ \searrow\bullet\nearrow \text{ min } \textit{achieved at the median}$$

Q2 SVM:

1. Linear kernel with C = 0.01- **image "D"**:

   The answer could be either image A or image D due to linear decision boundary. C controls how much do we want to penalize misclassification- what is the relative importance of the margin's size in front of the possibility to exceed the margins. A smaller C value will lead to a larger margin separation, while with a too small C value we will start to see misclassifications regardless if the data is linearly separable or not (in our case there wasn't any misclassifications). On the other hand, bigger C values will "push" the optimization problem to choose a smaller margin hyperplane (harder constraints on the optimization problem).

   Because of the explained above, and since 0.01<1 we will choose image "D" because it shows a larger margin than image "A".

   Another way to look at C is that it performs (in the soft SVM) like a weighting factor to the slack variables (ξ_i) which are offering a way to soften the constraints of the hard SVM optimization problem because in hard SVM- we are counting on that the data is linearly separable which is not always true. As bigger the ξ_i is- the "softer" the separation constraints will be. Because of the mentioned above we can also say that Slack variables (ξ_i) are related to overfitting and C is a hyper-parameter that we can use to control it.

2. Linear kernel with C = 1- **image "A"**:

   For the same explanation as before, we will choose image "A" to be the correct match because image "A" is showing a smaller margins than "D" which is achieved by bigger C value (1>0.01).

   Its critical to note that because (1) we do not now the other parameters of the problem, (2) the margins are not given to us, and (3) its hard to assess small distances in this pdf's image - its very hard to distinguish weather or not image A have data points in its margin. In that case one can say that the opposite choice between A and D is the true answer. Because of A contain points inside the margin it means that C is actually very small because its have a big margins (which supported by other vectors also).

3. 2nd order polynomial kernel- **image "C"**
   I am addressing the polynomial kernels in "F" and "C" after I matched all the other kernels to their images. So one way to address this problem is to say that with proof of contradiction we eliminated all of the other images (for each image I gave an explanation in its paragraph 1-2-5-6). Once, because both "C" and "F" are not linear and second because they do not resemble the structure of rbf kernel ,which I elaborated about in the rbf kernels paragraphs. Another way (without contradiction) to look at it is again by imagining a 3D shape (like I did with the rbf kernels). We cut this 3d shape with a plane to get our decision boundaries and this decision boundaries of images F and C does resemble polynomial behavior and not a linear or a gaussian like. From here we can see that image F contain a more complex decision

boundary than image C (F resemble a higher polynomial order than C). Therefore, I will classify image C to be the lower polynomial kernel order (2nd).

4. 10th order polynomial kernel- **image "F":**
   As explained above, from "C" and "F" I will classify image F to be the larger polynomial kernel order (10th) due to higher complexity.

5. RBF kernel with γ = 0.2- **E**

   γ is a kernel coefficient for RBF that controls how much we fit the training data. The bigger the γ value is- the more we fit the training data. Therefore, too high value of γ can cause overfitting and we can see that this is started to happen in "B" (one reason for choosing "E" here). Higher γ value will cause a higher curvature in the decision boundary than a smaller γ value. Also, we can demonstrate the decision boundary that made with the rbf kernel as a 3d gaussian that we are looking on it from above. In our case we can look at this gaussian shaped 3d template from above and cut it with a plane- the plane will intersect the paraboloid at a circle (this circle will become our boundary). Larger γ will lead to a sharper and narrower 3d "gaussian mountain". This 3d "mountain" will usually resemble a specific data point but in our case the data is linearly separable and the "mixing" of these mountains will look similar because they are in a specific region without any other data point inside of their area that comes from the other labeled group (so we do not get a valley in our images but for a much greater γ we would have get narrow gaussians that will perform circles around our samples) . Therefore, with RBF decision boundaries we are expecting to see circular forms (the intersection between the 3d "mountain" and a plane) on our data which indicates us that our RBF kernels are either image B or image E. As mentioned before, high value of γ can lead to overfitting and therefore we will classify B as a kernel with higher γ value (γ =1) than E as rbf kernel with a smaller γ value (γ =0.2). Another way to look at γ is as the parameter which indicates how strong the specific data point "pushes" the decision boundary. when low values of γ means that its "pushing hard"- the decision boundary is more far away from the data point (higher γ - closer decision boundary).This distance between the datapoints and the decision boundary is also a reason for choosing "E"

6. RBF kernel with γ = 1- **B**
   As explained above about the decision boundary structure for rbf kernels and about the effect of a higher γ, we will classify B as rbf kernel with a higher γ value than E (γ =1>0.2).

Q3 Capability of Generalization:

a. "Everything should be made as simple as possible but not simpler". The scientific term of the balance that Einstein meant to in machine learning aspect is Generalization (and the effect of overfitting). The balance his talking about is between the Goodness of fit (GOF) and the complexity of the model. The GOF describing how good the model fits to our training data, which is increasing when the complexity of the model increases (i.e. when the number of parameters is increasing). Good generalization model will have the ability to balance between them. In other words a model should not be too complex (to avoid overfitting) but still should not be too simple to avoid underfitting (another explanation about bias-variance tradeoff in this context will be given in chapter c). Hence, in this context of avoiding overfitting on the training examples (and still being aware of underfitting) the generalization ability of the model is taking place because it says how accurate is the model predictions for previously unseen data. Therefore, Einstein worn us from bad generalization performance which can be cause by overfitting (increasing our GOF too much on our training model due to high model complexity). so we should use a simple model (but not too simple because we do not want to underfit).

b. When choosing a model, we would prefer a model with the lowest AIC value (elaborated in the next sections). Lets look at the AIC equation: $AIC = 2p - 2\ln(\hat{L})$ .We can see that AIC decreases (rewarding) GOF which estimated by the likelihood function. Meaning that AIC rewards models that fits the data well when the $-2\ln(\hat{L})$ component decreasing as our model gets better in explaining the data. On the other hand, when we look at the 2p component in AIC we can see that it increases as the number of the model's parameters are increases. Meaning that AIC penalizing complex models. This penalty is made for helping us to avoid overfitting because more parameters in our model will improve the GOF.

   For conclusion: AIC design to balance between complex models and models that can explained the data very well. It penalizes **complexity** ("2p") and rewarding **GOF** ("$-2\ln(\hat{L})$").

c. The two options that are likely to happen if this balance was violated is overfitting and underfitting. As I started to explain in chapter a, this balance could be violated if we are increasing too much our ability to explain the data by adding more and more parameters that could "backfire" us in an increasing model complexity which will lead to overfitting. The same for the other side- this balance will be also violated if we will remove too many parameters to get simpler model, because again there will be some point in which the removal of a parameter will "backfire" us and we will decrease too much in our ability of explaining the data which can lead to underfitting.

   We can also look at the bias-variance tradeoff in this context:

   When we **increase the complexity** of a model, we are increasing its variance and reducing its bias- This can lead to **overfitting** because high variance can be seen as the model is learning the noise in the training set.
   Also, when we **reduce the complexity** of a model, we are increasing its bias and reducing its variance- This can lead to **underfitting** because high bias can cause our algorithm to miss the relationship between the target output and the features.

d. AIC is an information criterion that can tell us about the balance between GOF and complexity. As I said before, with AIC we are aiming to balance between complex models and models that can explain the data very well. It penalizes complexity (increasing for higher values of "2p") and rewarding GOF (decreasing for lower values of"$-2\ln(\hat{L})$"). Hence, we would prefer a model with the **lowest** AIC (balance). furthermore, we saw that the violation of the balance between these two components can cause overfitting or underfitting and therefore the value of this information criterion is important.