Machine learning in healthcare

HW3- Theoretical Questions

1. Clustering

a. The K-medoid algorithm is trying to minimize the sum of pairwise distances between the data points, as opposed to the sum of square distances in K-means. This means that the K-medoid will find the point that is the closest to other data points with the same label and will not be much affected by extreme outliers. In comparison, the K-means method takes the mean of all data points with the same label, which makes it more susceptible to outlier bias.

b. The term we want to minimize is a function of μ-

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

In order to find the minimum of the function, we calculate its derivative and compare it to 0:

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

$$0 = 2\sum_{i=1}^{m} y_i - 2\sum_{i=1}^{m} \mu \xrightarrow{yields} 2\sum_{i=1}^{m} y_i = 2\mu m$$

$$\xrightarrow{yields} \boldsymbol{\mu = \frac{\sum_{i=1}^{m} y_i}{m}}$$

And this is the definition of the mean. QED.

c. Bonus-

We now look to minimize a different loss function:

$$f(\mu) = \sum_{i=1}^{m} |y_i - \mu|$$

As we did earlier, we can compute the derivative:

$$f'(\mu) = \sum_{i=1}^{m} sign(y_i - \mu)$$

This term can only be equal to zero if and only if the number of data points larger than μ is equal to the number of points smaller than μ, which is the definition of the median.

2. SVM

a. ***Linear kernel with C=1.*** From the linear decision boundary we understand that this is a linear kernel. The larger C is, the smaller the margin gets and therefore the closer the decision boundary will be to the closest samples. In this graph we can see samples very close to the boundary, which indicate a large value of C.

b. ***RBF with $\boldsymbol{\gamma = 1}$***. RBF kernel compares each data point to the closest neighbors. γ determines the area of influence- the bigger it is the smaller the area gets and the more complex the model is. A high value of γ will create a more fitted model, and a small value will create a simpler, less fitted model. This graph shows a relatively fitted model, indicating a large value of γ.

c. ***2nd order polynomial kernel***. From the non-linear boundary condition we can see that this is a polynomial kernel- a polynomial kernel uses both the data samples and their combinations, leading to a non-linear decision boundary. The larger the order is, the more complex the model gets, with a higher risk to overfitting. A small order will be less accurate and could be somewhat similar to a linear kernel. This leads us to the conclusion that the graph shows a low order polynomial kernel.

d. ***Linear kernel with C=0.01***. From the linear decision boundary we understand that this is a linear kernel. The larger C is, the smaller the margin gets and therefore the closer the decision boundary will be to the closest samples. In this graph we can see a relatively large margin, indicating a smaller value of C.

e. ***10th order polynomial kernel***. Similarly to b, the non-linear boundaries suggest a polynomial kernel. As this graph shows significant overfitting, we understand that it is a high order polynomial kernel.

f. ***RBF with $\boldsymbol{\gamma = 0.2}$***. RBF kernel compares each data point to the closest neighbors. γ determines the area of influence- the bigger it is the smaller the area gets and the more complex the model is. A high value of γ will create a more fitted model, and a small value will create a simpler, less fitted model. This graph shows a relatively underfitted model, indicating a small value of γ.

3. Capability of generalization

a. Relevance-redundancy tradeoff. The solution is usually feature selection or feature engineering.

b. 2p gets larger as the number of features increases- this is the penalty for using too many parameters. ln(L) gets larger when the likelihood L increases, which is rewarding a better fit of the model to the data.

c. If the balance is violated and p is very large, we will get overfitting- the model is using too many parameters and therefor is creating a bias towards the training data, making it perform worse when generalized.
The other option, when we have ln(L) too small we get underfitting. This means that our model may be simplified, but its performance is poor and it makes many mistakes.

d. We should aim to get the lowest AIC value possible. Mathematically, this could be achieved only when the number of parameters is small- meaning our model is simple- and the likelihood is large- meaning the model fits the data well. The smaller a model's AIC value is, the more simple **and** efficient it is, and therefor models with a low AIC value are better at balancing simplicity and accuracy.