Machine Learning in Healthcare

336546

HW3

Dekel Brav

302362389

Q1.

a.

K-medoid is more robust to noise and outliers than K-means. This is because for an outlier (say a very large number compared to the rest of the data), the squared Euclidian distance will take into consideration the very large number, but the median will not. For example, if our data is valued between 0 and 10, and we have an outlier with a value of 1000, the K-means will use that distance in its algorithm, skewing the data immensely. The K-medoid on the other hand, doesn't care if the value is 1000, 1000000, or 11, as the output will be the same in any of the cases.

b.

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

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

$$\Rightarrow \sum_{i=1}^{m}x_i=\sum_{i=1}^{m}\mu_{\min}=m\cdot\mu_{\min}$$

$$\Rightarrow \mu_{\min}=\frac{1}{m}\sum_{i=1}^{m}x_i$$

$$\Rightarrow \mu_{\min}=\overline{x}=mean\left(x_i\right)$$

Bonus.

$$\frac{d}{d\mu}\left(\sum_{i=1}^{m}\left|x_i-\mu\right|\right)=-\frac{1}{m}\sum_{i=1}^{m}\operatorname{sgn}\left(x_i-\mu\right)\underbrace{=}_{want}0$$

$$\Rightarrow \sum_{i=1}^{m}\operatorname{sgn}\left(x_i-\mu\right)=0$$

We see that the left side of the equation is a sum of 1's and -1's, depending on the value of μ. Therefore, the μ that makes the left side of the equation equal to 0 is one that is larger than half of the values of x, and smaller than the other half. This way we get an equal number of 1's and -1's, which will lead to an overall sum of 0. This is the definition of a median (smaller than half and larger than the other half). Therefore, we arrive at the conclusion that:

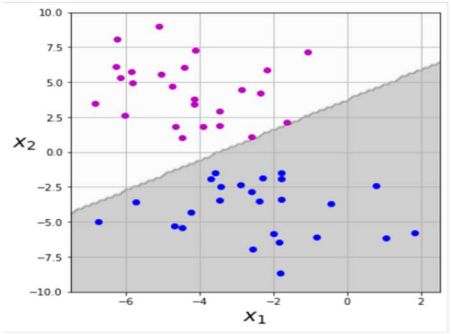
$$\mu_{\min}=median\left(x_i\right)$$

2.

- Explanations are after the images.

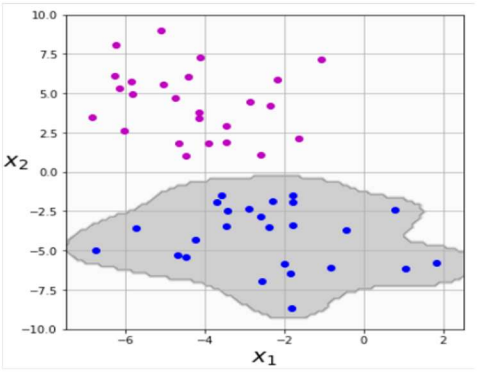
A. (2)

Linear kernel with C=1.



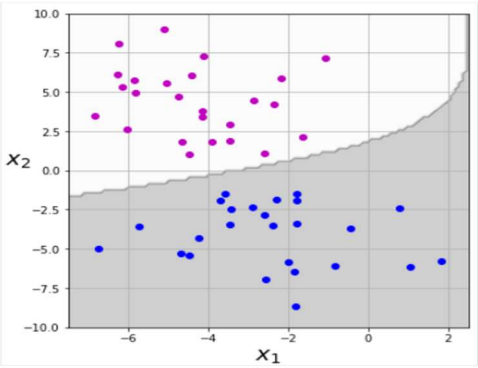
B. (6)

RBF kernel with $\gamma = 1$.



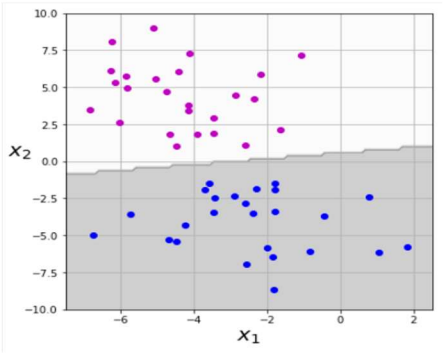
C. (3)

2nd order polynomial kernel.



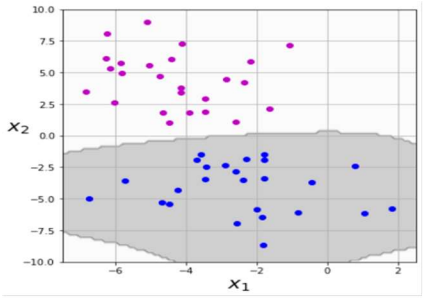
D. (1)

Linear kernel with C=0.01.



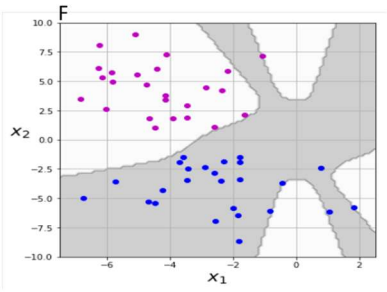
E. (5)

RBF kernel with $\gamma = 0.2$.



F. (4)

10th order polynomial kernel.



- **Explanations:**

Linear Kernels:

As we can clearly see, the two classification images that used a linear kernel are A and D. Now, the question is which one has C=1 and which one has C=0.01. As we know, the C hyperparameter tells the optimization how much it needs to avoid misclassifying an example. For larger values of C, the optimization wants to avoid misclassification as much as possible, so the margins are smaller. For smaller values of C, the optimization cares less about misclassification, and therefore the margins are larger. From here, we can see that the margins in A are very small (some data points are pretty much on the margins border), and in D they are larger. Therefore, we can come to the conclusion that A→2 and D→1.

RBF Kernels:

As we can see, the 2 classification images that use RBF kernels are B and E. The hyperparameter γ is defined as the inverse of the radius of influence of the samples that are selected by the model as support vectors. This means that for larger values of γ the hyperplane will be very fit to the data, and for smaller values of γ, the hyperplane will be less fit to the data (and for very small values the hyperplane will resemble one of a linear kernel). We can see from the images that the hyperplane in B is much more fit to the data than the hyperplane in E. Therefore, we can come to the conclusion that B→6, E→5.

Polynomial Kernels:

Now we are left with the two classification images C and F, which match with a polynomial kernel. We can clearly see that F is overfit. Of the two options (2^{nd} order and 10^{th} order), we can safely assume that if one of them were to be overfit, it would be the 10^{th} order polynomial. Therefore, we can come to the conclusion that C→3, F→4.

Q3.

a.

In a machine learning aspect, Einstein's interpretation of Occam's Razor can be attributed to the concept of overfitting/underfitting (in other words, the variance-bias tradeoff). For example, we can make a classifier very complex so that it fits the training data very well, but overfits it in the process. In essence, the way to "simplify" is to add generalization/regularization terms in order to prevent the overfitting. Even so, making it simpler than it should would cause underfitting. In conclusion, we should make our model the simplest possible one that suits our needs (to avoid overfitting), but not simpler than that (to avoid underfitting).

b.

Generally speaking, when applied as a scoring metric for the "goodness" of a model, the lower the AIC the better.

$2p$:

P is the number of parameters used in the model. Therefore, the AIC discourages large numbers of parameters, which makes sense because large numbers of parameters would fit the data more (increasing chances of overfitting).

$2\ln\left(\hat{L}\right)$:

The parameter $\hat{L}$ is the estimated maximum likelihood given P model parameters. This term exists to reward the goodness of fit.

Therefore, we have 2 terms, one that penalizes overfitting, and one that penalizes underfitting. This is basically an application of Einstein's interpretation of Occam's Razor, where we don't want to overfit the data, but also don't want to underfit it.

c.

If this balance was to be disturbed, we would either:

Overfit the data by having too many model parameters P

Or:

Underfit the data by having a low MLE value (this is done by not having enough parameters to explain the variance).

d.

As we said in 3.b, the aim is to find the minimal AIC. Basically, we want to find the balance between the using the smallest number of parameters and achieve the highest log likelihood. The AIC allows us to compare the two and find the optimal balance between them (the balance that would result in the smallest AIC).