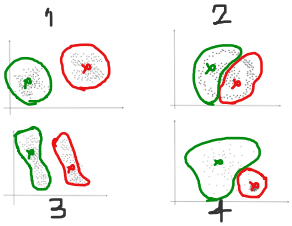
Mohammad Parham Bateni
Student number: 99105294

1 Clustering and K-means (35 points)

1.1 Intuitive Understanding

- In each sample below, draw the boundry that K-means finds for $K = 2$. Do you think the clusters separated by borders found by K means is meaningfull in each case? If not, what property of data causes this?

boundries are shown by red and green color and each cluster centroid is shown by x

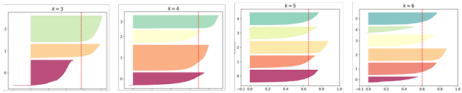


- In which case above, feature scaling can solve the problem? Why?

Except the first clustering, other clusters seperated by borders found by K means are not that meaningfull. Many properties related to data could cause this problem. For example, in the second clustering the data distribution is not Gausian. In the third clustering, data is anisotropic meaning that data is alongated along a specific axis. For the clustering number 4 although the data distribution is Gausian but there is a huge difference in standard deviation of data between two clusters.

Feature scaling makes sure that the features of the data-set are measured on the same scale and consists of two techniques called Normalization and Standardization. In Standardization the data is scaled to have mean of 0 and standard deviation of 1. This could be helpful when features in clustering show quite different variances like in clustering number 4. Moreover, features could be of incomparable units like clustering number 3 and hence Min-Max normalization could be beneficial in this case.

1.2 Finding proper value of K



- For finding a proper value for K, what is the advantage of using silhouette score in comparison to inertia?

- In the figure above, you can see silhouette diagram for different values of k. which value for K is better? Why?

- Silhoute score is a score for evaluating the clustering result by computing cohesion and seperation score for each point. This score lies between -1 and 1, the higher the score, the more suitable is k choice for clustering. However with inertia score we would compute the sum of distances between each point and its nearest cluster center and the result would be a number just telling us the sum of all distances. By taking the average of all silhouette scores of each point we can conclude immediately if the clustering is done good enough or not. On the other hand, inertia score is just a number that doesn't say that much except the sum of differences and in order to be able to evaluate the goodness of the clustering we need to compute this score for different scores and compare these scores with eachother, where in silhouetter score case we can make sure if a clustering is good or not by trying to find a clustering with a bigger silhouette score near to 1.
- K=4 and K=5 seem to be the best options as the silhouette plot shows the highest mean of sihouette score for this case (around %65) and there is no clusters having all silhouette scores bellow average (like K=3 and K=6). But if we want to choose a K as the best K we would go with K=5 because the clusters are more equally distributed. (width of the clusters in the figure K=5 are almost the same)

1.3 Applications of Clustering

Two applications of clustering are Active learning and semi-supervised learning. Explain usage of clusting in each of these approaches and the diffrences between them briefly.

Semi-supervised learning is an approach to machine learning in which a small amount of labeled data are combined with a huge amount of unlabeled data in the training process. In this approach we have a few labeled data and a lot unlabeled data. In this approach different algorithms like 'cluster assumption' which use clustering are used to give the unlabeled data labels. So to sum it up, in this approach of learning we need to label some of the data based on the known labeled data and this is could be mainly done by clustering for example we cluster the unlabeled data and then assign each cluster to its nearest labeled data point.

In semi-supervised machine learning, active learning allows the algorithm to pick the data it wants to learn from. This method allows the program to actively query an authority source, such as the programmer or a labeled dataset, to discover the accurate prediction for a particular problem. The process of assigning the data points labels is also done by clustering.

The difference between active learning and semi-supervised learning is that in semi-supervised learning we choose a small sample of data to be labeled and we train the model based on these data, however in active learning data is incrementally and dynamically labeld during training procedure. Somehow active learning dynamically chooses the label which would be most helpful for it to learn from.

2 Whitening Using PCA (25 points)

Whitening is one of the pre-processing techniques which is used in practical ML. By whitening, we mean that for a given feature matrix $\mathbf{X}$, this feature matrix should have zero mean vector and identity matrix as its covariance matrix. Explain that how we could transform $\mathbf{X}$ by using its covariance matrix principal components in order to have a whitened data set? After that, prove this new feature matrix has the desired properties, namely, zero mean vector and identity matrix as its covariance matrix.

By definition we know that the covariance matrix of a matrix like A is equal to E[(A-E[A])(A-E[A])^T) where E[A] is the mean vector of matrix A. In order to whiten the data we first have to transform X's covariance matrix into a diagnol matrix so the features would be uncorrelated.

$Cov(X) = E((X - E[X])(X - E[X])^T)$

$Y = X - E[X] \Rightarrow E[Y] = 0$, $Cov(Y) = E(YY^T)$

now we use diagonalization on $Cov(Y)$ (A)

$AV_i = \lambda_i V_i \Rightarrow AV = V\Lambda$ $\quad \Lambda = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & \lambda_n \end{bmatrix}$, $V = [V_1, V_2 \cdots V_n]$ V_i's are eigenvectors of A

$\Rightarrow A = A^T \Rightarrow V^T V = I \Rightarrow V^T A V = \Lambda$

now we consider $X' = V^T Y = V^T (X - E(X)) \Rightarrow Cov(X') = E(X'X'^T)$

$\Rightarrow Cov(X') = E(V^T (X - E(X))(X - E(X))^T V) = E(V^T A V) = \Lambda \Rightarrow$ diagonal

now if we consider $X'' = \Lambda^{-\frac{1}{2}} V^T (X - E(X))$

$\Rightarrow Cov(X'') = E(\Lambda^{-\frac{1}{2}} V^T \underbrace{(X - E(X))(X - E(X))^T}_{A} V \Lambda^{-\frac{1}{2}}) = I$

Λ

$\Rightarrow X'' = \Lambda^{-\frac{1}{2}} V^T (X - E(X))$ where Λ is a diagonal matrix with eigenvalues of $cov(X - E(X))$ and V is a matrix with eigenvectors of $cov(X - E(X))$ as its columns

We know that $E[AX + B] = AE[X] + B$

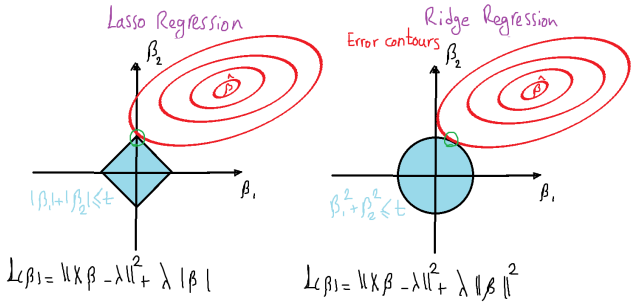
$\Rightarrow E[X''] = \Lambda^{-\frac{1}{2}} V^T \underbrace{E[X - E[X]]}_{0} \Rightarrow E[X''] = 0$

3 Linear Regression

3.1 Lagrange Multipliers (25 points)

Using Lagrange Multipliers give a geometric interpretation of why Lasso Regression results in more sparse solutions. Consider using $L_p$where $p < 1$ instead of $p = 1$. Show that we will get even more sparse solutions when we decrease p. Additionally explain why L_1 norm is used instead of smaller values for getting sparse solutions. (Hint: Analysis the convexity of unit ball with L_p norm.)

Lasso regression uses the L1 norm of the coefficients(Betha) as the additional constraint on the cost function where Ridge regression uses the L2 norm of the coefficients. The reason why Lasso regression could lead to more sparse results than Ridge regression can be deduced by looking at the provided figure.



In the illustrated example we have two features (betha1 and betha2) and considering the blue constraints shown we would like to minimize the error. The error contours are drawn in a red color and since Lasso regression has more sharp corners it is more likely that the contour hits the blue area constraint on the corners where one of the coefficients is zero. However, in Ridge regression the shape of the constraint is a circle and error contours would probably not hit where one coefficient is zero. The point where error contour and the blue constraint area meet is the point that has the least error considering the constraint conditions.

Considering Lp where p<1 as the constraint on coefficients. the shape of the constraint area would turn into a multidimensional diamond with many corners and the probablitiy that the solution point is on one of the corners would increase. So with p<1 we expect to get even more sparse solutions.

The reason why L1 norm is used instead of smaller values is because by using the L1 norm the cost function would be convex and we don't have to worry about local minimum answers as there is only one global minimum. But with Lp norm where p<1 the cost function would turn into a concave function and finding the true solution could get hard.
To prove that Lasso regression's cost function is convex pay attention that:
1- We can prove that L(B)=||XB-lambda||^2 is convex.
2- L(B)=lambda|B| is also convex.(This could be easily understood by looking at its shape)
3- And as we know that the sum of two convex functions are also convex (which is easy to prove), by adding the result of the two previous parts we conclude that Lasso regression is also convex.

However as the shape of L(B)=lambda|B|_p for p<1 turns into a diamond it is easy to notice that this function would no longer be convex and so for p<1 finding the actual best solution would be hard.

3.2 Ridge Regression (optional) (10 points)

Prove that the optimal value for the Ridge regression is $\mathbf{w}^* = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{y}$ and can be obtained by minimizing the Error function below:

$$L(\mathbf{w}) = \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 + \lambda\|\mathbf{w}\|_2^2 \tag{1}$$

$L(w) = \|Xw-y\|^2 + \lambda\|w\|^2 = (w^TX^T-y^T)(Xw-y) + \lambda w^Tw = w^TX^TXw - 2y^TXw + y^Ty + \lambda w^Tw$

$\frac{\partial L(w)}{\partial w} = (X^TX + X^TX)w - 2X^Ty + 2\lambda w = 0 \Rightarrow (X^TX + \lambda I)w = X^Ty$

$\Rightarrow w^* = (X^TX + \lambda I)^{-1}X^Ty$

4 Generalization Error (15 points)

- What is the effect of dimentionality reduction on overfitting? Why?
- We know that one solution for overfitting is using more data for training. However finding more data is not always possible, in this case we can alter the data that we already have so that by changing them slightly, we generate new data. This is called data augmentation. Search about data augmentation methods and give three examples of them.

- Dimentionality reduction helps to reduce overfitting. That's because the more feautres we have there is more chance that some of them would be correlated and hence there would be redundant factors in training set, leading to overfitting. Moreover when the number of features increases, the model will become more complex, thereby increasing the likelihood of overfitting.
- Depending on the data we are dealing with there are a bunch of different methods of data augmentation. For example for Image data we can add noise, crop, flip, zoom, scale, change the brightness, and we can use many other methods. With Audio data we can add noise, change the speed, and pitch the data. In case of text data we can replace the words or delete random words. When dealing with tabular data, it is easy to generate new data by randomly changing the variables or adding noise to them.

5 Kernels

5.1 Feature Space (15 points)

Consider $x, y \in \mathbb{R}^2$, the kernel function defined as

$$k(x, y) = (1 + cos(\angle x, y))^2, \qquad x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} \; y = \begin{bmatrix} y_0 \\ y_1 \end{bmatrix}$$

where $cos(\angle x, y)$ means the cosine of the angle between x and y vectors. Find the feature space $\phi(x)$ which corresponds to this kernel.

$\|x\| = \alpha \quad \|y\| = \beta$

$k(x,y) = \phi(x)\phi(y) = (1+\cos(\angle x,y))^2 = 1 + 2\frac{x^Ty}{\|x\|\|y\|} + \frac{(x^Ty)^2}{\|x\|^2\|y\|^2} = \frac{1}{\alpha^2\beta^2}(\alpha^2\beta^2 + 2\alpha\beta x^Ty + (x^Ty)^2)$

$= \frac{1}{\alpha^2\beta^2}(\alpha^2\beta^2 + 2\alpha\beta(x_0y_0 + x_1y_1) + x_0^2y_0^2 + x_1^2y_1^2 + 2x_0x_1y_0y_1)$

$= \frac{1}{\alpha^2} \begin{bmatrix} \alpha^2 \\ \sqrt{2}\alpha x_0 \\ \sqrt{2}\alpha x_1 \\ x_0^2 \\ x_1^2 \\ \sqrt{2}x_0x_1 \end{bmatrix}^T \frac{1}{\beta^2} \begin{bmatrix} \beta^2 \\ \sqrt{2}y_0 \\ \sqrt{2}\alpha y_1 \\ y_0^2 \\ y_1^2 \\ \sqrt{2}y_0y_1 \end{bmatrix} \Rightarrow \phi(x) = \frac{1}{x_0^2 + x_1^2} \begin{bmatrix} x_0^2 + x_1^2 \\ \sqrt{2}(x_0^2 + x_1^2)\, x_0 \\ \sqrt{2}(x_0^2 + x_1^2)\, x_1 \\ x_0^2 \\ x_1^2 \\ \sqrt{2}x_0x_1 \end{bmatrix}$

5.2 Kernel Matrix (20 points)

5.2.1

Suppose for now that k is indeed a valid kernel corresponding to some feature mapping ϕ. Now, consider some infinite set of m points, $\{x^{(1)}, ..., x^{(m)}\}$, and let a square m-by-m matrix K be defined so that its (i, j)-entry is given by $K_{i,j} = k(x^{(i)}; x^{(j)})$. This matrix is called the Kernel matrix.
Prove that K is a positive semi-definite matrix.

$x^T K x \geq 0 \quad \forall x \in \mathbb{R}^m \iff K$ is positive semi definite

$$x^T K x = \begin{bmatrix} x_1 & \dots & x_m \end{bmatrix} \begin{bmatrix} k_{11} & k_{12} & \dots & k_{1m} \\ & & \vdots & \\ k_{m1} & & \dots & k_{mm} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}$$

$$x^T K x = \begin{bmatrix} k_{11}x_1 + k_{21}x_2 + \dots + k_{m1}x_m & \dots & k_{1m}x_1 + \dots + k_{mm}x_m \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}$$

$$= k_{11}x_1^2 + 2k_{21}x_2x_1 + \dots k_{mm}x_m^2$$

$k_{ij} = \phi(x_i)^T \phi(x_j) \Rightarrow x^T K x = \|\phi(x_1)\|^2 x_1^2 + 2\phi(x_1)^T\phi(x_2) x_1 x_2 + \dots + \|\phi(x_k)\|^2 x_m^2$

$$\Rightarrow x^T K x = (\phi(x_1)x_1 + \dots + \phi(x_m)x_m)^T (\phi(x_1)x_1 + \dots + \phi(x_m)x_m)$$

$$= \|\phi(x_1)x_1 + \dots + \phi(x_m)x_m\|^2 \geq 0 \Rightarrow K \text{ is positive semi definite}$$

5.2.2

The positive semi-definitivity of kernel matrix turns out to be not only a necessary, but also a sufficient, condition for k to be a valid kernel, also called a Mercer kernel.
Prove that if K(kernel matrix) is a positive semi-definite matrix, then k is a dot product: $\exists \phi$ such that $k(x, y) = \phi(x).\phi(y)$
(Hint: Use diagonalization for real symmetric matrices.) First we prove that we can write a symmetric matrix as A=PDP^T

$$\left.\begin{array}{l} Av_1 = \lambda_1 v_1 \\ \vdots \\ Av_n = \lambda_n v_n \end{array}\right\} \Rightarrow A \underbrace{\begin{bmatrix} v_1 & \dots & v_n \end{bmatrix}}_{V} = \underbrace{\begin{bmatrix} v_1 & \dots & v_n \end{bmatrix}}_{V} \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \dots & 0 & \lambda_n \end{bmatrix} \Rightarrow AV = V\Lambda$$

$$\left.\begin{array}{l} Av_i = \lambda_i v_i \\ Av_j = \lambda_j v_j \end{array}\right\} \overset{T}{\Rightarrow} v_i^T A^T = \lambda_i v_i^T \overset{A=A^T}{\Rightarrow} v_i^T A v_j = \lambda_i v_i^T v_j \Rightarrow \lambda_j v_i^T v_j = \lambda_i v_i^T v_j$$

$$\lambda_i \neq \lambda_j \Rightarrow v_i^T v_j = 0 \Rightarrow \langle v_i, v_j \rangle = 0 \quad (i \neq j)$$

we can take λ_is such that $v_i^T v_i = 1$ so we would have:

$V^T V = I \Rightarrow A = V \Lambda' V^T$ where Λ' is a diagnol matrix where $\begin{cases} \lambda_i & i=j \\ 0 & i \neq j \end{cases}$

now we assume that k(x,y)=phi(x).phi(y) so k should be symmetric due to this assumption. K(x,y)=k(y,x)

So we can write k as PDP^T. $P = [P_1 \dots P_m]$, $D = \begin{bmatrix} D_1 & 0 & \cdots & 0 \\ \vdots & D_2 & & \vdots \\ 0 & \cdots & \cdots & D_m \end{bmatrix}$

$$\Rightarrow K = \begin{bmatrix} P_1 D_1 & P_2 D_2 & \cdots & P_m D_m \end{bmatrix} \begin{bmatrix} P_1^T \\ \vdots \\ P_m^T \end{bmatrix} = D_1 P_1 P_1^T + D_2 P_2^T P_2 + \cdots + D_m P_m^T P_m$$

$\Rightarrow$ P_{ij} : ith column jth row

$$k(x,y) = D_1 P_{1x} P_{1y} + D_2 P_{2x} P_{2y} + \cdots + D_m P_{mx} P_{my}$$

$$= \begin{bmatrix} \sqrt{D_1} P_{1x} \\ \vdots \\ \sqrt{D_m} P_{mx} \end{bmatrix}^T \begin{bmatrix} \sqrt{D_1} P_{1y} \\ \vdots \\ \sqrt{D_m} P_{my} \end{bmatrix} = \phi(x)^T \phi(y)$$

$$\Rightarrow \phi(x) = \begin{bmatrix} \sqrt{D_1} P_{x1} \\ \sqrt{D_2} P_{x2} \\ \vdots \\ \sqrt{D_m} P_{xm} \end{bmatrix}$$

where D_i is λ_i and P_{xi} is $v_{i(x)}$ (xth element of eigenvector v_i); k eigenvalue

6 Feature Expansion (20 points)

6.1

For each dataset below in figure 1, find a 1-dimensional transform $\phi : R^d \to R$ such that in the new feature space, 2 classes are linearly separable.

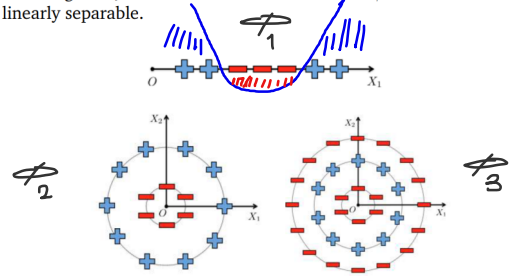


Figure 1: Datasets in original feature space

$$\phi_1(x) = (x-a)^2 - b$$

negative samples : N

$$a = \frac{\max(N) - \min(N)}{2}$$

$$b = \left(\frac{\max(N) + \min(N)}{2}\right)^2$$

$$\phi_2(x) = x_1^2 + x_2^2$$

$$\phi_3(x) = (x_1^2 + x_2^2 - a)^2 - b$$

positive samples : P, $x_1, x_2 \in P$

$\max(x_1^2 + x_2^2) = \text{I}$

$\min(x_1^2 + x_2^2) = \text{II}$

$a = \frac{\text{I} - \text{II}}{2}$, $b = \left(\frac{\text{I} + \text{II}}{2}\right)^2$

6.2

As you know, the equation of a circle in 2-d plane is defined as $(x_1 - a)^2 + (x_2 - b)^2 - r^2 = 0$. By expanding the former equation, show that any circular region is linearly separable in (x_1, x_2, x_1^2, x_2^2) feature space.

$x_1^2 + x_2^2 - 2ax_1 - 2bx_2 + a^2 + b^2 - r^2 = 0$

$= \begin{bmatrix} -2a \\ -2b \\ 1 \\ 1 \end{bmatrix}^T \begin{bmatrix} x_1 \\ x_2 \\ x_1^2 \\ x_2^2 \end{bmatrix} + a^2 + b^2 - r^2$

$A^T \quad \phi(x) \quad \varepsilon$

$\Rightarrow A^T \phi(x) + \varepsilon = 0$

$\Rightarrow$ circular region is linearly seperable in $\phi(x)$ feature space

7 SVM Decision Boundaries (30 points)

In Figure 2 we have plotted the decision boundaries and margins for SVM learned on the same data set using the following parameters (not in the same order as the figures):

(i) Linear kernel, C = 0.1

(ii) Linear kernel, C = 1

(iii) Linear kernel, C = 10

(iv) RBF kernel, $\gamma = 0.1$, C = 15

(v) RBF kernel, $\gamma = 1$, C = 3

(vi) RBF kernel, $\gamma = 10$, C = 1

Linear kernel: $L(\beta) = \frac{\|\beta\|^2}{2} + C \sum_i \max(0, 1 - y_i(\beta^T x + \beta_0))$

when C increases the margin shrinks

$\Rightarrow$ (i): c (ii): b (iii): f

RBF kernel: $K(x, y) = e^{-\gamma \|x - y\|^2}$

when $\gamma \uparrow$ variance $\uparrow$ bias $\downarrow$

$\gamma \downarrow$ variance $\downarrow$ bias $\uparrow$

$\Rightarrow$ (iv): d (v): e (vi): a

Match each one of the figures with one of these parameter settings. Explain your matchings in few sentences.

Linear kernels (b), (c), (f)

RBF kernels (a), (d), (e)

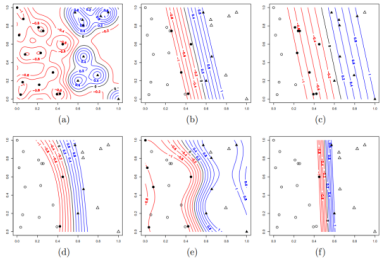


Figure 2: Circles and triangles denote Class 1 and 2 respectively, solid points are support vectors.

8 Monte Carlo Cross Validation (15 points)

In Monte Carlo Cross Validation(MCCV), also known as repeated random subsampling CV, you randomly select (without replacement) some fraction of your data to form the training set, and then assign the rest of the points to the validation set. This process is then repeated multiple times, generating (at random) new training and validation partitions each time. In other words, it simply splits the N data points into the two subsets n_t and n_v by sampling, without replacement. There exists $\binom{N}{n_t}$ unique training sets, but MCCV avoids the need to run this many iterations.

8.1

How does the choice of n_t affects the bias-variance trade-off?

By increasing nt the number of training samples would increase possibly leading to overfitting when training the model. Vise versa when decreasing nt we could possibly face underfitting. So overall by increasing nt the variance of the fitted model increases and the bias decreases, whereas when decreasing nt the variance of the fitted model would decrease and the bias increases.

8.2

Compare MCVV with k-Fold Cross Validation and state its pros and cons.

In MCVV as it was described we generate training partitions with size of nt multiple times, however in k-Fold Cross validation we devide the samples into k folds and each time we take one fold as validation samples and other k-1 as training samples. So in MCVV each data could be validated many times where in k-fold cross validation each sample is validated just once. In MCVV different combination of samples could be trained with eachother, but in k-fold CV each sample is always trained with the samples in its fold for sure. Selecting nt random samples as training samples at each iteration could be time consuming so MCVV would take much more time than k-fold CV to split the samples. MCVV is more repeatable so it could give more confidence and the variance is low, but the bias is high. On the other hand, in k-fold CV the variance is high and the bias is low. MCVV could lead to underfitting and k-fold CV could lead to overfitting. The reason why the variance in MCVV is low is because the same data could be chosen multiple times and hence it could lead to a biased model and underfitting.