Sharif University of Technology
Electrical Engineering Department

Machine Learning
HW 4

Amir Hossein Yari
99102507

June 26, 2023

Contents

1. Representer Theorem

1.1

- Hilbert Space ⇒ Is a complete inner product space. It is a vector space equipped with an inner product.
- Reproducing Kernel Hilbert Space ⇒ Is a positive definite function that characterizes the inner product structure of an RKHS. It captures the pairwise relationships between points in the input space. In the context of machine learning, the input space is often the space of feature vectors.
- Reproducing Kernel ⇒ Given a reproducing kernel, the RKHS associated with it is the Hilbert space of functions that can be represented as inner products with the kernel function. In other words, an RKHS is a space of functions where evaluation of a function at any point can be computed as an inner product with the kernel function.
- Mercer's Theorem ⇒ Assume Gram matrix $\mathbf{K}$ to be positive definite. Then from eigendecomposition we have $\mathbf{K} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$ where:

$$\mathbf{\Lambda} = diah(\lambda_1, \ldots, \lambda_N) \quad , \quad \lambda_i > 0 \quad \text{for } i = 1, \ldots, N$$

$$U = \begin{bmatrix} u1 & \ldots & u_N \end{bmatrix}$$

We can rewrite $\mathbf{K} = (\mathbf{\Lambda}^{\frac{1}{2}}\mathbf{U}^T)^T(\mathbf{\Lambda}^{\frac{1}{2}}\mathbf{U}^T) = \hat{\mathbf{U}}\,\hat{\mathbf{U}}^T$ where:

$$\hat{\mathbf{U}} = \begin{bmatrix} \lambda_1^{\frac{1}{2}}\, u_1 & \ldots & \lambda_N^{\frac{1}{2}}\, u_N \end{bmatrix}$$

Thus we can write $\mathbf{K}_{ij} = \hat{u}_i^T\, \hat{u}_j = \langle \hat{u}_i, \hat{u}_j \rangle$ and thus $\phi(x_i) = \hat{u}_i = \lambda_i^{\frac{1}{2}}\, u_i$. So we write the entries in form of inner product.

1.2

1.2.1

Mercer's theorem states that any positive definite kernel function can be expressed as an inner product in a reproducing kernel Hilbert space (RKHS). Based on the provided information, we can rewrite the function f(xj) using Mercer's theorem as follows:

$f(x_j) = \sum_{k=1}^{N} {}_k(x_k) + v(x_j)$
Here, $\Phi(.)$ represents a set of orthogonal functions, and v(.) represents an orthogonal function to $\Phi(.)$. The inner products between $\Phi(.)$ and v(.) with any function in the RKHS are zero.
Therefore, $v(.), (.) = 0$, and $v(.), v(.) = 0$
This form allows $f(x_j)$ to be expressed as a linear combination of the orthogonal functions (x_k) with corresponding coefficients ${}_k$, plus the orthogonal function $v(x_j)$.

1.2.2

The reproducing kernel property states that for any function f in the reproducing kernel Hilbert space (RKHS) with kernel K, we have:
$f(x) = f, K(x, .)$
Using this property, we can express the function $f(x_j)$ as a dot product of ${}'_k s$:
$f(x_j) = f, K(x_j, .)$
Now, let's substitute K(xj, .) with its expansion using the orthogonal functions ${}_k$:
$f(x_j) = f, \sum_{k=1}^{N} {}_k(x_k) + v(x_j), .$
Using linearity of the inner product, we can distribute it over the summation:
$f(x_j) = \sum_{k=1}^{N} kf, (x_k), . + f, v(x_j), .$
Since Φ(xk) are orthogonal, we can express the inner product as:
$f, (x_k), . = k$
Thus, we obtain:
$f(x_j) = \sum_{k=1}^{N} k(x_k) + f, v(x_j), .$
In this form, $f(x_j)$ is written as a dot product of the orthogonal functions Φk's, with coefficients k, and the term $f, v(x_j), .$, which accounts for the orthogonal function $v(x_j)$ in the RKHS.

1.3

To find a lower bound on the regularization term $R(||f||)$, we can utilize the orthogonality of ${}_k$ and v, as well as the monotonicity of R.
First, let's consider the regularization term $R(||f||)$. By the properties of the RKHS, we can express the norm of f as:
||f|| = √(f, f)
Using the expression of f(xj) as a linear combination of Φk's and v(xj), we have:

$$f, f = \sum_{k=1}^{N} k(x_k) + v(x_j), \sum_{l=1}^{N} {}_l(x_l) + v(x_j)$$

Expanding this inner product, we get:

$$f, f = \sum_{k=1}^{N} \sum_{l=1}^{N} {}_k l(x_k), (x_l) + \sum_{k=1}^{N} {}_k(x_k), v(x_j) + \sum_{l=1}^{N} {}_l v(x_j), (x_l) + v(x_j), v(x_j)$$

Now, since Φ(xk) and v(xj) are orthogonal, their inner products are zero:

$$(x_k), v(x_j) = v(x_j), (x_k) = 0$$

Thus, the above expression simplifies to:

$$f, f = \sum_{k=1}^{N} \sum_{l=1}^{N} {}_{kl}(x_k), (x_l) + v(x_j), v(x_j)$$

Now, by the monotonicity of the regularization term R, we have:

$$R(||f||)R(\sum_{k=1}^{N} {}_{kl}(x_k), (x_l) + v(x_j), v(x_j))$$

Since R is monotonically increasing, we can obtain a lower bound by evaluating R at the smallest possible value inside the brackets. In this case, the smallest value would be when all k and l are zero, except for one element, say ${}_1 = {}_1 = 1$, and $(x_1), (x_1) = 1$ (assuming normalized Φk's). Therefore, the lower bound on the regularization term $R(||f||)$ is:

$$R(||f||)R(v(x_j), v(x_j))$$

1.4

To jointly optimize both the loss terms and the penalty function with respect to v, we need to define the objective function and find its minimum with respect to v.

Let's consider an objective function that consists of the loss terms and the penalty function:

$$J(v) = L(v) + \lambda R(v)$$

Here, $L(v)$ represents the loss terms, λ is a hyperparameter that controls the trade-off between the loss and the penalty, and $R(v)$ is the penalty function.

To find the minimum of $J(v)$ with respect to v, we can take the derivative of $J(v)$ with respect to v and set it to zero:

$$\frac{dJ(v)}{dv} = \frac{dL(v)}{dv} + \lambda \frac{dR(v)}{dv} = 0$$

By solving this equation, we can find the optimal v that minimizes the objective function $J(v)$.

Now, let's move on to proving the representer's theorem. The representer's theorem states that for an optimization problem with a regularization term, the optimal solution can always be expressed as a linear combination of the training data points.

To prove the representer's theorem, let's assume that the optimal v, denoted as v^*, can be expressed as a linear combination of the training data points:

$$v^*(x) = \sum_{i=1}^{N} \alpha_i K(x, x_i)$$

Here, $K(x, x_i)$ represents the kernel function that measures the similarity between x and the training data point x_i, and α_i are the coefficients.

Now, we need to show that the optimal v^* satisfies the optimality conditions of the objective function $J(v)$.

Taking the derivative of $J(v)$ with respect to v and substituting v^* into the derivative, we have:

$$\left.\frac{dJ(v)}{dv}\right|_{v=v^*} = \left.\frac{dJ(v)}{dv}\right|_{v=v^*(x)} = \left.\frac{dJ(v)}{dv}\right|_{v=v^*(x)} = \left.\frac{dL(v)}{dv}\right|_{v=v^*(x)} + \lambda \left.\frac{dR(v)}{dv}\right|_{v=v^*(x)} = 0$$

Since the derivative is zero, we can conclude that the optimal v^* satisfies the optimality conditions of the objective function $J(v)$.

Thus, by assuming that v^* can be expressed as a linear combination of the training data points, we have shown that the optimal solution of the optimization problem can be represented in the form of $v^*(x) = \sum_{i=1}^{N} \alpha_i K(x, x_i)$, which is the representer's theorem.

1.5

The representer theorem provides insight into the form of the solution for kernel-based learning methods, such as Support Vector Machines (SVMs). It states that the optimal solution can be expressed as a linear combination of the training data points, with coefficients determined by the optimization process.

Specifically, for SVMs, the representer theorem implies that the decision function can be written as:

$$f(x) = \sum_{i=1}^{N} \alpha_i K(x, x_i) + b$$

Here, x represents the input feature vector, α_i are the coefficients determined during the optimization process, x_i are the support vectors, $K(x, x_i)$ is the kernel function that measures the similarity between x and x_i, and b is the bias term.

This form of the solution allows us to make predictions based on the similarity between the test point x and the support vectors. The coefficients α_i determine the importance or weight of each support vector in making predictions.

The final SVM solution is obtained by solving the optimization problem that aims to maximize the margin while minimizing the classification error. The optimization process involves finding the optimal values for the coefficients α_i and the bias term b that satisfy certain constraints.

In summary, the representer theorem provides a representation of the optimal solution in terms of the training data points and the kernel function, while the final SVM solution is obtained through the optimization process that determines the specific values for the coefficients and bias term that minimize the objective function.

2. Neural Networks Can be Seen as (almost) GPs!

2.1

2.1.1

$$E_\theta[f_k(x)] = E_b[b_k] + E[\sum_{j=1}^{H} v_{jk} h_j(x)] = 0 + \sum_{j=1}^{H} E[v_{jk} h_j(x)]$$

$$= \sum_{j=1}^{H} E_v[v_{jk}] E_u[h_j(x)] = \sum_{j=1}^{H} 0 \times E_u[h_j(x)] = 0$$

2.1.2

$$E[f_k(x) f_k(x')] = E_b[b_k^2] + E_\theta[\sum_{j=1}^{H} v_{jk} h_j(x)] E_b[b_k] + E_b[b_k] E_\theta[\sum_{j=1}^{H} v_{jk} h_j(x')]$$

$$+ E_\theta[\sum_{i=1}^{H} \sum_{j=1}^{H} v_{ik} h_i(x) v_{jk} h_j(x')] = \sigma_b^2 + \sum_{i=1}^{H} \sum_{j=1}^{H} E_u[v_{ik} v_{jk}] E_u[h_i(x) h_j(x')]$$

$$= \sigma_b^2 + \sum_{i=1}^{H} \sum_{j=1}^{H} \sigma_i^2 \, \delta_{ij} \, E_u[h_i(x) h_j(x')]$$

$$= \sigma_b^2 + \sigma_v^2 \sum_{j=1}^{H} E_u[h_j(x) h_j(x')] = \sigma_b^2 + \sigma_v^2 H E_u[h_j(x) h_j(x')]$$

2.1.3

Suppose we have H random variable that independent and $X_j = v_{jk} h_j(x)$. According to central limit theorem, we have:

$$\text{if } H \to \inf \Rightarrow X = \sum_{j=1}^{H} X_j$$

that has normal distribution with mean $H\mu$ and covariance matrix of $H\sigma^2$ where $\mu = E[X_j]$ and $\sigma^2 = Var(X_j)$.
According to 2.1.1 $\mu = 0$ and $\sigma^2 = Var(v_{jk} h_j(x)) = Var(v_{jk}) Var(h_j(x)) =$

$\sigma_u^2 E[h_j(x)h_j(x')]$

$$\sum_{j=1}^{H} v_{jk}h_j(x) = f_k(x) - b_k \approx N(0, \sigma_u^2 E[h_j(x)h_j(x')])$$

b has no effect $\Rightarrow f_k(x) \approx N(0, \sigma_u^2 E[h_j(x)h_j(x')])$

2.2

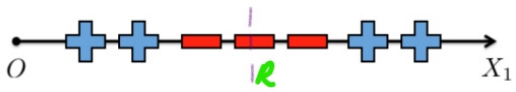
Neural tangent kernels (NTKs) are a way of studying the behavior and performance of deep neural networks (DNNs) using tools from kernel methods. A kernel is a function that measures the similarity between two inputs, and a kernel method is a technique that uses kernels to solve various learning problems. NTKs are kernels that are derived from the gradients of DNNs with respect to their parameters, and they describe how DNNs change during training by gradient descent.
One of the main results of NTK theory is that, in the limit of infinite width (i.e., when the number of neurons in each layer goes to infinity), the NTK becomes constant and equal to an explicit limiting kernel. This means that training a wide DNN by gradient descent is equivalent to performing kernel regression with the limiting NTK. This equivalence reveals a connection between DNNs and Gaussian processes, and allows for simple analysis of the convergence and generalization properties of DNNs. However, this equivalence may not hold for finite-width networks or for other training methods

3. SVM

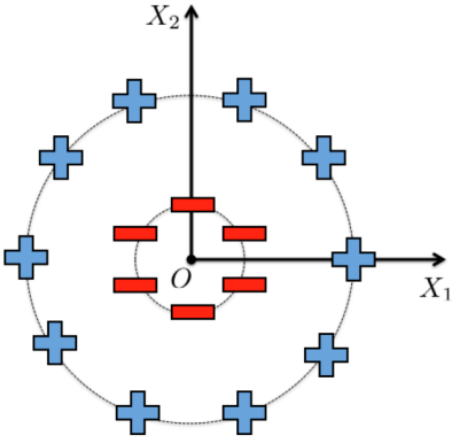
3.1

$$\begin{cases} \xi_i = 0 & \text{point lies on the margin or output of the margin and classified correctly} \\ 0 < \xi_i < 1 & \text{inside margin(not crossed hyperplane decision) and classified correctly} \\ \xi_i \geqslant 1 & \text{crossed hyperplane decision and classified incorrectly} \end{cases}$$

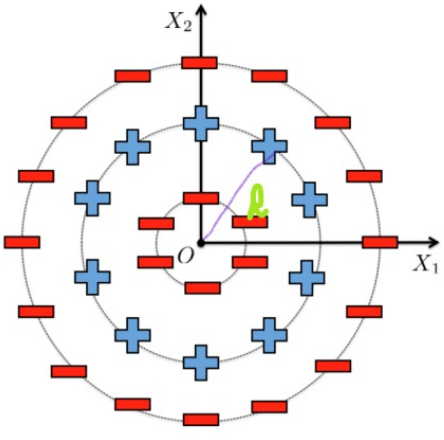
3.2



Proper mapping is $\Phi(X_1) = [(X_1 - R)^2]$



Proper mapping is $\Phi(X_1, X_2) = [X_1^2, X_2^2]$



Proper mapping is $\Phi(X_1, X_2) = [(X_1^2 + X_2^2 - R^2)^2]$

4. Interpretation Via Maximum Projection Spread

Suppose $X \in \mathbb{R}^{N \times D}$ is our data. So:

$$\bar{x} = \frac{1}{N} X^T \mathbf{1}$$

$$S = \frac{1}{N} X^T X - \frac{1}{N^2} X^T \mathbf{1}\mathbf{1}^T X = \frac{1}{N} X^T (I - \frac{1}{N}\mathbf{1}\mathbf{1}^T) X = \frac{1}{N} X^T H X$$

We can easily show that H is symmetric and idempotent, So it is projection matrix.

$$Hv = v - \frac{1}{N}\mathbf{1}\mathbf{1}^T v = v - \bar{v}$$

Thus H removes the mean of the vector from each coordinate.

$$S = \frac{1}{N} X^T H X = \frac{1}{N} X^T H^2 X = \frac{1}{N} X^T H^T H X = \frac{1}{N} (HX)^T (HX)$$

The problem for PCA can be formulated as:

max $\quad u^T S u$

subject to $\quad ||u|| = 1$

Using Spectral theorem, we can write S as $P \Lambda P^T$.

Assume the maximum spread direction is u and consider the following definition:

$$b = P^T u \Rightarrow u = Pb$$

$$u^T S u = (Pb)^T (P \Lambda P^T)(Pb) = b^T (P P^T) \Lambda (P^T P) b = b^T \Lambda b = \sum_{i=1}^{D} \lambda_i b_i^2 \leq \lambda_1 \sum_{i=1}^{D} b_i^2$$

$$||b||^2 = ||P^T u||^2 = (P^T u)^T (P^T u) = u^T P P^T u = u^T u = ||u||^2 = 1$$

$$\Rightarrow u^T S u = \sum_{i=1}^{D} \lambda_i b_i^2 \leq \lambda_1 \sum_{i=1}^{D} b_i^2 = \lambda_1 ||b||^2 = \lambda_1$$

Now check the variance for $u = p_1$:

$$b = P^T p_1 = e_1$$

$$\Rightarrow p_1^T S p_1 = b^T \Lambda b = \sum_{i=1}^{D} \lambda_i b_i^2 = \lambda_1$$

So $u = p_1$ is the direction of maximum spread.
We can show the following in an almost similar way:

$$\begin{aligned} p_2 \in \text{argmax} \quad & u^T S u \\ \text{subject to} \quad & ||u|| = 1 \quad , \quad u \perp p_1 \end{aligned}$$

$$\begin{aligned} p_3 \in \text{argmax} \quad & u^T S u \\ \text{subject to} \quad & ||u|| = 1 \quad , \quad u \perp p_1, p_2 \end{aligned}$$

$$\vdots$$

$$\begin{aligned} p_j \in \text{argmax} \quad & u^T S u \\ \text{subject to} \quad & ||u|| = 1 \quad , \quad u \perp p_1, p_2, \ldots, p_{j-1} \end{aligned}$$

5. Interpretation Via Reconstruction

$$||x_i - \sum_{j=1}^{K} z_{ij} v_j||^2 = (x_i - \sum_{j=1}^{K} z_{ij} v_j)^T (x_i - \sum_{j=1}^{K} z_{ij} v_j)$$

$$= x_i^T x_i - 2\sum_{j=1}^{K} z_{ij} x_i^T v_j + \sum_{t=1}^{K}\sum_{j=1}^{K} z_{ij} z_{it} v_j^T v_t$$

We know that:

$$v_i^T v_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

$$= x_i^T x_i - 2\sum_{j=1}^{K} z_{ij} x_i^T v_j + \sum_{j=1}^{K} z_{ij}^2 v_j^T v_j$$

Also we know that:

$$z_{ij} = x_i^T v_j$$

$$= x_i^T x_i - 2\sum_{j=1}^{K} v_j^T x_i x_i^T v_j + \sum_{j=1}^{K} x_i^T v_j v_j^T x_i v_j^T v_j$$

$$= x_i^T x_i - 2\sum_{j=1}^{K} v_j^T x_i x_i^T v_j + \sum_{j=1}^{K} x_i^T v_j v_j^T x_i = x_i^T x_i - 2\sum_{j=1}^{K} v_j^T x_i x_i^T v_j + \sum_{j=1}^{K} v_j^T x_i x_i^T v_j$$

$$= x_i^T x_i - \sum_{j=1}^{K} v_j^T x_i x_i^T v_j \quad \checkmark$$

6. Whitening Using PCA

We know that:

$$S = \frac{1}{N}(HX)^T(HX) = \frac{1}{N}\tilde{X}^T\tilde{X}$$

Assume we define $y = \Lambda^{-\frac{1}{2}}P^T\tilde{X} \Rightarrow \bar{y} = \overline{\Lambda^{-\frac{1}{2}}P^T\tilde{X}} = 0$ because $\bar{\tilde{X}} = 0$

$$S_y = \frac{1}{N}\sum_{n-1}^{N} y_n y_n^T = \frac{1}{N}\sum_{n=1}^{N} \Lambda^{-\frac{1}{2}}P^T\tilde{X}_n\tilde{X_n}^T P\Lambda^{-\frac{1}{2}} = \frac{1}{N}P^T\tilde{X}_n\tilde{X_n}^T P\Lambda^{-1}$$

$$= P^T(\frac{1}{N}\tilde{X}_n\tilde{X_n}^T)P\Lambda^{-1} = P^T S P\Lambda^{-1} = \Lambda\Lambda^{-1} = I$$

7. Principal Components with Missing Values

Iterative PCA is a method of imputing missing values in a dataset by using principal component analysis (PCA) to estimate the missing values as a function of other features. It involves the following steps:

- Fill the missing values with some initial values, such as the mean or median of each column.
- Perform PCA on the filled data matrix and obtain the principal components and scores.
- Use the principal components and scores to reconstruct the data matrix and update the missing values with the reconstructed values.
- Repeat steps 2 and 3 until convergence or a maximum number of iterations is reached.

For example suppose $X = \begin{bmatrix} 1.2 & 3.4 & 5.6 & 7.8 \\ 2.3 & 4.5 & 6.7 & \text{NAN} \\ 3.4 & 5.6 & 7.8 & 9 \\ 4.5 & 6.7 & 8.9 & 10.1 \\ 5.6 & 7.8 & 9 & 11.2 \end{bmatrix}$

First Step) $\frac{7.8+9+10.1+11.2}{4} = 9.525 \Rightarrow X = \begin{bmatrix} 1.2 & 3.4 & 5.6 & 7.8 \\ 2.3 & 4.5 & 6.7 & 9.525 \\ 3.4 & 5.6 & 7.8 & 9 \\ 4.5 & 6.7 & 8.9 & 10.1 \\ 5.6 & 7.8 & 9 & 11.2 \end{bmatrix}$

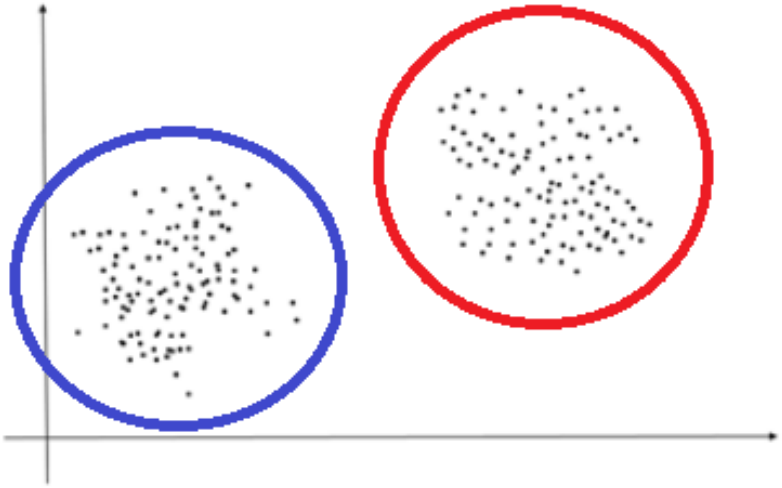
Second Step) PCA Component is $\begin{bmatrix} -0.377 & -0.467 & -0.557 & -0.577 \\ -0.577 & -0.377 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \end{bmatrix}$

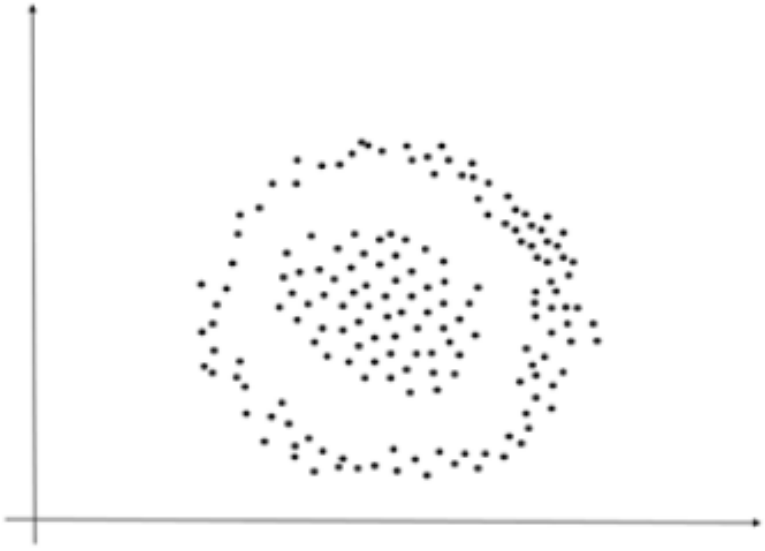
Third Step) reconstruct $X = \begin{bmatrix} 1 & 3 & 5 & 8 \\ 2 & 5 & 7 & 10 \\ 3 & 6 & 8 & 9 \\ 5 & 7 & 9 & 10 \\ 6 & 8 & 9 & 11 \end{bmatrix}$

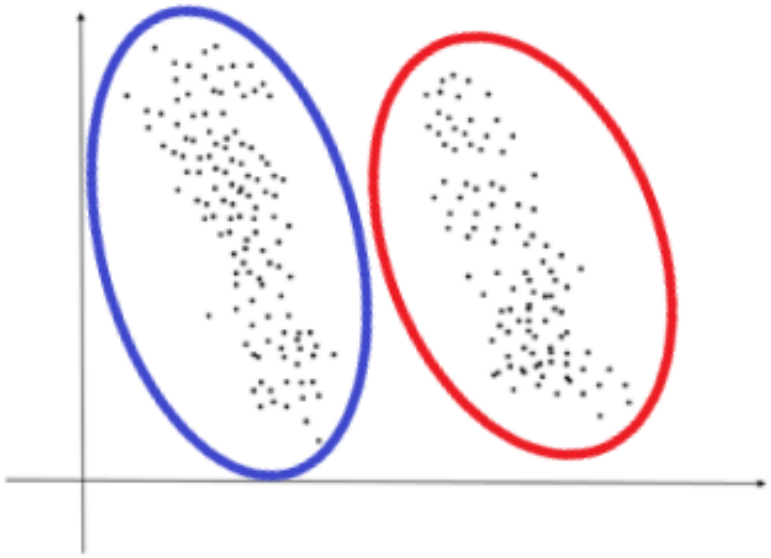
Fourth Step) Repeat steps 2 and 3

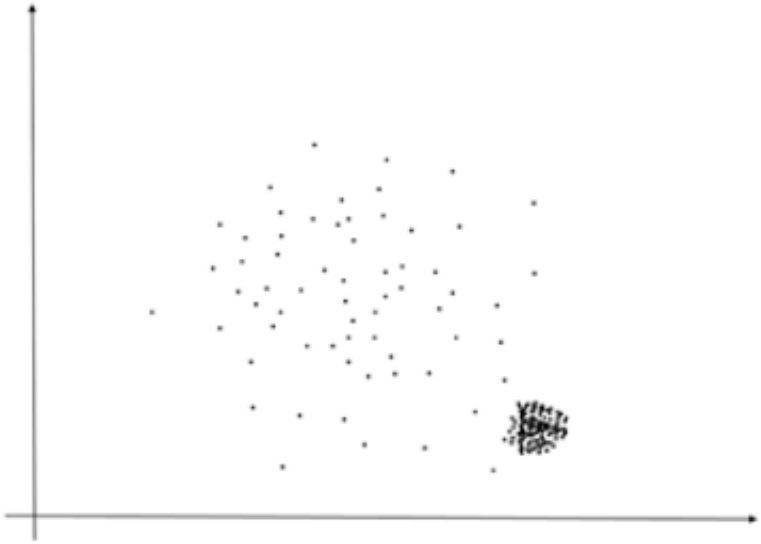
8. Clustering

8.1





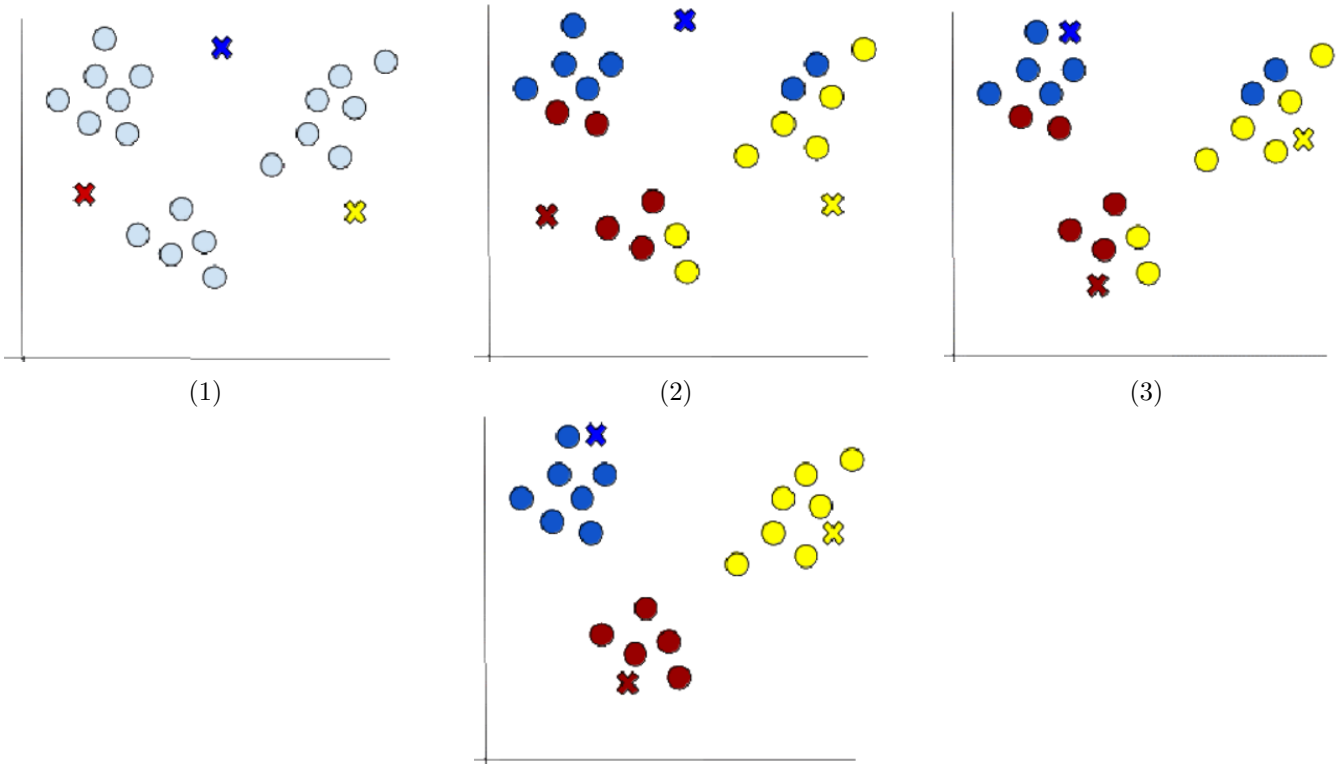




8.2

Yes, it is important to choose the initial points carefully in K-means clustering because the initial points can significantly affect the convergence and the quality of the clustering result. K-means is sensitive to the initial positions of the cluster centroids because the algorithm aims to minimize the within-cluster sum of squares,

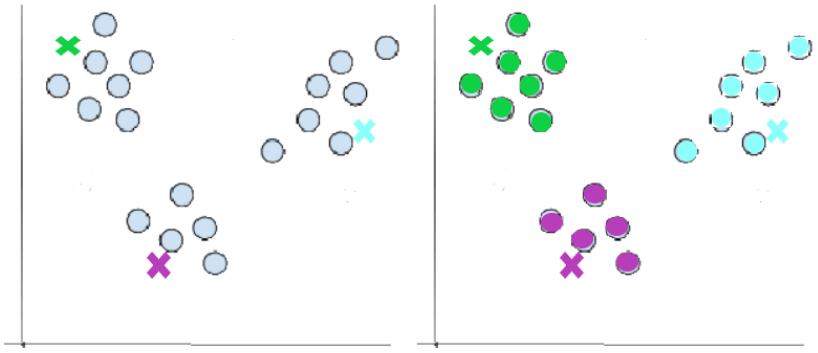
and the initial positions can influence the outcome of this optimization process.
Example for very bad choices for initial points



(1) (2) (3)

(4)

Example for very good choices for initial points



(1) (2)

8.3

The Kmeans++ algorithm is an improvement of the standard Kmeans algorithm that aims to avoid poor clusterings by choosing the initial cluster centers more carefully. The Kmeans++ algorithm works as follows:

- Initialize the first centroid randomly from the data points.
- For each remaining centroid, choose the next one based on a probability distribution that is proportional to the squared distance from each data point to its nearest centroid.
- Repeat step 2 until all centroids have been initialized.
- Proceed with the standard Kmeans algorithm using the sampled centroids as initial centers.

WCSS(Within-Cluster Sum of Squares) is a measure of the compactness of clusters in the K-means algorithm. It represents the sum of the squared distances between each data point and its assigned centroid within a cluster. WCSS is calculated for each cluster individually, and the overall WCSS is the sum of WCSS values for all clusters. The goal of K-means is to minimize the WCSS, as a lower value indicates that the data points within each cluster are closer to their centroid.
The elbow method is a technique used to determine the optimal number of clusters (K) in K-means clustering. It is based on evaluating the WCSS for different values of K. The idea is to plot the number of clusters against the corresponding WCSS values and observe the plot. The plot often forms a shape resembling an elbow. The ”elbow point” is the value of K where the WCSS starts to decrease more slowly, indicating that adding more clusters does not significantly improve the clustering quality. This elbow point is often considered as the optimal value of K.