ECE 479: IoT and Cognitive Computing

Spring 2024, Homework HW 1

Yanghonghui Chen/ yc47

Question 1: Edge Computing and Sensors (25 pts)

1. List two advantages and disadvantages of using Cloud Computing or Edge Computing for an IoT system. [**8 pts**]
   - Cloud Computing:
     - Advantages:
       (a) Scalability: Cloud Computing offers virtually limitless scalability, allowing IoT systems to easily accommodate growing data volumes and user bases without significant infrastructure changes.
       (b) Cost-Efficiency: Cloud services typically operate on a pay-as-you-go model, reducing upfront costs for hardware and maintenance. Additionally, the centralized nature of Cloud Computing can lead to cost savings in terms of administration and management.
     - Disadvantages:
       (a) Latency: Transmitting data to and from the cloud can introduce latency, which may not be acceptable for applications requiring real-time responses, such as industrial automation or autonomous vehicles.
       (b) Dependency on Internet Connectivity: Cloud-based IoT systems rely heavily on stable internet connections. Downtime or network issues can disrupt data flow and functionality, impacting critical operations.
   - Edge Computing:
     - Advantages:
       (a) Low Latency: Edge Computing brings computational resources closer to IoT devices, reducing latency by processing data locally. This is crucial for applications where real-time responses are essential, such as remote healthcare monitoring or smart grids.

(b) Data Privacy and Security: By processing data locally, Edge Computing can enhance privacy and security by minimizing the need to transmit sensitive information over networks, reducing exposure to potential cyber threats.

– Disadvantages:

(a) Limited Scalability: Edge Computing devices typically have limited computational power and storage capacity compared to cloud servers, which can pose challenges when dealing with large-scale IoT deployments or sudden spikes in demand.

(b) Complexity of Management: Managing a distributed network of edge devices can be more complex than centralized cloud infrastructure, requiring additional resources for monitoring, maintenance, and updates.

2. For the following applications, which paradigm (Cloud Computing or Edge Computing) is more suitable? Briefly explain why. **[6 pts]**

- Health data collected by a smartwatch to track your daily activities :
  - Suitable Paradigm: Cloud Computing.
  - Explanation:Cloud Computing offers several advantages for storing and processing health data collected by smartwatches. Firstly, cloud-based storage solutions provide virtually unlimited scalability, allowing for the storage of vast amounts of health data from multiple users over extended periods. Additionally, Cloud Computing enables centralized data management, making it easier to aggregate and analyze data from various sources, such as different types of smartwatches or wearable devices. This centralized approach facilitates data sharing between healthcare providers and researchers, supporting collaborative efforts in health research and analysis. Moreover, cloud-based solutions often incorporate robust security measures and compliance features to ensure data privacy and regulatory compliance, addressing concerns related to the security and confidentiality of sensitive health information.
- Temperature sensors are placed inside a refrigerated storage container to regulate the temperature during the shipping process:
  - Suitable Paradigm: Edge Computing.
  - Explanation: Temperature regulation in shipping containers demands real-time monitoring and control to ensure the integrity of perishable goods. Edge Computing enables local processing of sensor data within the container, facilitating immediate adjustments to maintain optimal

temperatures without reliance on continuous cloud connectivity. This approach enhances reliability and reduces the risk of disruptions during transit.

- License plate readers at toll plazas:
  - Suitable Paradigm: Cloud Computing.
  - Explanation: Cloud Computing can also offer advantages for storing and processing data from license plate readers at toll plazas. Cloud-based storage solutions provide centralized data management, enabling toll plaza operators to aggregate and analyze data from multiple toll plazas across different locations. This centralized approach facilitates efficient data management and analysis, supporting tasks such as toll collection, traffic management, and enforcement operations. Additionally, cloud-based solutions offer scalability, allowing toll plaza operators to handle fluctuations in data volume and traffic patterns effectively. Moreover, cloud-based solutions often incorporate redundancy and disaster recovery features, ensuring data availability and reliability even in the event of hardware failures or natural disasters.
- Medical wearable devices that detect when you fall:
  - Suitable Paradigm: Edge Computing.
  - Explanation: Fall detection using medical wearable devices necessitates immediate response and alerting in emergency situations. Edge Computing supports local processing of sensor data on the device itself, enabling rapid detection of fall events and timely notifications without depending on cloud connectivity. This approach ensures swift assistance for individuals in distress, even in environments where internet access may be limited or unreliable.

3. Wireless Sensors Network (WSN) Aggregation Strategy. [**11 pts**]

All nodes except A are sensors, and node A is the sink node (gateway). The numbers on the edges are the corresponding transmission latency. To find out the shortest path between two connected nodes in a graph, you may need the help of the Dijkstra's algorithm.

We want to collect data from sensors K, E, and G and transmit them back to sink node A. Please propose a plan, and calculate minimum **total** latency and the number of operations required in the transmission.

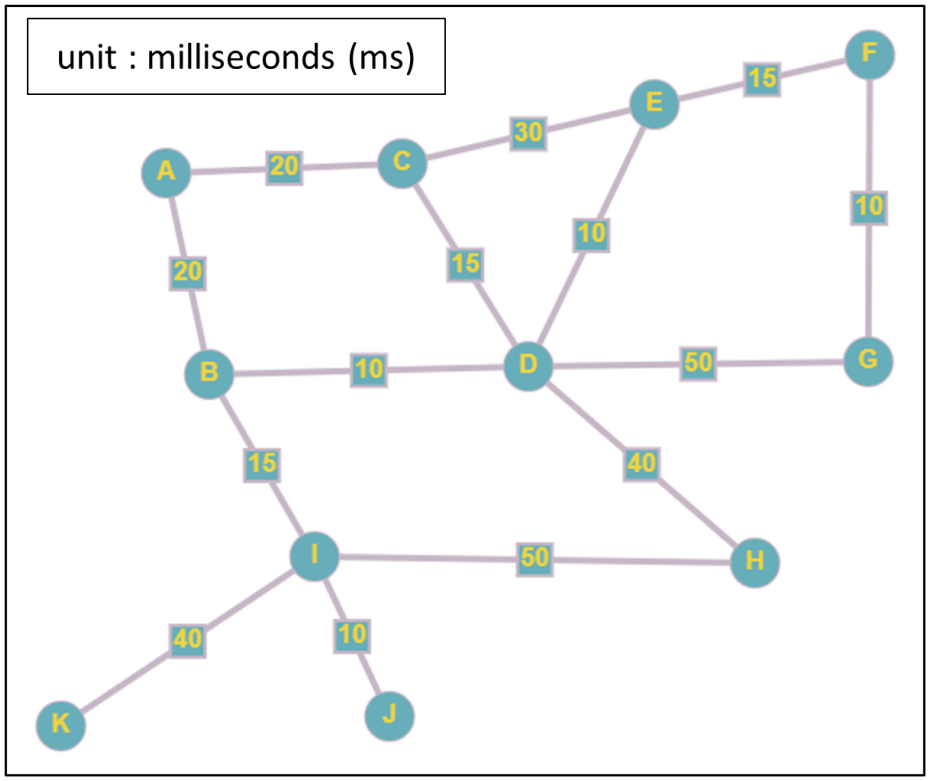

Figure 1: WSN topology

Assume each aggregation at one node requires 5 milliseconds (ms) to complete. Justify your answer and fill in the following table. (You may not need to fill in all columns.)

# of Operations	K	E	G	B	
Transmission	2	2	2	1	
Aggregation	1	1	1	0	
Total	3	3	3	1	

Table 1: Wireless Sensors Network Traversing

Question 2: Basic Python and NumPy programming (15 pts)

1. **Python One-Liners: Basics [6 pts]**

   (a) Write down **a single line** Python expression that calculates $\sum_{n=0}^{i} n!$ using only Python built-in functions. **[1 pts]**

   result= sum(math.factorial(n) for n in range(i+1))

   (b) Write down **a single line** of Python code to sum the 0, 3, 6, 9,...,(3n)th elements from a list A using only Python built-in functions. **[1 pts]**

   result = sum(A[::3])

   (c) Write down **a single line** of Python code to sum the 2, 5, 8, 11, (3n+2)th elements from a list A using only Python built-in functions. **[1 pts]**

   result = sum(A[2::3])

   (d) Write down **a single line** of Python code to reverse given list A and save it to list B. **[1 pts]**

   B = A[::-1]

   (e) Given two lists of numbers in Python, A and B, write down **a single line** of Python code to construct a new list of pairs of corresponding elements in two lists. For example, A = [1, 2, 3], B = ['a', 's', 'd', 'f'], then your code should give [(1, 'a'), (2, 's'), (3, 'd')]. **[1 pts]**

   pairs = list(zip(A, B))

   (f) Given two lists of numbers in Python, A and B, write down **a single line** of Python code to construct a new list of elements in A that appear in B as well. For example, A = [1, 2, 4, 4, 2, 1], B = [1, 4], then your code should give [1, 4, 4, 1] **[1 pts]**

   result = [x for x in A if x in B]

2. **More one-liners: List and String Operations [4 pts]**
   Use the following list and string methods:
   `append(), count(), extend(), insert(), join(), remove(), split()`
   Fill in the blank (parenthesis) to output the expected results. The comments are what you should expect after you have run the following line of code. You should only use one method in each blank.

```
# split string s0 by space and put into a list s0_list
# output: ['Oppenheimer', 'drinks', 'some', 'coffee', 'coffee',
   'coffee', 'coffee']
s0 = "Oppenheimer drinks some coffee coffee coffee coffee"
s0_list = (        s0.split()                              )

# count the occurrence of element 'coffee'
```

```
# output: 4
count = (       s0_list.count('coffee')
           )

# keep only the first four elements in the list
# output: ['Oppenheimer', 'drinks', 'some', 'coffee']
s0_list = (          s0_list[:4]                              )

# take out element 'some' from the list
# output: ['Oppenheimer', 'drinks', 'coffee']
s0_list.(        s0_list.remove('some')
         )

# put element 'much' in the list between 'drinks' and 'coffee'
# output: ['Oppenheimer', 'drinks', 'much', 'coffee']
s0_list.(           s0_list.insert(2, 'much')
           )

# put element 'everyday' at the end of the list
# output: ['Oppenheimer', 'drinks', 'much', 'coffee', 'everyday
   ']
s0_list.(           s0_list.append('everyday')
            )

# split string s1 by space and put into a list
# output: ['in', 'the', 'afternoon']
s1 = "in the afternoon"
s1_list = (       s1.split()                                  )

# put s1_list at the end of s0_list
# output: ['Oppenheimer', 'drinks', 'much', 'coffee', 'everyday
   ', 'in', 'the', 'afternoon']
s0_list.(        s0_list.extend(s1_list)
          )

# make a string from the list, connected with a space ' '
# output: Oppenheimer drinks much coffee everyday in the
   afternoon
s = (       ' '.join(s0_list)
      )
```

3. **NumPy Slicing [2 pts]**

The **shape** of a NumPy array *tmp* is (5, 3, 4, 2), which corresponds to dimensions $[i, j, k, l]$.

```
tmp = tmp[2:4, :-1, ::2, :1]
```

What is the shape of the NumPy array *tmp* and which part of the original NumPy array *tmp* has been extracted after this slicing?

The shape of the resulting array `tmp` is (2, 2, 2, 1)
Dimension i: Rows 2 and 3 are included.
Dimension j: All columns except the last one are included.
Dimension k: Every other row is included.
Dimension l: Only the first element is included

Please describe it in terms of i, j, k, l dimension. (for example : row 0 and row 1 of dimension i and .... of dimension j and ...)

4. **NumPy Operations [3 pts]**

We have `A = np.array([1, 2, 3, 4, 5, 6])` and a mystery NumPy array `B`. Printing `A*B` gives the output of `[2 14 9 20 30 24]`. What is this mystery NumPy array `B`?

Answer:
A = np.array([1, 2, 3, 4, 5, 6]);
result = np.array([2, 14, 9, 20, 30, 24]);
B = result / A;
print("Mystery array B:", B);
Mystery array B: [2. 7. 3. 5. 6. 4.];

Now suppose that NumPy array `A1` and `B1` are transformed from A and B using NumPy array manipulation functions.

Answer:
A1 = A.reshape(3, 2);
B= np.flip(B);
B1 = B.reshape(2, 3);
result = np.dot(A1, B1).ravel();
print(result);

Please find out a way to perform these transformations from `A` and `B`, using NumPy array operations, such that

```
print(np.dot(A1, B1).ravel())
```

gives the output of [10 20 9 24 46 23 38 72 37]

Question 3: Data exploration (20 pts)

1. **Data visualization**
   Suppose you are analyzing a dataset that contains information about the sales performance of a retail company over the past year. You want to gain insights into the data and communicate your findings effectively.

   (a) How would you create a scatter plot to visualize the relationship between the company's advertising expenditure and its monthly sales revenue? What do the resulting patterns on this scatter plot tell you about this relationship? **[3 pts]**

   (a) To create a scatter plot visualizing the relationship between the company's advertising expenditure and its monthly sales revenue, you would typically plot the advertising expenditure on the x-axis and the monthly sales revenue on the y-axis. Each point on the scatter plot represents a specific month's data. The resulting patterns on the scatter plot can reveal insights into the relationship between advertising expenditure and sales revenue. For example:
   If the points on the scatter plot form a clear upward trend, it indicates a positive correlation between advertising expenditure and sales revenue, suggesting that higher advertising expenditure is associated with higher sales revenue.

   – If the points appear scattered and do not exhibit a clear trend, it suggests a weak or no correlation between advertising expenditure and sales revenue

   (b) To understand the distribution of monthly sales revenue, what type of chart or graph would you use, and why? Provide a brief explanation of how you would construct this chart, and what insights it might provide.**[3 pts]**

   A histogram is an ideal choice to understand the distribution of monthly sales revenue because it provides a visual representation of the frequency distribution of continuous data. By dividing the range of sales revenue into intervals (bins) and plotting the frequency of observations within each bin, a histogram allows for insights into the shape, central tendency, spread, and potential outliers of the distribution.
   Constructing a histogram involves determining the range of sales revenue values, dividing this range into intervals, counting the frequency of observations in each interval, and plotting the histogram with intervals on the x-axis and frequency on the y-axis.

Insights from the histogram include:

- Shape of the distribution: Indicating whether the distribution is symmetric, skewed, uniform, or multimodal.
- Central tendency and spread: Providing information on the mean, median, standard deviation, and interquartile range.
- Outliers: Identifying observations that fall far from the main body of the histogram.
- Overall pattern: Revealing patterns such as seasonality, trends, or changes in sales revenue over time.

(c) You want to represent the contribution of each product category to the total annual revenue. What type of chart would be suitable for this purpose? Explain how you would create and interpret such a chart to effectively communicate the data.**[3 pts]**

A pie chart is a suitable choice for representing the contribution of each product category to the total annual revenue because it provides a clear visual representation of the proportions or percentages of each category relative to the whole.Constructing a pie chart involves:

- Calculating the contribution of each product category to the total annual revenue: Aggregate the revenue data by product category and calculate the percentage or proportion of revenue generated by each category relative to the total annual revenue.
- Creating the pie chart: Use a pie chart to visually represent the contributions of each product category. Each slice of the pie represents a product category, and the size of each slice corresponds to the proportion of revenue generated by that category.

Insights from the pie chart include:

- Relative importance of each product category: The size of each slice indicates the proportion of revenue contributed by each category, allowing stakeholders to quickly understand which categories are major contributors to the total revenue.
- Comparison between categories: Stakeholders can easily compare the contributions of different product categories and identify which categories have the largest and smallest shares of revenue.
- Total revenue visualization: The entire pie represents the total annual revenue, providing context for understanding the contributions

of individual categories within the overall revenue picture.

2. Finally, suppose you are interested in predicting future monthly sales revenue based on the past year's data. How would you go about fitting a regression line to the data points in the scatter plot you created earlier? Describe the steps involved and what information the regression line can provide.[**3 pts**]

    i. Data Collection: Gather past year's data on advertising expenditure and monthly sales revenue.

    ii. Data Preparation: Organize the data into a dataset with two columns: one for advertising expenditure (independent variable) and one for monthly sales revenue (dependent variable).

    iii. Scatter Plot: Create a scatter plot with advertising expenditure on the x-axis and monthly sales revenue on the y-axis. Each data point represents a specific month's data.

    iv. Regression Analysis: Perform linear regression analysis on the data points to fit a regression line to the scatter plot. This involves finding the equation of the line that best fits the data points using least squares regression.

    v. Interpretation: Examine the resulting regression line on the scatter plot. If the points on the scatter plot form a clear upward trend, it indicates a positive correlation between advertising expenditure and sales revenue, suggesting that higher advertising expenditure is associated with higher sales revenue. On the other hand, if the points appear scattered and do not exhibit a clear trend, it suggests a weak or no correlation between advertising expenditure and sales revenue.

    vi. Prediction: Once the regression line is fitted, it can be used to make predictions about future monthly sales revenue based on different levels of advertising expenditure. By plugging in different values of advertising expenditure into the regression equation, you can estimate the corresponding monthly sales revenue.

Fitting the Data

Suppose that we have collected some data points for training and plotted them in the scattered plots, shown below. To find the trend of these data points, we constructed three regression models. Later, we want to use one of these models to describe the incoming unknown data points. Observe the following three graphs and answer the questions.

1. Among the three models, which one do you think is the best? Why do you choose this model? [**2 pts**]

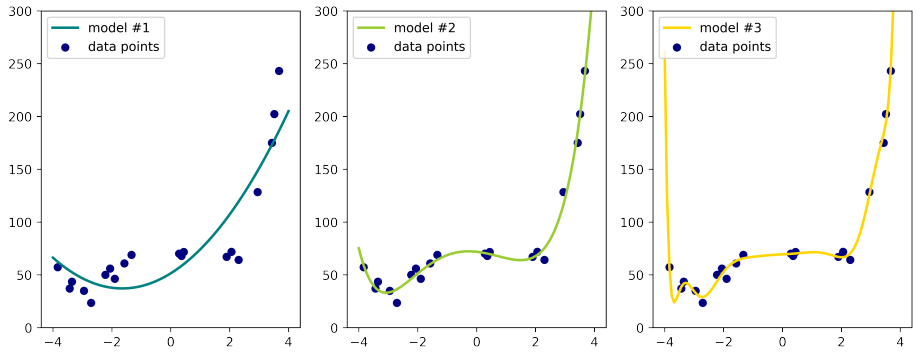

Figure 2: Three Different Models

I think the second one is the best since the second model fits most of the data appropriately and it helps to reveal a relatively good pattern instead of under-fitting or over-fitting.

2. What are the potential issues with the other two? Please explain in two to three sentences. **[2 pts]**

   The first one is a classical under-fitting model since it is too simple to fit most of the data and some data points are neglected. This will leads to high bias, poor predictive performance and inability to generalize. For the third model, it is a classical over-fitting model which tries to fit each single data point and thus becomes too sensitive. This will lead to high variance, poor generalization and difficulty in interpretation.

Training, Validation, and Testing

When developing a machine learning model, we usually need to evaluate the model's accuracy before testing it on real-world data. Since we only have access to the training dataset, we often have to split it into two parts, one for training and the other for validation.

1. In about three sentences, describe why we need a separate testing dataset to evaluate the trained model. **[2 pts]**

   We need a separate testing dataset to evaluate the trained model's performance on unseen data that the model has not been exposed to during training or validation. This ensures that the model's performance is unbiased and provides a reliable estimate of its generalization ability.

2. Why do we need validation/cross-validation if the testing set is enough to evaluate the real-world performance of a model? That is, why can't we tune the model based on the testing dataset directly? **[2 pts]**

   Validation or cross-validation is necessary even if we have a testing set

because tuning the model based on the testing dataset directly can lead to overfitting to the testing data. Validation/cross-validation allows us to assess the model's performance on multiple subsets of the training data, helping to prevent overfitting and providing a more accurate estimate of the model's real-world performance. Additionally, using a separate validation set allows us to iteratively adjust hyperparameters and select the best-performing model without biasing the final evaluation on the testing set.

Question 4: PCA and Preprocessing of data (18 pts)

1. **Curse of dimensionality[3 pts]**
   Briefly explain "the curse of dimensionality". Why high dimensional data is a "curse"? What makes it difficult to model high-dimensional data?

   Curse of Dimensionality: The curse of dimensionality refers to the phenomenon where the performance of machine learning algorithms deteriorates as the number of features or dimensions in the dataset increases. High-dimensional data is considered a "curse" because it becomes increasingly sparse, making it difficult to model accurately due to the exponential growth in the volume of the data space. This sparsity leads to challenges such as increased computational complexity, overfitting, and the need for larger datasets to maintain model performance.

2. **Statistical interpretation of PCA[3 pts]**
   In the lectures, you have learned the statistical interpretation of PCA. Briefly describe it below.

   Statistical Interpretation of PCA: Principal Component Analysis (PCA) is a statistical technique used to reduce the dimensionality of a dataset while preserving as much of the variability as possible. It achieves this by identifying the principal components, which are orthogonal vectors that represent the directions of maximum variance in the data. The statistical interpretation of PCA involves finding these principal components as linear combinations of the original features, where each component captures a certain amount of variance in the data. By selecting a subset of the principal components that capture the most variance, PCA effectively reduces the dimensionality of the dataset while retaining the most important information.

3. **PCA on simple data[12 pts]** Suppose you have a tiny dataset with two features $X1$ and $X2$. You want to perform PCA on this dataset to reduce its dimensionality (i.e., from 2 to 1).

   ```
   Data Point 1: X1 = 2, X2 = 4
   ```

```
Data Point 2: X1 = 3, X2 = 6
Data Point 3: X1 = 5, X2 = 10
Data Point 4: X1 = 7, X2 = 14
```

Please follow the steps below and find the projection of this dataset onto the principal component.

(a) Calculate the mean values of $X1$ and $X2$ and then center the dataset by subtracting these means from each data point.

Mean of X1 = (2 + 3 + 5 + 7) / 4 = 4.25
Mean of X2 = (4 + 6 + 10 + 14) / 4 = 8.5
Center the dataset by subtracting the means from each data point:
Data Point 1 (Centered): X1 = -2.25, X2 = -4.5
Data Point 2 (Centered): X1 = -1.25, X2 = -2.5
Data Point 3 (Centered): X1 = 0.75, X2 = 1.5
Data Point 4 (Centered): X1 = 2.75, X2 = 5.5

(b) Calculate the covariance matrix for the centered dataset. Show the full covariance matrix.

Covariance Matrix:
[[4.91666667 9.83333333]
[9.83333333 19.66666667]]

(c) Calculate the eigenvalues and eigenvectors of the covariance matrix you computed in the previous step. Provide both the eigenvalues and the corresponding eigenvectors. Select the most significant principal component (eigenvector) based on the eigenvalues calculated.

Eigenvalues:
[0. 24.58333333]
Eigenvectors:
[[-0.89442719 -0.4472136]
[0.4472136 -0.89442719]]
The largest eigenvalue is 24.58333333 so the most significant principal component is [0.4472136 -0.89442719]

(d) Project the centered data onto the chosen principal component. Show the projection of the data points along this component by taking the dot product of the centered data points and the principal component vector.

Projection of Data Points:
[[3.01869176]
[1.67705098]
[-1.00623058]
[-3.68951215]]

Question 5: kNN and Linear Regression (22 pts)

1. **Simple regression with kNN [2 pts]**
   In your own words, describe how the kNN algorithm calculates the regression. You can assume that all the dependent and independent variables are numerical.

   Description: The kNN (k-Nearest Neighbors) algorithm calculates regression by first identifying the k nearest neighbors of the query point (the point for which we want to make a prediction) in the feature space. It then predicts the target value for the query point by averaging the target values of its k nearest neighbors. In other words, the predicted value is a weighted average of the target values of the k nearest neighbors, where the weights are determined by the distance of each neighbor from the query point.

2. **kNN Classification**
   Given a dataset containing points in a 3D space belonging to one of three classes (Class A, Class B, and Class C), with each point having three features (x, y, z), calculate the Euclidean distance between a new point $P(4, 4, 4)$ and all the points in the dataset. The dataset is as follows:

   - Class A: $(1, 2, 1)$, $(2, 3, 2)$, $(3, 3, 1)$, $(2, 2, 2)$
   - Class B: $(7, 2, 1)$, $(8, 3, 2)$, $(9, 1, 3)$, $(7, 4, 1)$
   - Class C: $(5, 7, 7)$, $(7, 2, 5)$, $(6, 5, 3)$, $(5, 5, 6)$

   (a) **Calculate the Euclidean Distance [4 pts]**
   Calculate the Euclidean distance between the new point $P(4, 4, 4)$ and all the points in the dataset using the formula:

   $$d(P, Q) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

   Answer:
   Euclidean distances to Class A:
   [4.69041576 3.0 3.31662479 3.46410162]
   Euclidean distances to Class B:
   [4.69041576 4.58257569 5.91607978 4.24264069]
   Euclidean distances to Class C:
   [4.35889894 3.74165739 2.44948974 2.44948974]

   (b) **Determine the Optimal k [4 pts]**
   Based on the calculated distances, classify point P using kNN for $k = 3$, $k = 4$, and $k = 5$. With a tie between Class A and Class C for $k = 4$, additional criteria (e.g., weighted distance) might be needed for classification. Which value of k provides the most reasonable classification for point P? Discuss the impact of different k values on the classification

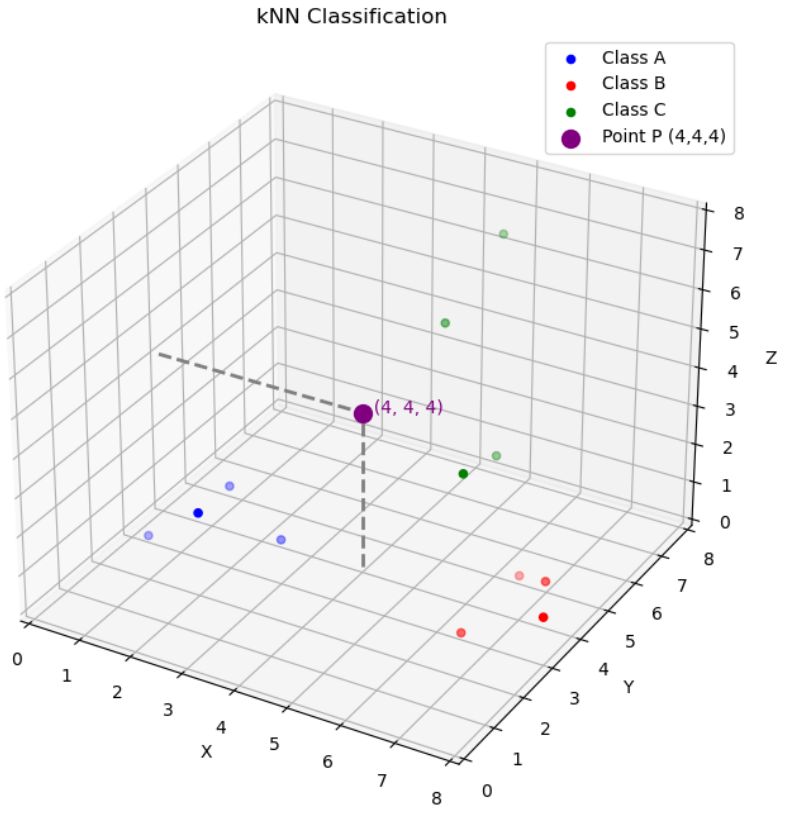

Figure 3: Demonstration of the dataset

outcome. Plot the results of the selected optimal k to verify your choice.

I think K=5 is the most reasonable choice. Since closer distances and larger number of points of the class included in the K neighbour points will lead to a higher tendency to be classified into that class, I design the following weighted distances method. First, we sum up the total length of the nearest 5 points' distances and use this number to divide each of the 5 points' distances. Then, we add each class's distance together to get the weighted distances. When K=5, class A has 3 points included while class C has only 2 points included but each has closer distance. After calculating, we get the weightedA=11.985929681828347 and weightedC= 13.55699922570096, so we decide that Point P should be Class C.

3. **Linear regression in greater depth [12 pts]**
Linear regression is one of the simplest, yet also an extremely effective algorithm. Interestingly, it can be interpreted from 2 different perspectives:

   (a) The first interpretation of this problem is straightforward: we simply want a linear function that can produce the minimal error of a given form, and

for linear regression, we usually pick the mean squared error (MSE). To begin with, we denote the set of parameters (weights) with $\boldsymbol{w}$ and the features of the i-th sample with $\boldsymbol{x}^{(i)}$. So the prediction for the i-th sample is:

$$\hat{y}^{(i)} = \boldsymbol{w}^T \boldsymbol{x}^{(i)}$$

Given the information above, write down the equation of the mean squared error (MSE) to be minimized, the first part of the equation is given to you. **[4 pts]**

$$\text{Minimize}_w f(w) = \frac{1}{n} \sum_{i=1}^{n} (y^{(i)} - \boldsymbol{w}^T \boldsymbol{x}^{(i)})^2$$

The linear regression can also be explained as a geometry/linear algebra problem. From this perspective, $\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\boldsymbol{w}}$ has an alternative interpretation: a linear combination of the columns of matrix $\boldsymbol{X}$. For example, a column vector with three elements can be written as:

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}$$

where v_1, v_2, and v_3 are the elements of the vector.
In this interpretation, the vector of truth values $\boldsymbol{y}$, can be seen as a vector that points out from the hyper-plane. So, minimizing the error becomes minimizing the distance between the vector constructed with the vectors on the plane $\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\boldsymbol{w}}$ and the ground truth vector $\boldsymbol{y}$, as demonstrated in the following figure extracted from Lecture 6. In geometry, this is the same as minimizing the magnitude of the vector $\boldsymbol{y} - \hat{\boldsymbol{y}}$. From pure geometry knowledge, we know that such a vector is shortest when it is perpendicular to the surface, in which case it will be orthogonal to all the vectors in that plane. Using this relationship, we can then derive the analytical expression of $\hat{\boldsymbol{w}}$, as explained in Lecture 6.

$$\boldsymbol{X}^T(\boldsymbol{X}\hat{\boldsymbol{w}} - \boldsymbol{y}) = 0 \tag{1}$$

$$\boldsymbol{X}^T\boldsymbol{X}\hat{\boldsymbol{w}} = \boldsymbol{X}^T\boldsymbol{y} \tag{2}$$

$$\hat{\boldsymbol{w}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{3}$$

1. In your own words, describe how you establish Equation (1) above and interpret this relationship. **[4 pts]**

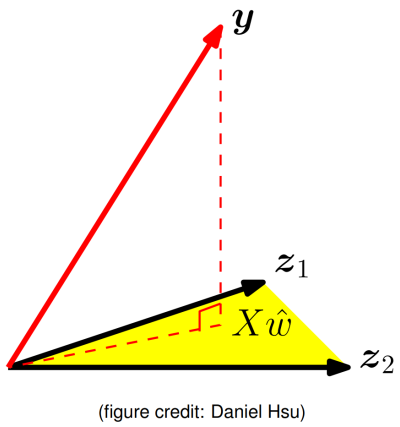

Figure 4: Demonstration of the least squares

Equation (1) is established by considering the geometric interpretation of linear regression. In this perspective, we view y'=Xw as a hyper-plane formed by a linear combination of the columns of matrix X. The vector y represents the true values, and it can be visualized as a vector pointing out from this hyper-plane. Now, let's break down Equation (1): Xw represents the predicted values obtained by multiplying the feature matrix X with the parameter vector w. X'T denotes the transpose of the feature matrix X. Xw-y computes the difference between the predicted values Xw and the true values y, then takes the dot product with the transpose of X. Setting this expression equal to zero implies that the difference vector Xw-y is orthogonal (perpendicular) to the hyper-plane defined by Xw.

Equation (3) takes a very different form from the answer in Part (a). Why do they describe the same problem? Think about the geometric interpretation and explain your understanding. **[4 pts]**

Interpreting this relationship geometrically, it means that the error vector (the difference between predicted and true values) is perpendicular to the hyper-plane of predicted values. This geometrically signifies that the error is minimized when it points directly towards the hyper-plane, making it orthogonal to all vectors in that plane. This condition leads to the shortest distance (magnitude) between the hyper-plane and the true values, resulting in the optimal solution for the parameter vector w.