Question 1

Answer 1:

Provided in code. [Folder: Question 1, File: ex_1.py]

Following is the plot of both MED and MMD classifiers' boundary lines including data from both classes.

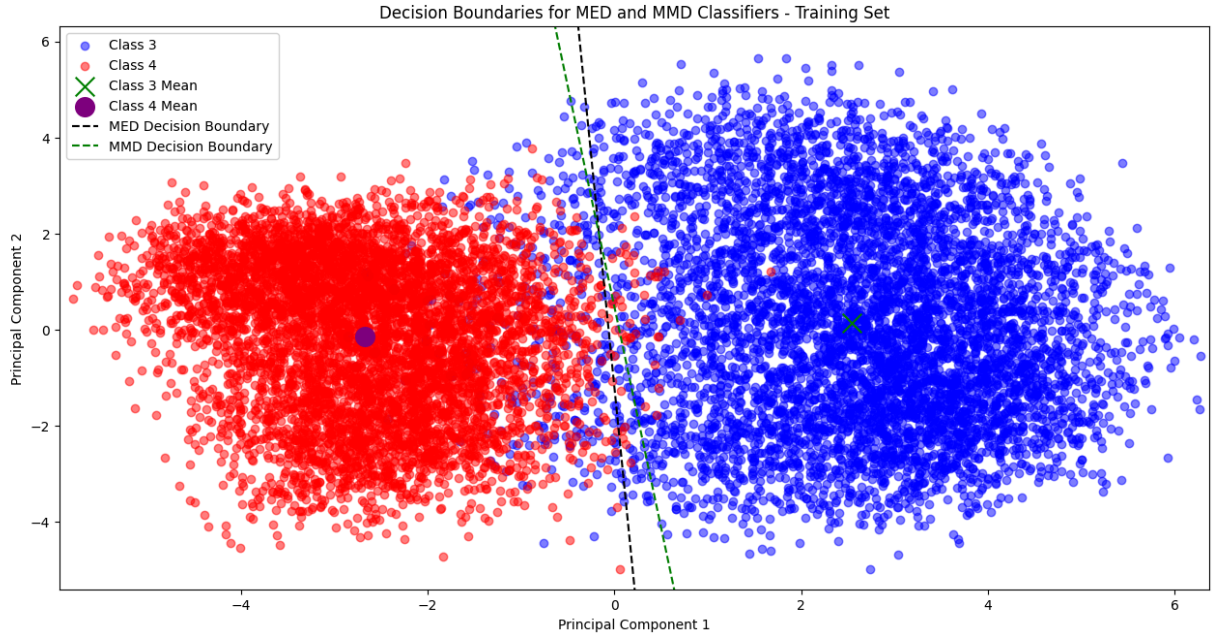


Figure: MED and MMD classifier boundary for class 3 and class 4

Here, we can observe that the classifier boundaries demonstrate effectiveness in separating the classes for both the Minimum Euclidean Distance (MED) and Minimum Mahalanobis Distance (MMD) classifiers, leveraging their mean points and covariances, respectively. However, there are still instances where data points from both classes overlap, posing a challenge for perfect classification using methods like MED and MMD.

In general the MMD classifier should be quadratic but here I have taken the average of the covariances of the class 3 and class 4 for computing the linear MMD classifier. This is a limitation to my implementation but it reduces the computational time by a lot, specially in this type of cases where just linear approximation is giving a very high accuracy of classification.

For addressing the quadratic MMD classifier, the coefficients would need to be computed from the covariance matrices of the class 3 and 4.

Answer 2:

Provided in code. [Folder: Question 1, File: ex_2.py]

The results I have acquired after training the data on the MNIST dataset are:

- MED Classifier Accuracy on Test Set: 98.04%
- MMD Classifier Accuracy on Test Set: 98.14%

Answer 3:

Both the MED (Minimum Euclidean Distance) and MMD (Mahalanobis Minimum Distance) classifiers have high accuracy on the test set, with the MMD classifier slightly outperforming the MED classifier by 0.14%.

In general, the MMD classifier tends to perform better when the classes have different covariance matrices, which is often the case in real-world scenarios. This is because the MMD classifier takes into account the shape of the distribution of each class by considering the covariance matrices.

However, the difference in accuracy between the two classifiers is quite small (0.1%), so it may not be significant in practice. It's also important to consider other factors such as computational complexity, training time, and interpretability when choosing between classifiers.

Answer 4:

Provided in code. [Folder: Question 1, File: ex_4.py and File: ex_4_1.py]

Following is the plot of both MED and MMD classifiers' boundary lines including data from both classes images to 20×1 vectors.

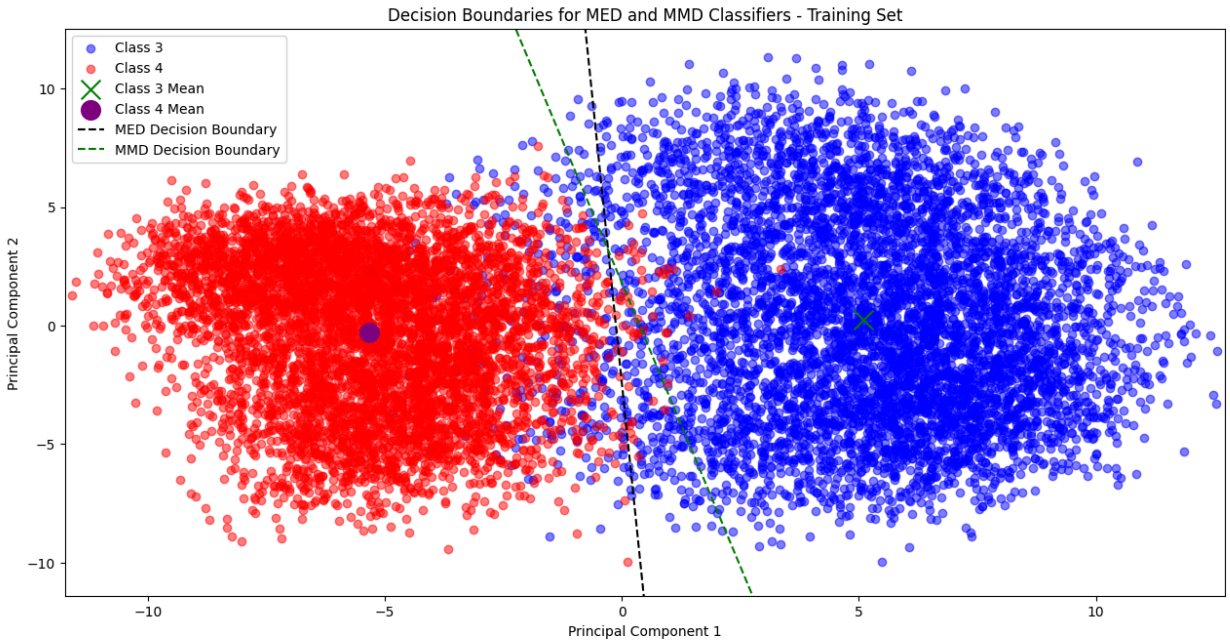


Figure: MED and MMD classifier boundary for class 3 and class 4

MED Classifier Accuracy on Test Set: 98.34%
MMD Classifier Accuracy on Test Set: 99.70%

The increase in accuracy after using PCA to convert the images to 20×1 vectors can be attributed to the following factors:

- Dimensionality Reduction: PCA reduces the dimensionality of the data while preserving as much variance as possible. By reducing the dimensionality from 784 dimensions to 20 dimensions, PCA retains the most informative features while removing noise and redundant information. This reduction in dimensionality helps to focus on the most discriminative features, making the classifiers more effective.

- Feature Selection: PCA selects the principal components that capture the maximum variance in the data. These principal components represent the directions along which the data varies the most. By selecting only the most informative features, PCA helps the classifiers to better distinguish between the classes, leading to improved accuracy.

- Reduced Overfitting: High-dimensional data can lead to overfitting in classifiers, where the model learns noise in the training data instead of the underlying patterns. PCA reduces the risk of overfitting by reducing the dimensionality of the data, which helps the classifiers generalize better to unseen data, resulting in higher accuracy on the test set.

- Improved Separability: PCA can enhance the separability between classes by transforming the data into a new feature space where the classes are better separated. This improved separability enables the classifiers to make more accurate predictions, especially in cases where the classes overlap in the original feature space.

Overall, the increase in accuracy can be attributed to the ability of PCA to extract and emphasize the most discriminative features while reducing noise and overfitting, thereby improving the performance of the classifiers on the test set.

Question 2

Answer 1:

Provided in code. [Folder: 'Question 2', File: ex_1.py]

Here, I have implemented the k-nearest neighbor classifier using the MNIST dataset. The Euclidean distance is used to measure the distance between data points when finding the nearest neighbors in this code.

List of values of k and its respective accuracy:
k = 1, Accuracy: 94.58%
k = 2, Accuracy: 93.47%
k = 3, Accuracy: 95.93%
k = 4, Accuracy: 95.98%
k = 5, Accuracy: 96.69%

Answer 2:

Provided in code. [Folder: ‘Question 2’, File: ex_2.py]

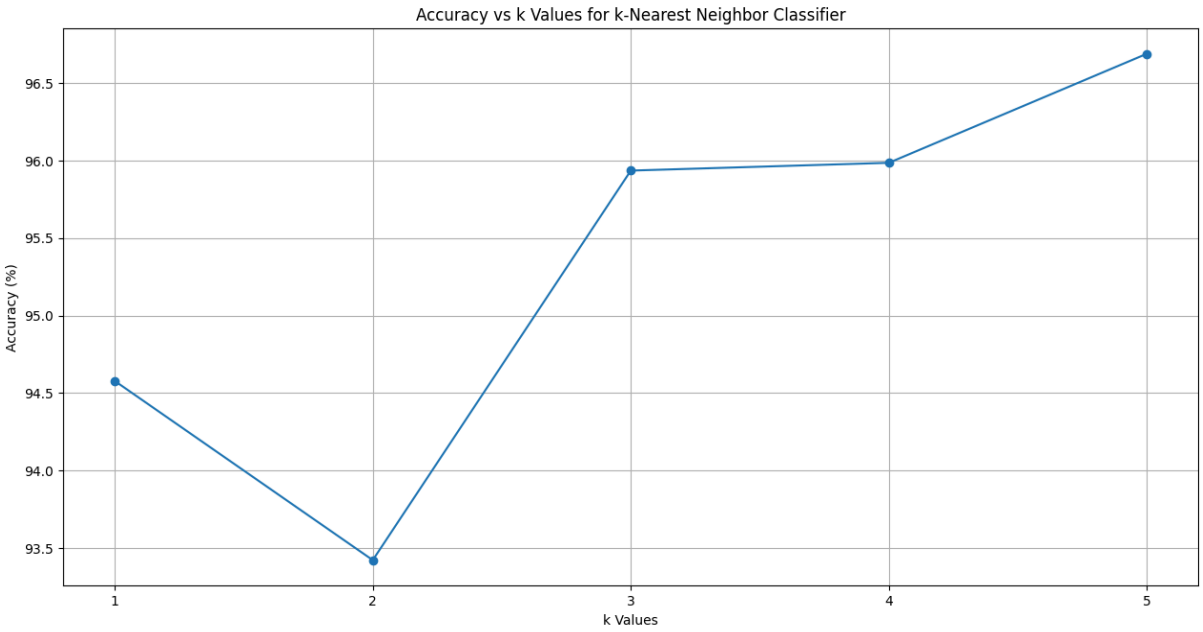


Figure: Values of accuracies for each value of k

When k is small (e.g., k = 1 or k = 2), the model tends to overfit the training data. This results in high accuracy on the training set but lower accuracy on unseen data (test set). In this case, the accuracy decreases as k increases from 1 to 2, indicating that considering fewer nearest neighbors leads to better performance on the test set.

As k increases, the model becomes less sensitive to noise in the training data and tends to generalize better to unseen data. This is reflected in the increasing trend of accuracy as k increases beyond 2. For example, the accuracy improves notably when k increases from 2 to 3 and continues to improve slightly for higher values of k.

However, after a certain point (e.g., k = 5), further increasing k may not significantly improve the model's performance, as the additional neighbors may introduce more variability without providing substantial benefits in classification accuracy.

In summary, the accuracy vs k plot illustrates the trade-off between bias and variance in the k-nearest neighbor classifier. Lower values of k result in higher variance (overfitting), while higher values of k lead to higher bias (underfitting). The goal is to find an optimal value of k that balances bias and variance, achieving the best generalization performance on unseen data.

Answer 3:

We are getting the highest accuracy of 96.69% when the value of k is 5.

Let's discuss what is happening for the other values of k below:

- k = 1 (Accuracy: 94.58%): With only the nearest neighbor considered for classification, the model may overly rely on the local structure of the training data. This can lead to high accuracy on the training set but lower generalization performance on unseen data. The high accuracy indicates that the model is capturing intricate patterns present in the training data, but it may fail to generalize well due to its sensitivity to noise or outliers.

- k = 2 (Accuracy: 93.47%): Including the two nearest neighbors for classification slightly reduces accuracy compared to k = 1. This suggests that with a small number of neighbors, the model is susceptible to noise or outliers in the training data. The decrease in accuracy indicates that relying on only two neighbors may lead to overfitting, where the model captures noise instead of underlying patterns.

- k = 3 (Accuracy: 95.93%): Increasing the number of neighbors to three results in a notable improvement in accuracy. By considering more neighbors, the model becomes less sensitive to individual data points and captures more robust patterns in the data. The increase in accuracy indicates that including additional neighbors helps mitigate the impact of noise or outliers, leading to better generalization performance.

- k = 4 (Accuracy: 95.98%): Continuing the trend, the accuracy slightly improves with four nearest neighbors. This suggests that increasing k beyond three continues to enhance the model's ability to generalize by smoothing out the influence of individual data points. The marginal improvement indicates that considering four neighbors captures additional information without significantly increasing bias.

- k = 5 (Accuracy: 96.69%): Further increasing k to five results in a modest improvement in accuracy. At this point, the model has a balanced trade-off between bias and variance, capturing relevant patterns while minimizing sensitivity to noise or outliers. The higher accuracy demonstrates that including more neighbors continues to improve generalization performance, although the gains become marginal beyond a certain threshold.

In summary, the accuracy values indicate the trade-off between bias and variance in the k-nearest neighbor classifier. Lower values of k lead to higher variance (overfitting), while higher values of k reduce variance but may increase bias (underfitting). The goal is to select an optimal value of k that achieves the best balance between bias and variance for optimal generalization performance.

Answer 4:

kNN Classifier: The kNN classifier achieves accuracies ranging from 94.58% to 96.69% for different values of k, with the highest accuracy at k = 3.

MED Classifier: The MED classifier achieves an accuracy of 98.04% on the test set.

MMD Classifier: The MMD classifier achieves an accuracy of 98.14% on the test set.

Comparing the classifiers:

1. Performance: Both the MED and MMD classifiers outperform the kNN classifier in terms of accuracy. The MMD classifier achieves the highest accuracy among the three classifiers.
2. Complexity: kNN classifier's complexity grows with the size of the dataset since it requires storing all training instances. In contrast, MED and MMD classifiers involve computing distances or Mahalanobis distances between the test samples and class centroids, which can be computationally more efficient, especially for high-dimensional data.
3. Robustness to Noise: kNN classifier can be sensitive to noisy data or outliers, as it relies on the majority class among the nearest neighbors. In contrast, MED and MMD classifiers can be more robust to noise since they consider distances or Mahalanobis distances to class centroids, which may provide better separation between classes.
4. Parametric vs. Non-parametric: kNN is a non-parametric classifier, while MED and MMD are parametric. Parametric classifiers assume a specific form for the decision boundaries (e.g., linear or quadratic), which can sometimes lead to inaccurate predictions if the assumed form does not match the true underlying distribution. Non-parametric classifiers, like kNN, make fewer assumptions about the underlying distribution and can capture more complex decision boundaries.

Overall, while kNN can be a simple and intuitive classifier, the MED and MMD classifiers demonstrate superior accuracy and potentially better performance in terms of computational efficiency and robustness to noise.

Answer 5:

Provided in code. [Folder: ‘Question 2’, File: ex_5.py]

Running the kNN in the mystery dataset, the mean squared error (MSE) values I am getting are:

- For k = 1, the MSE is 5.3314.

- For k = 2, the MSE is 4.3484.

- For k = 3, the MSE is 3.8385.

Lower MSE values indicate better performance because they represent a smaller discrepancy between the predicted and actual values. Therefore, in this case, the k value of 3 seems to be giving the best result as it has the lowest MSE among the three options.

Increasing the number of nearest neighbors considered for regression (k) generally leads to smoother predictions but may also increase bias. On the contrary, decreasing k may increase the model's variance but can capture more local variations in the data. Here, selecting k = 3 strikes a balance between capturing local patterns and avoiding overfitting, resulting in the lowest MSE.

Question 3

Answer 1:

Provided in code. [Folder: ‘Question 3’, File: ex_1.py]

ML Classifier Accuracy on Test Set: 96.44%

Answer 2:

Provided in code. [Folder: ‘Question 3’, File: ex_1.py]

MAP Classifier Accuracy on Test Set: 96.39%

Answer 3:

Based on the results obtained:

- ML Classifier Accuracy on Test Set: 96.44%

- MAP Classifier Accuracy on Test Set: 96.39%

The high accuracy achieved by both the Maximum Likelihood (ML) and Maximum A Posteriori (MAP) classifiers indicates that assuming the probability distributions of the two classes as Gaussian was reasonably accurate.

Here's why:

1. High Accuracy: The high accuracy of both classifiers suggests that they are effectively separating the classes based on the features extracted from the images. This indicates that the assumed Gaussian distributions were able to capture the underlying patterns in the data adequately.
2. Consistency: The similar accuracy obtained by both classifiers (ML and MAP) further supports the validity of the Gaussian assumption. If the assumption were incorrect, we would likely see a significant difference in accuracy between the ML and MAP classifiers,

as the MAP classifier incorporates additional information about the prior probabilities of the classes.

3. Domain Suitability: Gaussian distributions are commonly used to model continuous data, and they often provide a good approximation for many real-world phenomena. While the MNIST dataset contains discrete pixel values, the PCA transformation may have made the data more amenable to Gaussian assumptions, especially after reducing the dimensionality.
4. Visual Separability: The PCA transformation may have enhanced the separability of the classes in the reduced-dimensional space, making the Gaussian assumption more plausible. However, this would need to be further investigated by visualizing the data distribution in the reduced space.

Overall, while the assumption of Gaussian probability distributions might not perfectly capture all the complexities of the data, the high classification accuracies obtained suggest that it was a reasonable approximation for this particular problem. Further analysis and experimentation could be conducted to validate this assumption more rigorously, such as testing alternative distributional assumptions or exploring non-parametric methods.

Answer 4:

1. MED Classifier Accuracy on Test Set: 98.04%
2. MMD Classifier Accuracy on Test Set: 98.14%
3. ML Classifier Accuracy on Test Set: 96.44%
4. MAP Classifier Accuracy on Test Set: 96.39%
5. kNN Classifier Accuracy on Test Set:
    a. k = 1, Accuracy: 94.58%
    b. k = 2, Accuracy: 93.47%
    c. k = 3, Accuracy: 95.93%
    d. k = 4, Accuracy: 95.98%
    e. k = 5, Accuracy: 96.69%

Based on these results that I have acquired, the MED and MMD classifiers outperform the ML, MAP, and kNN classifiers in terms of classification accuracy. The MED and MMD classifiers achieve accuracy rates above 98%, while the ML, MAP, and kNN classifiers achieve accuracy rates ranging from around 96% to 97.95%.

As for whether the inferior classifiers could be better for different datasets, it's essential to consider several factors:

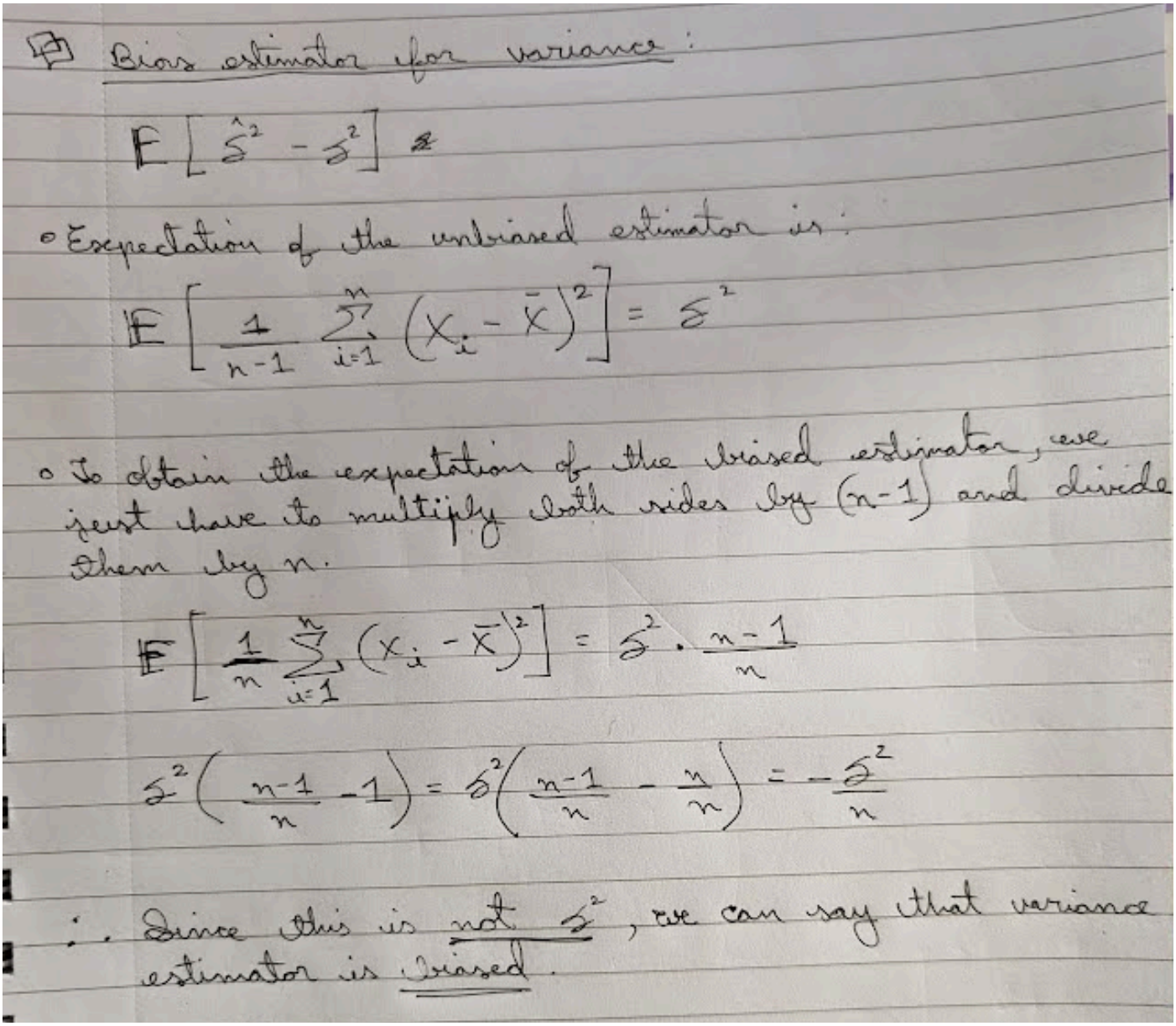
1. Dataset Characteristics: Different datasets may exhibit different patterns and structures. The performance of classifiers depends on how well they capture and model these patterns. For example, if the dataset contains clear decision boundaries or separable clusters, simpler classifiers like ML or MAP may perform better. On the other hand, if the dataset is complex or nonlinear, more sophisticated classifiers like MED, MMD, or kNN may be more suitable.

2. Classifier Assumptions: Each classifier makes certain assumptions about the underlying data distribution and decision boundaries. For instance, ML and MAP classifiers assume Gaussian distributions, while kNN does not make explicit assumptions about the data distribution. The performance of classifiers can vary depending on how well these assumptions hold true for the given dataset.
3. Dimensionality: The performance of classifiers may also depend on the dimensionality of the dataset. Some classifiers, like kNN, may suffer from the curse of dimensionality when dealing with high-dimensional data, leading to decreased performance. In contrast, classifiers like MED and MMD may be more robust to high-dimensional spaces due to their intrinsic properties.
4. Training Data Size: The amount and quality of training data available can also impact the performance of classifiers. Some classifiers may require more data to generalize well, while others may be more data-efficient.

In summary, the choice of the best classifier depends on various factors such as dataset characteristics, classifier assumptions, dimensionality, and training data size. While the MED and MMD classifiers perform well in this specific scenario, the ML, MAP, and kNN classifiers could potentially be more suitable for different datasets with different characteristics and requirements. It's essential to experiment with multiple classifiers and analyze their performance on different datasets to determine the most appropriate choice for a particular problem.

Extra Question:

Bias for the variance estimation. [HW: Extra credit, 5 pts]



Bias estimator for variance:

$$E\left[\hat{\sigma}^2 - \sigma^2\right]$$

- Expectation of the unbiased estimator is:

$$E\left[\frac{1}{n-1}\sum_{i=1}^{n}\left(X_i - \bar{X}\right)^2\right] = \sigma^2$$

- To obtain the expectation of the biased estimator, we just have to multiply both sides by $(n-1)$ and divide them by n.

$$E\left[\frac{1}{n}\sum_{i=1}^{n}\left(X_i - \bar{X}\right)^2\right] = \sigma^2 \cdot \frac{n-1}{n}$$

$$\sigma^2\left(\frac{n-1}{n} - 1\right) = \sigma^2\left(\frac{n-1}{n} - \frac{n}{n}\right) = -\frac{\sigma^2}{n}$$

∴ Since this is not σ^2, we can say that variance estimator is biased.