Statistical Pattern Recognition

Homework 2

Bayesian Classification & Dimensionality Reduction

Assignment Date: 1404/10/16 **Submission Deadline: 1404/11/6**

Objectives and Precautions:

In this homework, you will be introduced to two fundamental topics in pattern recognition: Binary classification using Bayesian classifiers. Dimensionality reduction using the PCA and Fisher Linear Discriminant algorithms. The primary objective of this homework is to develop a solid understanding of the theoretical foundations of these algorithms, implement them **from scratch**, and perform a statistical and conceptual analysis of the results. **You are restricted** to using only the following Python libraries in your implementation:

- Numpy, Pandas, Scipy, matplotlib
- From sklearn, you are allowed to use **only** the train_test_split, StratifiedKFold (if needed), StandardScaler (optional).

Additional Notes:

- you have **three weeks** to complete this assignment.
- The assignment instructions and datasets will be provided through your Quera class.
- You are allowed to use only **the Python programming** language.
- Save your results and answers in any format of your choice as a report, and submit it together with your code.
- Package your code files and the report into **a single ZIP file** and upload it to Quera. The ZIP file should be named using the format "**YourNames.zip**".
- If you have any questions, feel free to ask them in the **Telegram channel**.
- Ensure that all submitted work is original and written entirely by you; copying from others is strictly prohibited.
- You should be prepared to clearly explain every part of your code to the Teaching Assistants, especially if external tools (such as ChatGPT) were used for guidance.

Late Submission Policy

- Submissions delayed by one day will **receive 70%** of the total score.
- Submissions delayed by two days will **receive 50%** of the total score.
- Submissions submitted more than two days late **will not be graded.**

Good luck!

1. Binary Classification using Bayesian Classifiers

In this part, you will implement and analyze **Bayesian classifiers** for a **binary classification problem**. Three probabilistic models are considered: **Gaussian Naïve Bayes (GNB), Linear Discriminant Analysis (LDA / Gaussian LDA), and Quadratic Discriminant Analysis (QDA).** All three models are derived from Bayesian decision theory and belong to the class of **Bayesian generative classifiers**, as they explicitly model the class-conditional probability distributions of the data.

Although these models share a common Bayesian foundation, they differ in their underlying assumptions about feature independence and the structure of the class-conditional covariance matrices. Gaussian Naïve Bayes assumes conditional independence among features, which leads to a diagonal covariance structure. LDA assumes that samples from each class are drawn from a multivariate Gaussian distribution with a **shared covariance matrix** across all classes, resulting in linear decision boundaries. In contrast, QDA relaxes this assumption by allowing each class to have its own **class-specific covariance matrix**, producing more flexible, quadratic decision boundaries. By leveraging Bayes' rule, all three classifiers estimate posterior class probabilities and construct decision boundaries in a principled probabilistic manner.

1.1. Understanding the Data: Breast Cancer Wisconsin (Diagnostic)

The **Breast Cancer Wisconsin (Diagnostic)** dataset consists of numerical, continuous features extracted from microscopic images of breast mass tissue, used to classify tumors as **benign or malignant**. It contains 569 samples with 30 features and a binary class label.

1. Load the dataset and report the number of samples, features, and classes.
2. Compute the proportion of samples in each class and discuss any potential class imbalance and its potential impact on Bayesian classifiers.
3. Perform exploratory visual analysis by plotting **class-wise histograms** for a subset of features and visualizing the **feature correlation matrix** as a **heatmap**.
4. Identify the **three most highly correlated feature pairs**. Briefly explain why feature correlation plays an important role when comparing LDA and QDA with Naïve Bayes.
5. Using the plotted histograms, examine whether the feature distributions are approximately Gaussian, and explain why Gaussian-based Bayesian classifiers are still widely used in practice even when this assumption is only approximately satisfied **and comment on how deviations from Gaussian assumption may affect LDA and QDA differently.**

1.2. Bayesian Decision Theory and Distance Measures

1. Write down Bayes' rule and explain the role of each term: Prior probability, Likelihood, Posterior probability.
2. Explain why, for classification, maximizing $P(y \mid x)$is sufficient and why the normalization term can be ignored.

3. Define the Mahalanobis distance and explain intuitively how it differs from the Euclidean distance. Discuss why this distance naturally appears in LDA and QDA, but not explicitly in Gaussian Naïve Bayes.

1.3. Model Parameters:

1. Identify the parameters of Gaussian Naïve Bayes, LDA, and QDA. Explain the role and interpretation of each parameter within each model.
2. Compare the three classifiers in terms of number of parameters, modeling assumptions, and expected model flexibility.
3. Based on the dimensionality of the dataset, discuss which classifier is expected to require the largest amount of training data and why?

1.4. Model Implementation

1. Separate the dataset into feature matrix (X) and label vector (y), and then split it into training and test sets using an 80/20 train–test split.
2. Using the training data, estimate the following parameters:
    - Gaussian Naïve Bayes:
        - Class prior probabilities.
        - Feature-wise means and variances for each class.
    - LDA:
        - Class prior probabilities
        - Class mean vectors
        - A shared covariance matrix
    - QDA:
        - Class prior probabilities
        - Class mean vectors
        - Class-specific covariance matrices
3. Explain how the covariance structure differs between Naïve Bayes, LDA, and QDA, and discuss how these assumptions affect the shape of the decision boundary.
4. Implement the classifiers from scratch using the parameters estimated in the previous steps.
5. Use all three classifiers to classify both the training and test sets. For each model, apply the appropriate Bayesian decision rule (the Mahalanobis distance in the case of LDA and QDA). Discuss the numerical stability of QDA given the dimensionality of the dataset and the number of training samples per class. Explain whether any observed performance degradation may be attributed to covariance estimation issues rather than model inadequacy.
6. Using the classification outcomes from this section, analyze the effect of the conditional independence assumption and class prior probabilities on Gaussian Naïve Bayes, and evaluate its suitability as a baseline classifier for this dataset.

Note: The following analyses focus exclusively on **LDA and QDA**, as Gaussian Naïve Bayes assumes a diagonal covariance structure and does not involve full covariance matrix estimation.

1.5. Covariance Matrix Analysis

The covariance matrix plays a central role in Gaussian-based Bayesian classifiers such as LDA and QDA, as it captures feature variances and correlations that directly influence the shape of the decision boundaries.

1. Using **only the training data**, compute the empirical **covariance matrix of the feature** matrix X. What is the rank of this matrix? Based on the computed rank, is the covariance matrix singular or full-rank? Explain why a full-rank covariance matrix may still be numerically unstable in practice.
2. Compute the eigenvalues of the covariance matrix. Report the smallest and largest eigenvalues, and comment on the spread of the eigenvalue spectrum.
3. Compute the condition number of the covariance matrix and explain what it indicates about numerical stability. **Relate your findings to the practical reliability of matrix inversion in LDA and QDA.**
4. Based on your results from the previous questions, **clearly distinguish** between a **singular covariance matrix** and an **ill-conditioned covariance matrix**, and explain why a matrix can be full-rank yet numerically unstable.
5. Using the empirical evidence from this dataset, explain how a covariance matrix can be **full-rank** yet still be **ill-conditioned**, and discuss the implications of this behavior for Gaussian-based classifiers such as LDA and QDA.

1.6. Analysis of Variance and Feature Contribution

1. Compute the **variance of each feature** in the **training** dataset and identify features with **relatively low variance**.
2. Does the presence of low-variance features necessarily imply that the covariance matrix is singular? Justify your answer. Additionally, analyze how correlations between features may affect the numerical properties of the covariance matrix, even when all feature variances are non-zero.

1.7. Variance-Based Feature Elimination and Stability Assessment

Using the feature variance analysis performed in **Section 1.6**, investigate the effect of variance-based feature elimination on the numerical stability of the covariance matrix.

1. Based on the variance values obtained in Section 1.6, identify the features whose variance falls below a small percentile of the variance distribution (e.g., percentile-based thresholding).
2. Remove the identified low-variance features and construct the reduced feature set.
3. Recompute the empirical covariance matrix of the reduced feature set using the training data.

4. Compare the covariance matrices before and after feature elimination by analyzing the rank, the eigenvalue spectrum, and the condition number.
5. Based on your analysis, discuss whether variance-based feature elimination is sufficient to resolve covariance matrix singularity or significantly improve numerical conditioning, and explain its limitations.

1.8. Re-estimation and Classification with Updated Features

Using the reduced feature set obtained after variance-based feature elimination in Section 1.7, repeat the model training and classification process as follows:

1. Re-estimate the model parameters for LDA and QDA using the training data and the updated feature set. How do the model parameters change after feature elimination?
2. Classify both the training and test samples using the re-estimated LDA and QDA models and explain how does the updated feature set affect classification behavior?
3. Store the classification results and performance metrics for later comparison with those obtained using the original feature set.

1.9. Performance Evaluation and Error Analysis

Using the classification results obtained for **LDA** and **QDA**, evaluate and analyze the performance of both models as follows:

1. Report the following performance metrics on both the training and test sets: **Accuracy, Precision, Recall, F1-score, Confusion matrix**.
2. Based on the confusion matrices: report the total **number of misclassified samples** for each classifier, separate the errors into **False Positives (FP) and False Negatives (FN).**
3. Using the reported metrics and error types, determine which classifier achieves better generalization performance **If QDA shows inferior test performance, explain why this behavior may be expected given its parameter complexity and the dataset size.**

1.10. Model Parameters and Complexity Comparison

1. Report the learned model parameters for LDA and QDA, and compute the total number of free parameters required by each model.
2. Discuss how differences in parameter complexity relate to model flexibility, overfitting, and generalization performance observed in previous sections?

2. Dimensionality Reduction

Dimensionality reduction aims to transform high-dimensional data into a lower-dimensional space while preserving the most informative structure of the data, leading to reduced computational cost, improved numerical stability, and better generalization. Principal Component Analysis (PCA) and Fisher Linear Discriminant Analysis (Fisher LDA) are two classical linear dimensionality reduction techniques with different objectives. Both methods rely on linear transformations, eigenvalue decomposition, and covariance-based representations, and both are commonly used as preprocessing steps before applying classification algorithms.

The goal of this section is to study and implement linear dimensionality reduction techniques, specifically PCA and Fisher LDA, from scratch for face recognition. The focus is on understanding the geometric, statistical, and classification properties of these methods when applied to high-dimensional image data.

2.1. Dataset: The ORL face database

The ORL Database is a set of face images that includes ten different images of each of 40 distinct subjects. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position.

The files are in PGM format and the size of each image is 92x112 pixels, with 256 grey levels per pixel. The images are organized in 40 directories (one for each subject), which have names of the form sX, where X indicates the subject number (between 1 and 40). In each of these directories, there are ten different images of that subject, which have names of the form Y.pgm, where Y is the image number for that subject (between 1 and 10).

2.2. Principal Component Analysis (PCA)

PCA is an unsupervised method that projects data onto orthogonal directions of maximum variance, making it well suited for data compression and noise reduction, but it does not use class label information.

2.2.1. Data Loading & Preprocessing

1. Load ORL images and organize into a DataFrame.
2. Flatten images to 10304×1 vectors.
3. Compute the mean face for the dataset and normalize all images.
4. Randomly select one image per individual to visualize original vs mean-subtracted.
5. Apply **additional normalization options** (e.g., scale to unit variance) and compare effects.

2.2.2. PCA Implementation

1. Compute the covariance matrix of the zero-mean images, or equivalently, use an algebraically equivalent formulation of PCA that avoids explicit construction of the full covariance matrix when dimensionality is high.
2. Perform eigen decomposition of the covariance matrix to obtain eigenvalues and eigenvectors.
3. Sort eigenvectors by their eigenvalues in descending order and plot the eigenvalues to visualize the variance distribution.
4. Select the top 50 eigenvectors as principal components (eigenfaces and reshape and visualize these top 50 eigenfaces.

2.2.3. Projection & Reconstruction

1. Project all images onto a lower-dimensional subspace using the top eigenfaces and reconstruct selected images from the projection by adding back the mean face.
2. Visualize original vs reconstructed images for a few individuals.
3. Determine the number of principal components needed to retain 90% of the cumulative variance.
4. Project the images using the number of PCs determined.

2.2.4. Face Recognition/ Classification

1. Split dataset into training and test using **leave-one-out per individual**.
2. Project training and testing images into PCA space.
3. Implement and compare multiple classifiers in PCA space: QDA and LDA on PCA-projected features.
4. Evaluate and compare accuracy for all classifiers.
5. Analyze **how number of principal components affects classification accuracy** for each classifier. Plot results.
6. Discuss the trade-off between dimensionality reduction (number of PCs) and recognition accuracy.
7. Compare PCA-only vs PCA+LDA recognition performance.
8. Report and visualize **confusion matrices** for best-performing models.

2.3. Fisher Linear Discriminant Analysis

Fisher LDA is a supervised method that exploits class labels to find projections that maximize class separability by increasing between-class variance and minimizing within-class variance, and is therefore more directly tailored for classification tasks.

2.3.1. Data Preprocessing & Whitening

1. Apply any necessary preprocessing to the images and prepare your data matrix.
2. Consider whitening the data and analyze its effects on **feature correlations**.

3. Critically evaluate: is whitening essential in LDA, or can it be omitted? Justify your reasoning mathematically.

2.3.2. Between-Class Scatter (SB) and Within-Class Scatter (SW)

1. Compute both S_B and S_W for the ORL dataset.
2. Explain, with reference to the dataset structure, the difference between within-class and between-class scatter.
3. Discuss how the size and distribution of classes affect these matrices.

2.3.3. Eigenvalue Problem for LDA

1. Solve the generalized eigenvalue problem for $S_W^{-1}S_B$.
2. Plot the eigenvalues in descending order.
3. Analyze why descending order is chosen for selecting linear discriminants.
4. Provide an interpretation of the magnitude of eigenvalues in terms of class separability.

2.3.4. Evaluating Separability

1. Define a suitable separability metric and evaluate how it changes as you include more components.
2. Identify potential pitfalls when selecting too few or too many components.
3. Decide on an optimal number of discriminant components.
4. Explain how the number of classes in ORL constrains the maximum possible number of discriminants.
5. Reflect on how class overlaps or intra-class variability affect your selection.

2.3.5. Projection into Lower-Dimensional Subspace

1. Transform the original data into the selected LDA subspace.
2. Provide reasoning for why you selected this number of dimensions for the projection.
3. Discuss any trade-offs between dimensionality reduction and information retention.

2.3.6. Classification

1. Split the LDA-projected data into **train and test sets**, keeping a realistic proportion for testing.
2. Implement a Bayesian classifier (or any other probabilistic classifier of your choice) on the projected data.
3. Investigate how varying the number of LDA components affects classifier accuracy.
4. Plot the classification performance versus the number of components.
5. Critically analyze: which components contribute most to separability? Are there diminishing returns as more components are added?
6. Discuss any discrepancies between theoretical expectations (eigenvalues, scatter matrices) and practical classification results.