Project 1

Dimension Reduction in Classification

Maaut - Mathematics for Machine Learning

MECD & MMAC, 2nd Semester, 2024/2025

TÉCNICO LISBOA

Group 9

Members:

- Quian Blanco, Francisco (ist1115588)
- Sanguinetti, Maria Agustina (ist1115524)

Faculty:

- Maria do Rosário De Oliveira Silva
- Ana Moura Santos
- Pedro Alexandre Simões dos Santos

1. Introduction

Image classification is a fundamental task in computer vision, with applications ranging from medical diagnostics to autonomous driving. However, real-world image data often comes with challenges (high dimensionality, redundant features, and noise) that can make classification difficult. In this project, we tackle these challenges using dimensionality reduction techniques to simplify the data while preserving its essential structure.

We will be working with the Image Segmentation dataset, which contains 2,310 image regions from seven categories (like grass, sky, and brickface), each described by 19 numeric features. Our goal is to:

- Identify the most informative features using selection methods.
- Reduce dimensionality using Principal Component Analysis (PCA).

We then will be evaluating how these approaches affect classification performance using k-Nearest Neighbors (kNN), comparing metrics like accuracy and F1-score. The key question we explore: Is it better to carefully select individual features or let PCA create new composite features?

2. Preliminary Data Analysis

The dataset consists of image segmentation data described by 19 high-level numeric attributes, with 2310 instances evenly distributed across 7 classes: GRASS, PATH, WINDOW, CEMENT, FOLIAGE, SKY, and BRICKFACE. Each instance represents a 3x3 pixel region extracted from outdoor images, manually segmented to ensure accurate classification. The features include geometric properties (e.g., centroid coordinates, pixel count), texture metrics (e.g., edge detectors like vedge-mean and hedge-mean), and color-based attributes (e.g., RGB means, excess color values).

A notable characteristic is the balanced class distribution in the original dataset—300 test instances and 30 training instances per class—which avoids immediate concerns about class imbalance. However, since the data comes from just seven outdoor images, there's a risk of bias in real-world generalization. For example, if certain classes (like SKY) dominate typical outdoor scenes, the model might struggle with underrepresented classes (GRASS, WINDOW) in broader applications.

Potential issues emerged during the analysis:

- Feature Collinearity: Many attributes are inherently related, such as paired mean and sd (standard deviation) measures (e.g., vedge-mean/vedge-sd) or RGB values (rawred-mean, rawgreen-mean). While some forward feature selection methods (like CMIM or JMI) account for redundancy, others (e.g., MIM) might overlook these dependencies, affecting feature selection.

- Manual Segmentation Bias: Human-labeled data could introduce subjective patterns, limiting the model's ability to generalize to new images.

- Numeric Scale Disparities: Features like region-pixel-count (fixed integer) and color metrics (continuous floats) vary widely in range, suggesting normalization may be necessary before applying distance-based algorithms (e.g., kNN).

- Despite the dataset's cleanliness (no missing values, all features continuous), its small size raises challenges for training robust models, emphasizing the need for careful dimensionality reduction and validation.

3. Methodology

3.1 Data Splitting

By choosing a fixed seed, *123*, the dataset was randomly divided into 70% for training and 30% for testing. The training dataset was used to run the Forward Feature Selection methods that will be discussed in the next section.

3.2 Feature Selection

A total of eight Forward Feature Selection methods were applied to rank the relevance of the features: MIM, MIFS, mRMR, maxMIFS, CIFE, CMIM, JMI, and DMIM. The purpose of including this variety was to explore and highlight the key differences among them, particularly in how they compute and handle redundancy, and how much weight they assign to it. Based on their treatment of redundancy, these methods can be grouped into three categories:

- 1st group: focuses exclusively on the maximum relevance between each candidate feature and the class, computed as $MI(C, X_i)$, and completely ignores redundancy. MIM is the only method in this group.
- 2nd group: builds on this by introducing a penalty for inter-feature redundancy using $MI(X_i, S)$. This group includes MIFS, mRMR, and maxMIFS.

- 3rd group: further extends Group 2 by considering conditional dependencies among features given the class, using $MI(X_i, S|C)$. This group comprises CIFE, CMIM, JMI, and DMIM.

To calculate the mutual information, the following formulas were used:

- $MI(C, X_i) = h(X_i) + H(C) - h(C, X_i)$
- $MI(X_i, S) = h(X_i) + h(S) - h(S, X_i)$
- $MI(X_i, S|C) = h(X_i|C) + h(S|C) - h(X_i, S|C) = h(X_i, C) + h(S, C) - H(C) - h(X_i, S, C)$

Given X and S were discretized to calculate the entropy, some correction factors had to be applied.

- $\Delta = \frac{max-min}{k}$
- $H(X^{\Delta}) + ln(\Delta) \rightarrow h(X),\ k = ceiling(\sqrt{n})$
- $H(X^{\Delta}, C) + ln(\Delta) \rightarrow h(X, C),\ k = ceiling(\sqrt{n})$
- $H(X^{\Delta_1}, S^{\Delta_2}) + ln(\Delta_1) + ln(\Delta_2) \rightarrow h(X, S), k = ceiling(n^{1/4})$
- $H(X^{\Delta_1}, S^{\Delta_2}|C) + ln(\Delta_1) + ln(\Delta_2) \rightarrow h(X, S|C), k = ceiling(n^{1/4}/c)$

However, when k took these different values for each case, the MI was negative. As this is incorrect, it was decided to use exclusively $k = ceiling(\sqrt{n})$, no matter the case, and this solved the issue.

3.3 Classification

The k-Nearest Neighbour (kNN) classifier with k=3 neighbors was employed as the evaluation model. It was trained and tested using three distinct feature subsets derived from each Feature Selection (FS) method:

- The top ⅓ highest-ranked features
- The top ⅔ highest-ranked features
- The complete feature set

This classifier served as the benchmark to assess the effectiveness of each FS method across these subsets, quantifying performance through metrics such as accuracy, Macro Recall, Macro Precision and Macro F1.

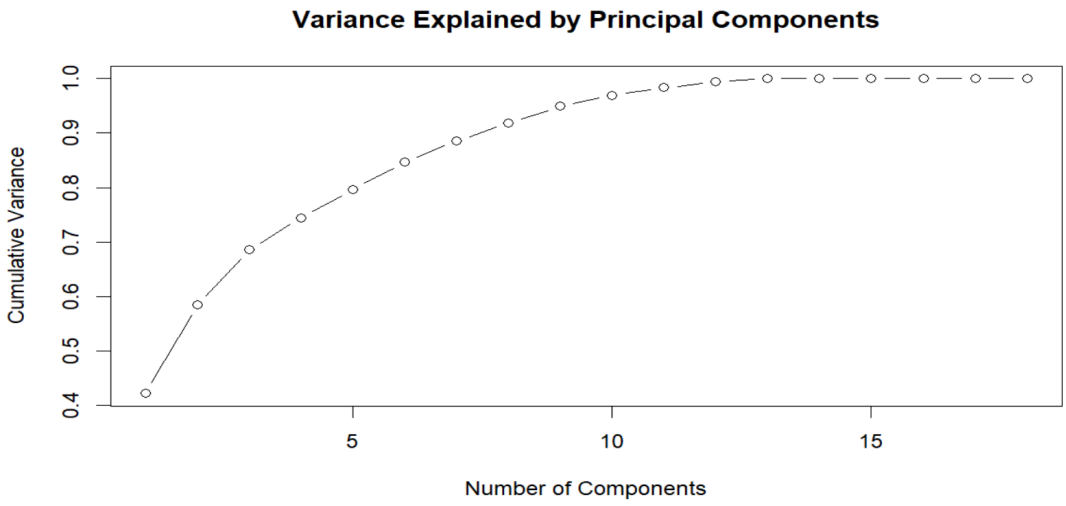
3.4 Principal Component Analysis

PCA is a method that simplifies complex data by combining features into new ones called principal components (PC). These new components are designed to capture the most important patterns in the data while reducing noise and redundancy. These PCs are linear combinations of the original features, ordered by how much variance they capture in the data. The first PC

accounts for the largest possible variance, the second PC (orthogonal to the first) captures the next largest, and so on.

Before running PCA, we ensured the data was suitable by removing constant columns, and the remaining features were standardized (centered and scaled) to prevent variables with larger scales from dominating the PC.

The PCA model processed the training data and, afterwards, we analyzed the variance to determine the optimal number of components. A common heuristic is to retain components that capture at least 95% of the total variance. In our case (see graph 1.1) this threshold was met after selecting 10 principal components.



Graph 1.1: Cumulative Variance for amount of PC

After transforming both the training and test sets using these components, we applied the k-Nearest Neighbors (kNN) classifier (with k=3) to evaluate performance. The results were compared using the same metrics as the forward feature selection methods (Accuracy, Macro Precision, Recall, and F1-score), allowing us to compare explicitly PCA and these methods.

4. Results and Discussion[1]

4.1 Feature Selection Outcomes

Firstly, it is interesting to highlight the different outcomes of the evaluated methods in predictive performance when using different numbers of features (1/3, 2/3, or all). This can be seen in the following table

[1] All results displayed where obtained using 'set.seed(123)'

Method	No. Features	Accuracy	Macro Precision	Macro Recall	Macro F1	Key Observations
MIM	6 (1/3)	87.59%	87.49%	87.67%	87.43%	Weakest among methods
MIM	13 (2/3)	95.38%	95.71%	95.63%	95.55%	Significant improvement with more features
MIM	All (19)	94.66%	94.86%	94.87%	94.85%	Consistent high performance
MIFS	6 (1/3)	87.88%	87.80%	87.95%	87.70%	Weakest among methods
MIFS	13 (2/3)	95.24%	95.52%	95.48%	95.40%	Excellent boost with 13 features
MIFS	All (19)	94.52%	94.77%	94.73%	94.73%	No gain beyond 2/3
MRMR	6 (1/3)	96.10%	96.32%	96.28%	96.26%	Best performance overall
MRMR	13 (2/3)	94.52%	94.65%	94.68%	94.65%	Slight drop but still strong
MRMR	All (19)	94.23%	94.45%	94.43%	94.42%	Stable performance
MaxMIFS	6 (1/3)	88.31%	88.20%	88.39%	88.13%	Better than MIM/MIFS at 1/3
MaxMIFS	13 (2/3)	95.24%	95.56%	95.48%	95.40%	Ties with MIFS at 13
MaxMIFS	All (19)	94.37%	94.60%	94.54%	94.55%	Plateau after 2/3
CIFE	6 (1/3)	92.21%	92.23%	92.40%	92.10%	The worst of 3rd group
CIFE	13 (2/3)	94.52%	94.83%	94.64%	94.71%	Good consistency
CIFE	All (19)	94.08%	94.31%	94.28%	94.26%	Slight drop
JMI	6 (1/3)	93.22%	93.65%	93.50%	93.45%	Solid choice with fewer features
JMI	13 (2/3)	95.38%	95.73%	95.62%	95.54%	Performs well with more features
JMI	All (19)	94.37%	94.56%	94.56%	94.54%	Small drop after 2/3
CMIM	6 (1/3)	95.53%	95.85%	95.74%	95.72%	Very strong at 1/3
CMIM	13 (2/3)	94.66%	94.89%	94.82%	94.84%	Tiny drop
CMIM	All (19)	94.52%	94.68%	94.71%	94.68%	Stable performance

DMIM	6 (1/3)	88.60%	88.44%	88.63%	88.46%	Similar to MaxMIFS at 1/3
DMIM	13 (2/3)	95.38%	95.68%	95.61%	95.55%	Strong with more features
DMIM	All (19)	94.08%	94.37%	94.28%	94.30%	Doesn't gain much from all features

Table 1.2: Accuracy and observation for every method and number of features chosen

4.2 PCA Results

Looking at the results of table 1.3 (found on Index), there are several key insights revealed about the dataset. First and foremost, The first principal component (PC1) alone captures 42.3% of the total variance in the data, while the first five components combined explain 79.7% of the variance. We observe diminishing returns after PC10, which collectively account for 96.9% of the variance. The remaining components contribute negligible information (each <2%). This suggests that most meaningful patterns in the original 19 features can be effectively represented by just 10 composite components, and is why we stopped at just PC10, as it had over 95% of the variance.

When using the first 10 components for kNN classification:

- Accuracy: 0.9422799
- Macro Precision: 0.9433686
- Macro Recall: 0.9442617
- Macro F1: 0.9425571

This indicates a very high accuracy (94.2%), so the modal correctly classified nearly all test instances as well as a balanced performance, as the other metrics show the model does not show substantial class-specific biases.

The most important takeaway is that PCA successfully reduced our feature space by nearly 50% (19 → 10) while preserving most discriminative information. In addition, the strong performance suggests PCA may have filtered out less meaningful variations present in the original features, which would have instead burden the precision.

5. Conclusion

Regarding feature selections, as it can be seen in the results of Table 1.2, increasing the amount of features does not guarantee better performance. And even increasing or decreasing this amount might as well improve or not the accuracy. Nonetheless, it should be highlighted that when passing from ⅔ to including all features, the accuracy decreases given it considers negative scores.

Moreover, although complex and robust methods such as JMI and CMIM aim to address feature redundancy and class relevance, they may underperform simpler approaches like MRMR when only a small number of features are selected. However, as the number of features increases, these robust methods tend to outperform the simpler ones. Furthermore, being part of the same methodological group does not guarantee similar performance — as illustrated by MRMR and MIFS: while MRMR achieves the highest accuracy with one-third of the features, MIFS performs the worst under the same condition. Method selection should thus be data-driven, weighing computational costs against expected gains while accounting for each algorithm's strengths and limitations.

Moving on PCA is very useful when there are lots of highly correlated features and you want to get rid of their shared patterns. Also extremely useful when computational efficiency is priority, as the reduced dimensionality speeds up training. Last but not least, PCA is recommended when you suspect that some features may be harmful (or have negative scores using some of the forward feature selection method), as PCA's variance maximization property ensures no component degrades the model's baseline performance.

To conclude, even though PCA did not outperform MRMR with 6 features, it still achieved similar or even better accuracy results than other feature selection methods, while reducing the dimensionality.

6. Index

-	PC1	PC2	PC3	PC4	PC5
Standard Deviation	2.758788	1.711161	1.343047	1.027322	0.9704948
Proportion of Variance	0.422830	0.162670	0.100210	0.058630	0.0523300
Cumulative Proportion	0.422830	0.585500	0.685710	0.744340	0.7966700

-	PC6	PC7	PC8	PC9	PC10
Standard Deviation	0.9463681	0.8427194	0.7613234	0.7443098	0.603466
Proportion of Variance	0.0497600	0.0394500	0.0322000	0.0307800	0.020230
Cumulative Proportion	0.8464200	0.8858800	0.9180800	0.9488600	0.969090

Table 1.3: Deviation in terms of total primary components