CSDS 313/413: Introduction to Data Analysis Fall 2025

Homework 4: Clustering, Dimensionality Reduction, and Predictive Modeling

©Vo Linh Chi Dao, Mehmet Koyutürk, Khoa Tran, Rui Yang, Crystal Zhu @ CWRU

Task 1: Clustering and Dimensionality Reduction

In this exercise, our aim is to analyze a high dimensional dataset using dimensionaly reduction techniques and unsupervised learning. For this purpose, we will use the following congressional votes dataset that is provided with the assignment:

- The file "p1_congress_1984_votes.csv" contains a matrix $X \in \{-1, 0, 1\}^{435 \times 16}$ indicating the votes of 435 U.S. House of Representatives congress members on 16 key issues in the congress of 1984. Here, -1, 0 and 1 denote *reject*, *neutral* and *accept* votes respectively.
- The file "p1_congress_1984_party_affiliations.csv" contains a vector Y of size 435 x 1 indicating the party affiliations (Republican or Democrat) of 435 congress members in the congress of 1984.
  See data source for more information.

Part A

- On the congressional votes dataset given in "p1_congress_1984_votes.csv", apply principal component analysis (PCA). Plot the cumulative variance explained by top k principal components (with highest eigenvalues) as a function of k (see "example_pca_figure.png"). How many principal components do you think are enough to sufficiently summarize the data (according to the explained variance)?

- Next, project the data onto the first 3 principal components with highest eigenvalues. For each of the three principal component pairs (PC1-PC2, PC1-PC3, PC2-PC3), draw a scatter plot of congress members colored according to their party affiliations
  (as given in "p1_congress_1984_party_affiliations.csv"). Which of the principal component pair separates the congress members best according to their party affiliations? Are the congress members with the same party affiliation seem to be clustered according to their votes on the congress?

Part B

- Use your favorite clustering algorithm to cluster the congress members into two groups based on their congress votes on 16 issues. Make sure to explain the clustering algorithm and the distance function that you use to cluster the congress members. Visualize these groups with scatter plots on the first two top principal components you identified in part (a). Are these groups seem visually separated? How much do they agree with the party affiliations?

- Assess the statistical significance of the clustering you found using permutation tests. For this purpose, define a score to measure the quality of the clustering (for example, this could be the objective function of the K-means algorithm). Compute that score on the clustering you found on the original dataset. Now, obtain a permuted dataset by permuting each congress member's votes randomly across different matters (this will make the votes random and independent of each other, but will preserve the distribution of Reject/Neutral/Accept for each individual member). Then cluster this permuted dataset using the same algorithm you used to cluster the original dataset. Compute the score of the clustering again. Repeat this randomization process a large number of times (as allowed by computation resources). Now compare the distribution of the clustering scores you obtained on the permuted instances to the score you obtained on the original dataset. Based on this comparison, can you conclude that the original dataset is significantly clustered? Explain why.

Part C

- Now quantify the agreement of the clusters with the party affiliations (for example, using the mutual information between cluster membership and party affiliation).
- Repeat the clustering analysis using the first two principal components (instead of all 16 votes). Again, quantify the agreement of the clusters with the party affiliations. Which clustering (using principal components vs. using all 16 votes) agrees with the party affiliations more? Comment on why this might be the case.

Task 2: Predictive Modeling

Can Your Model Taste Good Wine?

Imagine you're a data scientist with a table full of red and white wines. You can't taste any of them, but you do have lab reports showing acidity, sugar, pH, alcohol content... and one goal: predict which wines are actually good.

That's exactly what we'll do in this assignment, using the UCI Wine Quality dataset. You'll build classification models to predict wine quality, then evaluate how well these models generalize across two distinct types of wines: red and white. Along the way, you'll explore: Can your model learn what makes a wine good, or is it just memorizing the quirks of red or white?

Dataset

The dataset contains two subsets:

- **Red wine:** 1599 samples
- **White wine:** 4898 samples

Each record includes 11 physicochemical attributes: *fixed acidity*, *volatile acidity*, *citric acid*, *residual sugar*, *chlorides*, *free sulfur dioxide*, *total sulfur dioxide*, *density*, *pH*, *sulphates*, and *alcohol*. The output variable is quality, an integer score (typically ranging from 0 to 10) assigned by human wine tasters.

You are free to use all 11 features in your modeling process. You are welcome to analyze feature distributions, correlations, or importance to evaluate which variables contribute most to prediction. This is entirely optional and not required for full credit.

Part A

You will start by designing your own label for the classification task by discretizing the quality scores of the wine samples:

- Discretize the quality scores in both the red and white wine datasets into either two classes. You are free to choose the thresholds, but your label definition (i.e., the number of classes and cutoff values) must be applied consistently across both datasets. This ensures that models trained on one wine type can be meaningfully evaluated on the other.
- Visualize the original quality score distribution (e.g., using histograms), and plot the resulting class distribution after discretization for both red and white wines.
- Briefly describe the rationale behind your label design ($\leq$ 150 words). For example, did you aim for class balance, interpretability, or better model fit?

Part B

In this part, you will train two classification models and evaluate their performance both within the same wine type and across different wine types. You may choose any two classification models (e.g., neural networks, logistic regression, decision tree, random forest, SVM, or k-NN). Consider model complexity and generalization when making your selection. For each wine type (red and white), split the data into training and testing sets using a 70/30 or 80/20 split.

Since the white wine dataset is significantly larger than the red wine dataset, you must downsample the white wine training set to match the red wine sample size ($\sim$1600 rows). This ensures a fair comparison between models trained on red and white wines. Each model will be evaluated under two conditions:

- **In-domain testing:** Train and test on the same wine type (e.g., train on red, test on red).
- **Cross-domain testing:** Train on one wine type and test on the other (e.g., train on red, test on white).

For each evaluation setting, report at least one performance metric (e.g., accuracy, precision, recall). Your results in this part will be used for analysis and visualization in Part C.

Part C

In this part, you will interpret and compare your model results from Part B. Your answers should be concise ($\leq$ 100 words per question). You are expected to include relevant visualizations to support your analysis.

1. Across the in-domain settings (i.e., red $\rightarrow$ red and white $\rightarrow$ white), which model demonstrated more consistent performance? Did you observe any signs of overfitting?

2. Which model generalized better across wine types (i.e., red → white and white → red)? How much did its performance degrade compared to in-domain testing?

3. What factors might explain the performance differences observed during cross-domain testing? Consider possible causes such as dataset size, feature distribution shifts, label imbalance, or model sensitivity.
   Support your analysis with at least one informative visualization. For example, this could be a bar or line chart comparing performance across domains, a confusion matrix highlighting domain-specific errors, or an additional plot such as feature histograms, importance rankings, or misclassification breakdowns.