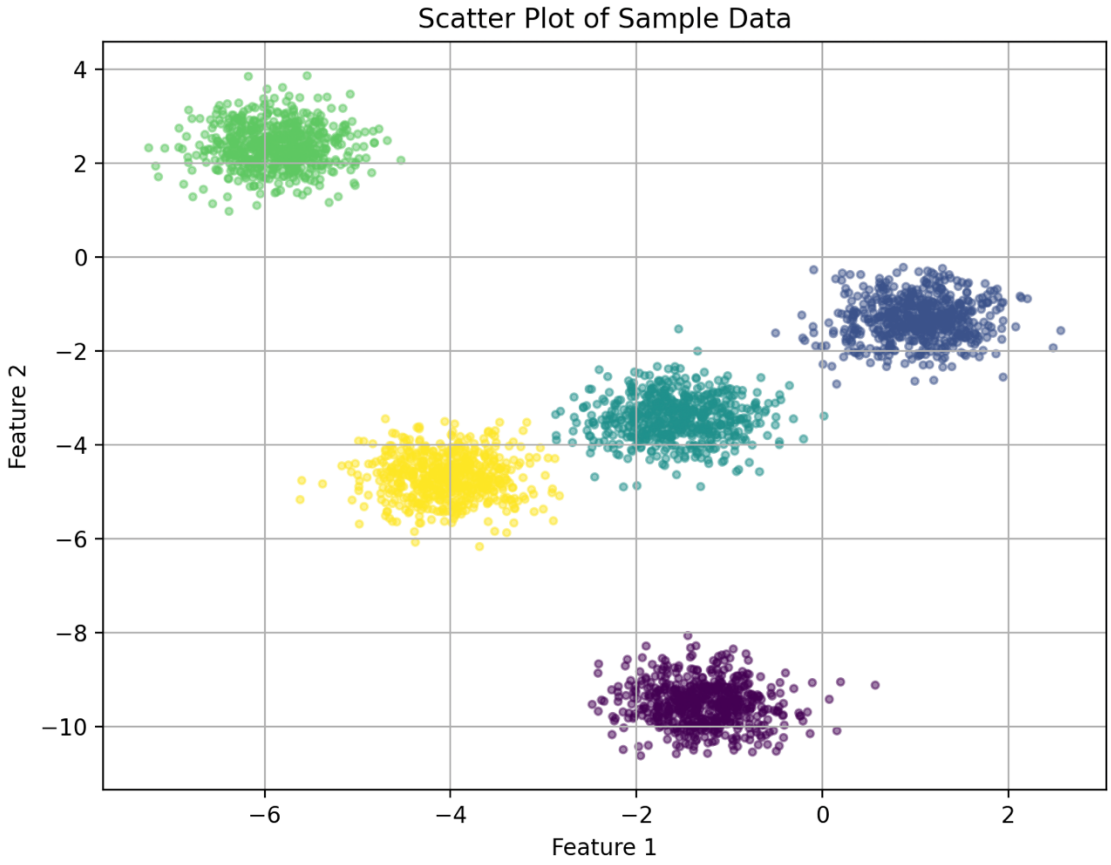
MAI612 - MACHINE LEARNING
Assignment 4 – Clustering & Anomaly Detection

Chrysis Andreou
(UC1366020)

Part A

Task 1: Visualize the Sample Data



Task 2: Implementing K-Means from Scratch

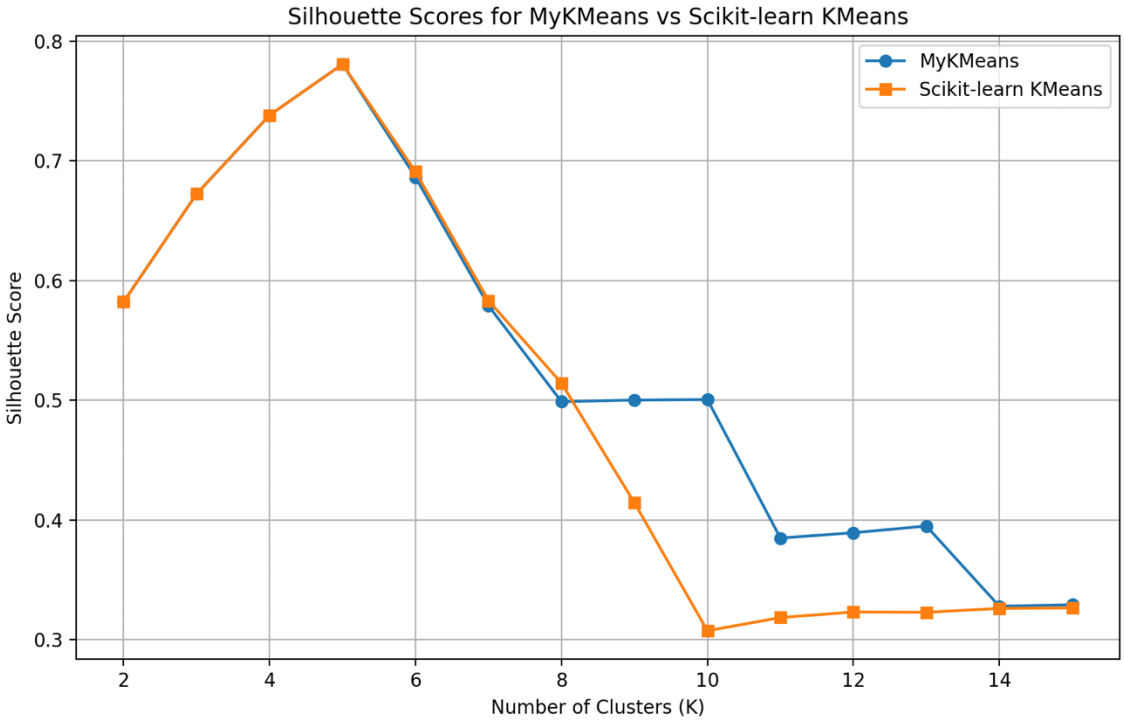
Final Centroids:
[[-4.04000168 -4.68322733]
 [-5.88689925 2.35538697]
 [-1.28529629 -9.51564383]
 [1.0284437 -1.32845473]
 [-1.57367699 -3.42492325]]

Final Inertia: 1218.7150

Task 3: Comparing Silhouette Scores

K=2: MyKMeans Silhouette Score = 0.5822, Scikit-learn KMeans Silhouette Score = 0.5822
K=3: MyKMeans Silhouette Score = 0.6723, Scikit-learn KMeans Silhouette Score = 0.6723
K=4: MyKMeans Silhouette Score = 0.7381, Scikit-learn KMeans Silhouette Score = 0.7381

K=5: MyKMeans Silhouette Score = 0.7808, Scikit-learn KMeans Silhouette Score = 0.7808
K=6: MyKMeans Silhouette Score = 0.6863, Scikit-learn KMeans Silhouette Score = 0.6910
K=7: MyKMeans Silhouette Score = 0.5790, Scikit-learn KMeans Silhouette Score = 0.5832
K=8: MyKMeans Silhouette Score = 0.4988, Scikit-learn KMeans Silhouette Score = 0.5143
K=9: MyKMeans Silhouette Score = 0.5001, Scikit-learn KMeans Silhouette Score = 0.4143
K=10: MyKMeans Silhouette Score = 0.5006, Scikit-learn KMeans Silhouette Score = 0.3074
K=11: MyKMeans Silhouette Score = 0.3849, Scikit-learn KMeans Silhouette Score = 0.3184
K=12: MyKMeans Silhouette Score = 0.3893, Scikit-learn KMeans Silhouette Score = 0.3231
K=13: MyKMeans Silhouette Score = 0.3949, Scikit-learn KMeans Silhouette Score = 0.3228
K=14: MyKMeans Silhouette Score = 0.3279, Scikit-learn KMeans Silhouette Score = 0.3261
K=15: MyKMeans Silhouette Score = 0.3291, Scikit-learn KMeans Silhouette Score = 0.3264

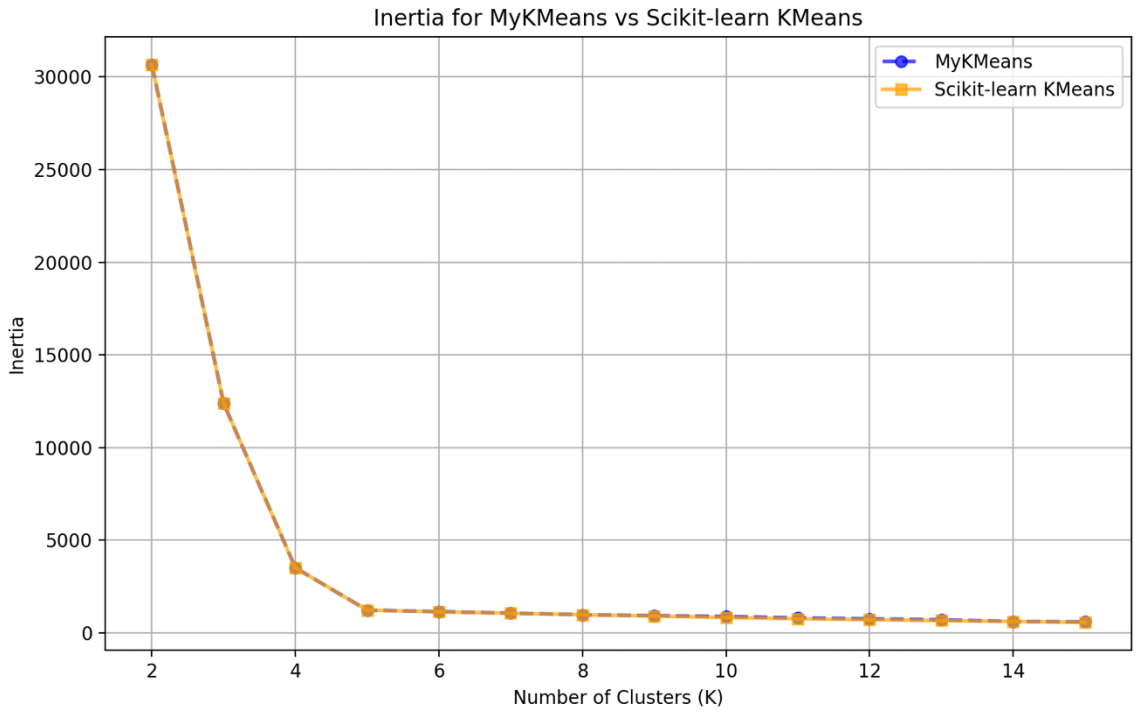


Comparison of MyKMeans and Scikit-learn KMeans:
1. Up to K=5, both implementations show similar silhouette scores.
2. For K=6 to K=8, Scikit-learn's KMeans generally has slightly higher scores.
3. For K=9 to K=13, MyKMeans achieves higher silhouette scores than Scikit-learn.
4. The optimal number of clusters remains K=5 based on the highest silhouette score.

Reasons Why MyKMeans Output Differs from Scikit-learn's KMeans:
1. Centroid Initialization: MyKMeans may not fully implement K-Means++ initialization, affecting initial centroid placement.
2. Empty Cluster Handling: MyKMeans retains old centroids when clusters are empty, whereas Scikit-learn reinitializes them to new positions.

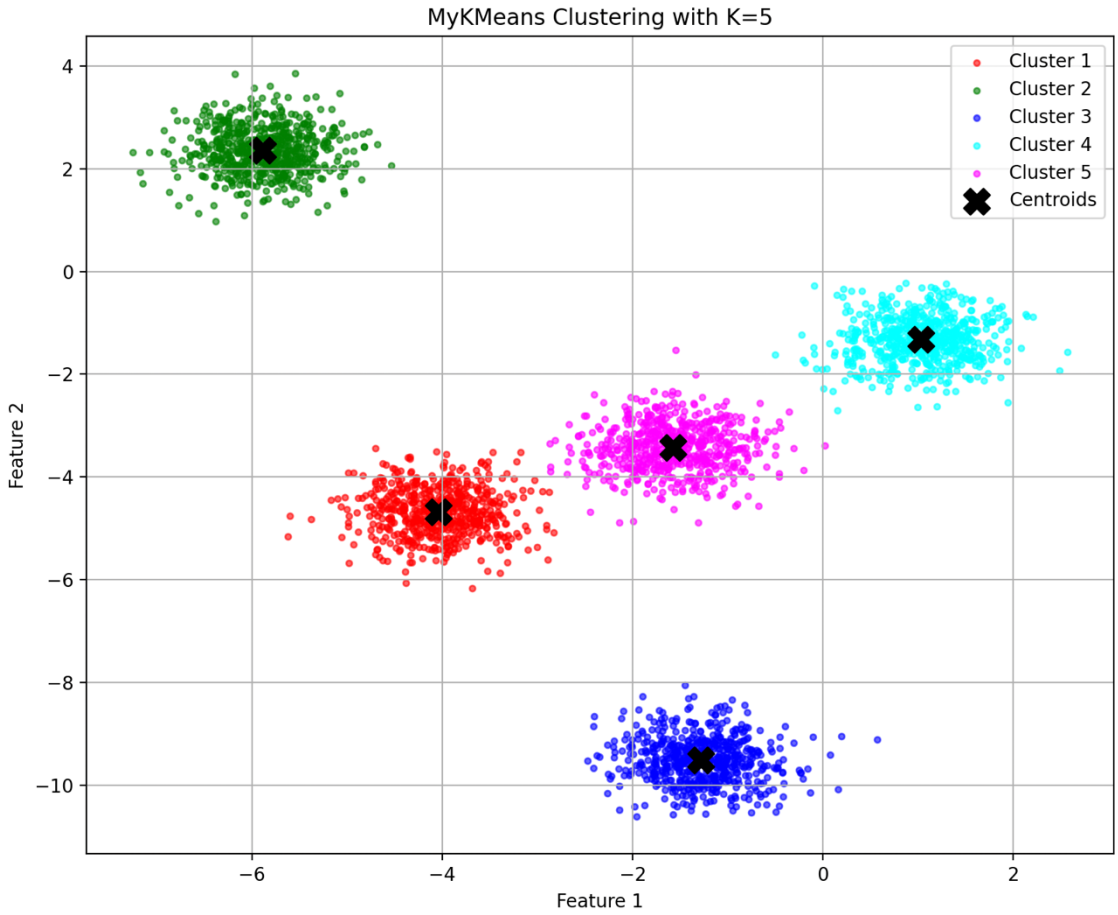
Task 4: Comparing Inertia using the Elbow Method

K=2: MyKMeans Inertia = 30660.73, Scikit-learn KMeans Inertia = 30660.73
K=3: MyKMeans Inertia = 12373.90, Scikit-learn KMeans Inertia = 12373.90
K=4: MyKMeans Inertia = 3500.83, Scikit-learn KMeans Inertia = 3500.83
K=5: MyKMeans Inertia = 1218.71, Scikit-learn KMeans Inertia = 1218.71
K=6: MyKMeans Inertia = 1139.73, Scikit-learn KMeans Inertia = 1133.40
K=7: MyKMeans Inertia = 1057.78, Scikit-learn KMeans Inertia = 1049.77
K=8: MyKMeans Inertia = 971.16, Scikit-learn KMeans Inertia = 969.17
K=9: MyKMeans Inertia = 919.40, Scikit-learn KMeans Inertia = 901.87
K=10: MyKMeans Inertia = 873.99, Scikit-learn KMeans Inertia = 820.33
K=11: MyKMeans Inertia = 798.39, Scikit-learn KMeans Inertia = 752.56
K=12: MyKMeans Inertia = 749.52, Scikit-learn KMeans Inertia = 700.29
K=13: MyKMeans Inertia = 696.55, Scikit-learn KMeans Inertia = 655.94
K=14: MyKMeans Inertia = 607.13, Scikit-learn KMeans Inertia = 608.09
K=15: MyKMeans Inertia = 583.08, Scikit-learn KMeans Inertia = 562.19



Comparison: Both implementations show similar inertia values, indicating effective clustering.
Optimal Clusters: The elbow point at K=5 suggests this is the optimal number of clusters.

Task 5: Training MyKMeans with K=5 and Plotting



Part B

Q1: Given that fraudulent transactions exist in the original/legitimate transactions, what kind of anomaly detection should we use and why? Outlier or Novelty?
A1: We should use outlier detection because we are identifying rare instances (frauds) within the known data distribution.

Q2: How many fraudulent and how many legitimate transactions exist in the original dataset?
A2: Number of fraudulent transactions: 487
Number of legitimate transactions: 28423

Q3: What percentage of transactions is fraudulent in the original dataset?
A3: Percentage of fraudulent transactions: 1.68%

Q4: How many fraudulent transactions were the IsolationForest and OneClassSVM able to capture separately?
A4: Number of fraudulent transactions detected by IsolationForest in training data: 377 out of 487

Number of fraudulent transactions detected by OneClassSVM in training data: 468 out of 487

Q5: How many legitimate transactions were incorrectly classified as fraudulent by IsolationForest and OneClassSVM separately in training data?
A5: Number of legitimate transactions incorrectly classified as fraudulent by IsolationForest in training data: 746
Number of legitimate transactions incorrectly classified as fraudulent by OneClassSVM in training data: 13986

Q6: Out of the total number of frauds in the original dataset,
what percentage of them were detected by IsolationForest and OneClassSVM respectively?
What is this metric called usually in machine learning?
A6: Percentage of fraudulent transactions detected by IsolationForest: 77.41%
Percentage of fraudulent transactions detected by OneClassSVM: 96.10%
This metric is commonly referred to as 'recall' or 'sensitivity' in machine learning.

Q7: How much time does it take to train IsolationForest and OneClassSVM for detecting fraudulent transactions?
What do you notice? Why is this happening?
A7: Time taken to train IsolationForest: 0.0963 seconds
Time taken to train OneClassSVM: 14.4200 seconds
Notice: OneClassSVM generally takes longer to train than IsolationForest due to its computational complexity.
IsolationForest isolates anomalies by constructing random trees,
which is efficient and scales well with data size.
In contrast, OneClassSVM uses kernel methods to transform data into higher dimensions
and solve complex optimization problems, leading to longer training times.

Q8: Algorithmically speaking, how do we classify a sample as an anomaly when using IsolationForest and OneClassSVM?
A8:
- IsolationForest: It isolates anomalies by randomly selecting a feature
and then randomly selecting a split value between the maximum and minimum values of that feature.
Anomalies are isolated quickly in fewer random partitions, so they have shorter path lengths in the tree structure.
- OneClassSVM: It uses a hyperplane to separate the data points in a high-dimensional space.
Points that lie on one side of the hyperplane are considered normal,
while those on the other side are considered anomalies.
It uses kernel methods to transform the data into higher dimensions to find this hyperplane.

Q9: Can you spot any anomalies in questions 1-8. If yes, how many and why?

Accuracy of IsolationForest: 0.38%
Accuracy of OneClassSVM: 0.07%
Comment:
- IsolationForest has a higher accuracy (0.38%) compared to OneClassSVM (0.07%) in this dataset.
- OneClassSVM shows a higher recall (96.10%), meaning it detects more fraudulent transactions (468 out of 487),
but it also has a significantly higher false positive rate (13986 false positives), leading to lower overall accuracy.
- This is an anomaly because while OneClassSVM seems to perform better in terms of recall,
its high false positive rate means it incorrectly classifies many legitimate transactions as frauds.
- The choice between these models depends on the specific requirements of the task,
such as whether minimizing false positives or maximizing recall is more important.
- The class imbalance affects accuracy because the dataset contains 28423 legitimate transactions and only 487 fraudulent transactions.
A model could achieve high accuracy by predicting all transactions as legitimate, but this would not be useful for detecting frauds.

Q10: Using your two trained models, detect anomalies on the new data (transactions_mini_validation):
Q10A. What is this anomaly detection method called? Outlier or Novelty?
A. This anomaly detection method is called 'Novelty Detection'
because we are applying the trained model to new, unseen data to identify anomalies.
In contrast, question 1 referred to outlier detection,
which involves identifying anomalies within the known dataset.

Q10B. How many anomalies are your trained models able to detect?
IsolationForest detected 2 anomalies in total, of which 2 were true positives (actual frauds), 0 were false positives (legitimate transactions incorrectly flagged), and 3 were false negatives (frauds missed).
OneClassSVM detected 11 anomalies in total, of which 5 were true positives (actual frauds), 6 were false positives (legitimate transactions incorrectly flagged), and 0 were false negatives (frauds missed).

Based on the results, we would choose OneClassSVM for this task because it has a higher recall, meaning it detects more fraudulent transactions. Although it has a higher false positive rate, we can manually review and dismiss false alarms, ensuring that we do not miss potential frauds.