CMPE 544 Assignment 2 Report

Ömer Coşkun
2024700024

Code Link: https://github.com/coskunomer/pattern-recognition-assignment-2

Classification

This section is about the first part of the assignment - Classification.

Part 1. SVM from scratch

A linear soft-margin SVM is implemented and solved using a quadratic programming solver.

Objective Function:

$$max_{\alpha} \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{m=1}^{N} \sum_{n=1}^{N} \alpha_n \alpha_m y_n y_m x_n^T x_m$$

Constraints:

$$s.t.\ 0 \leq \alpha_n \leq C \quad , \quad \forall n$$

$$\sum_{n=1}^{N} y_n \alpha_n = 0$$

The feature extraction process remains unchanged compared to the previous assignment. The dataset is filtered to only include rabbit and hand examples.

The training and test data is then transformed into the matrix format required by the “cvxopt” library that is used as a quadratic programming solver. Different C values are experimented against each other to cross validate and find the optimal C value. Below are the cross validation results:

C	Accuracy
0.001	0.9307
0.01	0.9415

0.1	0.9364
1	0.9224

The best value C = 0.01 is then used to train the final model using the whole training dataset. Training took **11 minutes and 36 seconds** and the resulting test accuracy was **0.9425**.

Part 2. SVM from scikit-learn

In this part, SVM's function was used to train the models. Linear, RBF, and Polynomial kernels are tested, hyperparameters for each of them are tuned using 5-fold cross-validation. The results are presented as below:

Linear Kernel

C = [0.001, 0.01, 0.1, 1] values are tested using 5-fold cross-validation. Best result was achieved using C = 0.01. Training took **6.05 seconds** and the resulting test accuracy was **0.9425**.

RBF Kernel

C = [0.01, 0.1, 1] and gamma = ['scale', 0.01, 0.001] are tested using 5-fold cross-validation. Gamma value represents how much influence a single training example has. It defines the shape of the decision boundary. Best result was achieved using C = 1 and gamma = 'scale'. Training took **15.97 minutes** and the resulting test accuracy was **0.9555**.

Polynomial Kernel

C = [0.01, 0.1, 1], degree = [2, 3, 4, 5] and coef0 (bias term) = [0, 1] are tested using 5-fold cross-validation. Best result was achieved using C = 1, degree = 3, coef0 = 0. Training took **11.6 minutes** and the resulting test accuracy was **0.9570**.

Comparison

The results of scikit-learn and part 1's model are exactly the same, while scikit-learn's kernel performed significantly faster (115 times faster). RBF kernel and polynomial kernel however, even though slower, performed better than the linear kernels, which suggests that the **dataset contains non-linearities that cannot be captured with a linear kernel**.

Part 3. Final Model and Results

Best performing model is used to train the final model with C = 1, degree = 3, coef0 = 0. Training took **11.6 minutes** and the resulting test accuracy was **0.9570**. Below are examples from support vectors and vectors that are far from the margin,

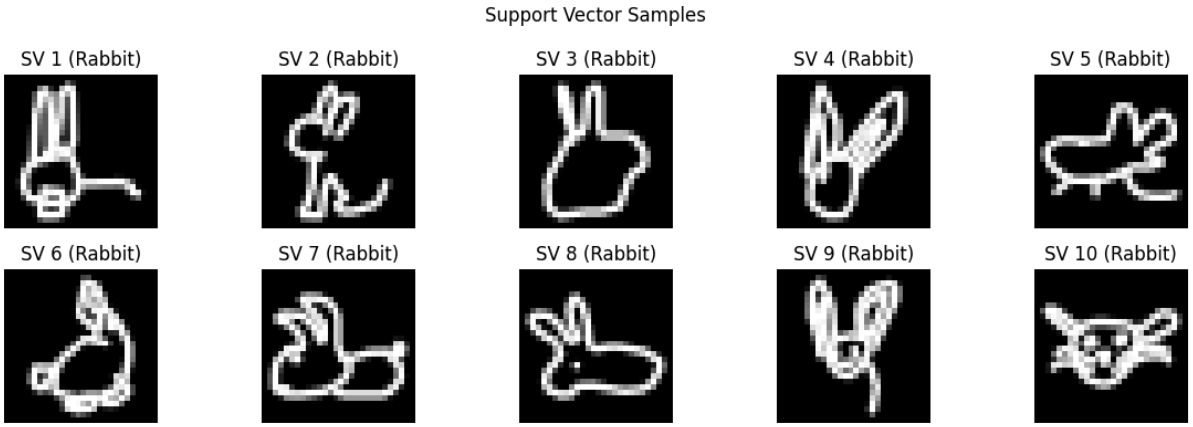

Fig 1. Support Vectors

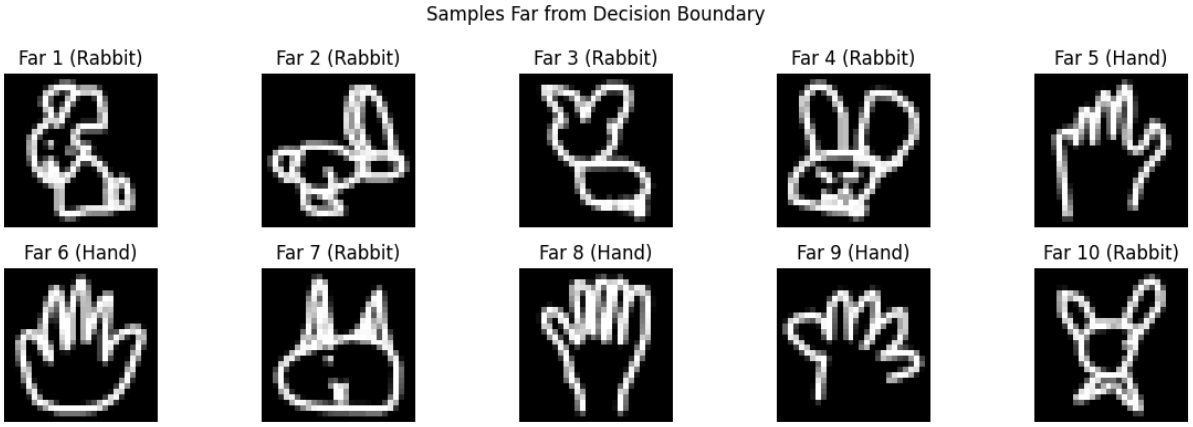

Fig 2. Vectors that are far from margin

Some of the support vector examples contain data points that do not look like rabbit or hand, which is consistent with what we would expect from samples that are hard to predict and therefore are used as support vectors. Samples that are far from the decision boundary, however, can be easily identified in the first glance, as we would expect, since they should be relatively easier to classify to be far from the decision margin.

Clustering

This section is about the second part of the assignment - Clustering.

In this part of the assignment, a custom K-Means clustering algorithm is implemented to group the dataset into 5 clusters, corresponding to the 5 image classes (Rabbit, Yoga, Hand, Snowman, Motorbike). The performance of the model is evaluated using multiple distance

metrics—Euclidean, Manhattan, and Cosine—to understand their effect on clustering quality. The following evaluation metrics are used to quantify the results:

1. Sum of Squared Errors (SSE)

   SSE works by summing the squared distances from each data point to its assigned cluster. It is one the most common ways of quantifying errors in K-Means.

$$SSE = \sum_{n=1}^{N} \sum_{x \in C_i} \|x - \mu_i\|$$

2. Accuracy

   Accuracy is measured by finding the best label permutation that aligns predicted clusters with true labels.

3. Silhouette Score [1]

   Measures how similar a sample is to its own cluster compared to other clusters. Ranges from -1 to 1. Higher values mean better-defined clusters.

$$s(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}}$$

   where a(i) is the mean intra-cluster distance and b(i) is the mean nearest-cluster distance.

4. Calinski-Harabasz Index [2]

   Higher values indicate better-defined clusters.

$$CH = \frac{TR(B_k)(N-k)}{TR(W_k)(k-1)}$$

   where B_k and W_k are the between-cluster and within-cluster scatter matrices.

5. Adjusted Rand Index (ARI) [3]

   Measures the similarity between predicted and true labels, adjusted for chance. Values range from -1 (poor) to 1 (perfect agreement).

$$ARI = \frac{RI - E[RI]}{max(RI) - E[RI]}$$

6. Normalized Mutual Information (NMI) [4]

   Quantifies mutual dependence between predicted and true labels, normalized between 0 (no mutual info) and 1 (perfect correlation).

$$NMI(U,V) = \frac{2\,I(U;V)}{H(U)+H(V)}$$

Implemented K-Means algorithm was run using Euclidean, Manhattan and Cosine distance metrics. The results are presented below:

Distance Metric	SSE	Accuracy	Silhouette	Calinski-Harabasz	ARI	NMI
Euclidean	400211	0.5040	0.0385	604.9	0.2375	0.2815
Manhattan	410055	0.5048	0.0529	470.4	0.2069	0.3097
Cosine	400436	0.5089	0.0391	601.8	0.2440	0.2852

Overall, Euclidean and Cosine distances were more consistent across all metrics, with Cosine distance performing slightly better.

References

[1] Rousseeuw, Peter J. "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis." *Journal of computational and applied mathematics* 20 (1987): 53-65.

[2] Caliński, Tadeusz, and Jerzy Harabasz. "A dendrite method for cluster analysis." *Communications in Statistics-theory and Methods* 3.1 (1974): 1-27.

[3] Hubert, Lawrence, and Phipps Arabie. "Comparing partitions." *Journal of classification* 2 (1985): 193-218.

[4] Strehl, Alexander, and Joydeep Ghosh. "Cluster ensembles---a knowledge reuse framework for combining multiple partitions." *Journal of machine learning research* 3.Dec (2002): 583-617.