

Project 2: Signature-fraud busting using K-means

Aniket Sharma (MS18126)

1 Introduction

The problem statement that we will attempt to solve in this project is stated like so:

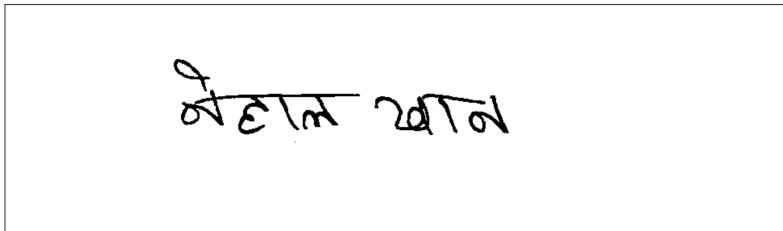
Given a small dataset consisting of signatures from multiple people, we would like to classify a newly introduced signature as either being a forgery or genuine replication of one of the existing signatures.

We will approach this by using **K-means clustering** to find clusters of signatures, each of which will ideally correspond to a single person's signature. A set of threshold values are then calculated for each cluster centroid, such that crossing the threshold will classify the sample as a forgery of that signature cluster. More details are presented further in our discussion.

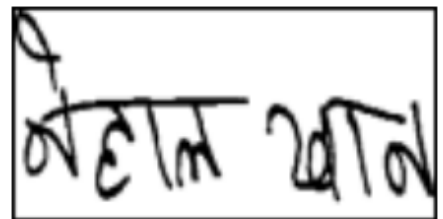
1.1 Data Scraping

We have obtained the dataset from the **Offline-Signature-Verification-using-Siamese-Network** project which contains signatures of 160 and 100 individuals in Hindi and Bengali, respectively. We proceed with only the Hindi dataset, as the Bengali script consistently yielded poor results. We considered 10 individuals, with 24 genuine signatures (19 of which were used to generate clusters, and 5 used for testing), and 30 forgeries used for testing. The key results discussed below holds the same even while considering the signatures of 50 individuals.

1.2 Image Pre-Processing



(a) Before pre-processing



(b) After pre-processing

The images were already in grayscale, which we converted to bit arrays (1 if pixel > threshold, 0 otherwise) to convert it entirely to black and white. Each image was then resized, reshaped and flattened to form a 1-dimensional vector of high dimensionality. No noise reduction was performed, as preliminary detection of noise in the images turned up negative.

2 Forming signature clusters

We perform the standard K-means clustering algorithm on 190 signature images (10 individuals) till the centroids have converged. The algorithm itself is unsupervised, so it cannot utilize labels (i.e. the individuals' name), but since we possess this information from the dataset, we create a composite datatype "**signature**" holding both the label (person's name) and the signature image. This is used for quantifying the accuracy and prediction rates after clustering is performed. Each centroid is assigned the label of the largest representative individual in its cluster ($> 50\%$). If this is not satisfied, the centroid is labelled "undefined", as it is diluted by too many varied signatures.

3 Optimizing free parameters

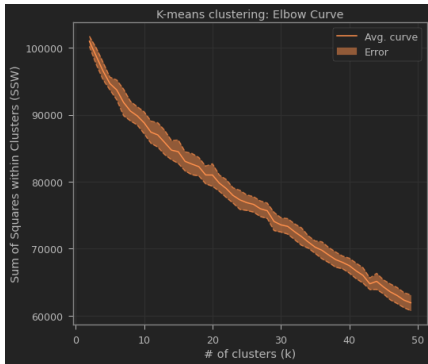
When working with K-means clustering, we have two parameters to set, namely, the number of clusters and the distance metric. We consider a range of k-values, and three distance metrics to draw some quick comparisons on which is most appropriate for our task.

3.1 Number of clusters, k

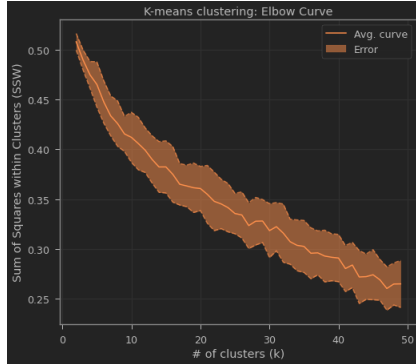
To quantify this, we plot the Sum-of-Squares-Within-Clusters (SSW) against the k-value over a range ($[2, 50]$) and observe the general trend for various distance metrics. Given a set of centroids $\{c_i\}$, and the set of points in its corresponding cluster, $\{x_{ij}\}$, the SSW is simply computed like so:

$$SSW(k) = \sum_{i=1}^k \sum_j [\text{dist}(x_{ij} - c_i)]^2$$

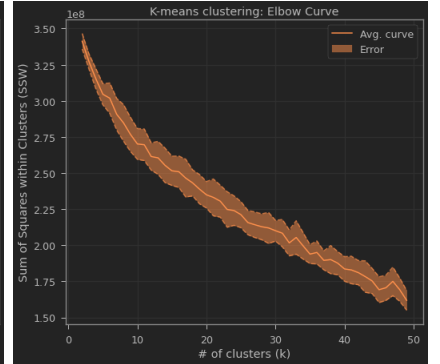
Intuitively, we expect the curve to have a knee at some k-value, that is, too many clusters subdivide the data into too many subgroups which reduces efficiency of classification.



(a) Euclidean metric



(b) Eisen-cosine metric



(c) Manhattan metric

These curves were plotted by averaging over 30 runs for each value of k. However, for our data, we were unable to determine an optimal K with certainty as we observe a strictly decreasing curve instead of an distinctive elbow-shaped curve for all the three distance metrics. Visually, we can roughly identify the elbow points as $k = 7, 11, 11$ for each of the three metrics. This was loosely verified by visual inspection of the clusters formed.

3.2 Distance metric for image comparison

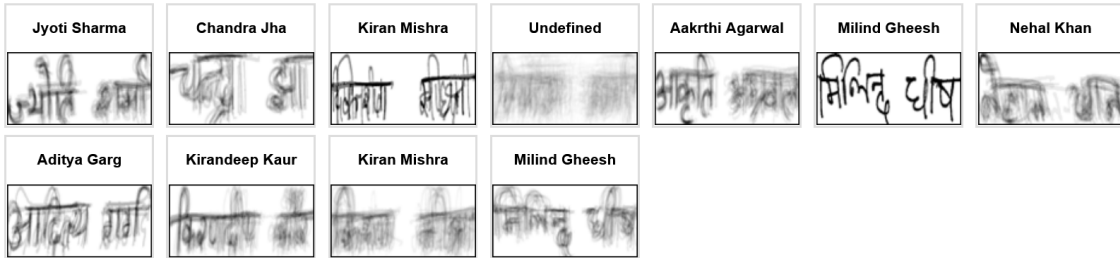
Here, we try geometric separation and correlation based distance metrics namely, Euclidean distance, Manhattan distance and Eisen-Cosine correlation distance. We perform clustering for $k = 11$ (as determined in the previous section), and compare the qualitative and quantitative results for each of these metrics to determine the most appropriate one. Each run produces a table which demonstrates the composition of various individuals in each of the clusters formed.

3.2.1 Euclidean metric

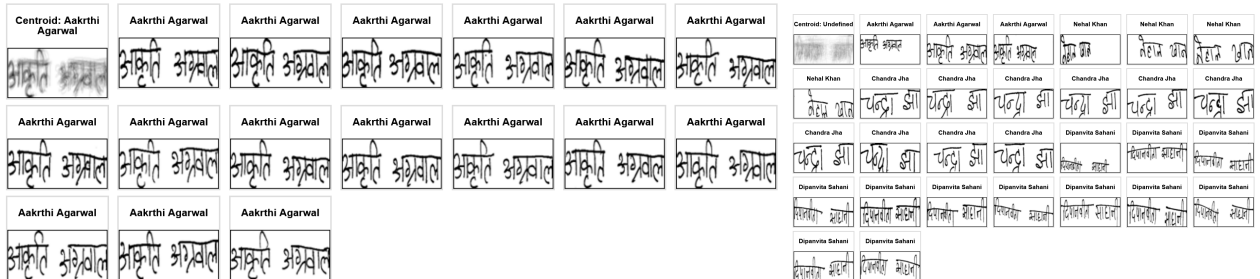
$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

Cluster label	#Points clustered	Chandra Jha	Aakrthi Agarwal	Aditya Garg	Dipanvita Sahani	Vikas Prasad	Milind Gheesh	Kirandeep Kaur	Jyoti Sharma	Kiran Mishra	Nehal Khan
0 Jyoti Sharma	12	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%
1 Chandra Jha	9	100.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
2 Kiran Mishra	2	0.0%	0.0%	0.0%	50.0%	0.0%	0.0%	0.0%	0.0%	50.0%	0.0%
3 Undefined	75	13.33%	4.0%	17.33%	22.67%	21.33%	5.33%	0.0%	9.33%	1.33%	5.33%
4 Aakrthi Agarwal	16	0.0%	100.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
5 Milind Gheesh	1	0.0%	0.0%	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%	0.0%	0.0%
6 Nehal Khan	14	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	100.0%
7 Aditya Garg	7	0.0%	0.0%	85.71%	0.0%	14.29%	0.0%	0.0%	0.0%	0.0%	0.0%
8 Kirandeep Kaur	19	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%	0.0%
9 Kiran Mishra	21	0.0%	0.0%	0.0%	4.76%	9.52%	0.0%	0.0%	0.0%	80.95%	4.76%
10 Milind Gheesh	14	0.0%	0.0%	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%	0.0%	0.0%

(a) Cluster composition



(b) Clusters formed



(a) Example of good cluster

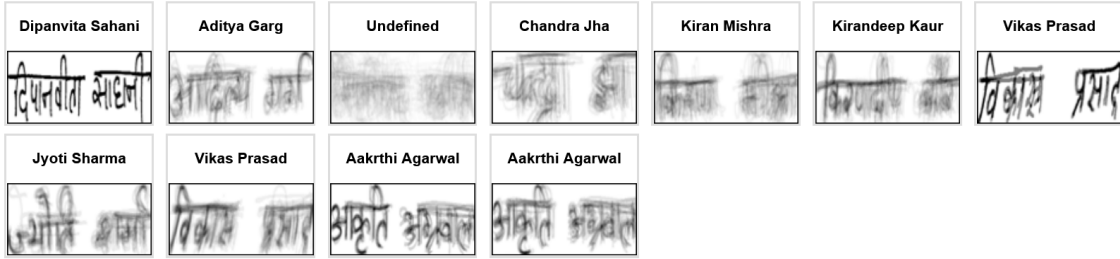
(b) Example of bad cluster

3.2.2 Eisen-Cosine correlation metric

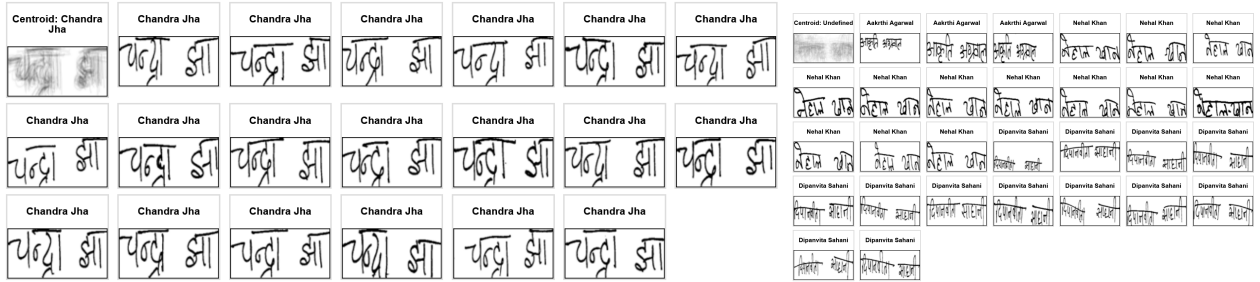
$$d(x, y) = 1 - \frac{|\sum_{i=1}^n x_i y_i|}{\sqrt{\sum_{i=1}^n x_i^2} \sqrt{\sum_{i=1}^n y_i^2}}$$

Cluster label	#Points clustered	Chandra Jha	Aakrthi Agarwal	Aditya Garg	Dipanvita Sahani	Vikas Prasad	Milind Gheesh	Kirandeep Kaur	Jyoti Sharma	Kiran Mishra	Nehal Khan
0	Dipanvita Sahani	1	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%	0.0%	0.0%	0.0%
1	Aditya Garg	17	0.0%	0.0%	94.12%	0.0%	0.0%	0.0%	0.0%	0.0%	5.88%
2	Undefined	62	0.0%	4.84%	4.84%	25.81%	1.61%	30.65%	0.0%	9.68%	1.61%
3	Chandra Jha	19	100.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
4	Kiran Mishra	22	0.0%	0.0%	0.0%	9.09%	0.0%	0.0%	0.0%	77.27%	13.64%
5	Kirandeep Kaur	20	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	95.0%	0.0%	5.0%
6	Vikas Prasad	2	0.0%	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%	0.0%	0.0%
7	Jyoti Sharma	14	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	92.86%	0.0%	7.14%
8	Vikas Prasad	17	0.0%	0.0%	0.0%	0.0%	94.12%	0.0%	0.0%	5.88%	0.0%
9	Aakrthi Agarwal	6	0.0%	100.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
10	Aakrthi Agarwal	10	0.0%	100.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%

(a) Cluster composition



(b) Clusters formed



(a) Example of good cluster

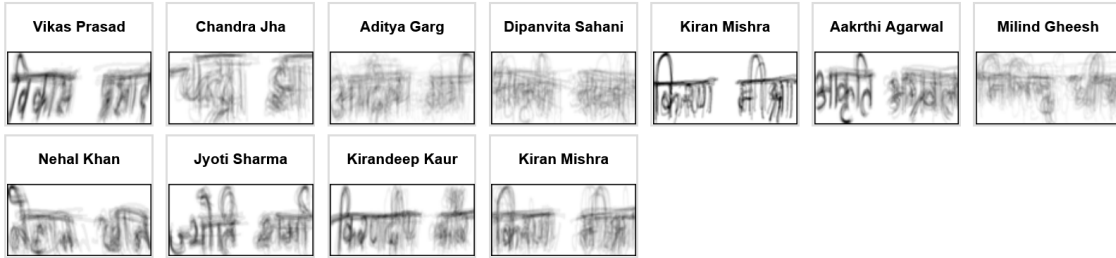
(b) Example of bad cluster

3.2.3 Manhattan metric

$$d(x, y) = \sum_{i=1}^n |x_i - y_i|$$

Cluster label	#Points clustered	Chandra Jha	Aakrthi Agarwal	Aditya Garg	Dipanvita Sahani	Vikas Prasad	Milind Gheesh	Kirandeep Kaur	Jyoti Sharma	Kiran Mishra	Nehal Khan
0 Vikas Prasad	17	0.0%	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%	0.0%	0.0%	0.0%
1 Chandra Jha	17	94.12%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	5.88%	0.0%	0.0%
2 Aditya Garg	30	10.0%	3.33%	63.33%	10.0%	0.0%	0.0%	0.0%	10.0%	0.0%	3.33%
3 Dipanvita Sahani	20	0.0%	20.0%	0.0%	70.0%	0.0%	0.0%	0.0%	0.0%	10.0%	0.0%
4 Kiran Mishra	4	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	100.0%	0.0%
5 Aakrthi Agarwal	13	0.0%	100.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
6 Milind Gheesh	25	0.0%	4.0%	0.0%	4.0%	4.0%	76.0%	0.0%	4.0%	0.0%	8.0%
7 Nehal Khan	15	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	100.0%
8 Jyoti Sharma	14	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%
9 Kirandeep Kaur	19	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%	0.0%
10 Kiran Mishra	16	0.0%	0.0%	0.0%	6.25%	6.25%	0.0%	0.0%	0.0%	81.25%	6.25%

(a) Cluster composition



(b) Clusters formed






3.3 Conclusion

Based on the results above we see that the efficiency of obtaining good clusters varies with the distance metric that is being used. In general, Manhattan distance should give more robust results, whereas Euclidean distance is likely to be influenced by outliers. Based on observations of the data provided above, the Manhattan metric was the best, closely followed by the Eisen-Cosine metric. The efficiency of Euclidean distance as distance metric yielded poor results. So, we proceed with the Manhattan metric for fraud detection.

We proceed with $k = 11$ clusters as it was determined to be optimal for the Manhattan metric. The following page presents each centroid along with its clustering points for a k-means run with the Manhattan metric, and $k = 11$. Moreover, each run converges to different cluster centroids, so we only consider the ones with minimum SSW.

3.4 Centroids and Clusters for Manhattan metric

<p>Centrol: Aditya Garg</p> 	<p>Aashrith Agarwal</p> 	<p>Nehal Khan</p> 	<p>Chandra Jha</p> 	<p>Chandra Jha</p> 	<p>Chandra Jha</p> 	<p>Dipavri Sahani</p> 
<p>Dipavri Sahani</p> 	<p>Dipavri Sahani</p> 	<p>Jyoti Sharma</p> 	<p>Jyoti Sharma</p> 	<p>Jyoti Sharma</p> 	<p>Aditya Garg</p> 	<p>Aditya Garg</p> 
<p>Aditya Garg</p> 	<p>Aditya Garg</p> 	<p>Aditya Garg</p> 	<p>Aditya Garg</p> 	<p>Aditya Garg</p> 	<p>Aditya Garg</p> 	<p>Aditya Garg</p> 
<p>Aditya Garg</p> 	<p>Aditya Garg</p> 	<p>Aditya Garg</p> 	<p>Aditya Garg</p> 	<p>Aditya Garg</p> 	<p>Aditya Garg</p> 	<p>Aditya Garg</p> 
<p>Aditya Garg</p> 	<p>Aditya Garg</p> 					

Centroid: Kiran Mishra	Kiran Mishra	Kiran Mishra	Kiran Mishra	Kiran Mishra
				

4 Fraud detection

Each centroid can be characterized by the distances of the points belonging to its cluster. So we plot the frequency of distances in each cluster to observe the general distributions.



(a) Distribution of distances

The distribution does not represent a normal/gaussian curve, but this might be due to the small sample size. We proceed by assuming that the actual distribution will follow a gaussian, and characterize the cluster by its mean and variance of distances. We can now define a threshold distance for each cluster ($t = \mu + \lambda\sigma$). A new signature can then be classified by computing distances to each cluster, and checking if it falls within the threshold. This value is determined to be $t = \mu + 0.5\sigma$, by trial and error over several runs.

Some typical results for Manhattan metric with 11 clusters are shown below. The table rows show the distance, tolerance and acceptance (True == Genuine, False == Forgery) of the sample from each centroid. By finding the nearest centroid, we can determine **which cluster it is closest to**, and ideally, whose signature it is trying to mimic (forge). The matched centroid and the sample image are also displayed.

4.1 Results and observations

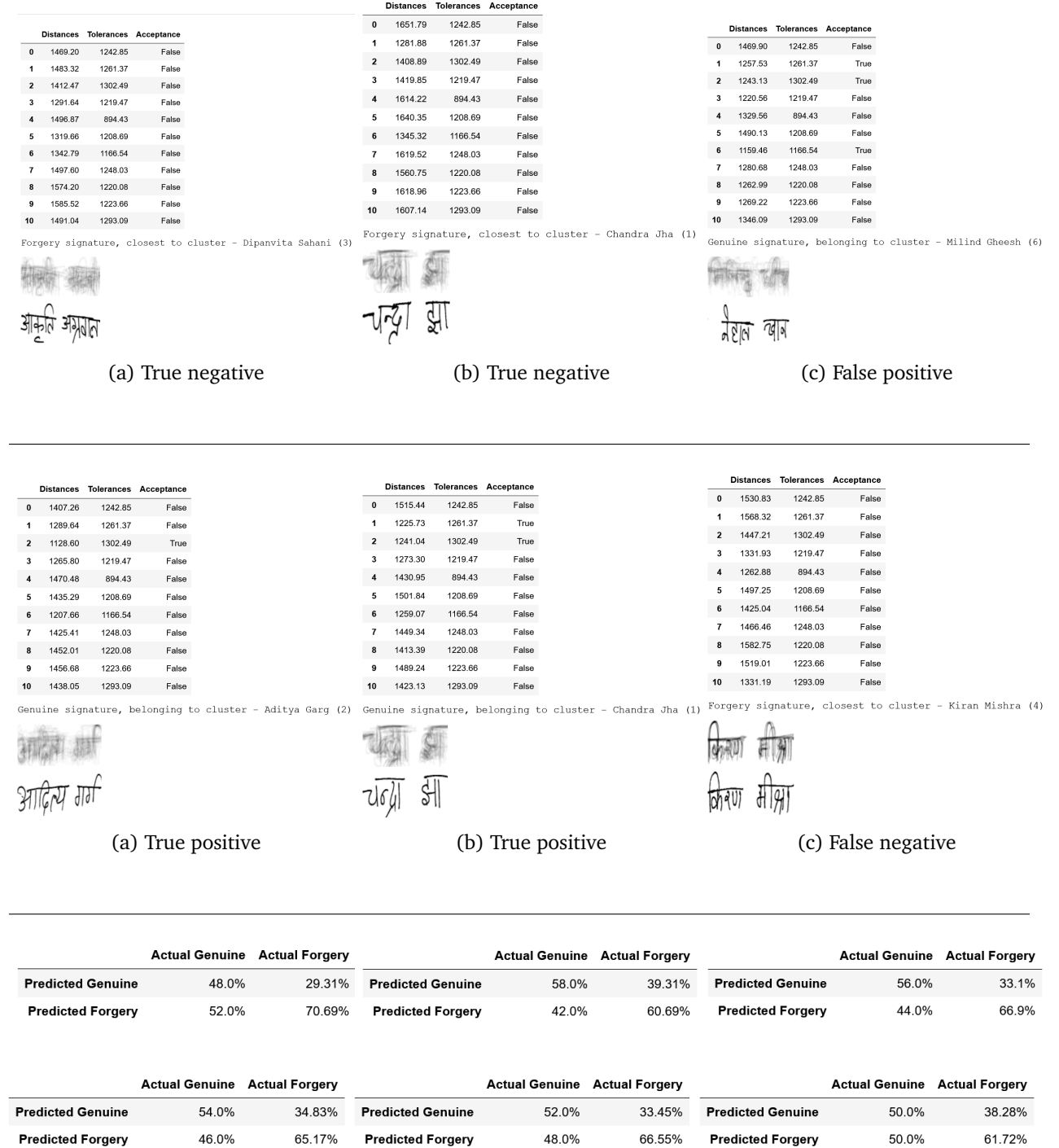


Figure 11: Confusion matrix for 6 trials

We have taken 50 (10 individuals, 5 per person) genuine samples and 300 (10 individuals, 30 per person) forgery samples. Using the tolerance method defined previously, we construct a confusion matrix to gauge the accuracy of our detection system. This gives varied results as the clusters formed are different each time.

There is a general trend of ≈ 50 -50 split of prediction of the true genuine samples. This might be because of the small sample size we considered due to limited data available. This is akin to a coin toss, and our classification of true genuines has a high margin of error. On the other hand, there is a 60-70% true negative rate of predicting the actual forgeries. This is a more reliable estimate as the number of samples used is larger. So we can conclude that our detection system has some merit, but can be vastly improved. This is discussed in the next section.

5 Limitations and Improvements

- The optimal number of clusters, K , could not be found accurately using the elbow method in our case. Using a better method, like Silhouette Scoring to determine optimal K can improve the results. However, this is computational intensive and we were not able to implement it.
- The number of data points in each cluster is different and the initial centroids are picked randomly which may be one of the factors responsible for moderate results. As a result, using K-means++ which tries to spread the initial centroids evenly might give us a better result.
- Many clusters had unequal number of points in them, leading to poor results. To fix the number of points in each cluster, we can get good initial estimates of the cluster centroids using fuzzy K-means and use global nearest neighbour assignment to associate the points with the cluster centroid. We can then calculate the distance matrix between each point and each cluster centroid, replicate each cluster centroid X times, and solve the linear assignment problem. Once done, we will get exactly X matches to data points for each cluster centroid, so that, globally, the distance between data points and centroids is minimised. Following which we can run the regular K-means.
- For determining the tolerance threshold for classification, we made an assumption that the points have been sampled from a gaussian distribution of distances from the centroid. However, this is not a well-founded assumption and better estimates can be made if no distribution is assumed apriori.

6 Data availability

Our code can be found in the py file (and html file) attached with this pdf. The dataset can be found in the following github link:

<https://github.com/hlamba28/Offline-Signature-Verification-using-Siamese-Network>.