MULTI-CLASS CLASSIFICATION OF HANDWRITTEN DIGITS USING SUPERVISED METHOD

Lecturer: Dr. Ndeye Niang Keita

Group 1

February 19, 2022

Group Members

Ibrahim Mbombo

Nana Phalonne

Alice Wachira

Osanyindoro Victor

Introduction

Discriminant analysis is a versatile statistical method often used by market researchers to classify observations into two or more groups or categories. In other words, discriminant analysis is used to assign objects to one group among a number of known groups we have.
Given the importance of these concepts in machine learning, we have been tasked to conduct a study on a given dataset using **Discriminant Analysis and K- Nearest Neighbours**.

About the Dataset

The dataset is a multi-feature digit dataset. It consists of features of handwritten numerals (0–9) extracted from a collection of Dutch utility maps.
A total of 2,000 patterns (200 patterns per class) have been digitized in binary images. These digits are represented in terms of the following six feature sets (files):

- mfeat-fou: 76 Fourier coefficients of the character shapes
- mfeat-fac: 216 profile correlations
- mfeat-kar: 64 Karhunen-Love coefficients
- mfeat-pix: 240 pixel averages in 2 x 3 windows
- mfeat-zer: 47 Zernike moments
- mfeat-mor: 6 morphological features

Part one:
Exploratory Data Analysis

Shape of each Dataset

Dataset	Columns	Rows
mfeat_fac	216	2000
mfeat_fou	76	2000
mfeat_kar	64	2000
mfeat_pix	240	2000
mfeat_zer	216	2000

Table: Shape of each dataset

Data Processing

The six datasets were then uploaded on python and dataframes created and stored in the dataframe into a CSV file format using *"Dataframe.tocsv()"* Method.
Next we combine the 6 dataset into one using the *"pd.concat()"* in python

Summary Statistic

	fac1	fac2	fac3	fac4	fac5	fac6
count	2000.000000	2000.000000	2000.00000	2000.000000	2000.000000	2000.00000
mean	272.051500	322.725500	774.82450	754.339000	640.460500	684.47200
std	91.275454	109.426588	140.14157	109.173768	48.940117	85.07886
min	67.000000	81.000000	500.00000	543.000000	437.000000	439.00000
25%	208.000000	247.000000	656.75000	667.000000	607.000000	642.00000
50%	274.500000	324.000000	766.00000	736.000000	636.500000	676.00000
75%	338.000000	406.000000	879.00000	830.000000	667.000000	716.00000
max	515.000000	565.000000	1264.00000	1134.000000	823.000000	1047.00000

8 rows × 649 columns

Figure: Summary Statistic of the first 6 column of the combined dataset

Response Variable Creation

We created a response variable column called 'response var' which contains the classes (0 - 9). The first two hundred rows belongs to "0" class followed by the next 200 rows which belongs to "1" class and so on to the last 200 rows which belongs to class "9". This response variable is created to help us to performed the supervised learning method

Part Two:
Descriptive Analysis

Dimensionality Reduction (PCA)

The goal of principle component analysis (PCA) is to minimize the dimensionality of a data set with a lot of correlated variables while keeping as much variance as possible. This is accomplished by transforming to a new collection of variables called principal components (PCs), which are uncorrelated and maintain the majority of the variance inherent in all of the original variables.

Data Standardization

To perform PCA on our data, a standard PCA was required since there was a high variability in the data set. To scale the data set, we standardized the data using *"StandardScaler()"* in python.

	0	1	2	3	4	5	6	7	8	9	...
0	-1.907359	-0.792743	-1.740279	-0.745228	-0.683874	-0.440549	-1.537618	-1.275246	-1.390997	-0.747908	...
1	-1.655311	-1.185799	-1.197835	-1.313272	-1.133515	-0.228927	-1.097040	-1.275246	-1.503546	-1.237697	...
2	-1.721063	-1.661122	-1.319171	-1.368244	-1.705786	-0.675683	-0.656462	-1.683411	-1.390997	-1.482591	...
3	-1.995027	-1.834798	-1.055086	-0.571150	-0.683874	-0.499332	-1.713849	-1.683411	-1.278448	-0.992802	...
4	-1.260802	-1.423461	-0.669665	-0.809362	-1.092639	-0.217171	-1.008924	-1.683411	-1.616096	-1.237697	...
...	...	...	...	...	...	...	...	...	...	...	...
1995	0.350110	-0.737898	1.086141	-1.175842	-1.562718	-0.123117	0.489042	-0.186804	-1.728645	-0.503013	...
1996	0.569282	0.066495	1.029042	-0.378748	-0.642997	0.100261	0.753389	-0.322859	-0.828250	-0.013224	...
1997	0.711744	-0.216871	0.550834	-0.277966	-1.215268	-0.287711	0.224695	-0.594969	-0.603152	0.231670	...
1998	-0.274529	-0.564222	0.650758	-1.111708	-1.542280	-0.816764	0.400926	-0.458914	-1.503546	-1.237697	...
1999	0.908998	0.514396	0.657896	0.656558	-0.050288	0.629315	0.665273	0.221362	0.072144	-0.992802	...

2000 rows × 649 columns

Figure: The Standardize Dataset

Principal Component Analysis

After standardizing the data, we performed the first PCA using 649 features.
Using the Kaiser's rule, we observed that 77% of the variability is explained by 30 principal components. Hence we decided to use 77% as our threshold variation which represents most of the information of the features.

Variability of the Principal Components

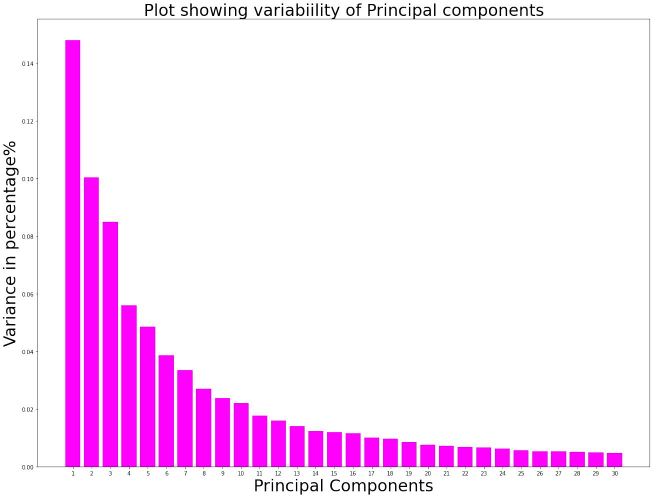

Figure: Variability Explained by the Principal Components

	0	1	2	3	4	5	6	7	8	9	...
0	-10.792587	-14.217229	-2.735840	-2.498813	-1.183472	7.075655	-2.578780	2.312856	-1.391230	1.417177	...
1	-8.030707	-13.084227	1.290484	-5.413479	-5.434129	7.035879	0.875896	2.192293	-3.160373	1.651192	...
2	-11.856732	-12.375021	0.499917	-0.850888	-4.471142	3.153493	-6.135362	6.616402	-4.272700	-1.366271	...
3	-8.377406	-13.201985	-1.619564	-2.736352	-2.866328	2.761106	-1.290269	6.484546	-3.559916	0.849158	...
4	-11.087021	-10.574100	-0.976119	-6.274910	-2.339075	7.777125	1.281137	2.372110	-2.397869	-3.129941	...
...	...	...	...	...	...	...	...	...	...	...	...
1995	1.769509	-8.680915	10.235653	2.825178	0.665071	-1.597781	-4.233299	-4.576576	2.839033	-9.605087	...
1996	9.287121	-8.811792	5.358049	0.754445	1.015373	2.737964	2.330241	0.712243	-4.391118	-4.845246	...
1997	3.091909	-7.749015	7.409622	7.716558	-4.053819	2.220747	2.002149	-8.093720	-0.343461	-5.732554	...
1998	0.014782	-8.775466	12.099608	1.365450	-4.051368	-1.909515	0.013557	-6.131533	-1.581514	-3.620881	...
1999	13.018259	-5.329831	0.880183	6.343388	-3.890295	2.526883	3.234955	-2.952298	-0.684106	-5.691923	...

2000 rows × 30 columns

Figure: The Principal Components

Hierarchical Agglomerative Method

Ward's method is an hierarchical method having a linkage function that specifies the distance between two clusters, such that the sum of squares between two clusters will increase when we merge them together. This method helps to determine the number of clusters to be used using dendongrams. In our case, it is evident that we can obtain 4 main clusters from the dendogram by cutting it across the dominating clusters.

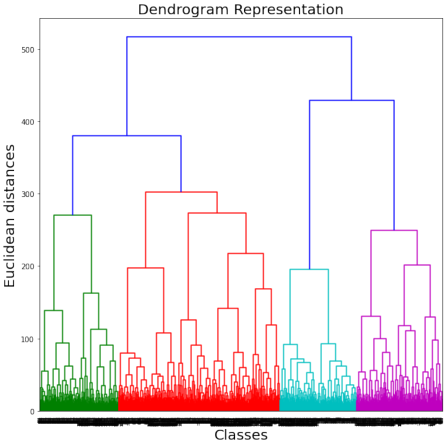

Figure: Dendogram representing the classification of clusters

Part Three:
Supervised Learning Methods

Discriminant Analysis

Linear Discriminant Analysis, or LDA for short, is a classification machine learning algorithm. It works by calculating summary statistics for the input features by class label, such as the mean and standard deviation. These statistics represent the model learned from the training data.
LDA assumes that the input variables are numeric and normally distributed and that they have the same variance (spread). If this is not the case, it may be desirable to transform the data to have a Gaussian distribution and standardize or normalize the data prior to modeling.

Steps for Discriminant Analysis

- To start, we first of all imported all the necessary libraries.
- We define our model using *"LinearDiscriminantAnalysis()"*
- We define model evaluation method using *"RepeatedStratifiedKFold()"*
- Finally, we evaluate our model and summarize the result.

Result of the Discriminant Analysis

```python
### Discriminant analysis

from sklearn.model_selection import RepeatedStratifiedKFold
from sklearn.model_selection import cross_val_score
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA

# define model
model = LDA()

# define model evaluation method
cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=15, random_state=1)

#evaluate model
scores = cross_val_score(model, X_train, y_train, scoring='accuracy', cv=cv, n_jobs=-1)
print("The accuracy for the Discriminant Analysis is : ",np.mean(scores))

# Predicting the Test set results
np.mean(X_test)
```

```
The accuracy for the Discriminant Analysis is :  0.98525
```

Figure: Result of the Discriminant Analysis

This figure show us the result of the LDA method and we can see that the accuracy is of 98.52%

K Nearest Neighbor(KNN)

Multi-class classification can be defined as the classifying instances into one of three or more classes. In this part, we are going to do multi-class classification using K Nearest Neighbours. KNN is a super simple algorithm, which assumes that similar things are in close proximity of each other. So if a datapoint is near to another datapoint, it assumes that they both belong to similar classes.

Steps for KNN

- To start, we first of all imported all the necessary libraries.
- Next, we split our input and output data into training and testing data. We choose a 80%-20% split for our training and testing.
- Training the KNN model on the Training set was next on the list, we used the *"KneighborsClassifier()"* and predict the test set results using *"classifier.predict(X test)"* function
- Finally, we make the confusion matrix, the classification report and the accuracy score to have the true and false prediction and the precision of prediction

Result of the KNN

From the figure below, The K Nearest Neighbor(KNN) has an accuracy score of 97.8%

```
Accuracy Score: 0.978
```

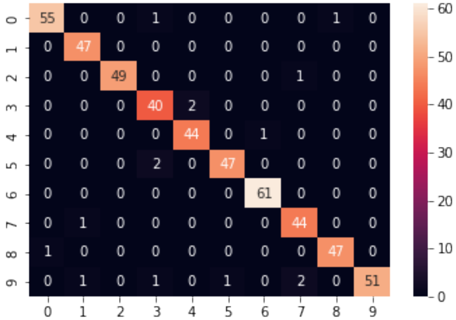

Figure: Heat map representing the prediction and actual value

Comparison

Supervised Method	Prediction Accuracy
Discriminant Analysis	98.52%
K Nearest Neighbor	97.8%

Table: Comparison of the two Supervised Method

From the table above, using the prediction accuracy, we can see that Linear Discriminant Analysis have a better accuracy than that of K nearest neighbour.

Conclusion

After all training with different methods, we can conclude that:

• We perform an hierarchical clustering on the PCA results and obtain 4 clusters at the end.

• The multi class Discriminant analysis algorithm was performed on the merged data set and we obtain an accuracy percentage of 98.52%.

• The K Nearest Neighbor algorithm was performed on the standardized PCA and we obtain an accuracy percentage of 97.8%.

• Then the multi class Discriminant analysis algorithm is performing better than the other method, therefore, it fits best for our objective.

Thanks for Listening!