Kinematic Features Final Project

**Sartaj Saadman**

**May 06, 2020**

# Introduction

The purpose of this study is to create a classification model based on a ‘labeled’ dataset and find out if it is reasonable to classify the ‘unlabeled’ dataset. A total of 16 models were created, four LDA and four K-NN models with PCA transformation, and four LDA and four K-NN models to classify all the dependent variables jointly and individually.

# Dataset

The ‘labeled’ dataset contains features from the MoVAlyzeR® system. The dataset consists of 103311 observations of 30 variables. The main difference between the ‘labeled’ and ‘unlabeled’ data is that, there are no character observation columns for each observation in the ‘unlabeled’ data set. For analysis, I am going to model the three dependent variables using 23 independent variables from the ‘labeled’ dataset. The dependent variables are:

1. Group. This is the writing style, which two classes, ‘Cursive’ and ‘Print’.
2. Subject. This variable consists of all the writers, having 40 different classes.
3. Condition. The phrase written, which has 6 classes.

Now, lets take a look at the independent variables used for analysis.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Variable | Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. |
| Direction | -0.11111 |  |  |  |  |  |
| Duration |  |  |  |  |  |  |
| VerticalSize |  |  |  |  |  |  |
| PeakVerticalVelocity |  |  |  |  |  |  |
| PeakVerticalAcceleration |  |  |  |  |  |  |
| HorizontalSize |  |  |  |  |  |  |
| StraightnessError |  |  |  |  |  |  |
| Slant |  |  |  |  |  |  |
| LoopSurface |  |  |  |  |  |  |
| RelativeInitialSlant |  |  |  |  |  |  |
| RelativeTimeToPeakVerticalVelocity |  |  |  |  |  |  |
| RelativePenDownDuration |  |  |  |  |  |  |
| AbsoluteSize |  |  |  |  |  |  |
| AverageAbsoluteVelocity |  |  |  |  |  |  |
| Roadlength |  |  |  |  |  |  |
| AbsoluteyJerk |  |  |  |  |  |  |
| NormalizedyJerk |  |  |  |  |  |  |
| AverageNormalizedyJerkPerTrial |  |  |  |  |  |  |
| AbsoluteJerk |  |  |  |  |  |  |
| NormalizedJerk |  |  |  |  |  |  |
| AverageNormalizedJerkPerTrial |  |  |  |  |  |  |
| NumberOfPeakAccelerationPoints |  |  |  |  |  |  |
| AveragePenPressure |  |  |  |  |  |  |

# Methodology

The first step was to transform the dataset by taking the mean of all the segments of each trial, reduced the dataset to 1440 observations. Out of the 1440 observations, 90% of the data was used as the training set for the models, while 10% was used as the test set.   
The goal was to create the best possible model to classify the ‘unlabeled’ dataset. In order to do so, I created 16 models in total, using LDA and K-NN classifiers with/without PCA transformation.

* **Principal component analysis (PCA)** is a technique for dimensionality reduction and minimizing information loss (Ian T. Jolliffe, 2016), which might be helpful in this scenario to achieve better accuracy using LDA and K-NN. PCA is an unsupervised learning method in that it finds patterns without reference to the prior knowledge about the dataset.
* **Linear discriminant analysis (LDA)** is a commonly used machine learning technique for predicting categories. LDA is a generalization of Fisher's linear discriminant, a method used to find a linear combination of features that characterizes or separates two or more classes of objects or events. Compared to logistic regression, linear discriminant analysis does a better job at classifying well separated or multiple classes (Brownlee, 2016), which is the case with the ‘labeled’ dataset.
* **K-nearest neighbors (k-NN)** is a supervised machine learning algorithm that can be used to solve classification problems. The K-NN algorithm assumes that similar things exist in close proximity (Harrison, 2018), which makes it an ideal model for classifying objects properly into their respective character class. For this project, Euclidean distance-based weighting for identifying the nearest k- neighbors was used.

The models were created using the train dataset and 10-fold cross validation. The independent variables were scaled when used in the model since it provided better accuracy compared to non-scaled data. For K-NN models, the accuracy rate reported is attained with the value of ‘k’ which yields the highest accuracy.

# Model Accuracy

The models constructed are shown below:

* Joint classification(**Group:Subject:Condition**) : Four models (LDA with and without PCA transformation, and K-NN with and without PCA transformation)
* Individual classification (**Group**): Four models (LDA with and without PCA transformation, and K-NN with and without PCA transformation)
* Individual classification (**Subject**): Four models (LDA with and without PCA transformation, and K-NN with and without PCA transformation)
* Individual classification (**Condition**): Four models (LDA with and without PCA transformation, and K-NN with and without PCA transformation)

**LDA with PCA**Below are the *model accuracies* of joint, group, subject and condition using the train set and 10-fold cross validation:

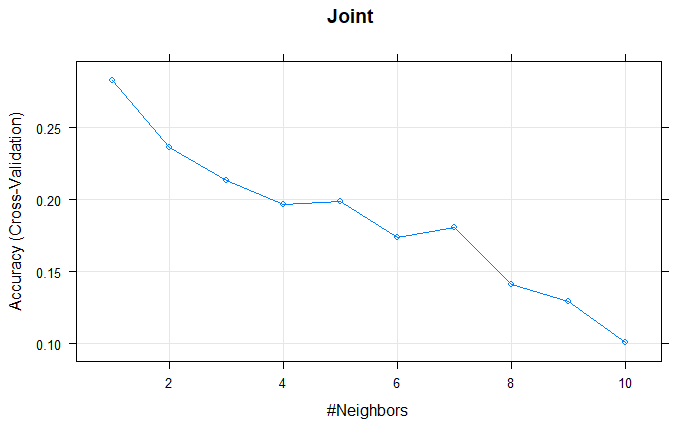
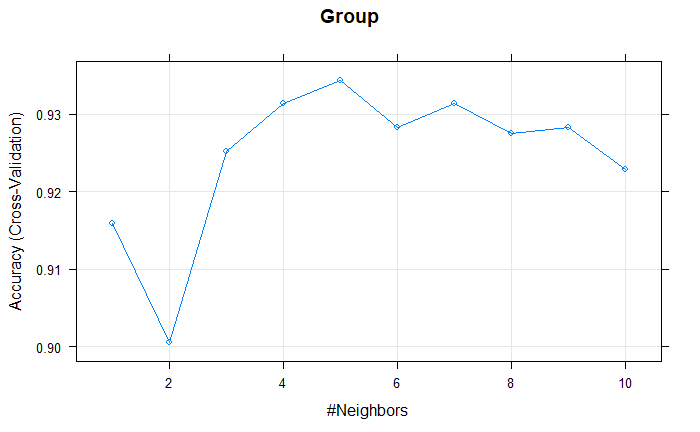
|  |  |  |  |
| --- | --- | --- | --- |
| Joint | Group | Subject | Condition |
| 0.3277916 | 0.9058607 | 0.5254351 | 0.3162893 |

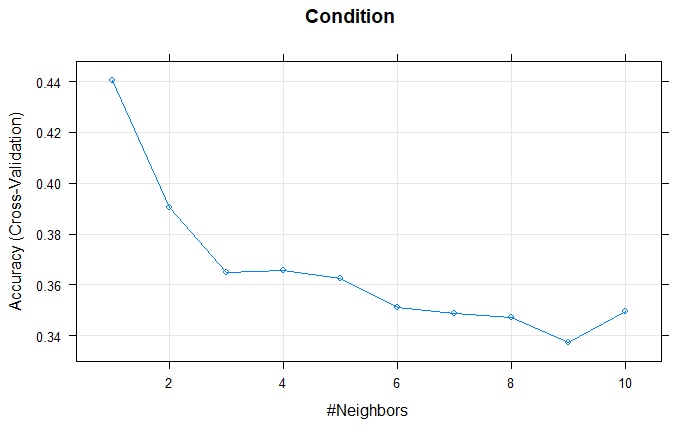
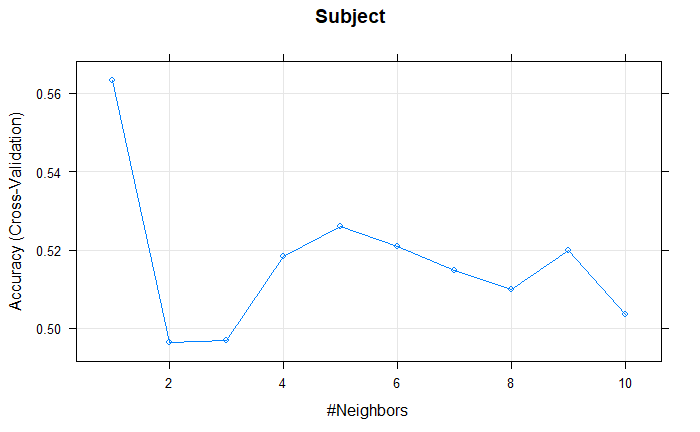
Now, coming to the *prediction accuracies* on the test set:

|  |  |  |  |
| --- | --- | --- | --- |
| Joint | Group | Subject | Condition |
| 0.2916667 | 0.8889 | 0.6042 | 0.3889 |

**K-NN with PCA**The *model accuracies* of joint, group, subject and condition using the train set and 10-fold cross validation:

|  |  |  |  |
| --- | --- | --- | --- |
| Joint | Group | Subject | Condition |
| 0.2831819 | 0.9343948 | 0.5633986 | 0.4405951 |

The value of k, which yields the highest accuracy for each model can be observed from the plots below.  
****

****

The *prediction accuracies* on the test set:

|  |  |  |  |
| --- | --- | --- | --- |
| Joint | Group | Subject | Condition |
| 0.2916667 | 0.9375 | 0.5833 | 0.4097 |

**LDA**The *model accuracies* of joint, group, subject and condition models constructed without PCA transformation, using the train set and 10-fold cross validation:

|  |  |  |  |
| --- | --- | --- | --- |
| Joint | Group | Subject | Condition |
| 0.5420244 | 0.9243879 | 0.6974875 | 0.3958013 |

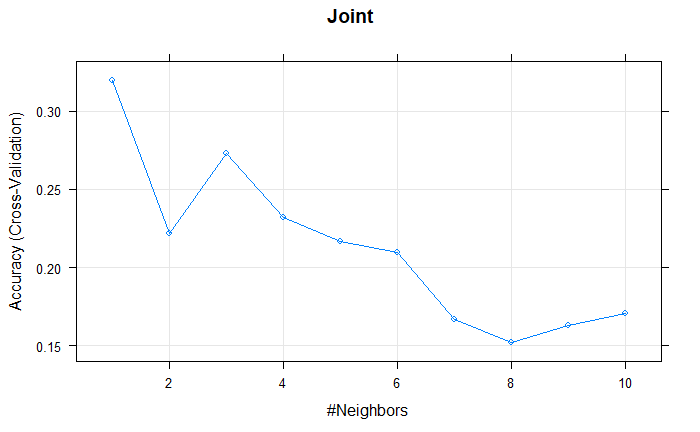
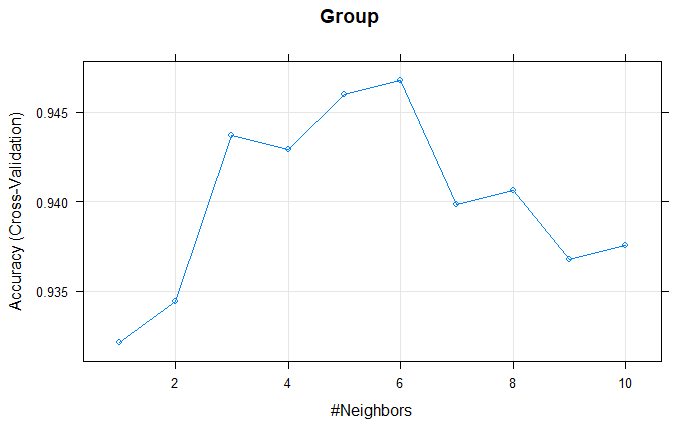
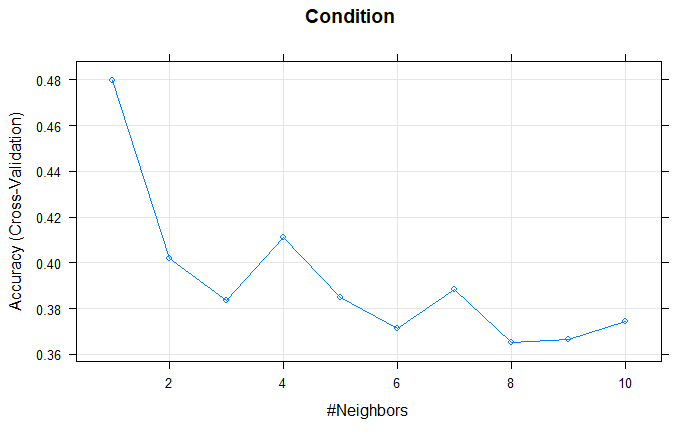
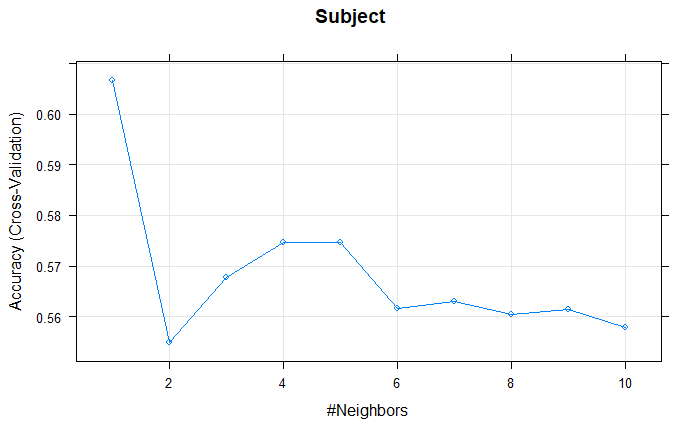
Now, coming to the *prediction accuracies* on the test set:

|  |  |  |  |
| --- | --- | --- | --- |
| Joint | Group | Subject | Condition |
| 0.4722222 | 0.9097 | 0.7292 | 0.375 |

**K-NN**Below are the *model accuracies* of joint, group, subject and condition models constructed without PCA transformation, using the train set and 10-fold cross validation:

|  |  |  |  |
| --- | --- | --- | --- |
| Joint | Group | Subject | Condition |
| 0.3199144 | 0.9467919 | 0.6067233 | 0.4799637 |

The value of k, which yields the highest accuracy for each model can be observed from the plots below.

**** ****Now, coming to the *prediction accuracies* on the test set:

|  |  |  |  |
| --- | --- | --- | --- |
| Joint | Group | Subject | Condition |
| 0.3541667 | 0.9306 | 0.6597 | 0.4792 |

# Model comparison

Now, coming to comparison of the models, let us break it down to which model is best in classifying the independent variables jointly or individually.

* Joint classification (**Group:Subject:Condition**): The model with best prediction accuracy when it comes to classifying all the independent variables together is **LDA without PCA** transformation. Its prediction accuracy (0.4722222) is significantly above the rest of the models (LDA with PCA (0.2916667), K-NN with PCA (0.2916667), K-NN without PCA (0.3541667)).
* Individual classification (**Group**): When it comes to classifying ‘Group’ individually, **K-NN with PCA** transformation has the highest prediction accuracy (0.9375) compared to the other models (LDA with PCA (0.8889), LDA without PCA (0.9097), K-NN without PCA (0.9306)).
* Individual classification (**Subject**): The model with best prediction accuracy when it comes to classifying ‘Subject’ individually is **LDA without PCA** transformation. Its prediction accuracy (0.7292) is the highest among the models (LDA with PCA (0.6042), K-NN with PCA (0.5833), K-NN without PCA (0.6597)).
* Individual classification (**Condition**): Finally, the model with the best prediction accuracy when it comes to classifying ‘Condition’, **K-NN without PCA** transformation is the best. Its accuracy (0.4792) compared to the other models (LDA with PCA (0.3889), LDA without PCA (0.375), K-NN with PCA (0.4097)) is the highest.

# Conclusion

The models show that it is quite reasonable to predict the writer, writing style and phrase from an unlabeled dataset, as the accuracy rates are reasonably decent. When it comes to predicting all the dependent variables together, LDA is more effective compared to K-NN, with PCA playing no part in it. On the other hand, K-NN is more useful when it comes to individually predicting the dependent variables. Given the dataset and its number of independent variables, PCA did not play a major role in achieving a better accuracy rate. If the goal is to predict the dependent variables together, the LDA model is the best with 47.2% accuracy. Similarly, if the goal is to predict the writer (subject), again the LDA model is the best with 72.9% accuracy. On the other hand, if the goal is to predict the writing style (group), K-NN model with PCA transformation has the highest accuracy of 93.7%, while for predicting the phrase (condition), K-NN model is the best with an accuracy rate of 47.9%.

# References

Brownlee, J. (2016, April 06). *Linear Discriminant Analysis for Machine Learning*. Retrieved from Machine Learning Mastery: https://machinelearningmastery.com/linear-discriminant-analysis-for-machine-learning/

Harrison, O. (2018, 09 10). *Machine Learning Basics with the K-Nearest Neighbors Algorithm*. Retrieved from Towards Data Science: https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761

Ian T. Jolliffe, J. C. (2016). Principal component analysis: a review and recent developments. *Royal Society*.