Pison Data Scientist Challenge
Technical Report

Kevin Egan

October 28, 2022

I started the process by downloading the data set and loading it into my Google Colaboratory notebook. To familiarize myself with the data further, I researched what each column represented and how they related to the sensor worn by the user. Although I found a significant amount of information regarding accelerometer and gyroscope data, creating new variables in the data set was unnecessary.

I initially checked to ensure that there were no missing values in the data frame before assessing the time evolution of each column. Although the sensor data provided timestamps in milliseconds, I found that there were lags at specific points in the data. I then cross-referenced the lags with each body movement and found that the lags occurred when the user switched from each body movement. Next, I reviewed the accelerometer, gyroscope, and quaternion data and found that each body movement was performed three times. This information later proved helpful in my analysis when trying to understand each identified feature.

Based on the data set, I knew unsupervised learning was necessary to determine the number of wrist-motions performed during each whole-body movement. Therefore, I applied k-means clustering to identify the number of features. k-means clustering originates from signal processing and partitions the data into k different clusters by placing a collection of data points aggregated by certain similarities. The number of clusters represents the number of centroids (centermost point in the cluster) necessary in the data set. The unsupervised learning algorithm assigns every data point to the nearest cluster and tries to keep the centroids as small as possible.

For clustering, I decided to focus on the accelerometer, gyroscope, and quaternion columns since they provide the most meaningful information for motion sensor data. To ensure that the clustering algorithm correctly generalizes the data points for each column so that the distance between each point will be lower, I scaled the data using the `RobustScaler` function of `sklearn` before clustering. As a popular scaling method from `sklearn`, the scaler removes the median and scales all columns of the data according to the quantile range. Rather than scaling each body movement separately, I scaled the entire data frame since the rate the user was moving did not significantly influence the data, and the wrist-motion readings were consistent.

After scaling the data, I used a `for-loop` to apply `sklearn`'s k-means clustering algorithm to several potential values of k and identified the optimal value of k that reduces the within-cluster sum of squares while generalizing the data. I reviewed the plot between the number of clusters and sum of squares to assess the 'elbow' in the plot, allowing me to determine the optimal number of clusters. After reviewing the cluster plot and using `kneed`'s `KneeLocator` function, I found that four was the optimal number of clusters, meaning there were four wrist-motion classes in the data.

Before performing classification, I developed a new data frame with the cluster labels as my target variables and the scaled data set as my independent variables. I then split the data into a 70/30 train/test split to develop my

training and test sets. By splitting the data, I trained the model on 70% of the data and saved 30% to assess how well the model predicted the labels in the test set. Although the data set contains time-series data, I decided that randomly splitting the data was acceptable in this situation because we are more focused on generalizing the data to understand wrist-motions rather than predicting the data forward in time.

I chose Gradient Boosting with `sklearn`'s `XGBoost` to build my classifier. Gradient boosting is a powerful tool that develops classification prediction models by generating an ensemble of weak prediction models using decision trees. The algorithm builds models sequentially and uses subsequent models to reduce the errors of the previous model. Upon performing `XGBoost`, the algorithm identified a prediction model that classified the data with 99.13% accuracy and a similar F1-score of 99%, representing the average between the precision and the recall values. These values show that the classifier can accurately determine what the user does during each whole-body movement because it identifies the high percentage of true positives out of all positive values (precision). At the same time, the recall results show the model's ability to identify relevant data since it is the ratio between true positives and true positives plus false negatives.

After building the classifier, I plotted the clusters by color along the scaled data to determine what each feature represented. By assessing the plots for each body movement, I could see a pattern for the features in the data. I returned to the information provided about the data set and tried to understand how the data represented the sensor's movements via acceleration and angular rotation. I also created a table providing the cluster centers to determine the most important points in the clusters because I assumed that would help determine what each feature represented. Using this information, I made assumptions about the angular velocity and rotations based on the positive and negative values in the table and plots. By assessing these values, I could better understand the potential wrist-motions based on the information provided.

From the plots and the table, I concluded that the four wrist-motions were:

1. Accelerating east, south, and up, while increasing in angular rotation east, north, and down.

2. Accelerating west, south, and down while increasing in angular rotation west, rapidly south, and slightly up.

3. Accelerating east, south, and down while increasing in angular rotation east, rapidly north, and up.

4. Accelerating west, south, and up while increasing in angular rotation west, south, and up.

Ultimately, I concluded that the four wrist-motion classes represented the user moving their wrist diagonally to the upper right, lower left, lower right, and upper left.

The code for this challenge is available online at `https://github.com/kevinegan31/Pison_DS_Challenge`.