CS260 Winter 2018
Machine Learning Algorithms
Dr. Yutao He
Due: March 25th, 2018

Project 2 – *BMLCO*: Solving Real ML Problems

Student Name: Jennifer MacDonald
SID: 604501712

Research Paper-Style Description of the Study

Introduction: With the advent of fitness-tracking devices like Fitbit, the ability to monitor, collect, and perform analysis on people's exercise behavior has increased. The Brunie ML Consulting (BLMCO) had been contracted out by the healthcare company *MyBody Inc*. to perform machine-learning and data analysis on the data set "Smartphone-Based Recognition of Human Activities and Postural Transitions Data Set" from the Urvine Machine Learning Repository. This dataset contains data from 30 participants from ages 19 to 48 years old, tracking their data on a wearable smartphone corresponding to six activities: standing, sitting, lying, walking, walking downstairs, and walking upstairs. Data on six kinds of postural transitions between these activities were also recorded. *MyBody Inc*. wanted BLMCO to train a model that could accurately predict which of the six activities was being performed based on the data set.

Methodology: The dataset contained training and testing sets for both the tracked data (X) and the results (y). The training set contained 7767 instances, or roughly 70% of the total instances collected, and the testing set contained 3162 instances (~30%). The X-datasets contained arrays of 561 measurements each, or the number of different attributes that were tracked with each activity. The models used were the classifiers instructed by the project: KNN classifier, which implemented the k-nearest neighbors vote; Naïve Bayes classifier, which applies Bayes theorem with assumptions of independence between the features; and Decision Trees classifier, which is a supervised learning method that uses data features to split data and create rules from those data splits. In addition, in preparation for this project, I also read "A Public Domain Dataset for Human Activity Recognition Using Smartphones", which analyzed the same dataset using a multiclass SVM generalized through a One-Vs-All (OVA) approach with a 10-fold Cross Validation procedure. I implemented this same model in addition to the three listed above because I was curious to see if it would perform similarly to that of the data in the paper.

After loading and studying the dataset, I attempted to use feature selection to narrow down the 561 attributes measures to a smaller number of the most relevant features using variance threshold. However, the results using feature selection were not as good as when I did not use feature selection, so I left my experimental code in my project, but it was not included in my final results. After developing the models by creating the classifiers listed above, we trained the model using the X and y data, then validated and evaluated the models.

Experimental Results: The results from the study are as below. The measurements I took were for accuracy, which was simply the number of accurate predictions over the total numbers measured. I also checked for cross validation accuracy as well as checking for over- and under-fitting by performing a cross-validation score. All the results used the default method except for the linear SVM classifier, which used the 10-fold cross-validation specified in the research paper.

KNN Classifier

Accuracy: 88.6%
Cross Validation Accuracy, Check for Over/Underfitting: 87% (+/- 2%)
Confusion Matrix: see Table 1

```
[[487  46  53   0   0   0   1   0   0   0   2   1]
 [  1 419  51   4   2   1   3   0   0   0   4   0]
 [  8   6 316   0   0   0   0   0   0   0   1   0]
 [  0   0   0 431  59   2   1   0   0   0   0   0]
 [  0   0   0  73 495   2   0   0   0   0   1   0]
 [  0   0   0   0   0 540   0   0   0   0   1   0]
 [  0   0   0   0   0   0  18   0   0   0   2   0]
 [  0   0   0   0   0   0   0  10   0   0   0   0]
 [  0   0   0   0   0   0   0   0  29   0  15   1]
 [  0   0   0   0   0   0   0   0   0  22   0  13]
 [  0   0   0   0   0   0   0   0   3   1  23   2]
 [  0   0   0   0   0   0   0   0   0   2   0  10]]
```

Table 1: Confusion Matrix of the classification results on the test data using the KNN Classifier.

Naïve Bayes Classifier

Accuracy: 74.7%
Cross Validation Accuracy, Check for Over/Underfitting: 71% (+/- 7%)
Confusion Matrix: see Table 2

```
[[416   8  80   0   0   0   0   0   0   0   0   0]
 [ 38 442  83   0   0   0   1   0   0   0   0   0]
 [ 42  11 257   0   0   0   1   0   0   0   0   0]
 [  0   0   0 457 311  62   0   0   0   0   0   0]
 [  0   0   0  35 220   0   0   0   0   0   1   0]
 [  0   0   0   1   1 467   0   0   0   0   0   0]
 [  0   9   0   8  22   0  15   0   0   0   1   1]
 [  0   0   0   5   0   0   2   9   1   0   0   0]
 [  0   0   0   1   0   0   3   0  24   0  18   0]
 [  0   0   0   1   0  11   0   1   0  21   2  15]
 [  0   1   0   0   2   1   1   0   7   1  27   3]
 [  0   0   0   0   0   4   0   0   0   3   0   8]]
```

Table 2: Confusion Matrix of the classification results on the test data using the Naïve Bayes Classifier.

Decision Trees Classifier

Accuracy: 80.9%
Cross Validation Accuracy, Check for Over/Underfitting: 82% (+/- 2%)
Confusion Matrix: see Table 3

```
[[375  50  36   0   0   0   0   0   0   0   1   1]
 [109 354  72   0   0   2   1   0   2   0   2   0]
 [ 12  55 312   0   0   2   0   0   0   0   0   0]
 [  0   0   0 407  84   0   1   1   1   0   1   0]
 [  0   0   0  97 470   0   0   0   1   0   1   0]
 [  0   0   0   1   0 540   1   0   1   0   0   0]
 [  0   3   0   2   1   0  15   0   2   1   5   2]
 [  0   0   0   0   0   0   2   8   0   0   0   0]
 [  0   0   0   0   1   0   0   1  16   0   6   0]
 [  0   0   0   0   0   0   0   0   0  15   0  10]
 [  0   9   0   1   0   0   3   0   9   1  33   0]
 [  0   0   0   0   0   1   0   0   0   8   0  14]]
```

Table 3: Confusion Matrix of the classification results on the test data using the Decision Trees Classifier.

Linear SVM Classifier

Accuracy: 94.4%
Cross Validation Accuracy, Check for Over/Underfitting: 94% (+/- 1%)
Confusion Matrix: see Table 4

```
[[492  19  13   0   0   0   0   0   0   0   2   0]
 [  0 450  29   3   0   1   6   1   0   0   5   1]
 [  4   2 378   0   0   0   0   0   0   0   0   0]
 [  0   0   0 435  40   0   2   1   0   0   1   0]
 [  0   0   0  70 516   0   0   0   0   0   1   0]
 [  0   0   0   0   0 544   0   0   0   0   2   0]
 [  0   0   0   0   0   0  15   8   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0  20   1  11   0]
 [  0   0   0   0   0   0   0   0   0  23   0  20]
 [  0   0   0   0   0   0   0   0  12   0  27   3]
 [  0   0   0   0   0   0   0   0   0   1   0   3]]
```

Table 4: Confusion Matrix of the classification results on the test data using the Linear SVM Classifier.

Conclusions: From the results, I conclude that the best model to use out of the four is the linear SVM classifer. It has the highest accuracy and there is high cross validation. The confusion matrix produces a strong diagonal, which is indicative of an accurate performance. The most to least accurate performances were as follows: linear SVM, KNN, Decision Trees, and Naïve Bayes. The fact that the Naïve Bayes was the lowest suggests that there are dependent relationships between the attributes. However, more analysis could be done on these models to really look at possible overfitting, like the ROC curve and k-fold validation.

Design Review: I designed the experiment in typical ML design fashion. In summary, I first studied the dataset, attempted to use feature selection, developed the model, trained the model, and then validated and evaluated the results. I went back and performed the experiment again without the feature selection to see if it produced better results and it did, so I left the feature selection method out. The description of the study section goes into more detail.

What I Have Learned: For this type of experiment, linear SVM worked the best because it took advantage of the multi-class OVA, which gave more precision in the results. However, I learned that while you can narrow down the information to make good guesses one what models would work best, sometimes the best way to go about is testing. For example, I thought decision trees would perform better than KNN, but I suppose that the pre-selected number of groups helped the algorithm with the precision. Additionally, I learned that feature selection does not always lead to more accurate results. Because of the split between the training and the testing, it was necessary that the features selected in both were the same, which made using the variance threshold method difficult to do above 65%. Future testing could use a different method for feature selection.

Problems Encountered During Implementation & Workarounds: The project was a fairly easy one, so the only problem that I encountered was poor results when implementing feature selection. After tinkering around with different values and methods, it seemed that the best results were when I didn't use it at all, which I suppose can be considered my workaround. Future implementations of this experiment should delve in to other approaches to feature selection that I did not attempt.

Approach That I Came Up With: The approach that I came up with was to follow the typical ML workflow. I added an additional classifier to try because background reading for this experiment yielded a paper that used the same dataset but with a linear SVM multi-class classifier. I achieved the highest accuracy out of all of my models at 94.4%, but the results in the paper got up to 96%. I suspect that they used additional methods, parameters, and a Gaussian kernel to achieve those results.

Most Important Aspects: The most important aspects of this experiment for me was learning about how the error measurements worked with the classifiers that we learned about in class. Actually writing the programs made it clear of the relationship between everything.

Where I Spent the Most Time: I spent the most time on trying out different feature selections and how it affected the results of my data. Ultimately, the results without any sort of feature selection we're the best, so I just commented out that section for my final submission. It wasn't really struggling, but more testing to see how the data would react to different strategies.

Suggestions: I really liked the open-endedness of this experiment, although it would be better to get specifications on what is going to be graded so that I don't have to guess what exactly is supposed to go into the report and how much each section is worth. In addition, I could have spent more time on this assignment, but the due date made it difficult since I'm travelling and unavailable after the final, and I spent the bulk of time before the final (but after it was assigned that Tuesday) studying for the exam.