Learning Task

We were required to create a ML model based on our insights derived from the data analysis section in the first part of the assignment, to predict the activity variable ('gt').

The only constraint the model must stand by is:

" *Your model should get at least 40 percent accuracy when training on 70 percent of the data and testing on 30 percent of the data.* "

Based on insight 1 we found, showing the X, Y, Z values to have certain values for different activities, we decided to start off by building a model using only the X, Y, Z data as the feature data.

First, we needed to convert our DataFrame loaded from the JSON file to a DataFrame containing 2 columns, a 'feature' column, which contains a vector object and a 'label' column containing a DoubleType representing the label of the feature vector.

To achieve these data conversions, we used:

- StringIndexer – to convert the different activities to a numeric DoubleType label.
- VectorAssembler – to convert the X , Y, Z columns to a feature Vector obj in one column.

Second, we chose the model we wanted to use, Random Forest Classifier. The reason we chose to use this model is because we saw the model was achieving the best results in original paper referenced by the project, named: "Smart Devices are Different: Assessing and Mitigating Mobile Sensing Heterogeneities for Activity Recognition".

Third, we split the data into 70% - train data, 30% test data. Trained the model on the 70% train data and tested the model on the 30% test data. All that was left was to calculate the accuracy on the predictions of the test data as well as on the train data.

The results:

```
Calculating accuracy scores...
---
Accuracy score on test set: 42.87%
Accuracy score on train set: 42.78%
```

Achieving the required accuracy scores, on the training set and the test set, on the assignment which isn't an assignment in a ML course, was enough for us to move on to the next question. But we felt that we didn't utilize the 3 insights we found from section 1.

We decided to run the Random Forest model on a feature vector which will contain the information regarding: 1. Norm of XYZ. 2. User ID. 3. Arrival_Time - representing the time the measurement arrived to the sensing application.

In order to fulfil this, we needed to calculate the norm of XYZ, use the StringIndexer on the User column along with the OneHotEncoder on the output to eliminate any numeric meaning to the users number representation. Once we convert the features to a vector we are now working with a Sparse Vector object, also supported by the classifiers the ml.lib supply.

The results from the new feature, trying to use as much of the information found from the insights:

```
Calculating accuracy scores...
---
Accuracy score on test set: 47.35%
Accuracy score on train set: 47.16%
```

Although not a huge improvement, we were happy to see the insights helped direct us to a better model.

RandomForestClassifier hyperparameters choice

We decided to investigate the default “max depth”, “max Bins” and “num of trees” params used in the Random Forest Classifier, trying to further Improve our model’s accuracy. We ran a sequence of different variables, calculated the accuracy of each model, and reached the optimal choice of parameters:

```
# create random forest classifier
rfClassifier = RandomForestClassifier(numTrees=10, maxDepth=30, maxBins=70)
```

And by doing so, we achieved the following results:

```
Calculating accuracy scores...
---
Accuracy score on test set: 75.14%
Accuracy score on train set: 75.13%
```

We were very satisfied with this improvement.

The tools we acquired using sparks ML libraries, all in a distributed environment will for sure come in handy in the future.

Thank you for the opportunity to learn these tools.