APPLIED MACHINE LEARNING AND DATA MINING

Human Activity Recognition Assignment

Jose Salinas

Data set: Smartphone-Based Recognition of Human Activities and Postural Transitions Data Set

Tasks:

1. A model that can accurately predict the activity (moving, stationary and transitory) based on features. If you use multiple models, please report the accuracy (or other relevant score) for all models and motivate which model works best. Note that different models might work better for different activities. This motivates steps 2 and 3.

2. Feature Engineering: Identify the minimal set of features (you might need to create or estimate other features from the dataset) that can predict each activity. You might find that different features predict different activities.

3. Based on step 2, report which features can seperate the different activities. For example, which feature (and value) can seperate stationary and moving activities.

1. A model that can accurately predict the activity (moving, stationary and transitory) based on features. If you use multiple models, please report the accuracy (or another relevant score) for all models and motivate which model works best. Note that different models might work better for different activities. This motivates steps 2 and 3.

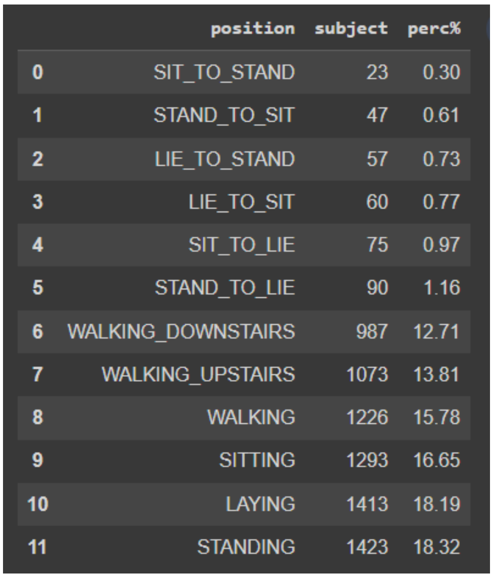
General Overview

At first, I took the job to see how the different models could adapt to the whole dataset. This was intended to see if it was truly necessary to separate the data frame into separate ones like stationary, moving, and transitory.

I cleaned the datasets looking for duplicates or missing values. Non were found.

In identifying the balance between the different activities, I noticed that there was a great bias between absolute movements(such as sitting, standing, and laying)and transitioning movements such as sitting to standing or standing to sit.

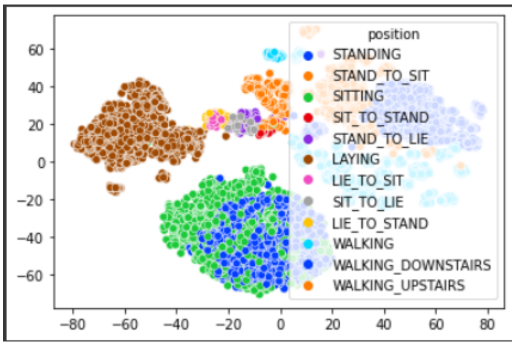
These were the results:



	position	subject	perc%
0	SIT_TO_STAND	23	0.30
1	STAND_TO_SIT	47	0.61
2	LIE_TO_STAND	57	0.73
3	LIE_TO_SIT	60	0.77
4	SIT_TO_LIE	75	0.97
5	STAND_TO_LIE	90	1.16
6	WALKING_DOWNSTAIRS	987	12.71
7	WALKING_UPSTAIRS	1073	13.81
8	WALKING	1226	15.78
9	SITTING	1293	16.65
10	LAYING	1413	18.19
11	STANDING	1423	18.32

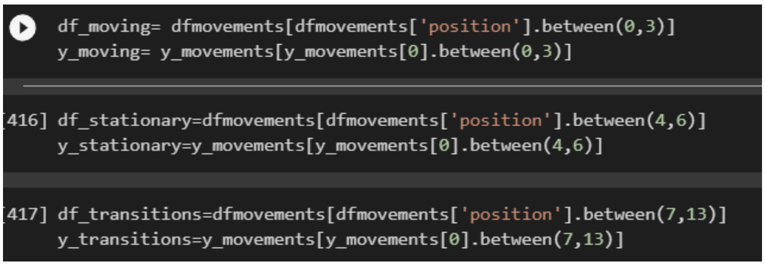
These may have different explanations, and maybe is easier to collect data on the absolutes than on the rest of the activities.

I also wanted to see how well distributed the data frame was, and how easy would be to separate between different activities. As seen in the figure below the data frame seems well distributed, but some activities such as standing or sitting may be confused by models.



I separated the data frame in the next way:

Separating the dataframe



```
df_moving= dfmovements[dfmovements['position'].between(0,3)]
y_moving= y_movements[y_movements[0].between(0,3)]

[416] df_stationary=dfmovements[dfmovements['position'].between(4,6)]
      y_stationary=y_movements[y_movements[0].between(4,6)]

[417] df_transitions=dfmovements[dfmovements['position'].between(7,13)]
      y_transitions=y_movements[y_movements[0].between(7,13)]
```

The first three activities are **walking, walking upstairs, and walking downstairs**. These activities I considered as moving.

The second three activities are **sitting, standing, and standing.** These activities I considered these as stationary activities.

The next group is activities from **stand to sit** until **lie to stand.** I thought of these activities as transitional activities.

Model Selection

The models used were KNeighborsClassifier, LogisticRegression, and SVM.

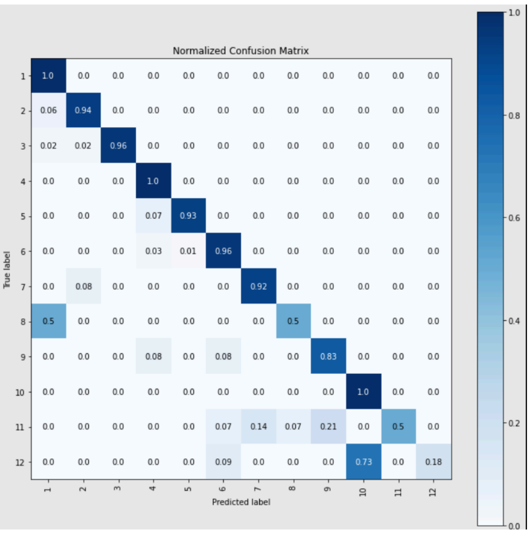
General Data frame

For the general data frame(all 12 variables), the best model for the data is KNeighborsClassifier.

The results were as follows:

KNeighborsClassifier

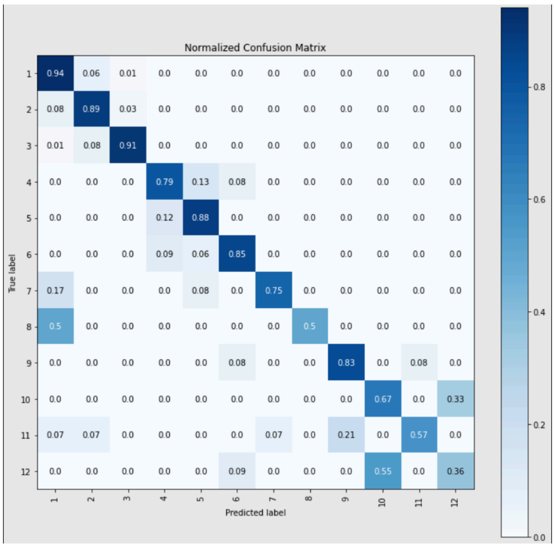
- Accuracy: 0.953024453024453
- Weighted F1-Score: 0.9512510607895533
- Balanced Accuracy Score: 0.8095222503342289
- Kappa Score: 0.9441778757555644



	1	2	3	4	5	6	7	8	9	10	11	12
1	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.06	0.94	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.02	0.02	0.96	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	0.0	0.0	0.0	0.07	0.93	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.03	0.01	0.96	0.0	0.0	0.0	0.0	0.0	0.0
7	0.0	0.08	0.0	0.0	0.0	0.0	0.92	0.0	0.0	0.0	0.0	0.0
8	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0
9	0.0	0.0	0.0	0.08	0.0	0.08	0.0	0.0	0.83	0.0	0.0	0.0
10	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
11	0.0	0.0	0.0	0.0	0.0	0.07	0.14	0.07	0.21	0.0	0.5	0.0
12	0.0	0.0	0.0	0.0	0.0	0.09	0.0	0.0	0.0	0.73	0.0	0.18

The second one was SVM which results were as follows

- Accuracy: 0.861003861003861
- Weighted F1-Score: 0.8606865758617115
- Balanced Accuracy Score: 0.7444006490666292
- Kappa Score: 0.8348631165070467



Conclusions on the General Data Frame

As explained on the balance review we could see that the transitional activities were not well enough represented. Therefore the model would not work as smoothly in these activities, given this, I noticed the reasoning for choosing different features to explain these specific activities.

Choosing Model for the “Moving” Data Frame

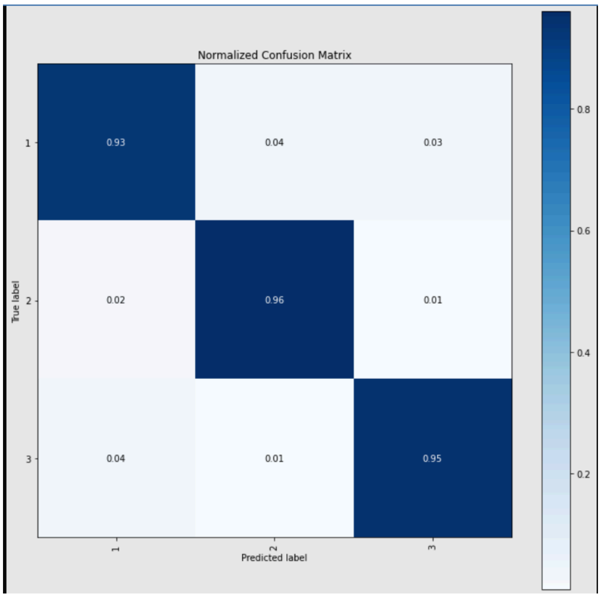
As mentioned before, the activities that compose this data frame are **walking, walking upstairs, and walking downstairs.**

Again the models I used to analyze are KNeighborsClassifier, LogisticRegression, and SVM.

The results are:

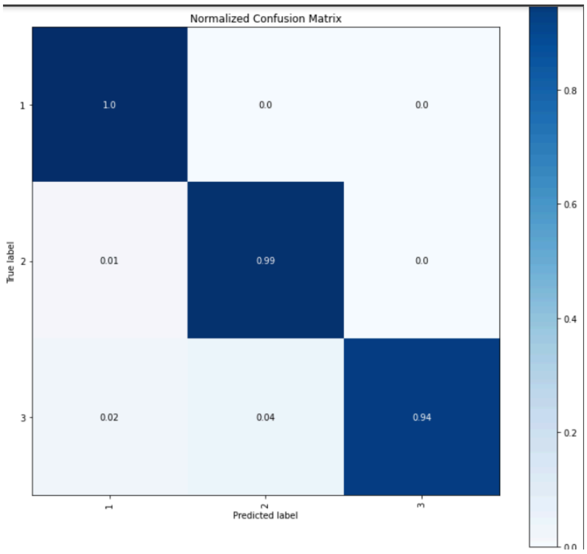
LogisticRegression

- Accuracy: 0.9498480243161094
- Weighted F1-Score: 0.9498113244995644
- Balanced Accuracy Score: 0.9497559300190878
- Kappa Score: 0.9247087378640777

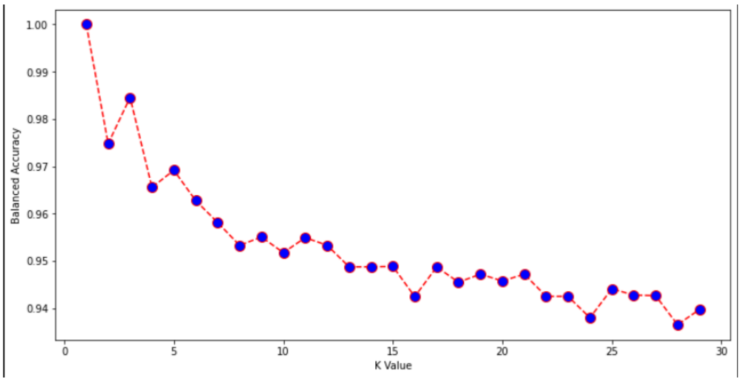


KNeighborsClassifier

- Accuracy: 0.9756838905775076
- Weighted F1-Score: 0.975563483235925
- Balanced Accuracy Score: 0.974780701754386
- Kappa Score: 0.963472347512317

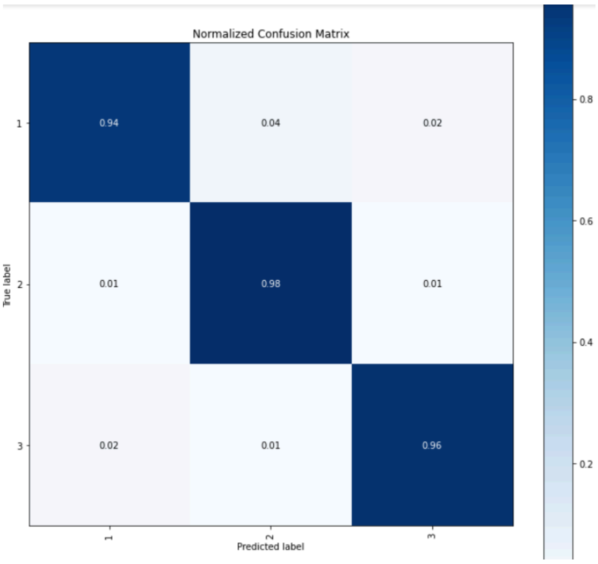


The model works properly, since a good number k would be for instance 8, we could get an accuracy over 96% as shown below.



SVM

- Accuracy: 0.9620060790273556
- Weighted F1-Score: 0.9619561220486202
- Balanced Accuracy Score: 0.9618120144435934
- Kappa Score: 0.9429524615405956



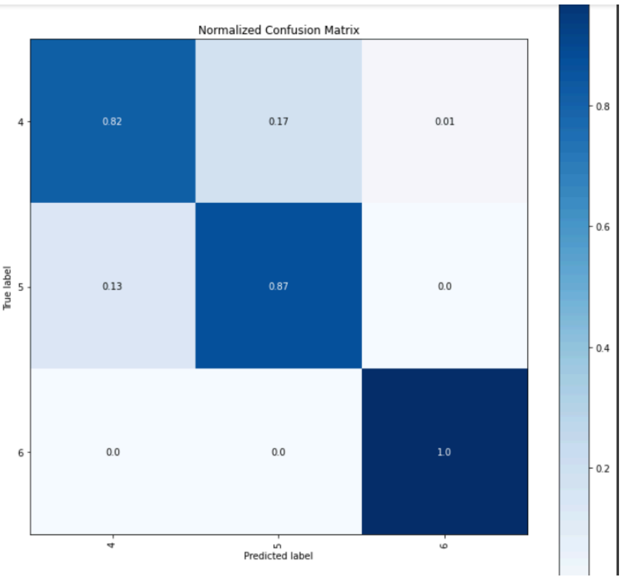
In conclusion the best model for the moving activities is Kneighbors classifiers with a score of almost 96% accuracy

Choosing Model for the “Stationary” Data Frame

Again using the same types of models the results are as follows:

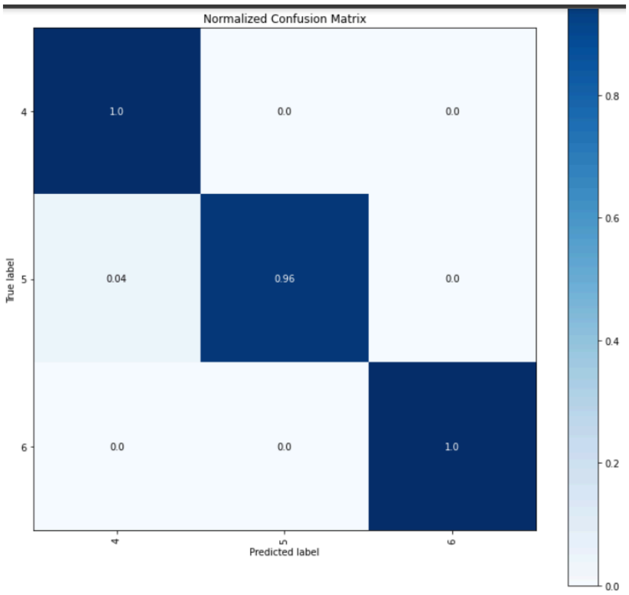
Logistical Regression

- Accuracy: 0.8983050847457628
- Weighted F1-Score: 0.8980024823037477
- Balanced Accuracy Score: 0.8972302754652409
- Kappa Score: 0.8472740719870395

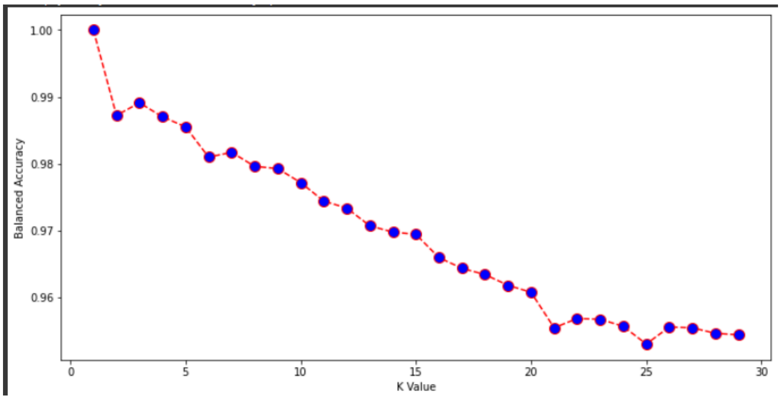


KNeighborsClassifier

- Accuracy: 0.986682808716707
- Weighted F1-Score: 0.986689617526181
- Balanced Accuracy Score: 0.9872241579558653
- Kappa Score: 0.9800223830223237

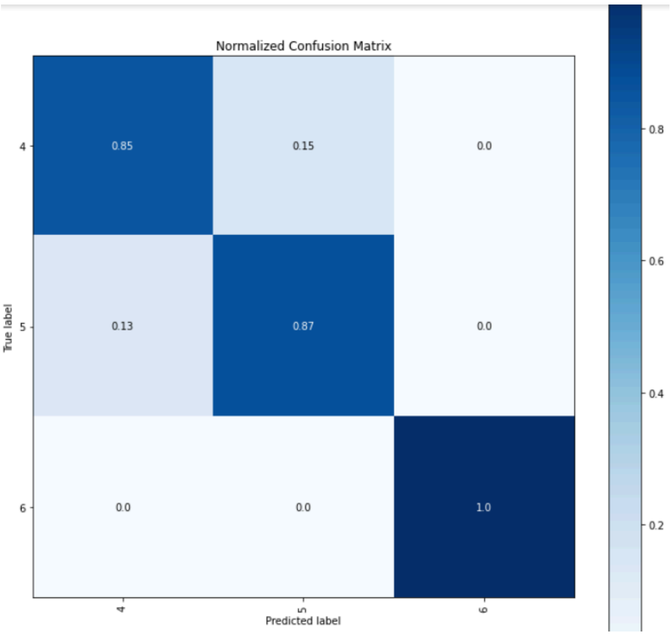


The model works properly, since a good number k would be for instance 10, we could get an accuracy over 97.5%



SVM

- Accuracy: 0.9067796610169492
- Weighted F1-Score: 0.9067536128943394
- Balanced Accuracy Score: 0.9061361278825437
- Kappa Score: 0.8600489813162464



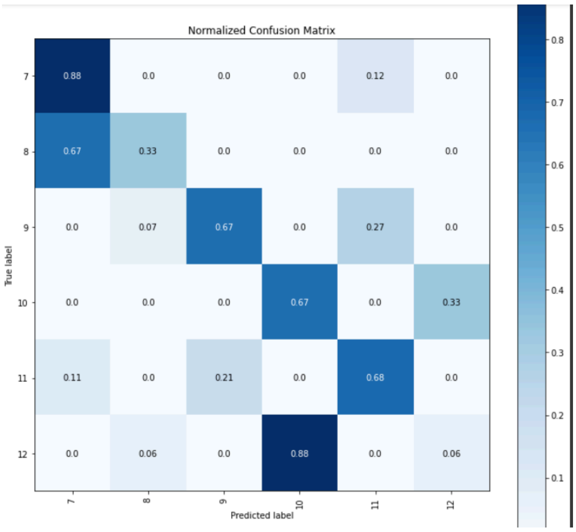
Conclusion

After analyzing the three accuracy scores and Kappa scores, I determined that without a shadow of a doubt, the best model for the stationary data frame is Kneighbors classification.

Choosing Model for the “Transitory” Data Frame

Logistical Regression

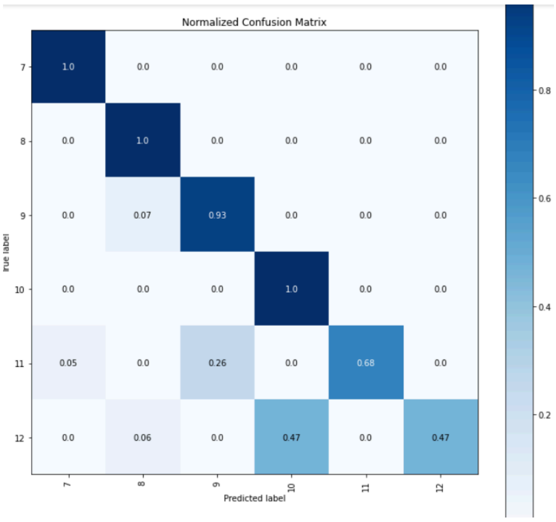
- Accuracy: 0.5352112676056338
- Weighted F1-Score: 0.5043656816719878
- Balanced Accuracy Score: 0.5474501203990367
- Kappa Score: 0.43337363966142683



This model worked poorly to say the least.

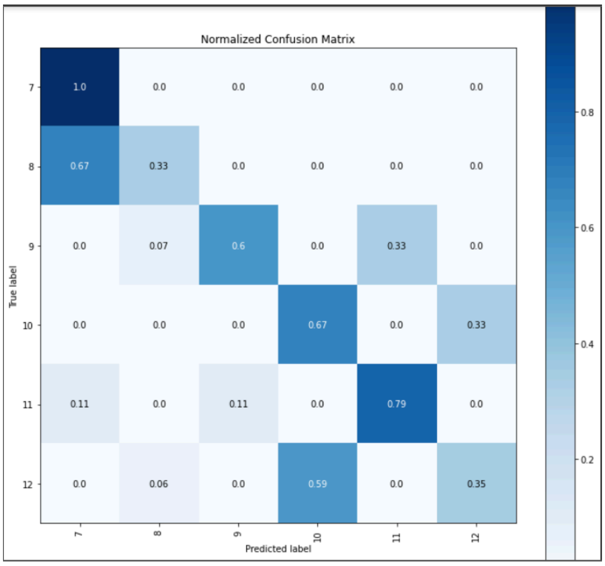
KNeighborsClassifier

- Accuracy: 0.7746478873239436
- Weighted F1-Score: 0.7701496080555732
- Balanced Accuracy Score: 0.8480220158238735
- Kappa Score: 0.7251391241229131



SVM

- Accuracy: 0.6338028169014085
- Weighted F1-Score: 0.6276923076923077
- Balanced Accuracy Score: 0.6237358101135191
- Kappa Score: 0.5490962383976551



Conclusion

After reviewing the models it is obvious that the best one is KneighborsClassifier, nonetheless. It has a low accuracy score when analyzing the last two activities. It was already mentioned that this might happen because of the big bias that the original df had in the different activities.

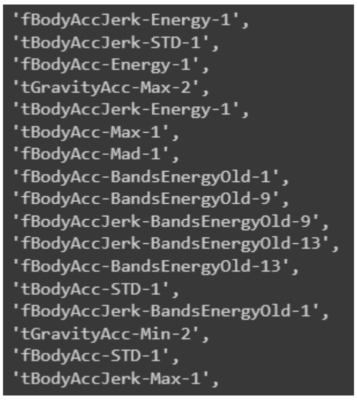
2. Feature Engineering: Identify the minimal set of features (you might need to create or estimate other features from the dataset) that can predict each activity. You might find that different features predict different activities.

The method used for feature selection was ANOVA, given that the data frame lacked many categorical variables that could make correlations easier. Besides having an enormous amount of features that made it hard to understand where could there be a correlation between variables.

For choosing the minimal set of features I considered it based on the information gained on them. This was subjective which made me hesitant if it was the right way to approach the problem. Nonetheless not knowing another way I chose this way to answer the question.

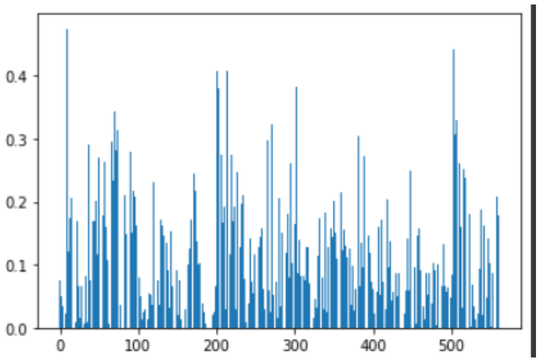
For the first group, I noticed that there were low scores on information gain, which is why the standard for selecting the features was set at 40% information gained. From this criteria, we selected 17 features

The resulting features are:



```
'fBodyAccJerk-Energy-1',
'tBodyAccJerk-STD-1',
'fBodyAcc-Energy-1',
'tGravityAcc-Max-2',
'tBodyAccJerk-Energy-1',
'tBodyAcc-Max-1',
'fBodyAcc-Mad-1',
'fBodyAcc-BandsEnergyOld-1',
'fBodyAcc-BandsEnergyOld-9',
'fBodyAccJerk-BandsEnergyOld-9',
'fBodyAccJerk-BandsEnergyOld-13',
'fBodyAcc-BandsEnergyOld-13',
'tBodyAcc-STD-1',
'fBodyAccJerk-BandsEnergyOld-1',
'tGravityAcc-Min-2',
'fBodyAcc-STD-1',
'tBodyAccJerk-Max-1',
```

On the “Moving” data frame

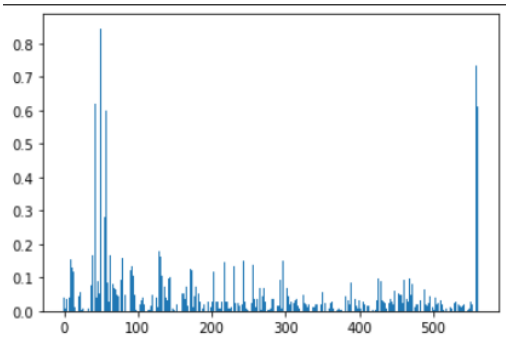


As you can see there are very few amounts of variables that have a piece of information gained bigger than 40%. But not wanting to lower the standard we used 17 features to explain the moving category.

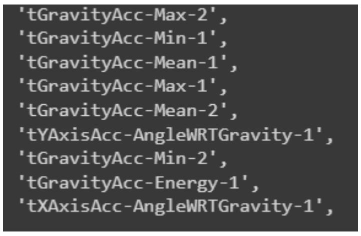
On the “Stationary” data frame

On this selection, I was a little more strict on the feature selection based on the argument that there was more recollection of data in this one. Besides the fact that these features have greater information gained scores. The requisite to get the feature list was 70% information gained.

The results were:



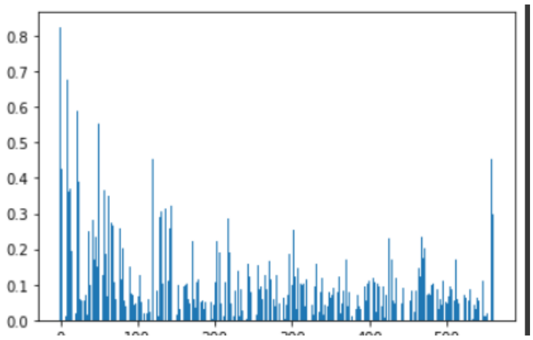
After these criteria, I selected 9 variables that explained the data frame. These were:



```
'tGravityAcc-Max-2',
'tGravityAcc-Min-1',
'tGravityAcc-Mean-1',
'tGravityAcc-Max-1',
'tGravityAcc-Mean-2',
'tYAxisAcc-AngleWRTGravity-1',
'tGravityAcc-Min-2',
'tGravityAcc-Energy-1',
'tXAxisAcc-AngleWRTGravity-1',
```

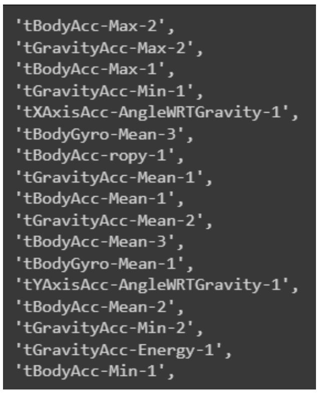
On the “transitionary” data frame

After viewing the performance of the model on the past data frame I decided to go back and accept an information score of almost 40% again. After this modification, I got interesting results



There were really few variables that could gain information better than 40%. This is clear because of the bias of the data recollection.

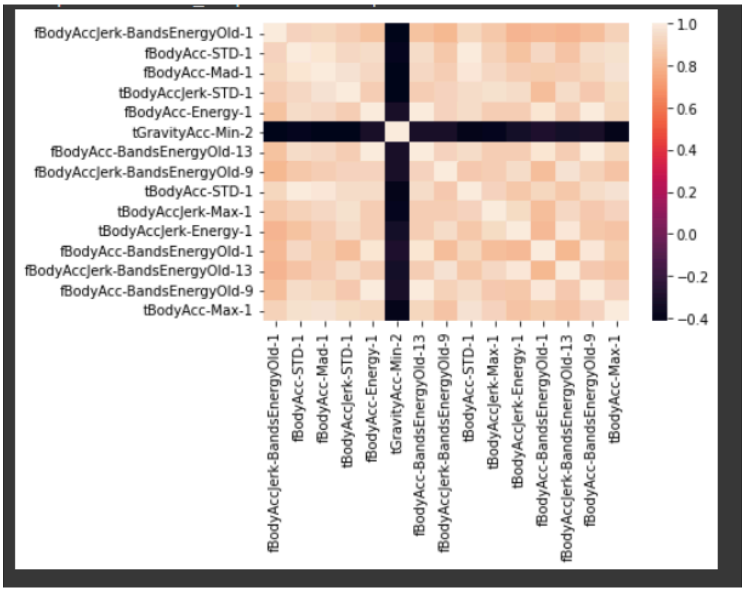
The features were 18:

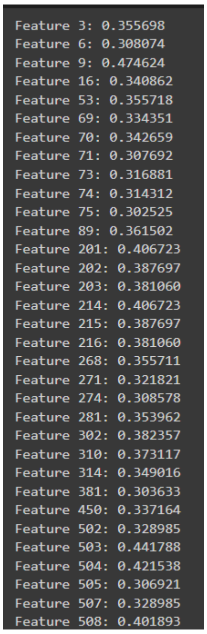


```
'tBodyAcc-Max-2',
'tGravityAcc-Max-2',
'tBodyAcc-Max-1',
'tGravityAcc-Min-1',
'tXAxisAcc-AngleWRTGravity-1',
'tBodyGyro-Mean-3',
'tBodyAcc-ropy-1',
'tGravityAcc-Mean-1',
'tBodyAcc-Mean-1',
'tGravityAcc-Mean-2',
'tBodyAcc-Mean-3',
'tBodyGyro-Mean-1',
'tYAxisAcc-AngleWRTGravity-1',
'tBodyAcc-Mean-2',
'tGravityAcc-Min-2',
'tGravityAcc-Energy-1',
'tBodyAcc-Min-1',
```

3. Based on step 2, report which features can seperate the different activities. For example, which feature (and value) can seperate stationary and moving activities.

“Moving”

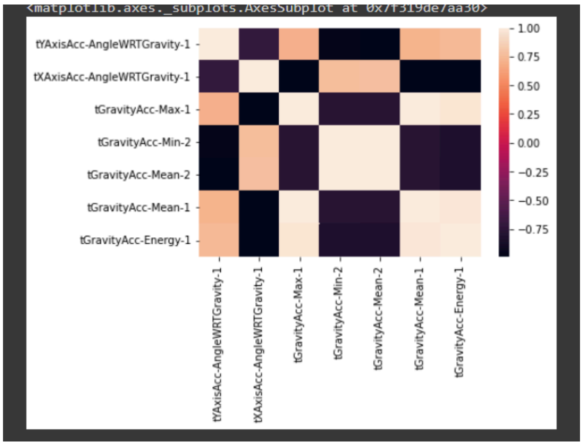




```
Feature 3: 0.355698
Feature 6: 0.308074
Feature 9: 0.474624
Feature 16: 0.340862
Feature 53: 0.355718
Feature 69: 0.334351
Feature 70: 0.342659
Feature 71: 0.307692
Feature 73: 0.316881
Feature 74: 0.314312
Feature 75: 0.302525
Feature 89: 0.361502
Feature 201: 0.406723
Feature 202: 0.387697
Feature 203: 0.381060
Feature 214: 0.406723
Feature 215: 0.387697
Feature 216: 0.381060
Feature 268: 0.355711
Feature 271: 0.321821
Feature 274: 0.308578
Feature 281: 0.353962
Feature 302: 0.382357
Feature 310: 0.373117
Feature 314: 0.349016
Feature 381: 0.303633
Feature 450: 0.337164
Feature 502: 0.328985
Feature 503: 0.441788
Feature 504: 0.421538
Feature 505: 0.306921
Feature 507: 0.328985
Feature 508: 0.401893
```

As seen here all variables work smoothly to separate the activities, I think this one is hard to pinpoint so I would say all variables work almost just as well between 0.8/1 .

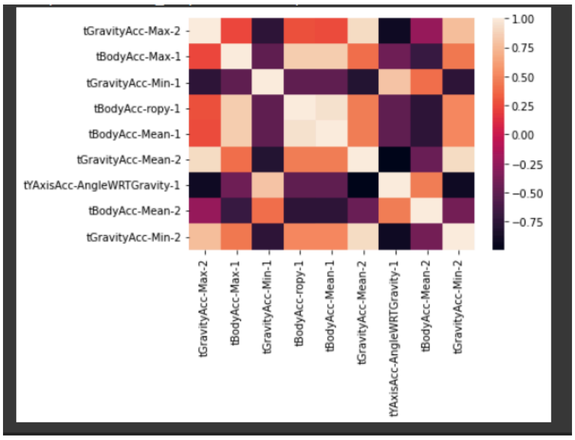
“Stationary”



```
<matplotlib.axes._subplots.AxesSubplot at 0x7f319de7aa30>
```

For this one, it was quite obvious that the one that explains best and separates properly are feature white and skin color.

“Transitory”



On this one again it is easy to identify which are the features that separate properly the data frame