Machine Learning Assignment 2
2020 Spring

2015143508
Sojeong Lee

1. Purpose of the Assignment

The purpose of the assignment is to practice the overall process of how the machine learning techniques could be applied to the real/generated datasets. To fulfill the purpose of the assignment, to try the knowledge that covered in the class as much as we can is highly recommended. Important steps for the assignment would be: choosing a suitable model for classification/regression problem, feature engineering, final evaluation (figuring out the accuracy).

2. Dataset & Goals

1) Dataset Explanation

The name of the chosen dataset is "Bar Crawl: Detecting Heavy Drinking Data Set"[i], collected in May 2017 by Jackson A Killian from Harvard University and Danielle R Madden and John Clapp from University of Southern California. Two different data, accelerometer and transdermal alcohol content data (TAC), were collected by thirteen volunteers from a college bar crawl event to predict heavy drinking episodes via mobile data.

- **Accelerometer data**: It was collected from mobile phones at a sampling rate of 40Hz. The five columns are: a timestamp, a participant ID, and a sample from each three axis of the accelerometer.
- **Transdermal Alcohol Content data (TAC)**: Cleaned TAC readings were used, which was processed raw TAC data with a zero-phase low-pass filter to smooth noise without shifting phase. The two columns are: a timestamp and the value of TAC.

2) Variables

- **Accelerometer data**

  There are five features in the accelerometer data:

  time: integer, unix timestamp, milliseconds

  pid: symbolic, 13 categories listed in pids.txt

  x: continuous, time-series

  y: continuous, time-series

  z: continuous, time-series

  According to the pearson correlation coefficient between the four (Time, x, y, z), it seems that time is independent to the other variables and x, y and z have an extremely slight correlation from each other since the accelerometer records(x, y, z) are regarding one person's moves at the exact time. However, since the value of the correlation coefficient is small, it is safe to call them independent.

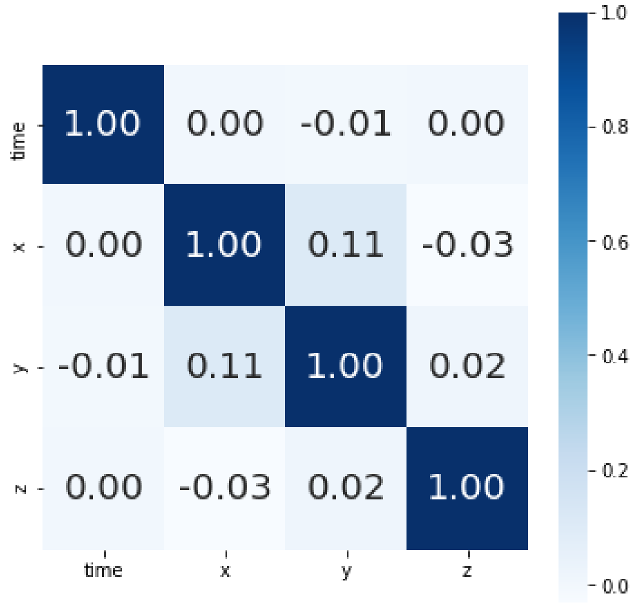


Figure 1. Correlation Coefficient between the variables of Accelerometer Data

- **Transdermal Alcohol Content data (TAC)**

  There are two features in TAC:

  timestamp: integer, unix timestamp, seconds

  TAC_Reading: continuous, time-series

According to the calculated pearson correlation coefficient between timestamp and TAC_Reading, it seems that they have an extremely slight positive correlation between the two. However, since the value is small, it is safe to say that they are independent variables.

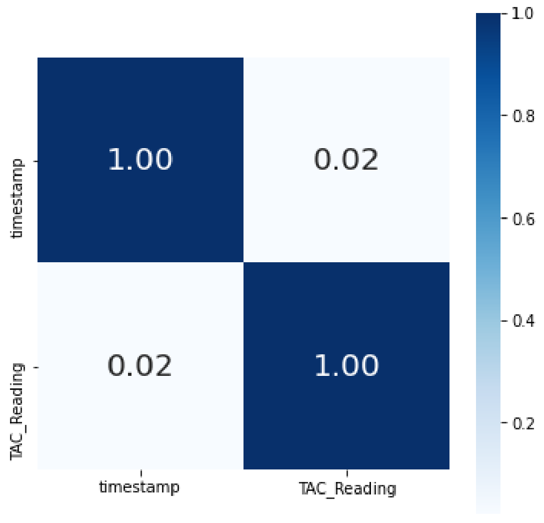


Figure 2. Correlation Coefficient between the variables of TAC Data

3) Goals

The goal is to classify sober (TAC < 0.08) vs. intoxicated (TAC >= 0.08) people using data. TAC values are in grams per deciliter where 0.08 is the legal limit for the suspension of the driver's license not only in the United States but a lot of different countries such as South Korea, Canada, Singapore and so on.

3. Overall Approach

1) Model Selection

Support Vector Machine (SVM) is selected to solve the problem. The reason for selecting SVM is that our main problem is to classify the movement data whether if the person is drunk or not. SVM has been a strong classifier since late 50s, which is also supported by scikit-learn libraries. Plus, some kernel tuning would be possibly applied to the machine.

2) Data Preprocessing

Since the data is recorded based on the timestamps, it could be considered as time series data that some preprocessing steps were needed. Before processing the data, I mapped the accelerometer data to the 3D space to see the dispersion.

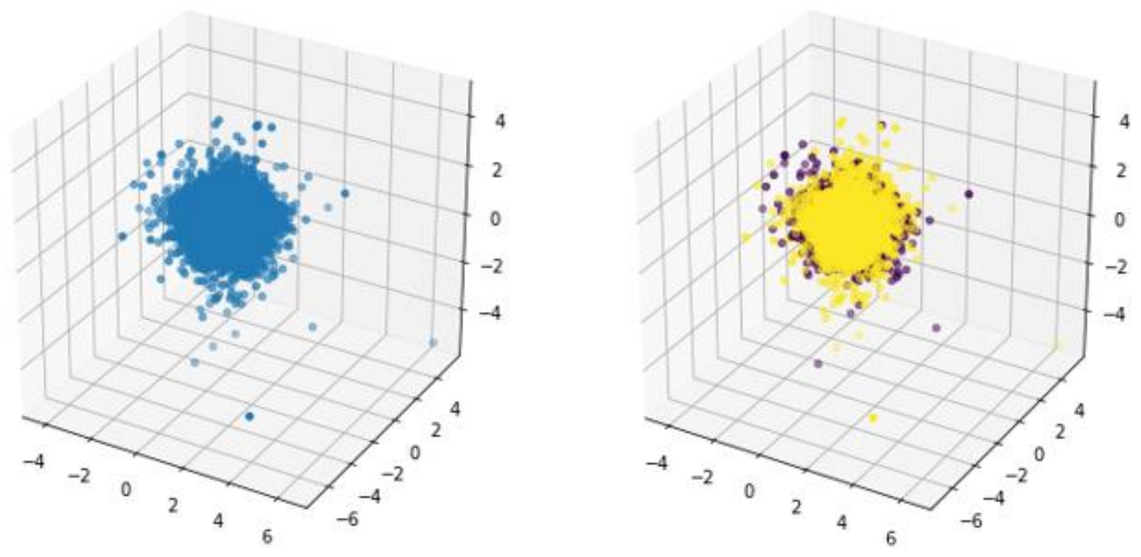


Figure 3. Before and After of labeling the accelerometer data (visualized in 3D space)

As we can see the left graph of Figure 3, the acceleration of each axis usually moves between -3 to 3 except for some values. Even the unusual values would not exceed 6.

I preprocessed the whole data by cutting down the accelerometer data every 30 minutes and compare it with TAC data so that we can label if the accelerometer data is collected while sober or intoxicated. The right graph of Figure 3 shows the labeled result of the data. The purple dots show the sober ones and the yellow dots show the intoxicated ones.

3) Preliminary Approach – Before Feature Engineering

Before doing feature engineering, the svm was applied based on the preprocessed datasets. The labeled accelerometer data was put to train svm model. (Since the size of the data was extremely big, it was cut down randomly.) The accuracy score of the model was 0.63.

4) Feature Engineering Ideas

Not only the TAC value matching but also the other features was calculated. It was mostly figured out based on the accelerometer location values.

- The average of raw data (accelerometer signals)
- Number of Recorded Time (how many steps did the walker stepped in 30 mins?) – It was calculated by counting the recorded time based on the 30min. timeframe.
- Rate of Change (How fast the walker changes its fastness?) – It was calculated by computing the total acceleration based on the accelerometer values
- Max accelerometer value (How fast and large the walker changes its direction?)

5) Feature Selection – Recursive Feature Elimination[ii]

The features above were applied to proceed next approach, which hopefully would lead the model to the better accuracy results. However, feature engineering was not very successful. Therefore, feature selection method, Recursive Feature Elimination was applied. It helped to pick out four main considered-to-be-effective features.

6) Final Approach – Scaling up the number of data

The selected features were applied to proceed final approach. However, even the feature selection was not very effective. Scaling up the number of data were the final approach to get the higher accuracy, and gladly, it worked.

4. Program Source Codes

1) Preprocessing Codes

```
def labeling(accTime):
  lowerIndex = bisect_left(tacTimestamps, accTime / 1000, hi=len(tacTim
estamps)-1)
  closest = BK.index.get_loc(min(tacTimestamps[lowerIndex], accTimestam
ps[lowerIndex+1],key=lambda x: abs(x - accTime / 1000)))
  if BK.iloc[closest]['TAC_Reading'] > 0.08:
    return True;
  else:
    return False;

BK7610['label'] = BK7610['time'].apply(labeling)
```

This is the main part of preprocessing source codes. (BK1670 is the accelerometer data of the smartphone whose pid is "BK7610".)

Since the gap of the timestamps are different between TAC(30min) and accelerometer data(1/40 sec), some process were needed to match the time gap. The source code above is to match the accelerometer data to the nearest TAC timestamp to figure out the TAC value at the certain time written in accelerometer data.

- function labeling: it gets accTime as an input. accTime is the timestamp of one accelerometer datum. The function carries out binary search to find out the closest TAC timestamp.

2) SVM training Source Code

```
from sklearn import model_selection
from sklearn import svm
from sklearn import utils

bklist = BK_for_svm.columns.tolist()
bklist.remove('label')
X = BK_for_svm.loc[:, bklist].values
Y = BK_for_svm.loc[:, ['label']].values
X_train, X_test, y_train, y_test = model_selection.train_test_split(X,
Y, test_size = 0.3, random_state = 1)
model = svm.SVC(kernel='linear', C=1.0, random_state=0).fit(X_train, y_
train)
y_pred = svm.predict(X_test)
print('Accuracy: %.2f' % accuracy_score(y_test, y_pred))
```

This is the main part of SVM training source codes.

- The first three lines are the codes to import scikit learn packages to train svm.
- bklist for the features (Features of the data are identified as column names, and they would be the value of X). 'label' feature would be removed because it is the target value, which would be the value of Y.
- The data were split into train set and test set – the portion of the test set is 30%, as we can see the value of test_size is 0.3

- the model was trained and fitted + accuracy would be calculated.

- while performing the final approach, the kernel would turned into ‘poly’, to get the best result

3) Feature Selection Code

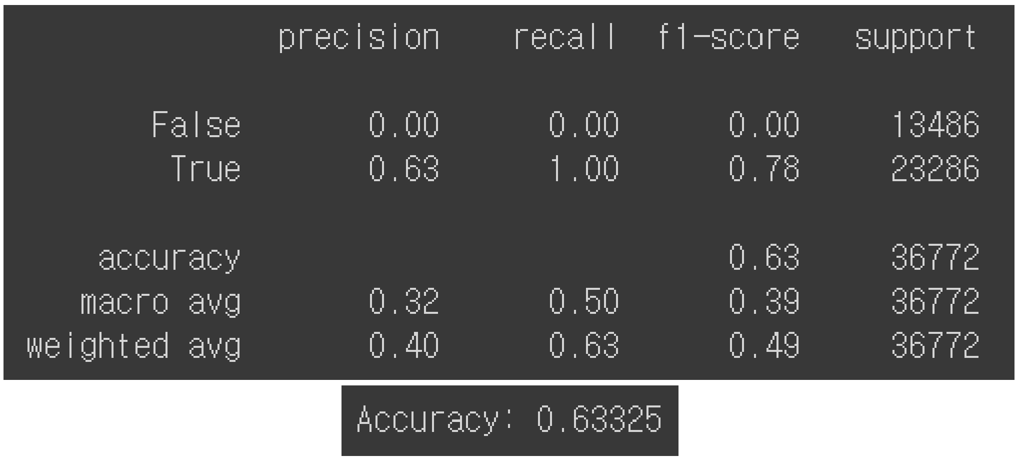
```
model_ex = svm.SVC(kernel='linear', C=1.0, random_state=0)
fe = RFE(model_ex, 4)
model_fe = fe.fit(X_fin_train, y_fin_train)
print("Number of Features: %d" % model_fe.n_features_)
print("Selected Features: %s" % model_fe.support_)
print("Feature Ranking: %s" % model_fe.ranking_)
```

Since the accuracy does not change after feature engineering, Recursive Feature Elimination (RFE) was used to extract features from the feature engineered dataset. RFE was imported from sklearn.feature_selection and gave four main features to use in the model.

5. Execution Results

Since it was recorded 40 times per second(40Hz) by 13 students for 18 hours, the amount of data was too large that I could not proceed the assignment in my desktop. It even took more than two hours to be processed in Google Colaboratory, so the data was cut down randomly into 10% to execute the codes and train the model.

1) The first execution result was:



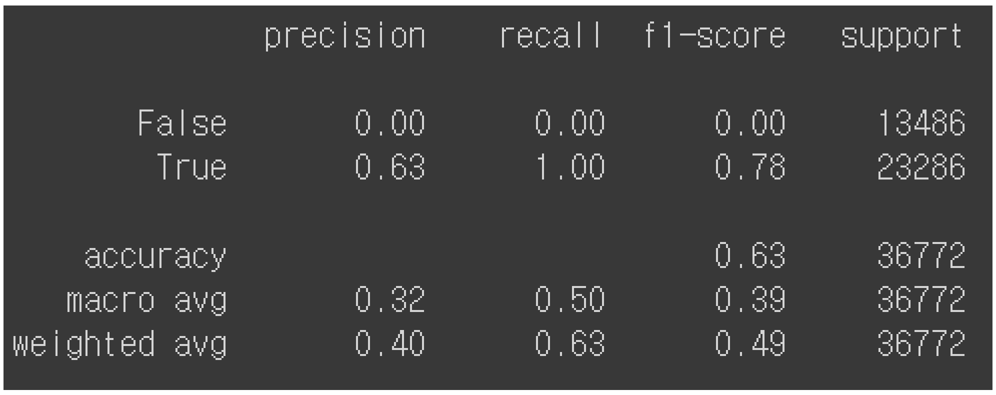
```
              precision    recall  f1-score   support

       False       0.00      0.00      0.00     13486
        True       0.63      1.00      0.78     23286

    accuracy                           0.63     36772
   macro avg       0.32      0.50      0.39     36772
weighted avg       0.40      0.63      0.49     36772
```

```
Accuracy: 0.63325
```

Figure 4. Precision, recall, f1-score, support, accuracy result of the first execution

The model has a precision of 0.63 and recall of 1. (The accuracy with more detailed precision was 0.63325.) The f1-score was 0.78 which could be said the model is somehow helpful to detect if the person is intoxicated or not. However, the model could do nothing for the False one, which was the sober one. I guess there is less pattern when the person is in sober state and not intoxicated.

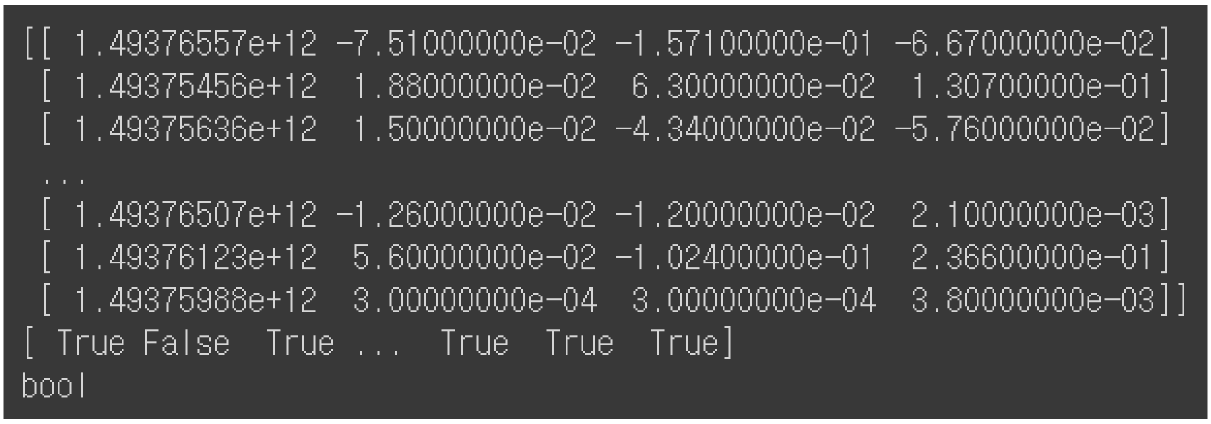
2) The result after feature engineering was:



```
              precision    recall  f1-score   support

       False       0.00      0.00      0.00     13486
        True       0.63      1.00      0.78     23286

    accuracy                           0.63     36772
   macro avg       0.32      0.50      0.39     36772
weighted avg       0.40      0.63      0.49     36772
```

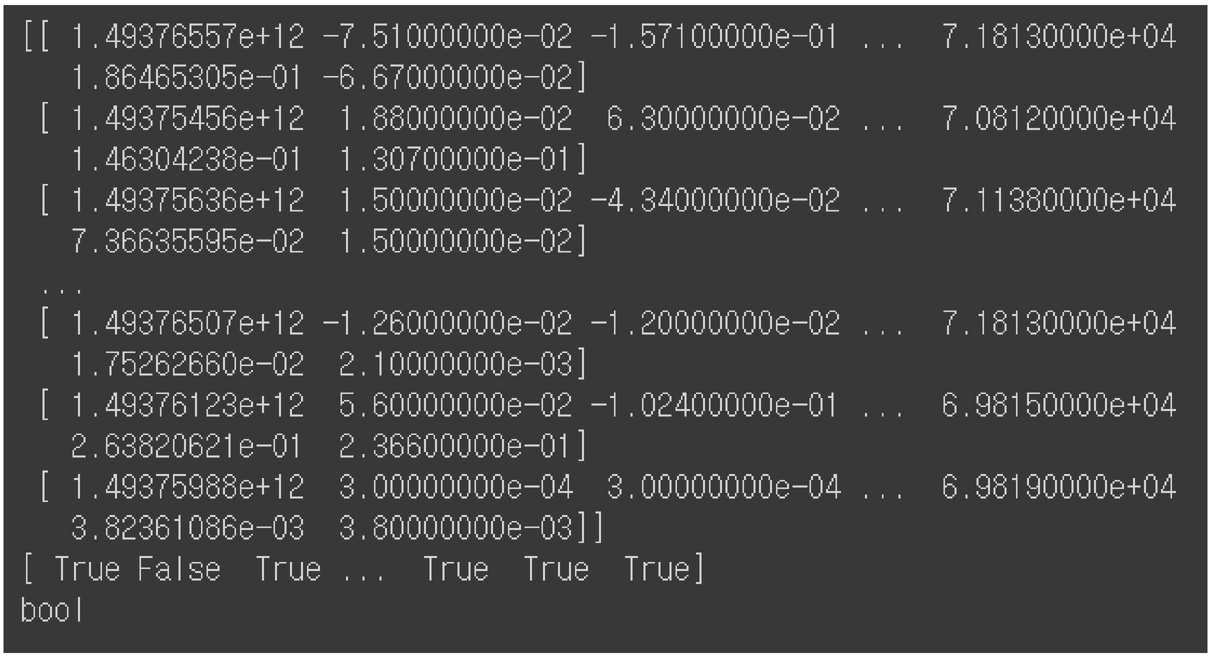
Figure 5. Precision, recall, f1-score, support, accuracy result of after feature engineering

The result was exactly the same as the first execution, which was weird. The input vectors between the two execution approaches were different, as shown in the picture below:



```
[[ 1.49376557e+12 -7.51000000e-02 -1.57100000e-01 -6.67000000e-02]
 [ 1.49375456e+12  1.88000000e-02  6.30000000e-02  1.30700000e-01]
 [ 1.49375636e+12  1.50000000e-02 -4.34000000e-02 -5.76000000e-02]
 ...
 [ 1.49376507e+12 -1.26000000e-02 -1.20000000e-02  2.10000000e-03]
 [ 1.49376123e+12  5.60000000e-02 -1.02400000e-01  2.36600000e-01]
 [ 1.49375988e+12  3.00000000e-04  3.00000000e-04  3.80000000e-03]]
[ True False  True ...  True  True  True]
bool
```

Figure 6. The input vectors of the first approach

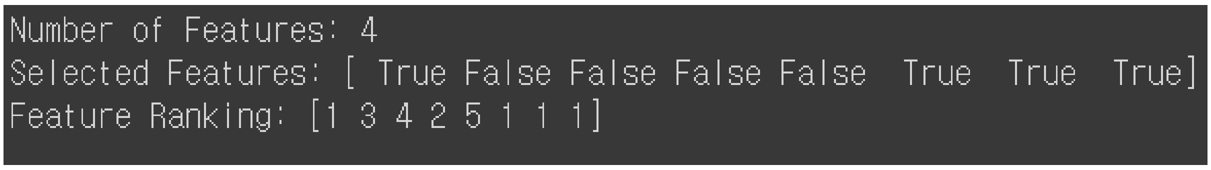


```
[[ 1.49376557e+12 -7.51000000e-02 -1.57100000e-01 ...  7.18130000e+04
   1.86465305e-01 -6.67000000e-02]
 [ 1.49375456e+12  1.88000000e-02  6.30000000e-02 ...  7.08120000e+04
   1.46304238e-01  1.30700000e-01]
 [ 1.49375636e+12  1.50000000e-02 -4.34000000e-02 ...  7.11380000e+04
   7.36635595e-02  1.50000000e-02]
 ...
 [ 1.49376507e+12 -1.26000000e-02 -1.20000000e-02 ...  7.18130000e+04
   1.75262660e-02  2.10000000e-03]
 [ 1.49376123e+12  5.60000000e-02 -1.02400000e-01 ...  6.98150000e+04
   2.63820621e-01  2.36600000e-01]
 [ 1.49375988e+12  3.00000000e-04  3.00000000e-04 ...  6.98190000e+04
   3.82361086e-03  3.80000000e-03]]
[ True False  True ...  True  True  True]
bool
```

Figure 7. The input vectors after feature engineering

Since the feature engineering could not improve the model's accuracy, feature selection was proceeded. By Recursive Feature Elimination(RFE), features were selected as below:

3) The result after feature selection was:

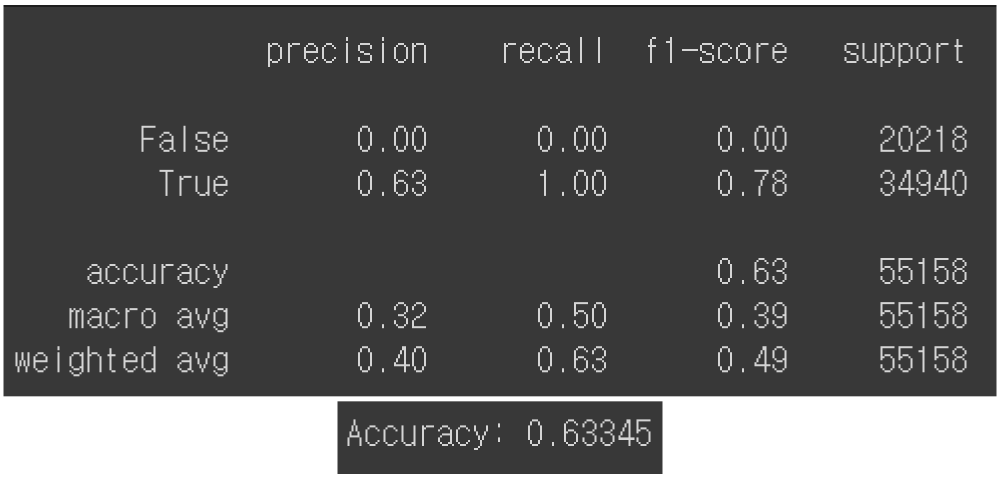


```
Number of Features: 4
Selected Features: [ True False False False False  True  True  True]
Feature Ranking: [1 3 4 2 5 1 1 1]
```

The list of features was: [time, x, y, z, average, recorded time, acceleration rate, maximum of xyz]. Surprisingly, the original values x, y and z were not selected as important features, but the other features derived from feature engineering (recorded time, acceleration rate, maximum of xyz) were chosen as the important features.

However, the feature selection process could not provide dramatic change of accuracy, so other approaches were needed. Since the number of data was cut down into 10% because of the lack of computing resources, it was chosen to scale up the number of data a bit (20%) until it would not experience the lack of computing resources.

4) The final result was:



```
              precision    recall  f1-score   support

       False       0.00      0.00      0.00     20218
        True       0.63      1.00      0.78     34940

    accuracy                           0.63     55158
   macro avg       0.32      0.50      0.39     55158
weighted avg       0.40      0.63      0.49     55158
```

```
Accuracy: 0.63345
```

Figure 8. Precision, recall, f1-score, support, accuracy result of the final execution

The accuracy *slightly increased* as 0.00020. Therefore, it is expected that the model would get higher accuracy if there is enough computing resources and time with the features.

6. Result Analysis

1) Overall Result

Based on the accelerometer and TAC data collected from people's smartphones, a model to predict if the person is sober or intoxicated was derived. Used features to train a model were: recorded time of accelerometer data, the number of successful recorded times of accelerometer in 30 min, the acceleration rate calculated based on the coordinate values of accelerations, and maximum of x, x, and z. The final accuracy was ~~~.

2) The success of feature engineering may be difficult to evaluate

The feature engineering was worked, because based on the feature selection, it turned out that the derived features were more effective than the original values of x, y, and z. However, the result did not show the dramatic changes of the accuracy. It seems that due to the lack of computing resources, the scale of data was small to show the changes of the accuracy. Therefore, sometimes the feature engineering could be considered as successful regarding the feature selection process, but it may not turn into the development of the accuracy.

3) The number of data to train a model is important.

At the first approach, because of the lack of computing resources, the number of data were cut down to 10% because it was the proportion that I could manage in Google Colaboratory. (My computer was even poorer than Google Colab.) To increase the accuracy, the increment of the number of data was proceeded to 15% and it was successful.

4) Difficulties

The lack of computing resources throughout the Assignment and the feature engineering process were suffering. However, within the help of Google Colaboratory, it successfully worked.

---

[i] Dataset from: https://archive.ics.uci.edu/ml/datasets/Bar+Crawl%3A+Detecting+Heavy+Drinking

[ii] Feature Selection For Machine Learning in Python. May 20, 2016. https://machinelearningmastery.com/feature-selection-machine-learning-python/. Searched on June 24, 2020.