
# COMP 9417

## Machine Learning and Data Mining

## Project Report

# 1 INTRODUCTION

Our project used the provided data to do classification prediction of students' performance on two main scores: PANAS and flourishing. The basic methods and goal are presented as follows:

**Methods:**

1. To select the useful information from the original database.
2. Group and extract new features from the selected information.
3. Divide the new features into several phases on the basis of the progress of the term.
4. Try and examine different models and parameters.
5. Compare models for better performance.
6. Try different optimization methods and functions to improve performance.
7. Select the top three models with best results.

**Goals:**

8. Find the relatively best combination of features from the original database.
9. Find the relatively best model and parameter for prediction.
10. Improve the performance of the model and get the best results.

**Problems to consider:**

The original dataset contains a lot of sensing information from the 10 weeks recording and unprocessed data usually mixed in a lot of worthless and disruptive information. Therefore, information filtering is a crucial process, where carefully studying on each raw data provided is necessary to excavate useful information. Otherwise, it is necessary to extract related (several data to infer the user behaviour) information pieces from the different data sets and recombine and summarized them into new features.

Model and parameter selection is the next process after feature selection. Moreover, various optimization and ideas are applied to improve performance.

# 2 DATA SET

## 2.1 TRAINING DATABASE DESCRIPTION:

The whole *StudentLife* dataset keeps track of all students from 3.27 to 6.2(65 days regardless of the missing value) to predict the mental state change evaluated by two result scores. The original data contains all the sensor data, EMA data, survey responses, and educational data. The data selected is a subset of *StudentLife* consist of activity, audio, bluetooth, conversation, dark, gps, phonecharge, phonelock, wifi, and wifi_location.

We aimed at those data which can represent the changes in student's activities, conversations and sleep duration. In order to conclude these three main features of each data, we extensively studied every possible useful attribute.

As an example, activity data describes one's physical activity inferences including stationary, walking, running and unknown. Expect for activity data, bluetooth, gps, wifi, and wifi_location data can represent a user's activity state as well and other main features are also selected and grouped by vary original data.

## 2.2 DATA GROUPING:

### 2.2.1 Activity

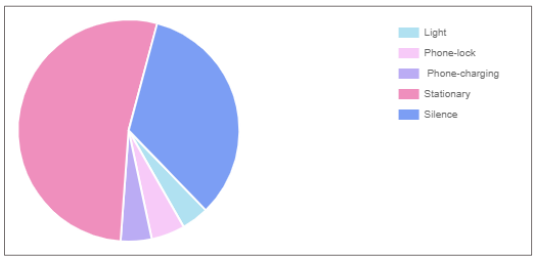
Activity data grouped data from the folder 'activity' and 'gps'.

### 2.2.2 Conversation

Conversation data grouped data from the folder 'conversation' and 'audio'.

### 2.2.3 Sleep duration

In order to compute the sleeping duration, we referenced the linear sleep model. It combines the data from dark(to compute light time), phonelock, phonecharge, activity(for stationary) and audio(for silence).




$$Sleep\ duration = 0.0415 * Light + 0.05 * PhoneLock + 0.0469 * PhoneCharging + 0.55 * Stationay + 0.3484 * Silence$$

## 2.3 RESULT SET

We need to predict two main results: Flourishing and PANAS.

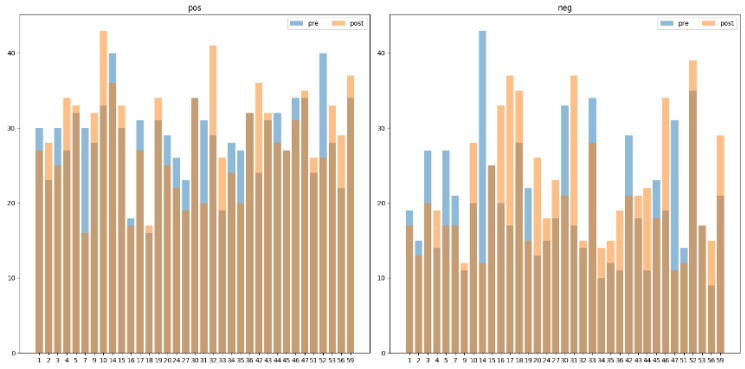



figure 2-3-1 PANAS result

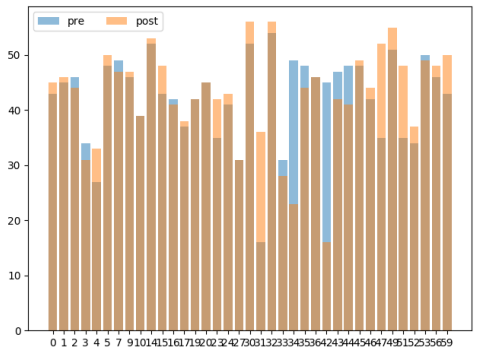



figure 2-3-2 Flourishing result

The figures for Flourishing and PANAS results are shown in figure 2-3-1 and 2-3-2. Figure 2-3-1 shows that the amount of positive value is decreased while negative value increased for the majority of students. In figure 2-3-2, we can see that for major students, the number for the post flourishing value is larger than the pre value.

In both these files, several missing values are presented for a few students. Some of students did not even attend the research. Hence, we decid to remove the student data from the training set and after filtering there are 35 valid results for PANAS and 34 valid results for Flourishing at last.

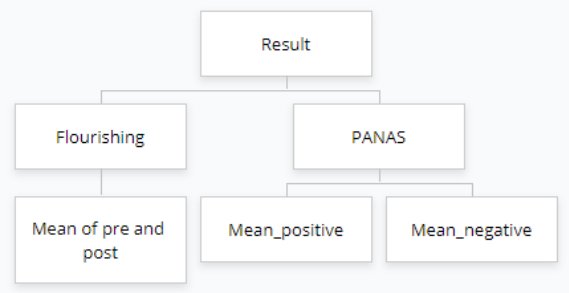



figure 2-3-3 Results prediction

In order to represent the changes in each student and the problems consideration mentioned above:

For Flourishing, we predict the mean of the pre and post.

For PANAS, we divided the results into two parts: positive and negative, predict mean between pre and post.

## 2.4 BINARIZATION METHOD:

To divide the classes, we use the median as the threshold. If the predicted score is larger than the threshold, it will be labelled as ‘High’ and otherwise ‘Low’.

# 3 METHODS

## 3.1 FEATURE (TRAIN SET) EXTRACTION:

According to introduction in *StudentLife*, students' stress increased and the conversation activity and sleep duration decrease with the pace of the term. They become less activate. Thus, we select the data that are mainly focus on these three aspects.

The original data record start from 3.27 to 6.2. In order to present the changes during the process of the term, we split the semester into three main phases: the first two weeks (3.27-4.10), median (4.11-5.17), and the last two weeks (5.18-6.2). All the following features selected are allocated into these three phases.

On the activity aspect, we used the information under 'activity' and 'gps' folders. In 'Activity', we calculate the total time duration which are not inferenced as 'stationary' state of each student. In 'gps', we calculate the total time duration of each student which the travel state is 'moving'.

On the conversation aspect, we used the data under 'conversation' and 'audio' folders. For each student, we calculated the total time duration of conversation and that which inferenced as 'voice'.

On the sleep aspect, we grouped the data from 'dark', 'activity', 'phonelock', 'phonecharge', and 'audio'. Firstly, calculate the total time of dark time, stationary time, phone charge and lock duration, audio are inferenced as silence. These characters are used to define that someone is sleep. By comparison, we found that for most students, the features are not complete and in some attribute records there were lots of missing dates. Therefore, for each student, we only selected the date with all these five valid records to calculate his average daily sleep duration.

After dividing the data into three-time phases and data grouping, the initial dataset we used is:

| conversation | | | activity | | | gps | | | audio | | | sleep | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| p1 | p2 | p3 | p1 | p2 | p3 | p1 | p2 | p3 | p1 | p2 | p3 | p1 | p2 | p3 |

## 3.2 Missing and Noise

It is obvious that there are lots of missing or meaningless data, which will influence the training of model. Thus, dealing with these missing data and noise in pre-process is especially crucial.

### 3.2.1 Missing values

There are lots of missing values. For example, the data of the count of activity data by days for each user is shown in figure 3-3-1.

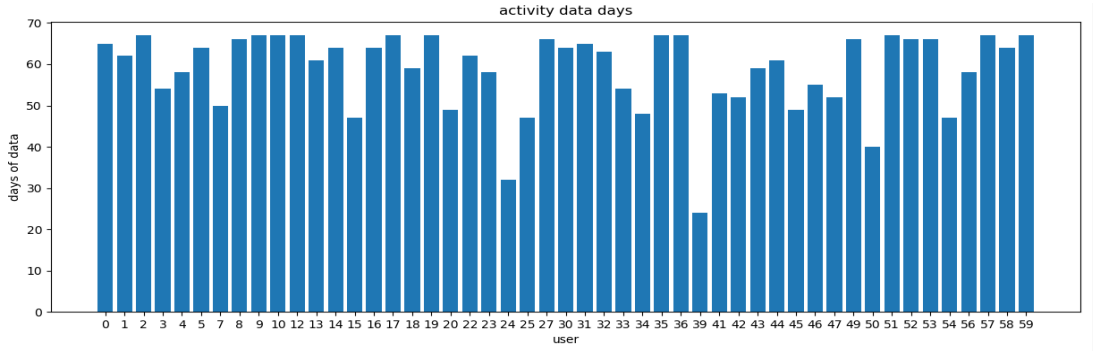
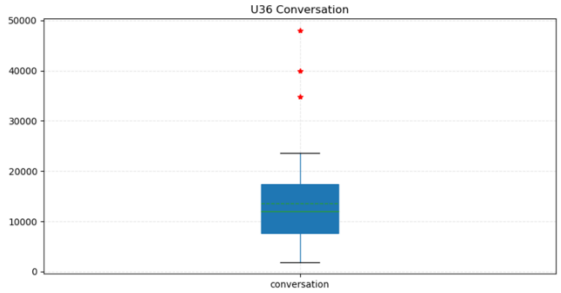



figure 3-2-1 number of record days in activity for each student

The figure above indicates that the total number of valid record dates varies from person to person hence the mean values of the recorded valid data are used to fill the missing for each feature.

### 3.2.2 Noise

Gaussian distribution is applied to determine if the data is unnormal. By means of box plot we can visualize these outliers as red star points.




To dealing these noise values, we replace them with the mean value. The floor of data is mean – 3 * standard deviation and the ceiling of data is mean + 3 * standard deviation. Then the normal distribution can be drawn like U00 conversation and U36 conversation. Data between red dotted line and green dotted line are normal while the data out of these two lines we will be replaced with the mean value.

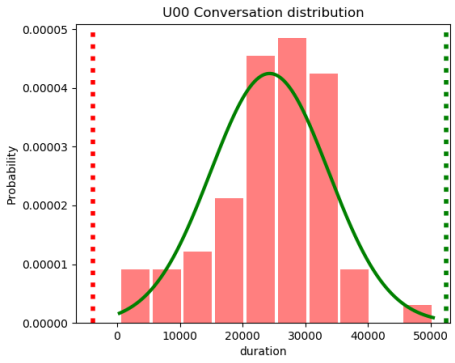



figure 3-2-2-2 u00 **Conversation distribution**

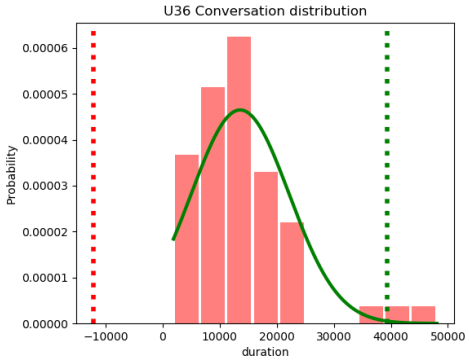



figure **3-2-2-3 u36 Conversation distribution**

## 3.3 PRE-PROCESSING OF TRAINING SET:

### 3.3.1 At the beginning

For the result set, we divide the class into two parts by the **median** as threshold. Due to that, there are lots of incomplete results. We stablish a new list to keep the valid user and only using these data as training sets.

Before we put all the data into next step, we will filter the data in training set that has the valid corresponding result data. As we mentioned before, we only got 34 valid Flourishing record and 35 for PANAS, thus the training set should be consistent with the result set.

### 3.3.2 Normalization

We tried two normalizations: MinMaxScaler and StandardScaler.

MinMaxScaler uses the following formula to scale the feature values into the range of [0,1]:

$$x_{new} = \frac{\mathrm{x} - \min(\mathrm{x})}{\max(\mathrm{x}) - \min(\mathrm{x})}$$

StandardScaler uses the following formula to scale the feature values into the range of [-1,1]:

$$x_{new} = \frac{\mathrm{x} - \mathrm{mean(training\ samples)}}{\mathrm{standard\ deviation\ of\ the\ training\ samples}}$$

## 3.4 PARAMETER AND FEATURE SELECTION:

### 3.4.1 Hyper-parameter selection

For those models with multiple optional parameters, we used sklearn.model_selection.GridSearchCV to find the best hyper-parameter combination and set scoring = ‘roc_auc’. This process is able to exhaustively search over specified parameter values for an estimator and return the model with the best status.

```
def svm_cross_validation(X_train, Y_train):
    model = SVC(probability=True)
    param_grid = {'C': [1e-3, 1e-2, 1e-1, 1, 10, 100, 1000],
                  'gamma': [1,0.1,0.01, 0.001, 0.0001],
                  'kernel': ['rbf','linear']}
    grid_search = GridSearchCV(model, param_grid,scoring='roc_auc')
    grid_search.fit(X_train, Y_train)
    best_parameters = grid_search.best_estimator_.get_params()
    model = SVC(kernel=best_parameters['kernel'], C=best_parameters['C'], gamma=best_parameters['gamma'], probability=True)
    return model
```

figure 3-4-1-1 GridSearchCV sample for SVM

### 3.4.2 Feature selection

We select the final dataset by referencing the feature_importance_ and experiment based on the RandomForestClassifier.

```
r_forest = Random_forest_classifier(X, result_class)
feature_importance = r_forest.feature_importances_
print(data.iloc[0, :])
print(feature_importance)
feature_importance = 100.0 * (feature_importance / feature_importance.max())
sorted_idx = np.argsort(feature_importance)
print(sorted_idx)
```

We used sklearn.feature_selection.SelectFromModel to optimize the features for partial models. It is applied for selecting features based on important weights. However, this function is not suitable for all model. The estimator must come with either a '*feature_importances_'* or '*coef_'* attribute after fitting. In the models we selected, KNeighborsClassifier and GaussianNB are not suitable.

## 3.5 MODEL SELECTION:

| Model | Parameter selection | Feature selection |
|---|---|---|
| SVM.SVC | GridSearchCV: 'C', 'gamma', 'kernel' | SelectFromModel |
| DecisionTreeClassifier | GridSearchCV: 'max_depth' | SelectFromModel |
| KNeighborsClassifier | GridsearchCV: 'n_neighbors', 'weights' | None |
| RandomForestClassifier | GridsearchCV: 'n_estimators', 'max_depth' | SelectFromModel |
| MultinomialNB | None | SelectFromModel |
| LogisticRegression | Solver = 'liblinear' | SelectFromModel |
| GaussianNB | None | None |

## 3.6 EVALUATION METHODS

We evaluate the model mainly based on ROC_AUC:

ROC_AUC: Compute Area Under the Receiver Operating Characteristic Curve (ROC AUC) from prediction scores.

Besides, we employ cross validation to further evaluated the models based on these two metrics by take the mean of the results score because the size of the training set is small:

```
score = np.mean(cross_val_score(model X, result_class, cv=9, scoring=roc_auc))
```

# 4 RESULTS

## 4.1 IMPORTANT FEATURE

Initially, we experiment with our database by employing the *feature_importance_* to rank the selected data and the result is showing in figure 4-1-1 left.

We decided to regroup out dataset to emphasis the features to enlarge the proportion of the three aspects we mentioned and also to avoid the situation such as the importance of some features was zero. Moreover, we reduced the dimension of the data to avoid over-fitting and then calculated the mean of the valid days of each data rather than divided them into three time periods. After we optimize the dataset according to the former database, the latest data importance is as showing on right.

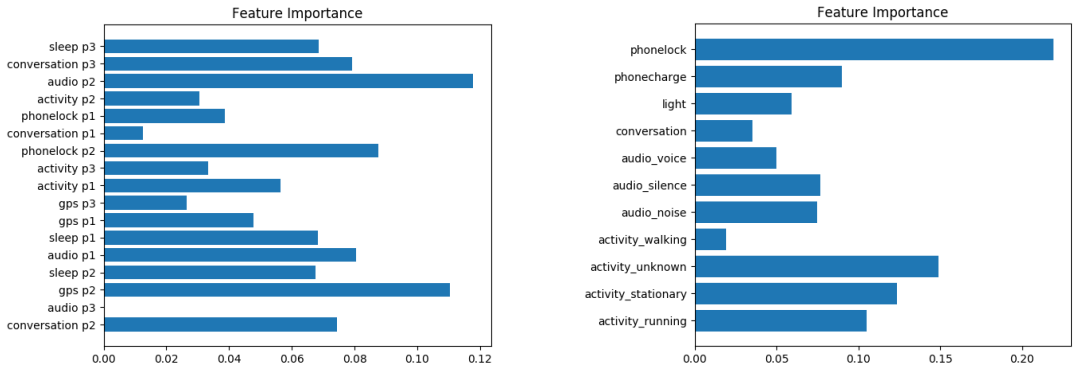



figure 4-1-1 Feature Importance ranking

The final feature set we use to train the model is

| activity_running | activity_stationary | activity_unknown | audio_noise | audio_silence | audio_voice | conversation | light | phonecharge | phonelock |
|---|---|---|---|---|---|---|---|---|---|

## 4.2 HYPER-PARAMETERS SELECTION

After the gridsearchCV and comparison by cross_val_score(cv = 9,scoring = 'roc_auc'), the best parameters acquired for each model are showing as follow:

Flourishing:

| Model | Best Parameter Combination |
|---|---|
| SVM.SVC | C=1000, Degree=3, gamma=1, kernel=linear, probability=True |
| DecisionTreeClassifier | Criterion=entropy, max_depth=2 |
| KNeighborsClassifier | Algorithm =auto, leaf_size=30, n_neighbors 1, weights = uniform |
| RandomForestClassifier | max_depth=2, n_estimators=8 |
| MultinomialNB | None |
| LogisticRegression | Solver = 'liblinear' |
| GaussianNB | None |

PANAS positive mean:

| Model | Best Parameter Combination |
|---|---|
| SVM.SVC | C=100, Degree=3, gamma=0.01, kernel=rbf, probability=True |
| DecisionTreeClassifier | Criterion=entropy, max_depth=2 |
| KNeighborsClassifier | Algorithm =auto, leaf_size=30, n_neighbors 7, weights = distance |
| RandomForestClassifier | max_depth=1, n_estimators=5 |
| MultinomialNB | None |
| LogisticRegression | Solver = 'liblinear' |
| GaussianNB | None |

PANAS negative mean:

| Model | Best Parameter Combination |
|---|---|
| SVM.SVC | C=0.1, Degree=3, gamma=1, kernel=rbf, probability=True |
| DecisionTreeClassifier | Criterion=entropy, max_depth=3 |
| KNeighborsClassifier | Algorithm =auto, leaf_size=30, n_neighbors 2, weights = distance |
| RandomForestClassifier | max_depth=3, n_estimators=1 |
| MultinomialNB | None |
| LogisticRegression | Solver = 'liblinear' |
| GaussianNB | None |

## 4.3 FINAL RESULTS

After the cross validation, the final results acquired are showing below:

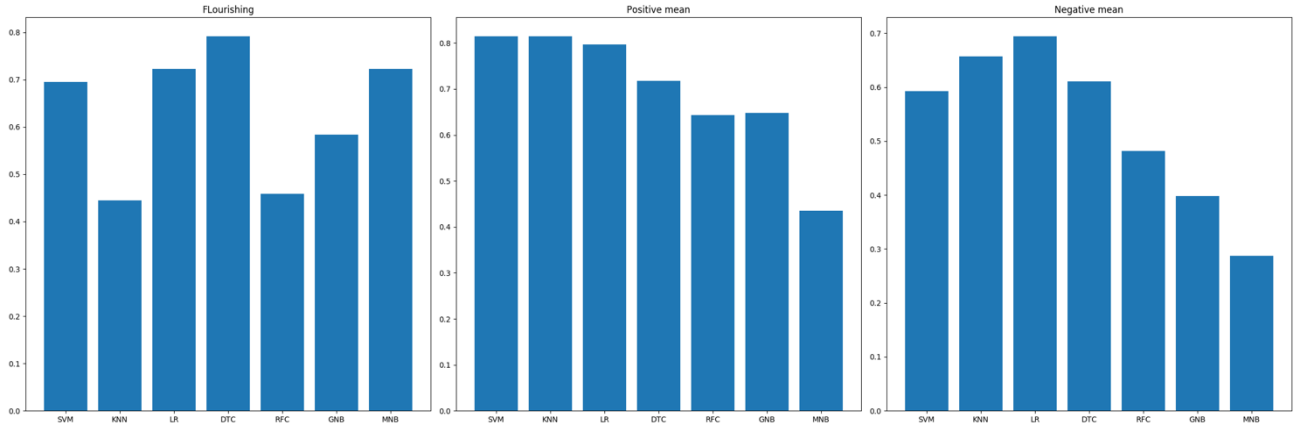



figure 4-3-1 Final results

DecisionTreeClassifier, LogisticRegression, MNB and SVM perform well for flourishing prediction.

SVM, KNeighborsClassifier, LogisticRegression and DecissionTreeClassifier perform relatively well than other models for PANAS positive prediction.

LogisticRegression, KNeighborsClassifier, SVM and DecisionTreeClassifier perform better on PANAS negative prediction.

As a conclusion, the top three model in our project based on the mean of PANAS and Flourishing are **DecisionTreeClassifier, SVM** and following is **KNeighborsClassifier and LogisticRegression**.

# 5 DISCUSSION: COMPARISON AND DISCUSSION ON EACH METHODS

## 5.1 FEATURE EXTRACTION

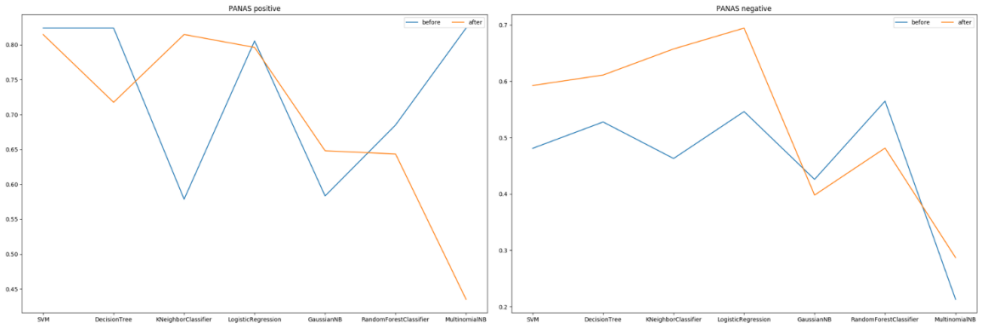



figure 5-1-1 Feature comparation

As the figure 5-1-1-1 shows, the most scores have been increased after the optimization of the data as mentioned in 4-1. The application of the mean of the daily mean value as the metric of feature makes the datasets more rational and more reliable. We find that low dimension features can avoid overfitting as it shows above, however it is not guaranteed to improve every model.

Besides, the pre-processing process is extremely necessary for data cleaning. Replacing missing data with average can prevent data loss and it is very efficient when the size of data is small. Also, processing noise can reduce the interference caused by meaningless data.

Admittedly, this pre-process is not perfect. Firstly, we have noticed that the durations of each collected features are not exactly consistent. In addition to the date of each student during sensing their data are not identical. Therefore, using daily average as the metric of each feature is not sufficiently rigorous. Secondly, we choose to replace null value with mean based on the small size of the dataset, but it may still cause the increase of variance and bias of model. Lastly, we deal with noise depend on Gaussian distribution. It can filter some outlier when their probabilities are very low, however, the threshold can be more detailed instead of $3\sigma$ if we can do more training and research.

## 5.2 FEATURE SELECTION

As we mentioned above in 3-4-2, we employ sklearn.feature_selection.SelectFromModel to improve the feature performance while this method cannot work on KNeighborsClassifier and GaussianNB. Thus, the comparison of the left models and results are as follows:

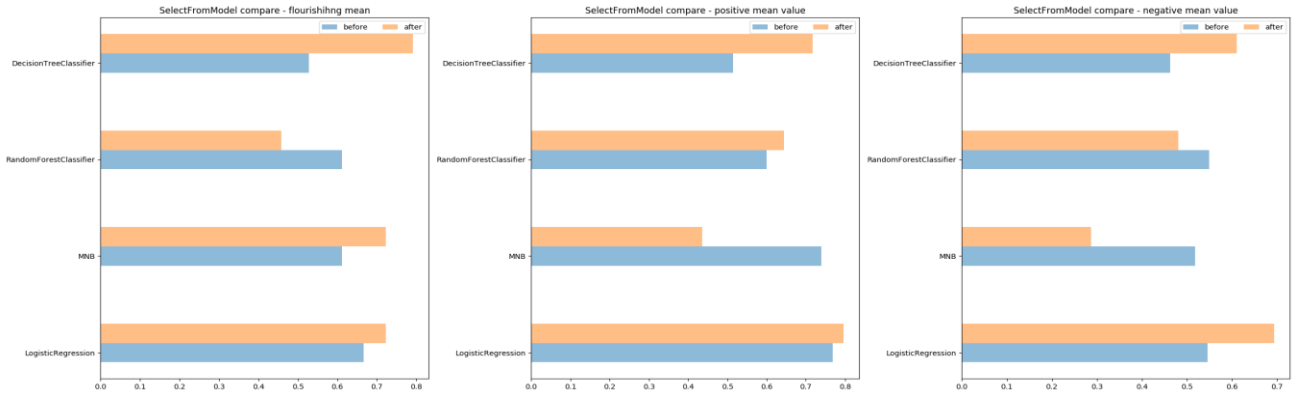



figure 5-2-1 SelectFromModel Function comparation

As we can see in figure 5-1-2-1, roc score has increased for most models after employ SelectFromModel. So we used this function to improve features for those models.

## 5.3 NORMALIZATION

We tried MinMaxScaler and StandardScaler and the roc score after cross validation showing in figure 5-1-3-1. It shows that most model perform well under MinMaxScaler. More importantly, the MinMaxScaler scales the data to [0,1] while StandardScaler scales to [-1,1]. As we planned to experiment on MultinomialNB while the input must be non-negative for MNB. According to this we think MinMaxScaler is better for this project.

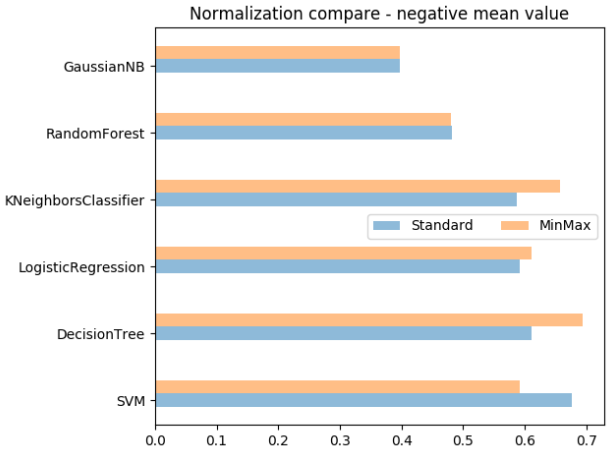



figure 5-3-1 ROC_AUC score by using different normalization

## 5.4 MODEL SELECTION

We tried lots of classifier at the very beginning. We do some rough prediction to early evaluate the model and discard those models with inferior performance. Them we used the left models to do further experiments.

We chose KNeightborsClassifier, RandomForestClassifier, SVM, GaussianNB, MultinomialNB, LogisticRegression, BernoulliNB, BaggingClassifier, DecisionTreeClassifier, AdaBosstClassifier.

Here are the early results of the three-prediction rank based on our initial dataset:

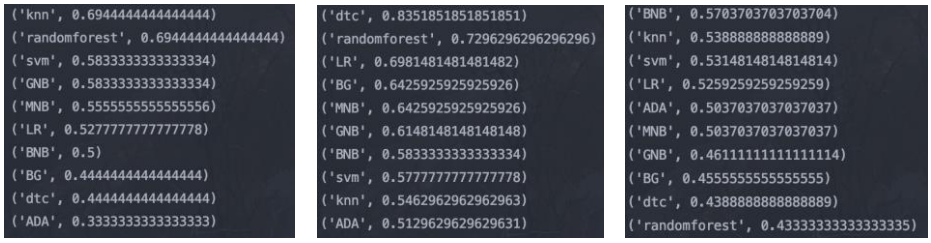



figure 5-4-1 Model selection

The mean the results are as follows:

| KNN | RFC | SVM | GNB | MNB | LR | BNB | DTC | ADA | BG |
|---|---|---|---|---|---|---|---|---|---|
| 0.593 | 0.619 | 0.564 | 0.553 | 0.5672 | 0.5839 | 0.5511 | 0.5727 | 0.449 | 0.514 |

The BNB, ADA, BG has the least roc_auc score on those predictions.

Therefore we chose KNeighborsClassifier, RandomForestClassifier, SVM, GaussianNB, MultinomialNB, LogisticRegression and DecisionTreeClassifier(though we noticed that dtc has severe oscillations) as the main classifiers to do further comparation.

## 5.5 Evaluation and performance

We chose roc_auc as the evaluation for our models. We also make some comparisons with accuracy on each prediction as following.


figure 5-1-5-1 Final ROC_AUC results

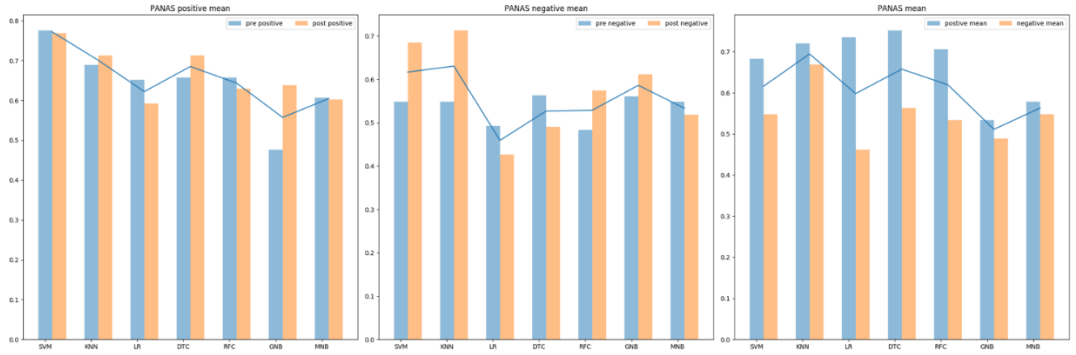



figure 5-1-5-2 Final Accuracy results

The accuracy rate is more balanced, but according to the algorithm, roc is more suitable to highlight the strengths and weaknesses of a model.

# 6 Conclusion

In this project, we tried many new ways to improve the model from feature selection to evaluation and there are several findings:

First of all, most early part and the most importance progress is data selection and cleaning. The training dataset is crucial and directly related to the final result. Secondly, pre-processing. In this part, we have to try as many as we

got the relatively best score on each model. This process further optimizes our dataset and reduces bias (deal with noise and miss). Thirdly, model and parameter selection. We selected the model by testing the average performance of each model and employ the grid-search and cross-validation to select the best parameter combination. Finally, further improvement. After all these above procedures done, we have to test again and again on each step to improve and try more new ways on each step in order to find more useful functions or ideas to improve the performance.

In conclusion, this project enhanced the ability of machine learning. There are still many ways to explore and our model is still not good enough. We need keep learning and researching in future study.

# 7 REFERENCE LIST


1. **StudentLifeDataset2014.http://studentlife.cs. dartmouth.edu/.**

2. **Z. Chen, M. Lin, F. Chen, N. D. Lane, G. Cardone, R. Wang, T. Li, Y. Chen, T. Choudhury, and A. T. Campbell. Unobtrusive sleep monitoring using smartphones. In *Proc. of PervasiveHealth*, 2013.**

3. **Medium. (2019). Scale, Standardize, or Normalize with Scikit-Learn. [online] Available at: https://towardsdatascience.com/scale-standardize-or-normalize-with-scikit-learn-6ccc7d176a02 [Accessed 22 Nov. 2019].**

4. **Scikit-learn.org. (2019). sklearn.feature_selection.SelectFromModel — scikit-learn 0.21.3 documentation. [online] Available at: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectFromModel.html [Accessed 22 Nov. 2019].**

5. **Rasmussen, C. and Williams, C. (2008). Gaussian processes for machine learning. Cambridge, Mass: MIT Press.**

6. **García, S., Ramírez-Gallego, S., Luengo, J. et al. Big Data Anal (2016) 1: 9. https://doi.org/10.1186/s41044-016-0014-0**

7. **Scikit-learn.org. (2019). Classifier comparison — scikit-learn 0.21.3 documentation. [online] Available at: https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html [Accessed 24 Nov. 2019].**