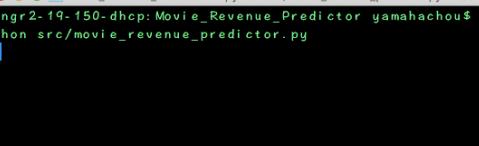


# Result

## I. Process of running code

## 1. Run RMSE test



Project

- Movie\_Revenue\_Predictor
  - .git
  - datasets
  - key\_result
  - notebook
  - src
    - .gitignore
    - README.md

classification.py clustering.py movie\_revenue\_predictor.py preprocessing.py

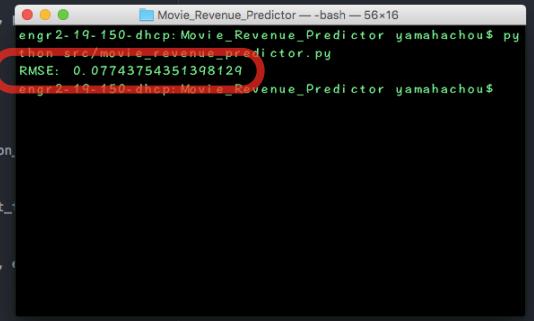
```
173 def classificationTest(test_times, classification_method,   
174     # err, total, diff = trainAndTest(df, 1, True)  
175     df = pd.read_csv('datasets/clustered.csv')  
176     err_t = 0  
177     diff_t = 0  
178     total = 0  
179     for i in range(test_times):  
180         err, total, diff = trainAndTest(df, classification,   
181             err_t = err_t + err  
182             diff_t = diff_t + diff  
183             print('avg correctness: ' + str(((total - (err_t/test_1)) / test_1)))  
184             print('avg diff: ' + str((diff_t/test_times)/total)))  
185  
186 def RMSETest(test_times, classification_method, plotting,   
187     df = pd.read_csv('datasets/noclustered.csv')  
188     err_t = 0  
189     for i in range(test_times):  
190         rmse = trainAndTest(df, classification_method, plotting, evaluation_method)  
191         err_t = err_t + rmse  
192         print('RMSE: ' + str(err_t/test_times))  
193  
194 classification_method = 1 # boosting  
195 plotting = True # don't plot results  
196 evaluation_method = 0 # class  
197 test_times = 1  
198  
199 preprocess()  
200 generateClusteredCSV()  
201 #classificationTest(test_times, classification_method, plotting, evaluation_method)  
202  
203 evaluation_method = 1 # RMSE  
204 RMSETest(test_times, classification_method, plotting, evaluation_method)  
205
```

src/movie\_revenue\_predictor.py ⏺ 0 ▲ 0 ⏻ 0 201:2

• LF UTF-8 Python ⌂ master ⌂ Fetch ⌂ GitHub ⌂ Git (2) 2 updates

2. After it complete the calculation, it would show the plotting picture

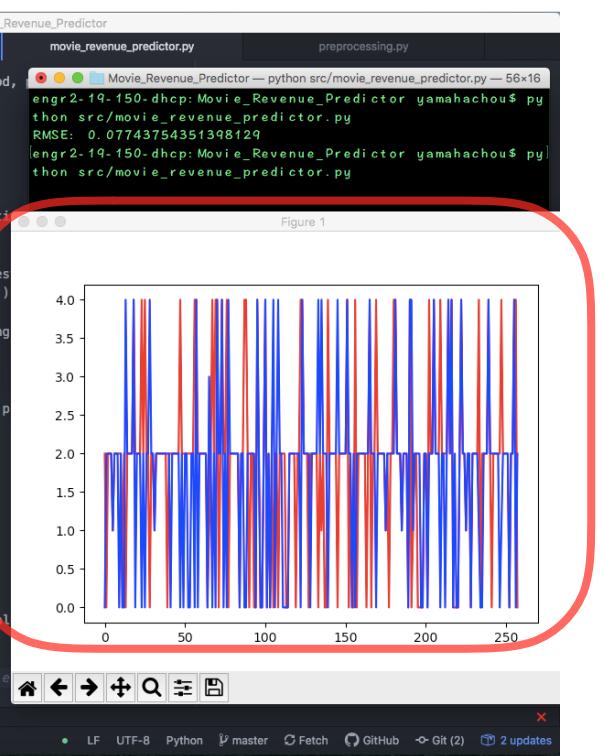
### 3. After close the plotting picture, the terminal would show the computation result of RMSE



```
movie_revenue_predictor.py -- ~/Documents/Movie_Revenue_Predictor
Project
Movie_Revenue_Predictor
src
classification.py clustering.py movie_revenue_predictor.py preprocessing.py
173 def classificationTest(test_times, classification_method,
174     # err, total, diff = trainAndTest(df, 1, True)
175     df = pd.read_csv('datasets/clustered.csv')
176     err_t = 0
177     diff_t = 0
178     total = 0
179     for i in range(test_times):
180         err, total, diff = trainAndTest(df, classification,
181             err_t = err_t + err
182             diff_t = diff_t + diff
183             print('avg correctness: ' + str(((total - (err_t/test_
184             print('avg diff: ' + str((diff_t/test_times/total)))
185
186     def RMSETest(test_times, classification_method, plotting,
187         df = pd.read_csv('datasets/noclustered.csv')
188         err_t = 0
189         for i in range(test_times):
190             rmse = trainAndTest(df, classification_method, plotting, evaluation_method)
191             err_t = err_t + rmse
192             print('RMSE: ' + str(err_t/test_times))
193
194     classification_method = 1 # boosting
195     plotting = True # don't plot results
196     evaluation_method = 0 # class
197     test_times = 1
198
199     preprocess()
200     generateClusteredCSV()
201     #classificationTest(test_times, classification_method, plotting, evaluation_method)
202
203     evaluation_method = 1 # RMSE
204     RMSETest(test_times, classification_method, plotting, evaluation_method)
205
+
src/movie_revenue_predictor.py ① 0 ▲ 0 ③ 0 201:2
```

Movie\_Revenue\_Predictor -- bash -- 56x16  
enr2-19-150-dhcp:Movie\_Revenue\_Predictor yamahachou\$ python src/movie\_revenue\_predictor.py  
RMSE: 0.07743754351398129  
enr2-19-150-dhcp:Movie\_Revenue\_Predictor yamahachou\$

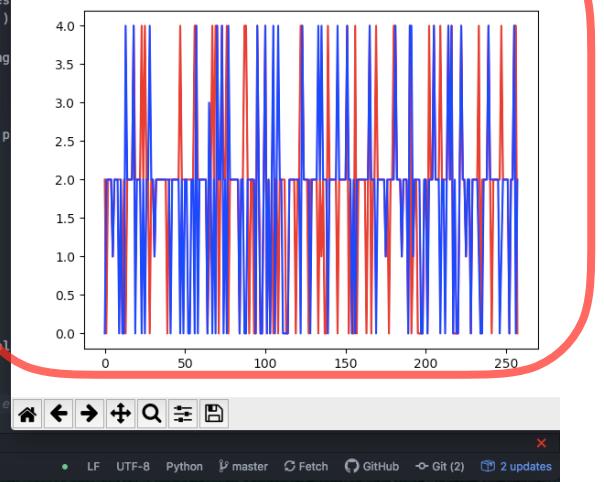
### 4. Start to run the classification error evaluation, it would show the plotting picture



```
movie_revenue_predictor.py -- ~/Documents/Movie_Revenue_Predictor
Project
Movie_Revenue_Predictor
src
classification.py clustering.py movie_revenue_predictor.py preprocessing.py
173 def classificationTest(test_times, classification_method,
174     # err, total, diff = trainAndTest(df, 1, True)
175     df = pd.read_csv('datasets/clustered.csv')
176     err_t = 0
177     diff_t = 0
178     total = 0
179     for i in range(test_times):
180         err, total, diff = trainAndTest(df, classification,
181             err_t = err_t + err
182             diff_t = diff_t + diff
183             print('avg correctness: ' + str(((total - (err_t/test_
184             print('avg diff: ' + str((diff_t/test_times/total)))
185
186     def RMSETest(test_times, classification_method, plotting,
187         df = pd.read_csv('datasets/noclustered.csv')
188         err_t = 0
189         for i in range(test_times):
190             rmse = trainAndTest(df, classification_method, plotting, evaluation_method)
191             err_t = err_t + rmse
192             print('RMSE: ' + str(err_t/test_times))
193
194     classification_method = 1 # boosting
195     plotting = True # don't plot results
196     evaluation_method = 0 # class
197     test_times = 1
198
199     preprocess()
200     generateClusteredCSV()
201     classificationTest(test_times, classification_method, plotting, evaluation_method)
202
203     evaluation_method = 1 # RMSE
204     RMSETest(test_times, classification_method, plotting, evaluation_method)
205
+
src/movie_revenue_predictor.py ① 0 ▲ 0 ③ 0 204:2
```

Movie\_Revenue\_Predictor -- python src/movie\_revenue\_predictor.py -- 56x16  
enr2-19-150-dhcp:Movie\_Revenue\_Predictor yamahachou\$ python src/movie\_revenue\_predictor.py  
RMSE: 0.07743754351398129  
enr2-19-150-dhcp:Movie\_Revenue\_Predictor yamahachou\$ python src/movie\_revenue\_predictor.py

Figure 1



5. After close the plotting picture, the terminal would show the computation result of Classification error method

The terminal window shows the command being run: `Movie_Revenue_Predictor -- bash`. The output includes:

```
engr-2-19-150-dhcp: Movie_Revenue_Predictor yamahachou$ python src/movie_revenue_predictor.py
RMSE: 0.07743754351398129
engr-2-19-150-dhcp: Movie_Revenue_Predictor yamahachou$ python src/movie_revenue_predictor.py
errors: 66
total: 258
diff: 181
avg correctness: 0.7441860465116279
avg diff: 0.7015503875968992
engr-2-19-150-dhcp: Movie_Revenue_Predictor yamahachou$
```

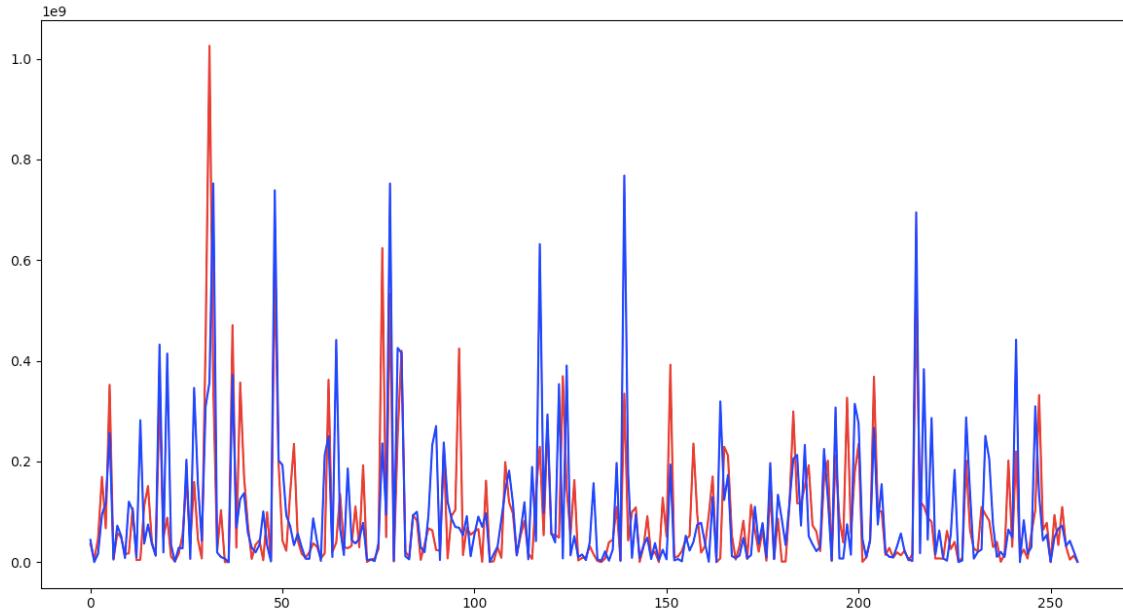
The code in classification.py contains several functions for classification and RMSE testing, including `classificationTest`, `RMSETest`, and `preprocess`.

## II. Evaluation result for each classification

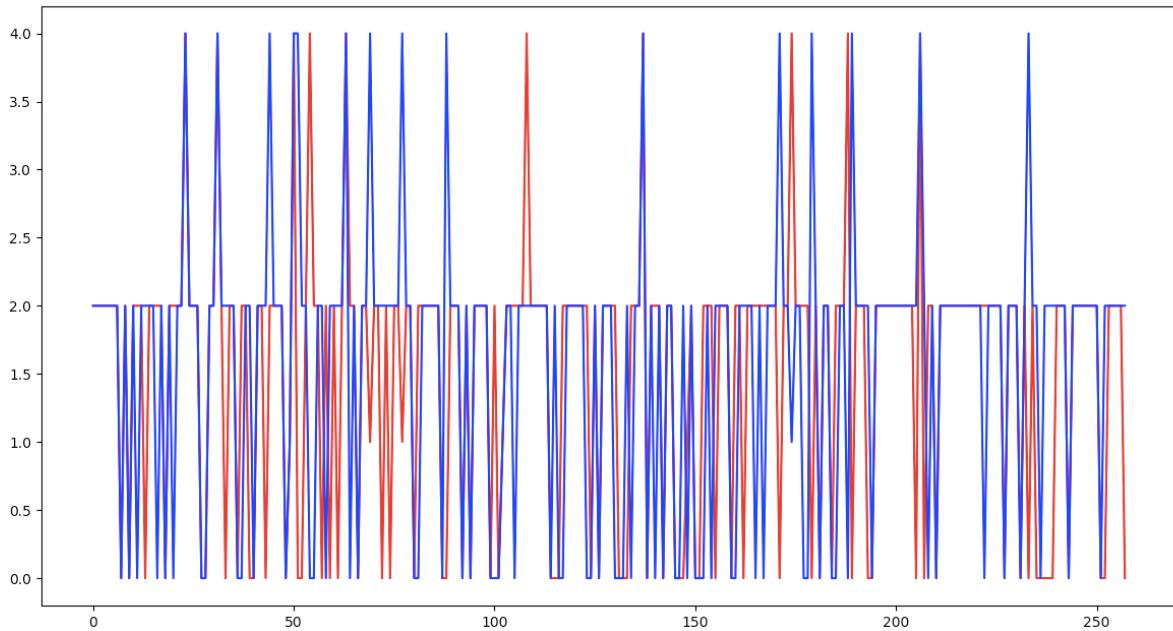
After we ran the evaluation method on each classifier, we would get the plotting picture for each classifier, the red part means **our prediction**, and blue part means the **real situation**.

### 1. Decision Tree Classification

\* RMSE

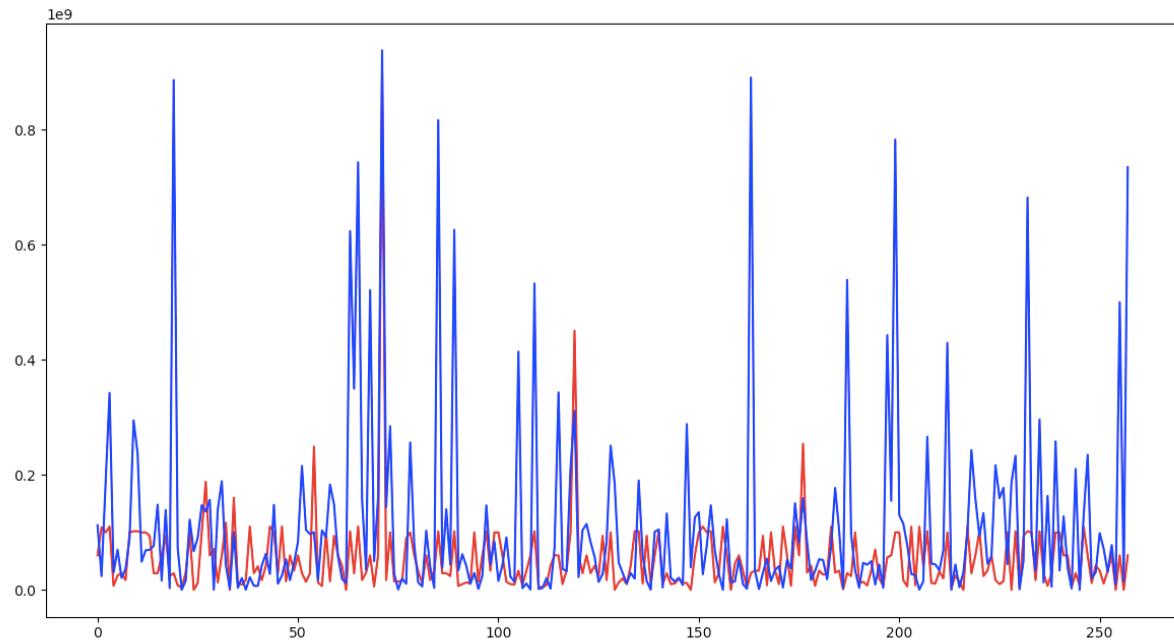


\* Classification error

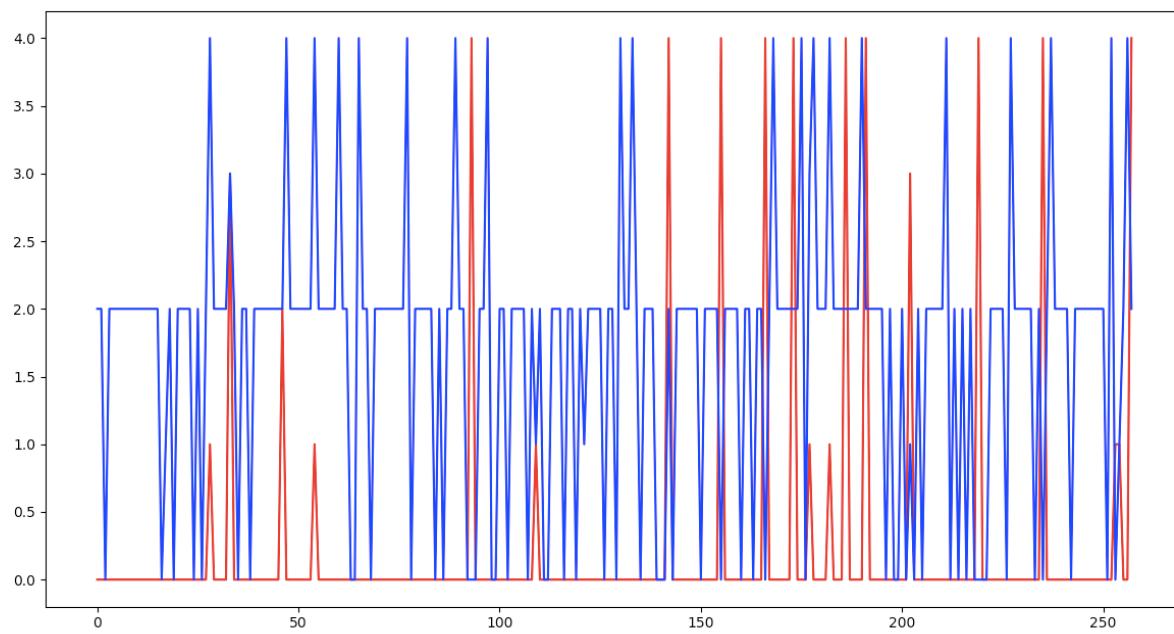


## 2. Gaussian Naive Bayes Classification

\* RMSE

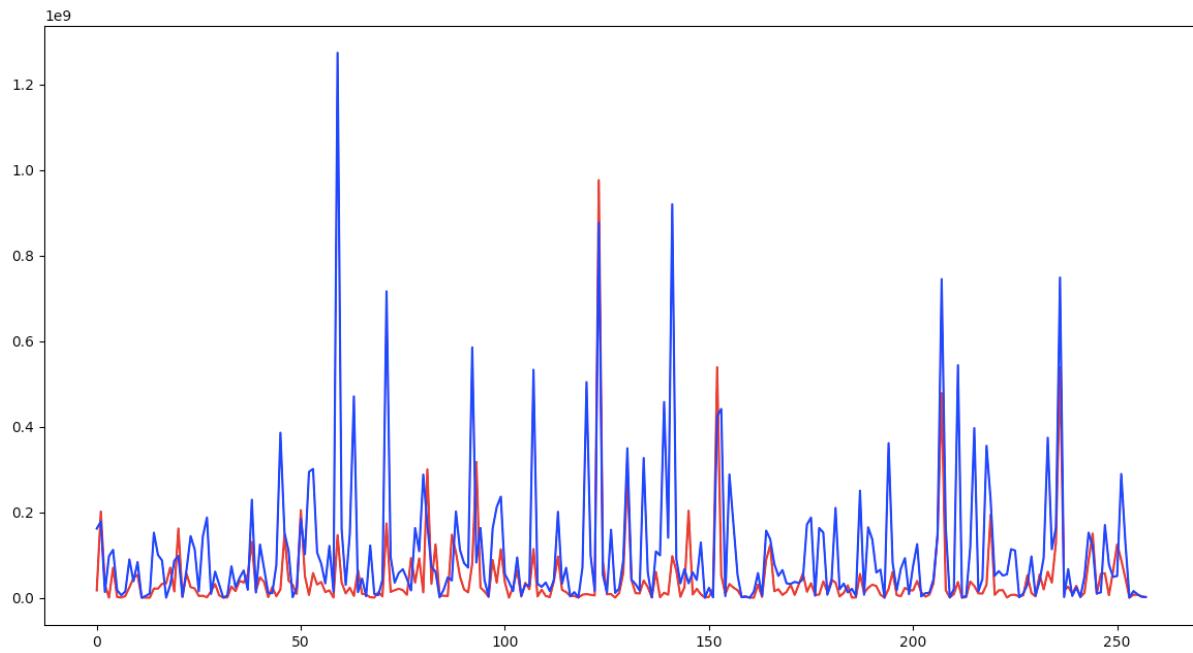


\* Classification error

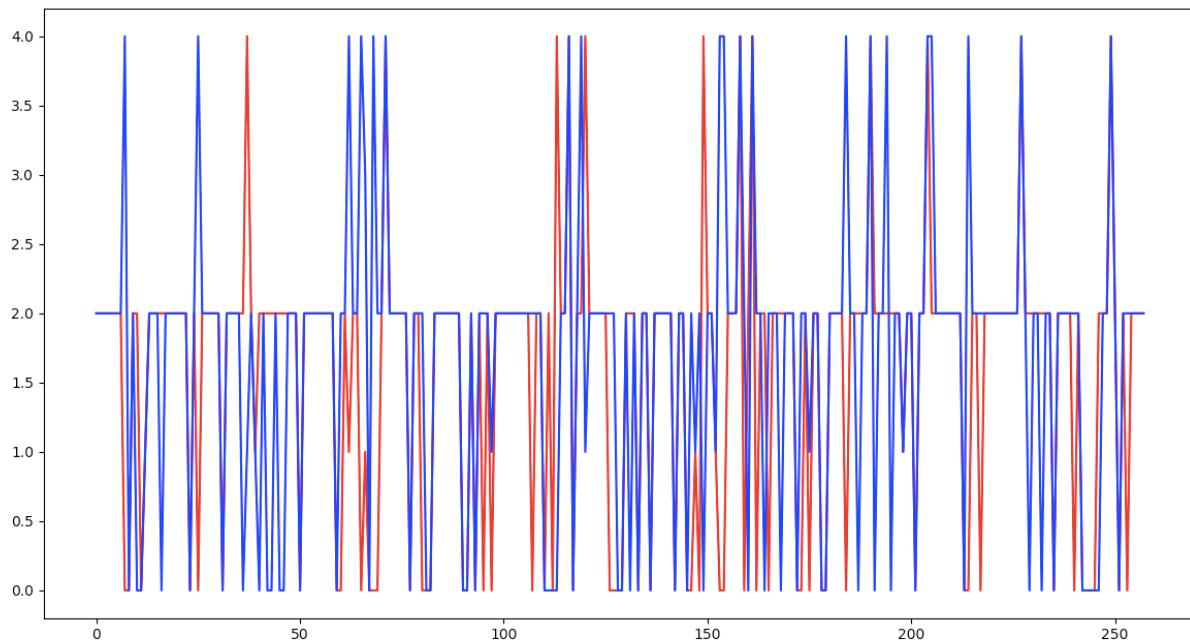


### 3. Random Forest Classification

\* RMSE

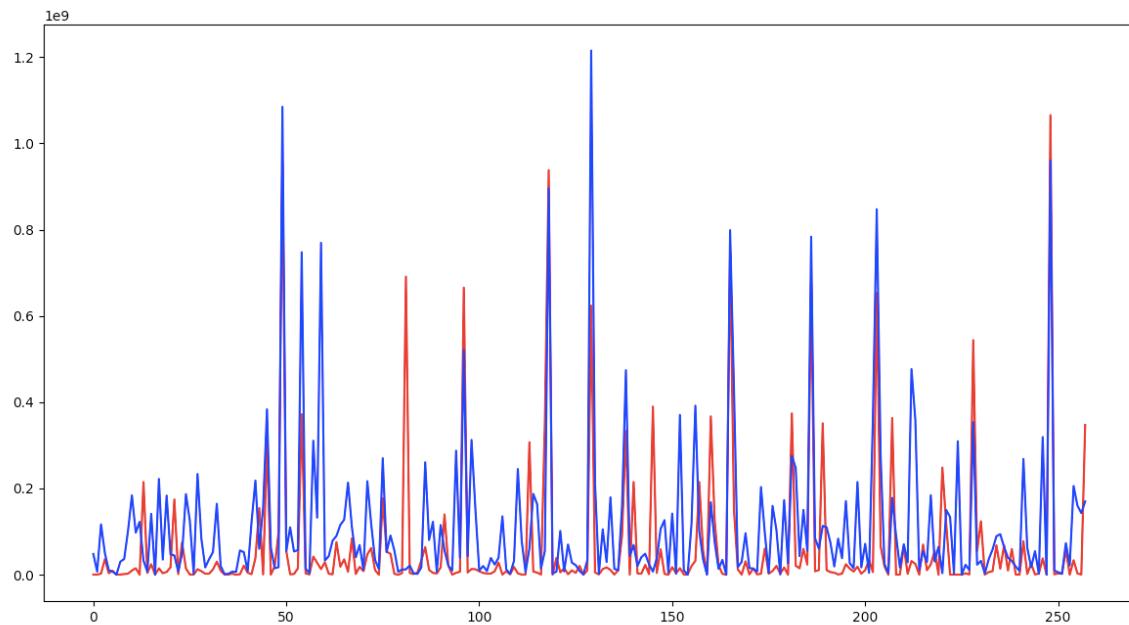


\* Classification error

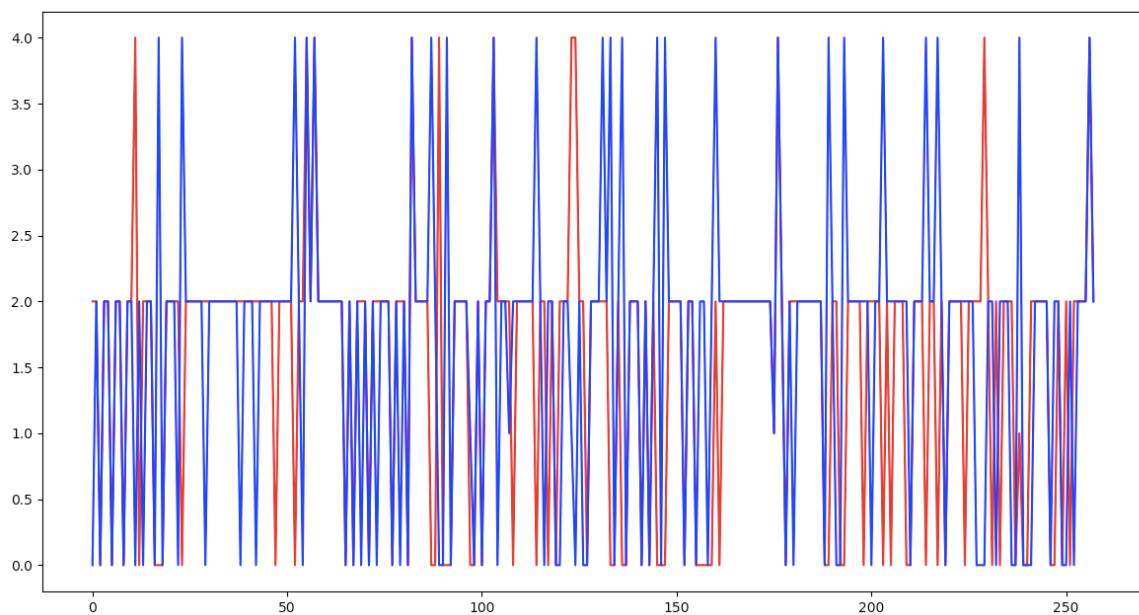


#### 4. Bagging (with Decision Tree as base estimator)

\* RMSE

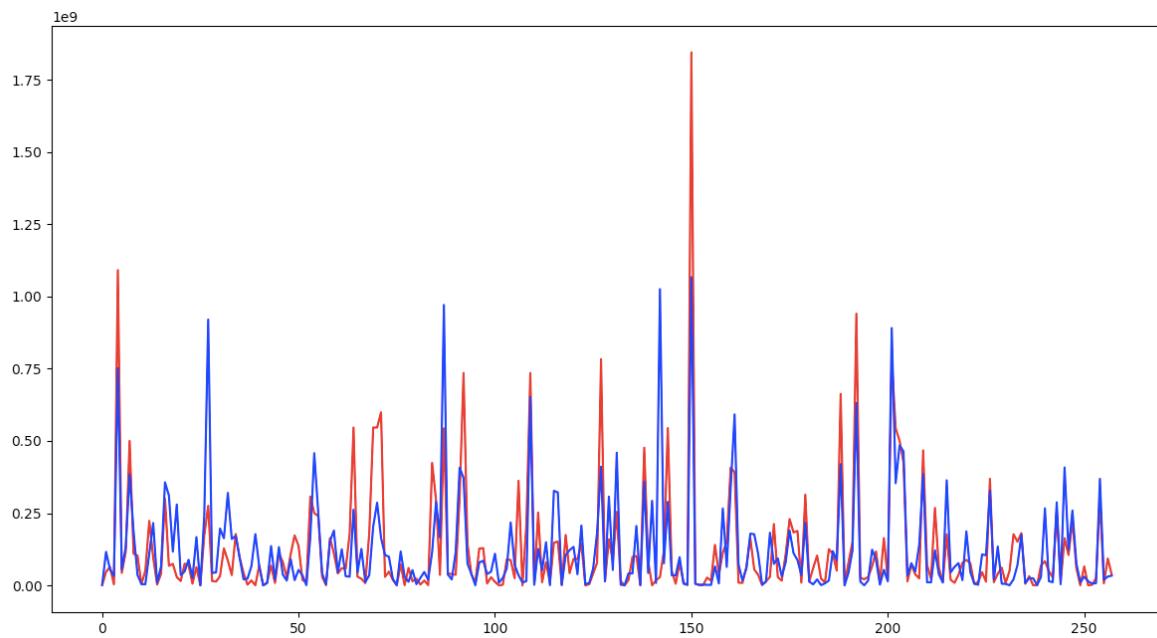


\* Classification error

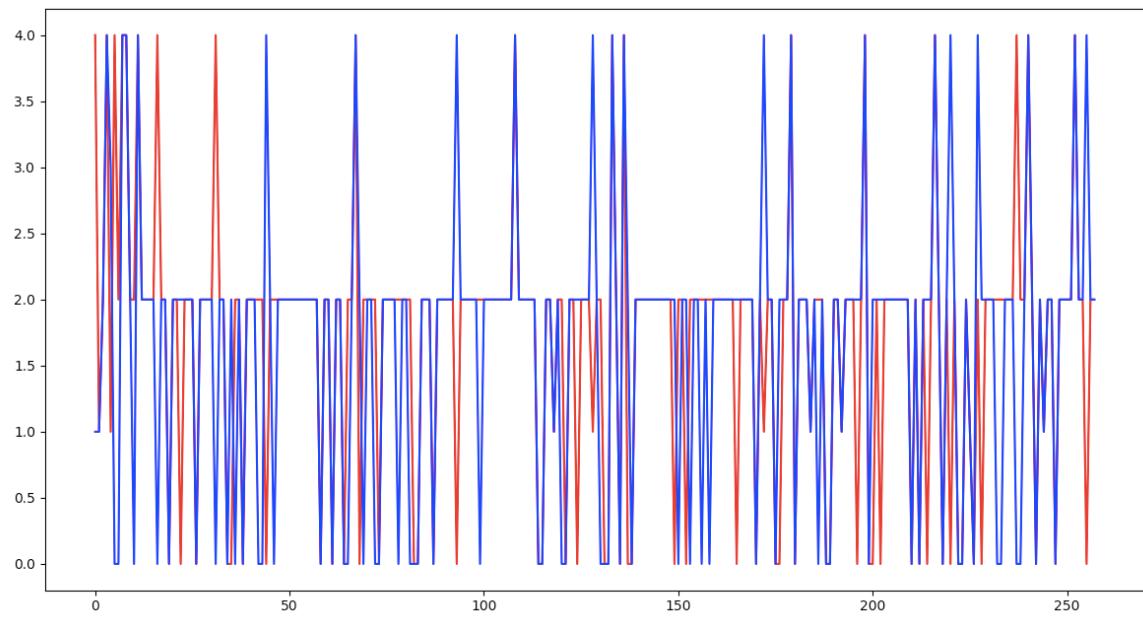


## Bagging (with Random Forest as base estimator)

\* RMSE

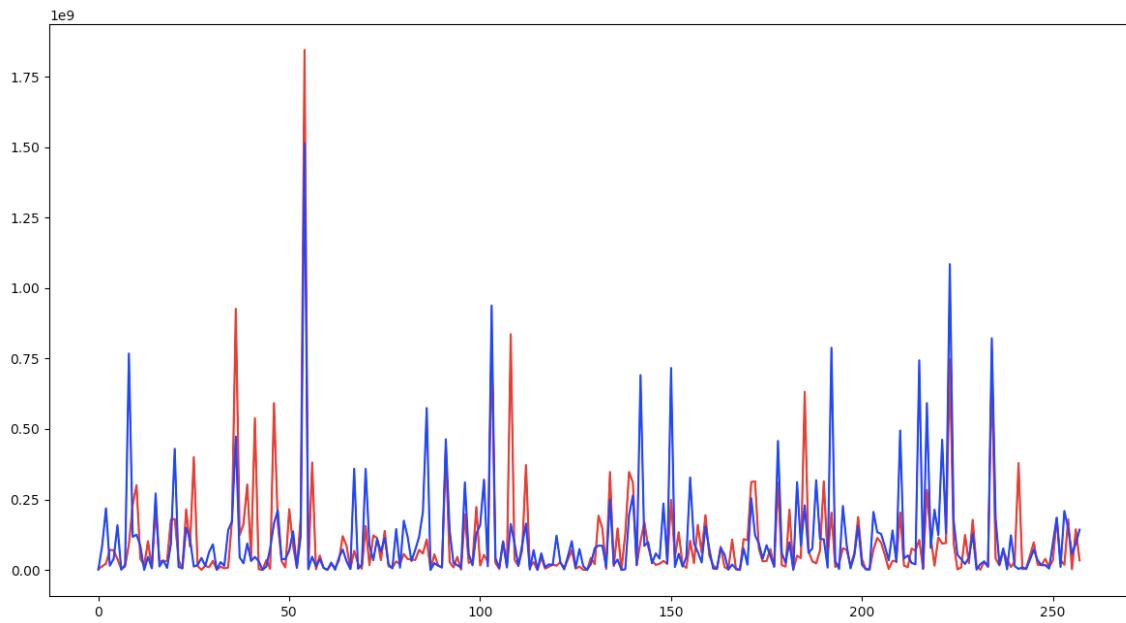


\* Classification error

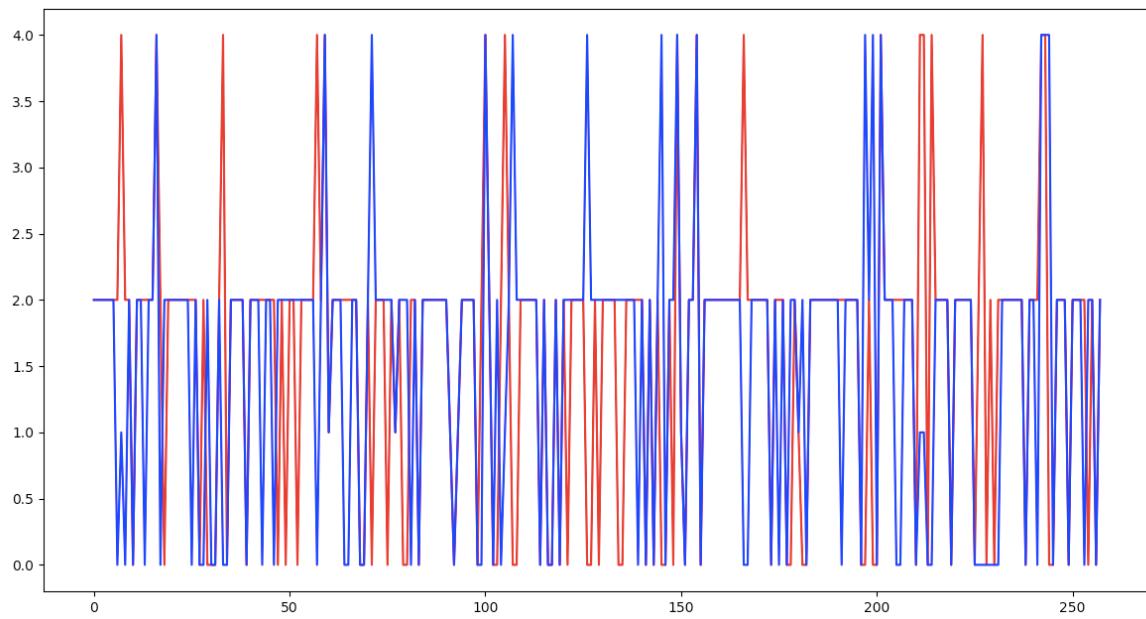


## Boosting (with Decision Tree as base estimator)

\* RMSE

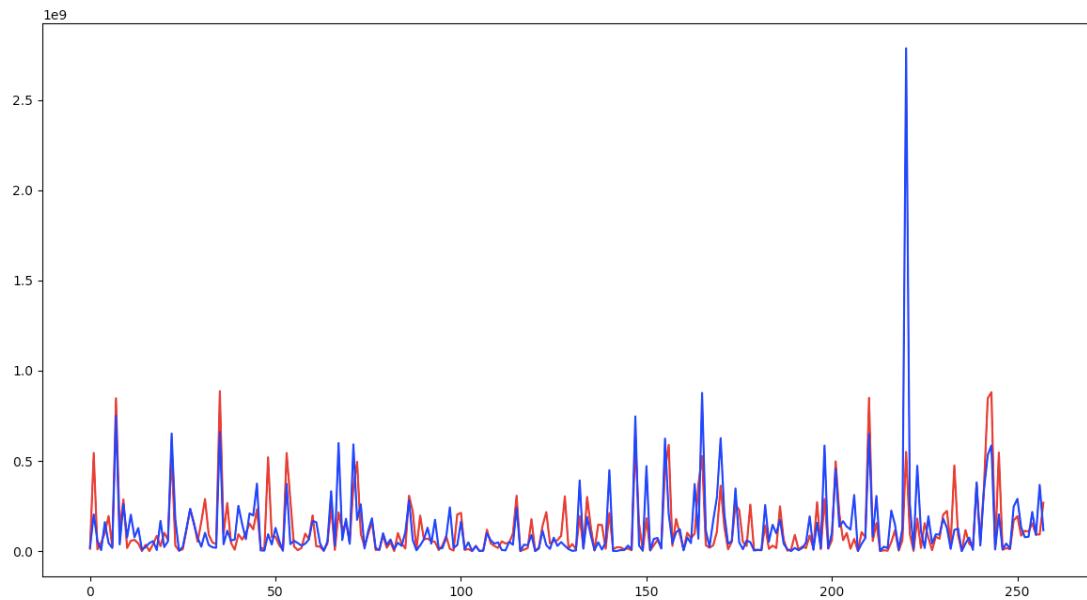


\* Classification error

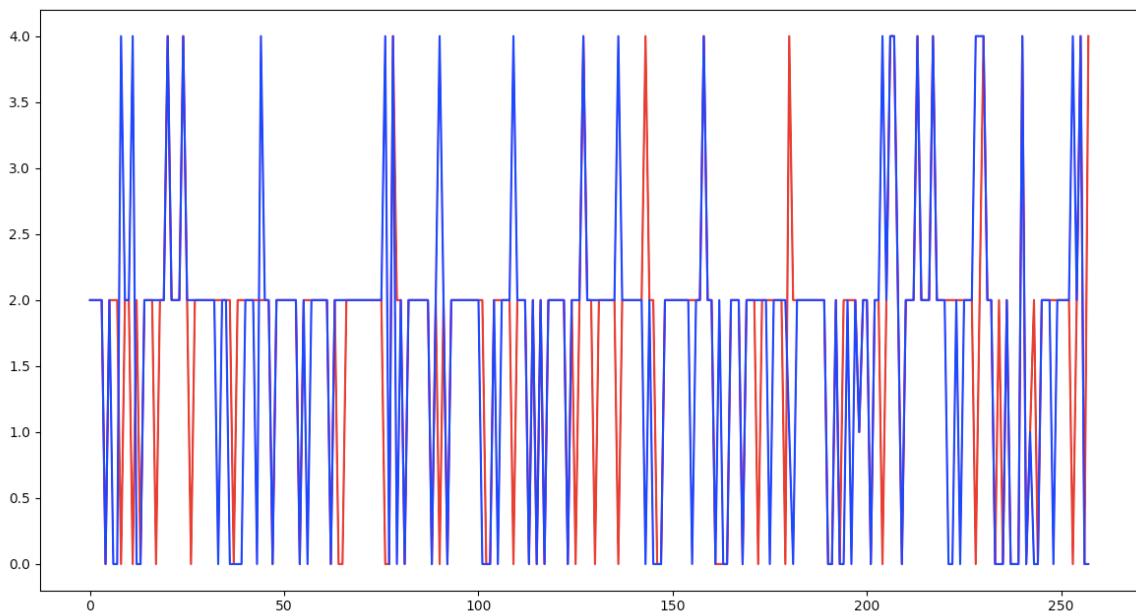


## Boosting (with Random Forest as base estimator)

\* RMSE



\* Classification error



### III. Comparison of the evaluation result between each classifier

In this evaluation, we would split our original dataset into a training set and testing set. After we trained our training set we implemented two ways to test our result in testing set. First one is RMSE, it is a frequently used measure of the difference between values predicted by a model and the values actually observed from the environment that is being modeled. RMSE serves to aggregate them into a single measure of predictive power. We also normalized the result of RMSE since we wanted to facilitate the comparison between different datasets, and we used the range (defined as the maximum value minus the minimum value) of the measured data.

Second evaluation method is classification error method. In the beginning, we used K-Means clustering to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. Then we trained training set and tested the result in testing set to see whether it can fall in the right partition, if it fall into the right section, we calculated how far it is from the right section, and divided total data number to see the diff rate. Below is a comparison table which record the two result from each classifier algorithm, the red part is the best result.

	Decision Tree	Gaussian Naive Bayes	Random Forest	Bagging		AdaBoosting		w/o Hashtags
base estimator				Decision Tree	Random Tree	Decision Tree	Random Tree	Random Tree
RMSE	0.131	0.175	0.087	0.113	0.0868	0.098	<b>0.0706</b>	0.0764
<i>Classification</i>								
avg correctness	0.75	0.2	0.79	0.74	0.81	0.79	<b>0.82</b>	0.75
avg diff	0.55	1.75	0.51	0.65	0.45	0.51	<b>0.45</b>	0.55