Data preprocessing

In [1]:
```
#Some preparation
import pandas as pd
import numpy as np
from data_preprocessing import *
from prediction_algorithm import *
#Load the data
df = load_data('database\score_data_set.csv')
```

In [2]:
```
#Check the shape of data
print('Shape of the original data: ', df.shape)

#Try to remove missing rows
df = remove_missing_value(df)
print('Shape of the processed data: ', df.shape)
```

```
Shape of the original data:  (2000, 20)
Shape of the processed data:  (1969, 20)
```

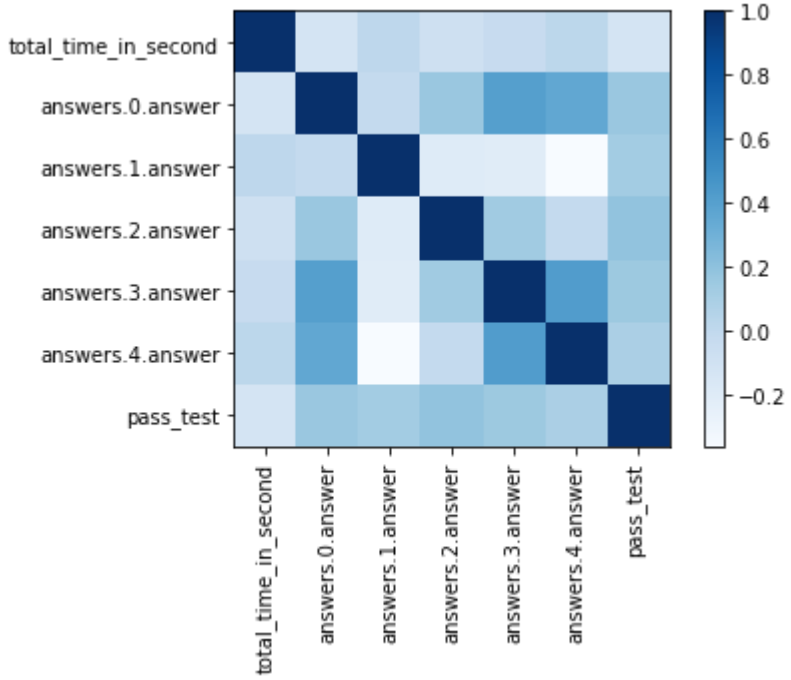
=> After removing the missing rows, we got 1969 row left, that means the number of missing data is quite small. So, we can use this way to handle missing data.

I. Examine some of the correlation between these variables.

Hypothesis: The values on answers.0/1/2/3/4.answer reflect the evaluation results for the answer.

=> Hence, We only examine some important numeric variables: total_time_in_second,pass_test, answers.0/1/2/3/4.answer.

In [3]:
```
#Create a new dataset for correlation computing:
new_df_corr = df[['total_time_in_second','answers.0.answer', 'answers.1.answer',
                  'answers.2.answer', 'answers.3.answer', 'answers.4.answer','pas
#Calculate and visulize the correlation result:
corr(new_df_corr)
new_df_corr.corr()
```



Out[3]:

	total_time_in_second	answers.0.answer	answers.1.answer	answers.2.answer
total_time_in_second	1.000000	-0.120255	0.014657	-0.080141
answers.0.answer	-0.120255	1.000000	-0.014738	0.163782
answers.1.answer	0.014657	-0.014738	1.000000	-0.196267
answers.2.answer	-0.080141	0.163782	-0.196267	1.000000
answers.3.answer	-0.029101	0.405196	-0.207265	0.135362
answers.4.answer	0.016201	0.358256	-0.362838	-0.016663
pass_test	-0.127719	0.162530	0.125053	0.189693

Some conclusion from the correlation table:

1. Candidates have a slightly similar result between question 0, 3 and 4.

2. Pass test does not depends on any single question result or total time.

=> Not many information could be seen here, if we want to predict pass test, a machine learning algorithm is necessary. If you want me to perform more variables' correlation, it is ok. But I don't think it will be a good ideal. it is often not useful when we try to find some knowledge hidden in single variables.

II. Build up a model to predict the Pass/Fail of the student and then give out the conclusion

We have only nearly 2k datas to build the model. Hence, to avoid overfitting I only choose of the important features of the machine learning algorithm.

Note: Choosing features step based on the hypothesis that the values on answers. 0/1/2/3/4 .answer reflect the evaluation results for the answer.

In [4]:
```
#Choose some important features:
data =  df[['test_id','objective','total_time_in_second','pass_test','answers.0.a
                  'answers.2.answer', 'answers.3.answer', 'answers.4.answer']]
```

In [5]:
```
#Divide the data set into 2 set: Training set (70%) and test set (30%)
#X will be the predicting model and y will be the test result.
X_train, X_test, y_train, y_test = prepare_data(data)
```

In [6]:
```
#Build the model based on decision tree algorithm:
model = prediction_model(X_train, y_train)

# Test score on training set:
print('Test score on training set:')
prediction_result(model, X_train, y_train)

#Test score on test set:
print('Test score on test set: ')
prediction_result(model, X_test, y_test)
```

```
Test score on training set:
f1 score:  0.999534667287
Accuracy score: 0.999274310595
Test score on test set:
f1 score:  0.941311852704
Accuracy score: 0.913705583756
```

Conclusion:

1. The accuracy score on test data is highly acceptable (~94%). It also means my hypothesis could be true.

2. The difference between test score on training data and test data is roughly 5%, which is caused by a little overfitting due to the shortage of data.

3. To be honest, we can slightly increase a bit of the test score on test data by some tricks. However, it does not really matter and may cause some risks when applying the model to the real data because we only test on 30% of 2k datas (600 samples), extremely few to conclude something.

=> If you want me to carefully explain about my using machine learning algorithm, just tell me.