**IS590 assignment 5**

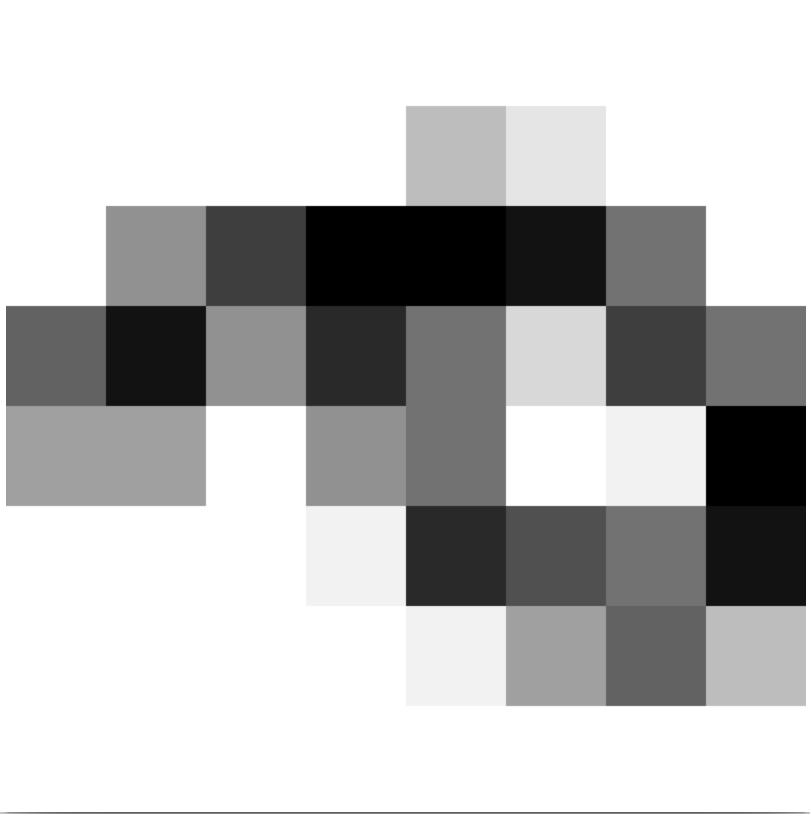
**yuweic3**

**2017/10/6**

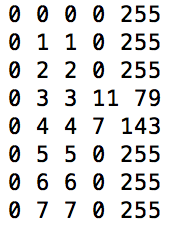
**b)**

(Actually the program and the visualization is really misleading as the visualization is rotated.)

Here is the first instance with digit of 6. The 8x8 grid below is what I get after running the program



The following 8 tuples are corresponding with 8 pixels in the first column of the grid.

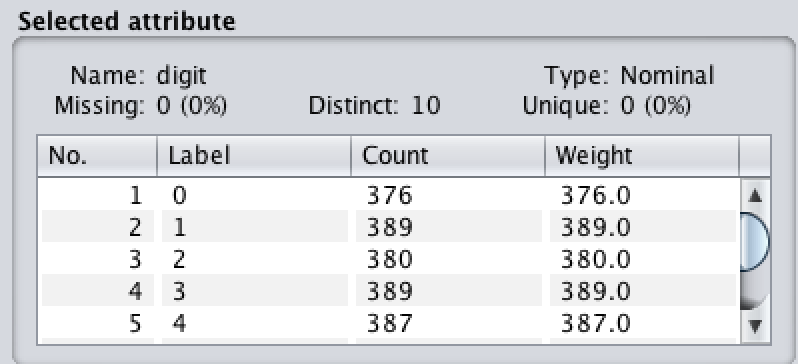


The first two digits represent the column and row info. The third digit represents the index of pixel. The forth value is used to produce the intensity which is the fifth value. The color of pixel is determined by the intensity.

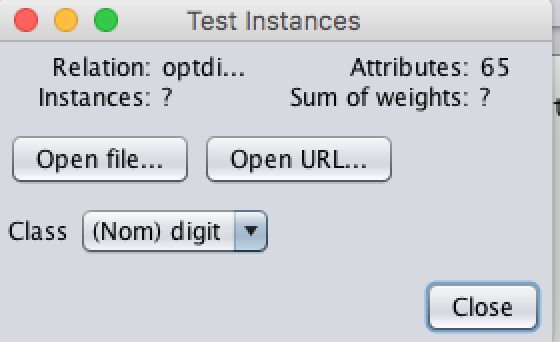
For example, the color of the first pixel in the first column is white (RGB: 255,255,255) with corresponding intensity of 255. Similarly, the forth pixel of the first column is gray with RGB value of (79, 79, 79) and intensity of 79.

**c)**

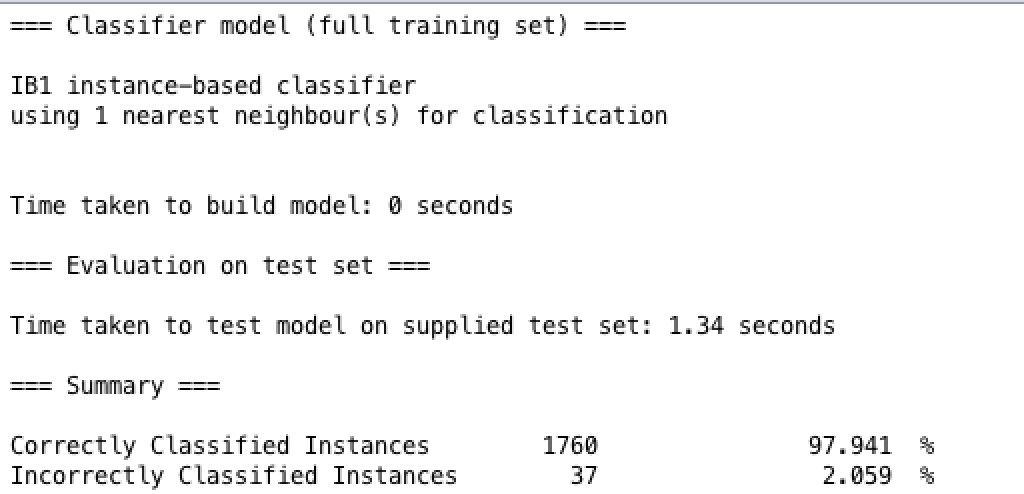
First we need to preprocess the training dataset and test dataset by converting the type of “digit” attribute from numeric to nominal.



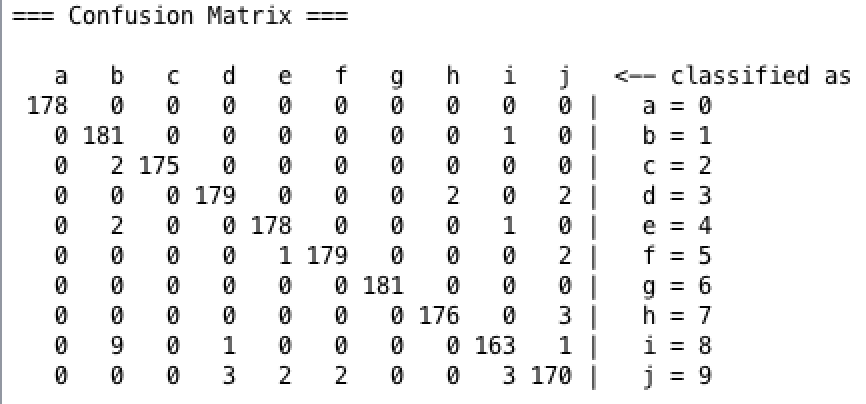
Then we use IBK with different k to train a classifier with training set and test on test set by setting “Supplied test set” option.



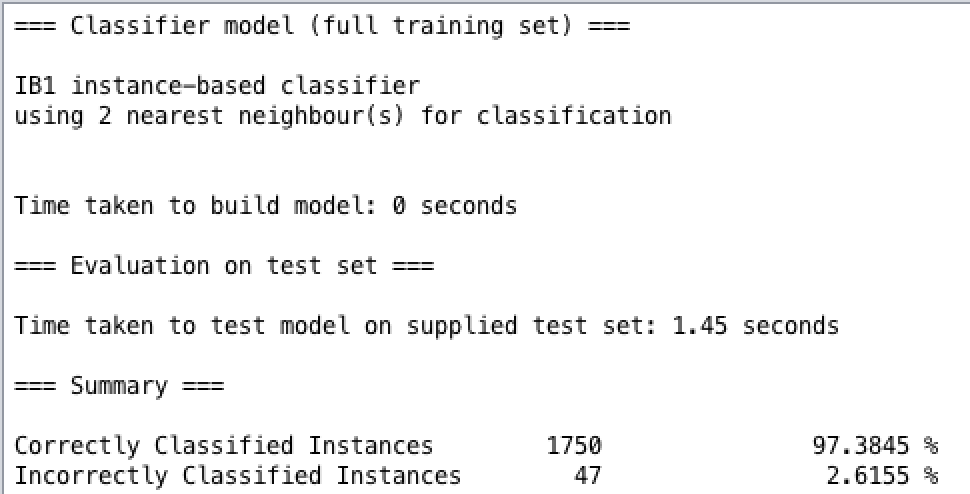
**# When k = 1, the accuracy = 97.941%**



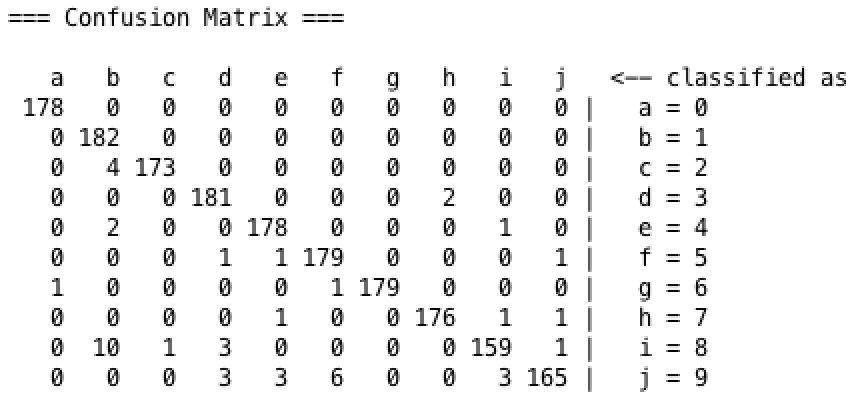
The corresponding confusion matrix: (The highest wrong classification is 8)



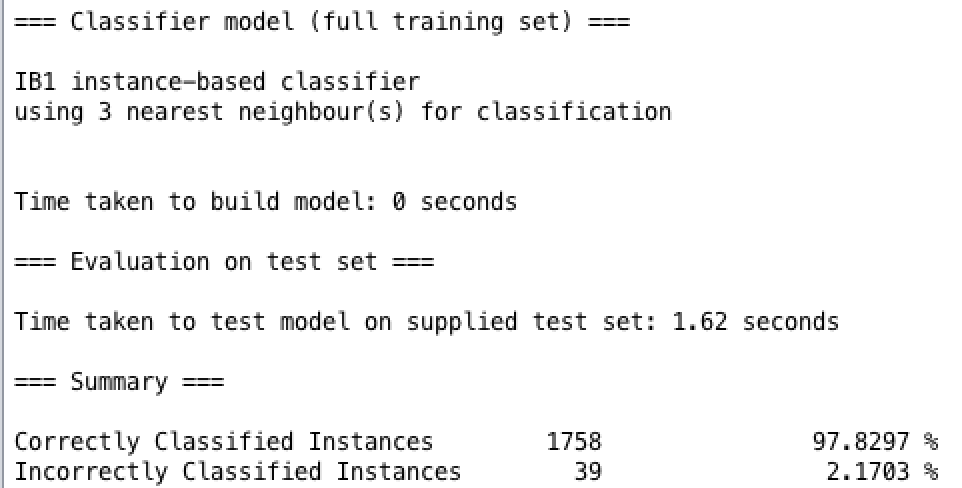
**# When k = 2, the accuracy = 97.3845%**



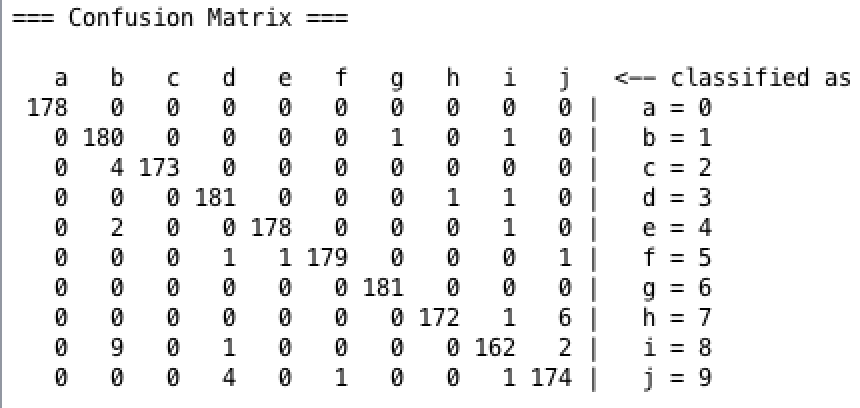
The corresponding confusion matrix: (The highest wrong classification is 8)



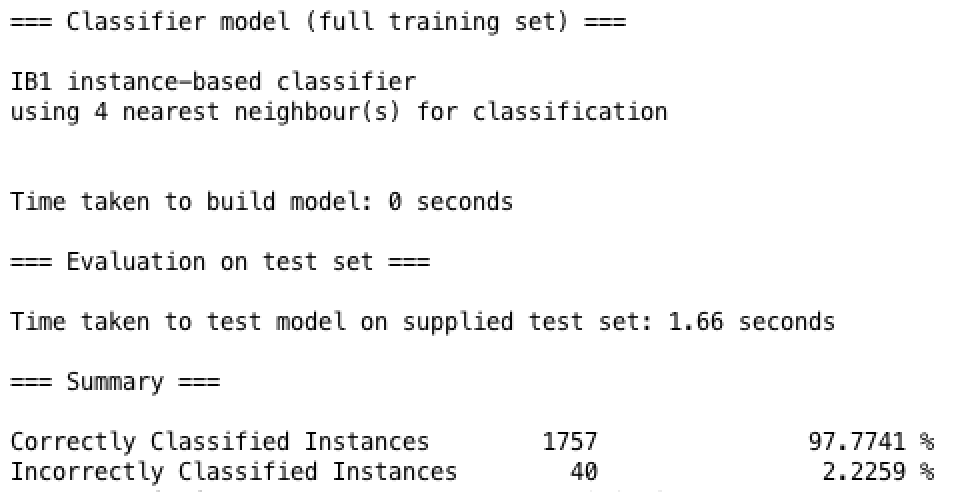
**# When k = 3, the accuracy = 97.8297%**



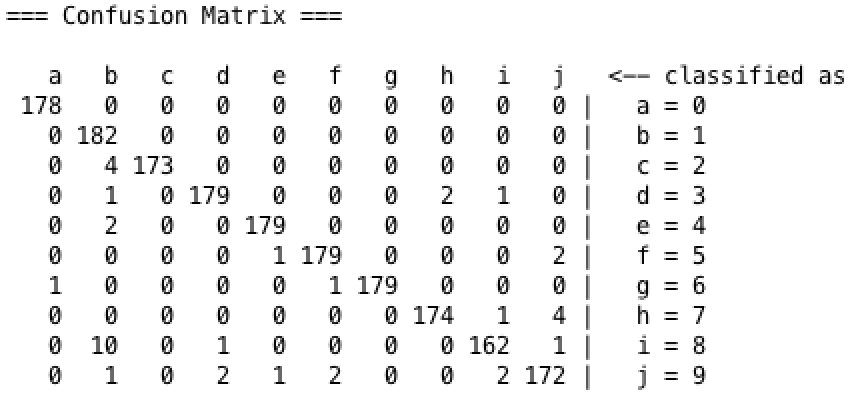
The corresponding confusion matrix: (The highest wrong classification is 8)



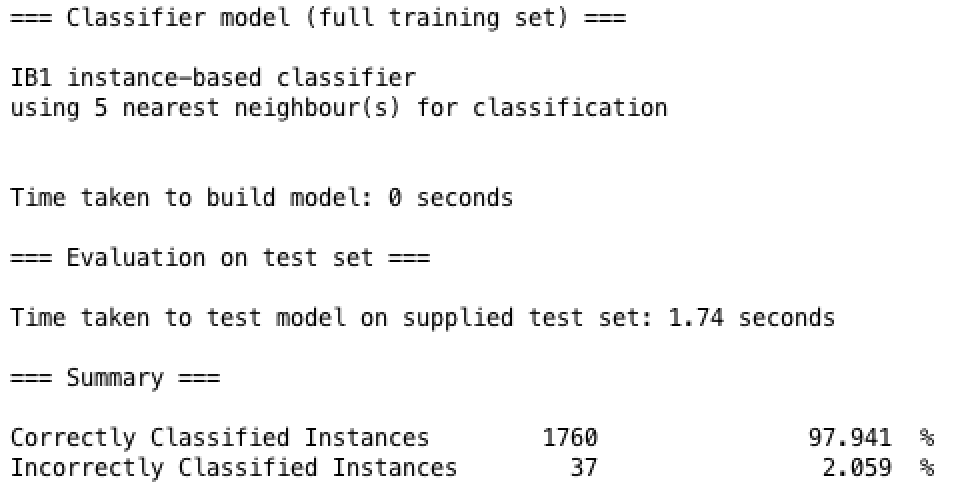
**# When k = 4, the accuracy = 97.7741%**



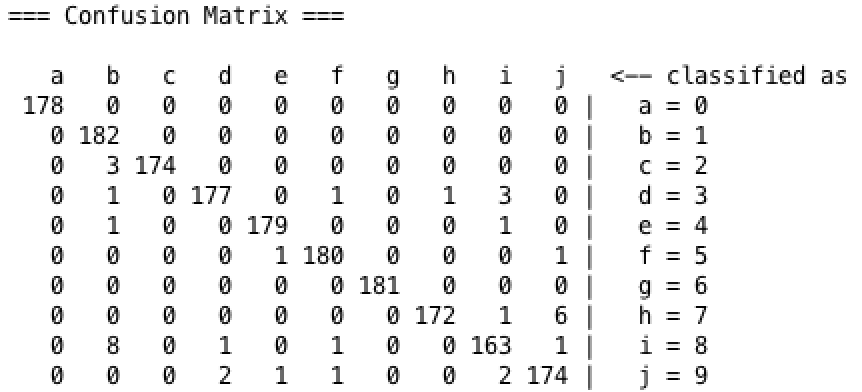
The corresponding confusion matrix: (The highest wrong classification is 8)



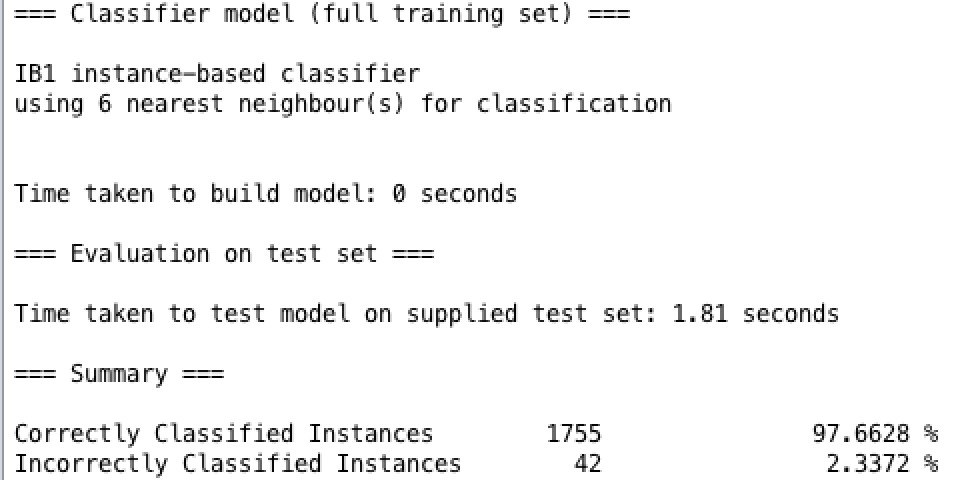
**# When k = 5, the accuracy = 97.941%**



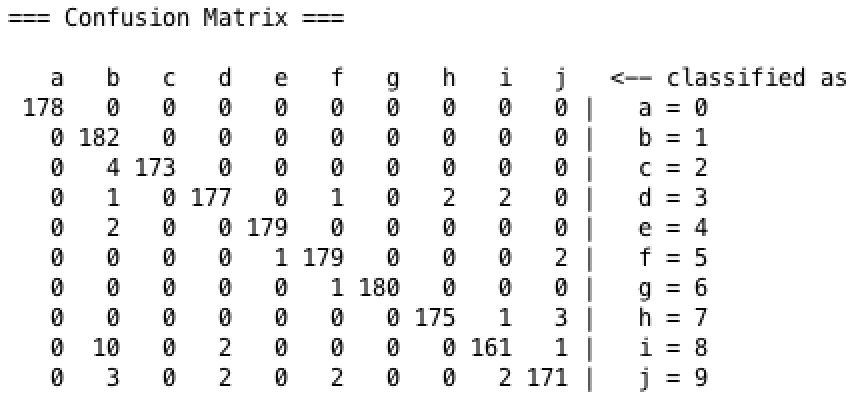
The corresponding confusion matrix: (The highest wrong classification is 8)



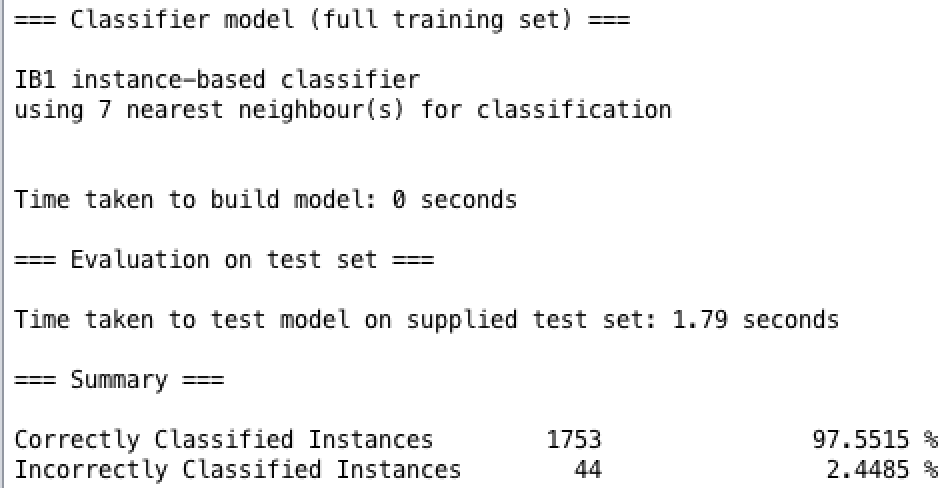
**# When k = 6, the accuracy = 97.6628%**



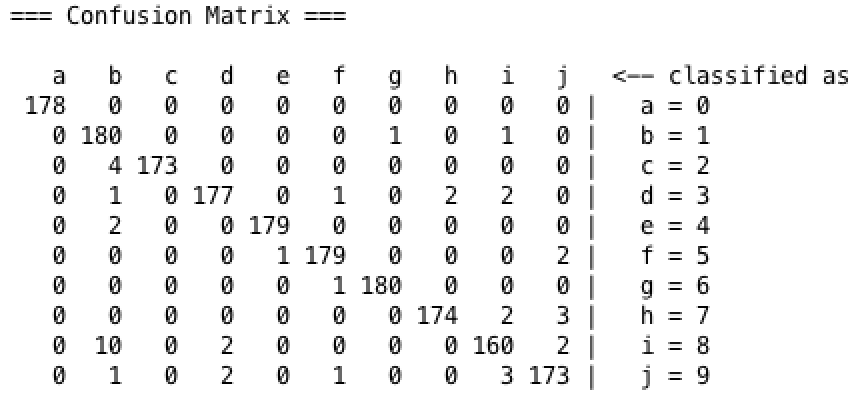
The corresponding confusion matrix: (The highest wrong classification is 8)



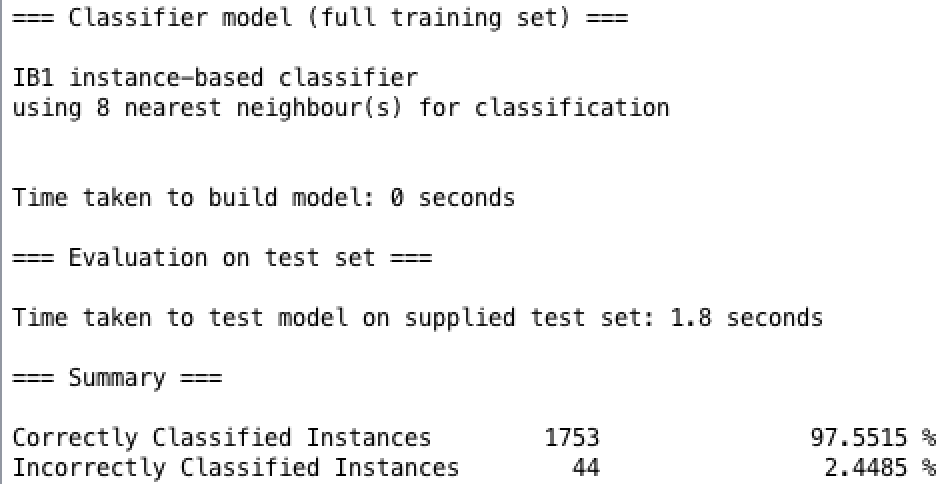
**# When k = 7, the accuracy = 97.5515%**



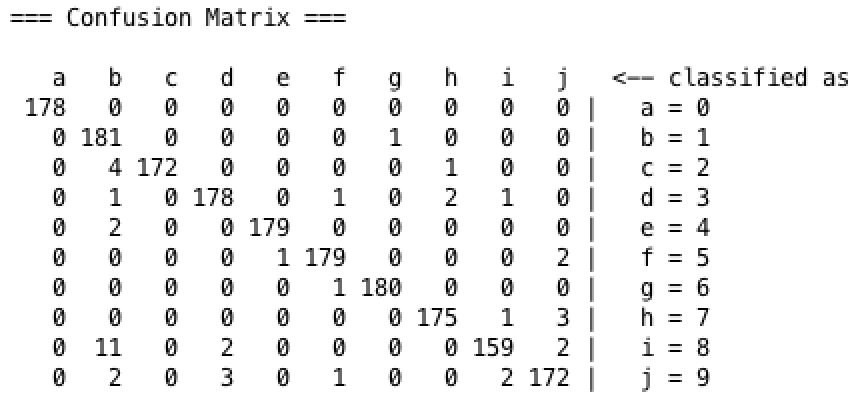
The corresponding confusion matrix: (The highest wrong classification is 8)



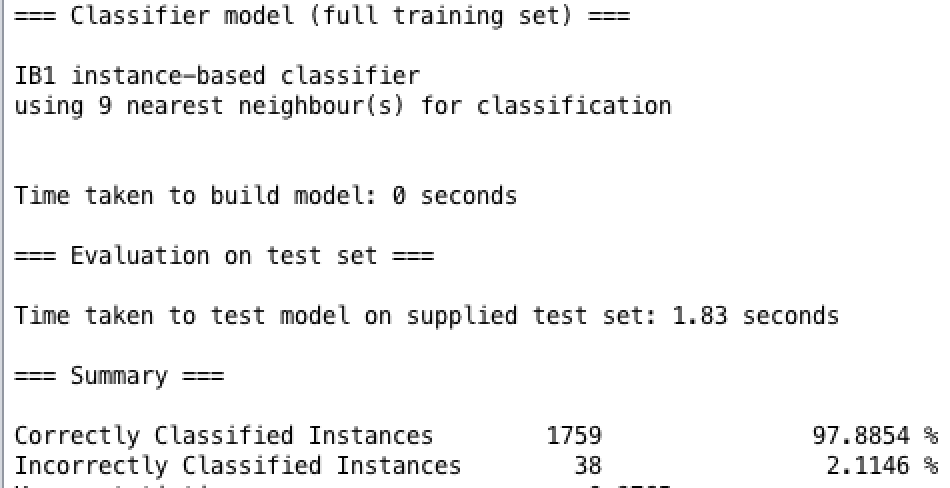
**# When k = 8, the accuracy = 97.5515%**



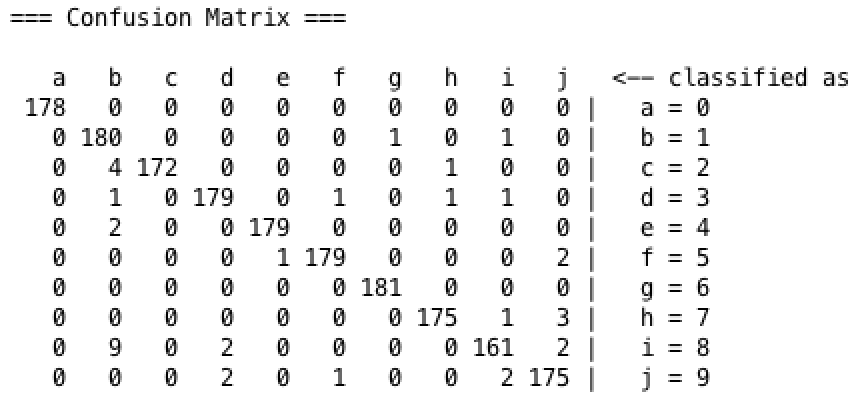
The corresponding confusion matrix: (The highest wrong classification is 8)



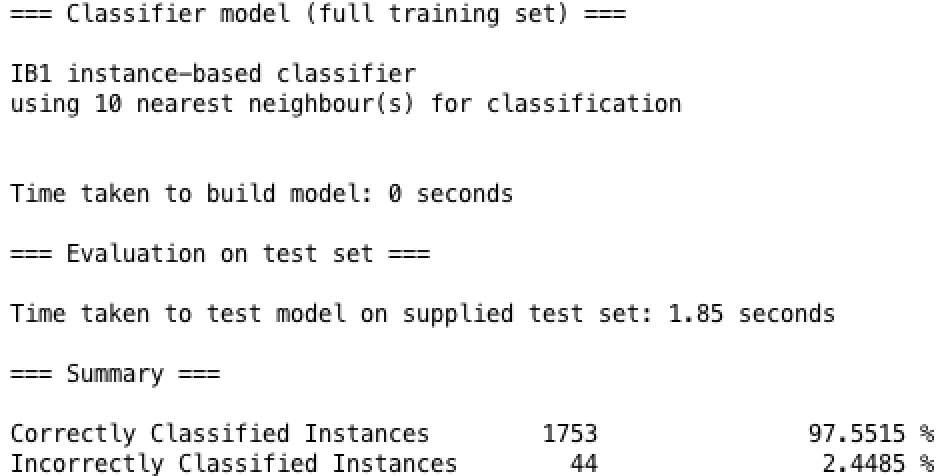
**# When k = 9, the accuracy = 97.8854%**

****

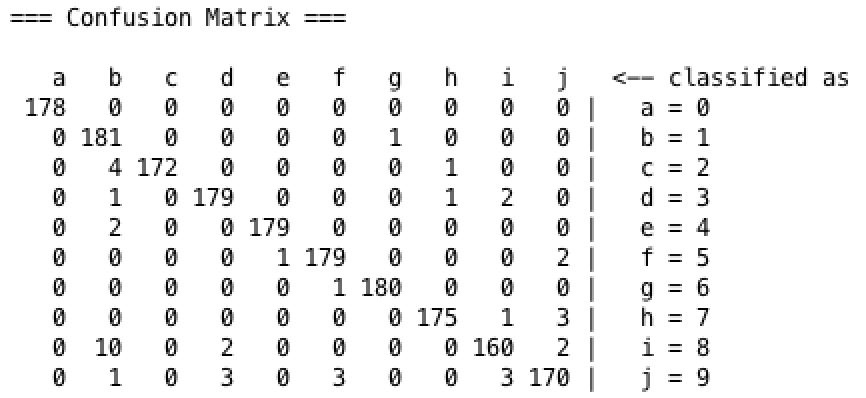
The corresponding confusion matrix: (The highest wrong classification is 8)



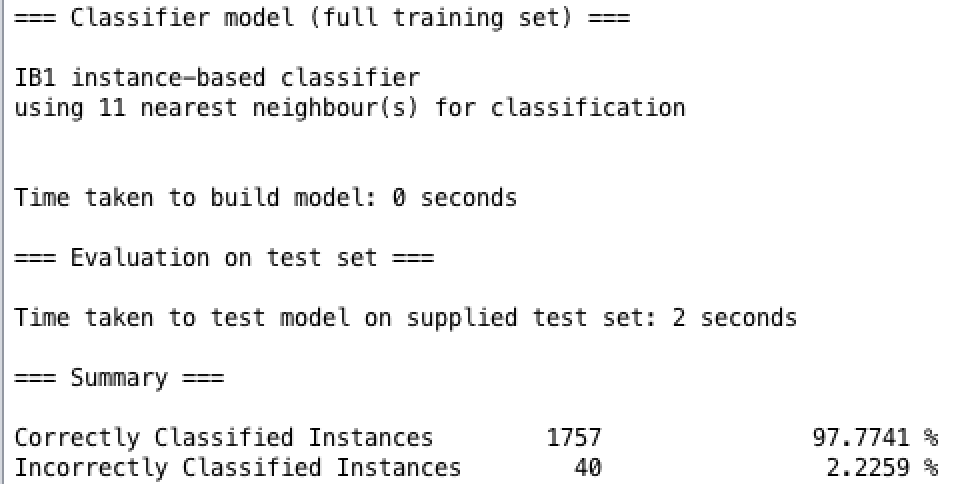
**# When k = 10, the accuracy = 97.5515%**



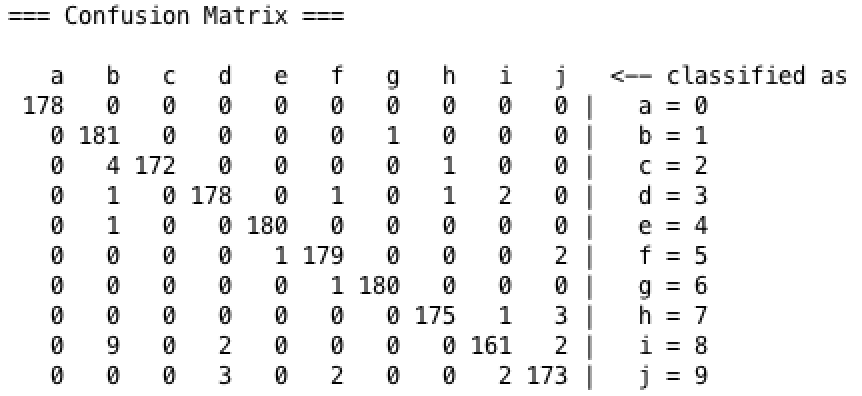
The corresponding confusion matrix: (The highest wrong classification is 8)



**# When k = 11, the accuracy = 97.7741%**



The corresponding confusion matrix: (The highest wrong classification is 8)



According to the experiment results, we find that when k=1, we have the best accuracy performance with 97.974% since we fit our model to the 1-nearest point and have a very low bias. However, the corresponding variance is really high as optimizing on only 1-nearest point means that the probability that we also model noise in the data is really high.

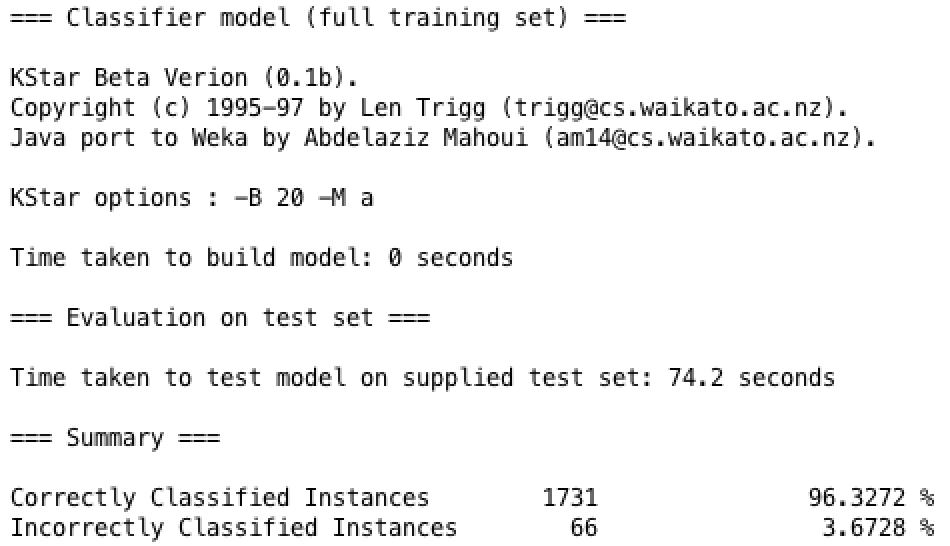
Also from the confusion matrices we get, we find that digit 8 is the easiest one to be classified into wrong class, while digit 1 is the easiest one to be distinguished.

**d)**

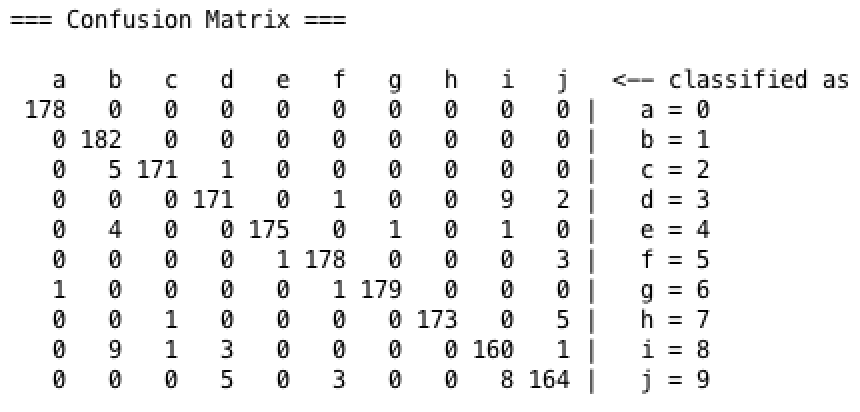
**KStar classifier**

*(K\* is an instance-based classifier, that is the class of a test instance is based upon the class of those training instances similar to it, as determined by some similarity function.)*

The accuracy performance is 96.3272%



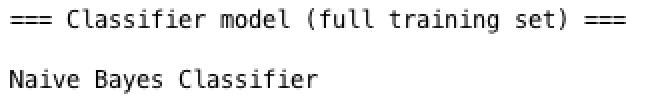
The corresponding confusion matrix is as following:

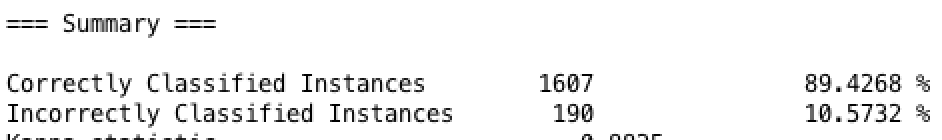


**Non instance based classifier**

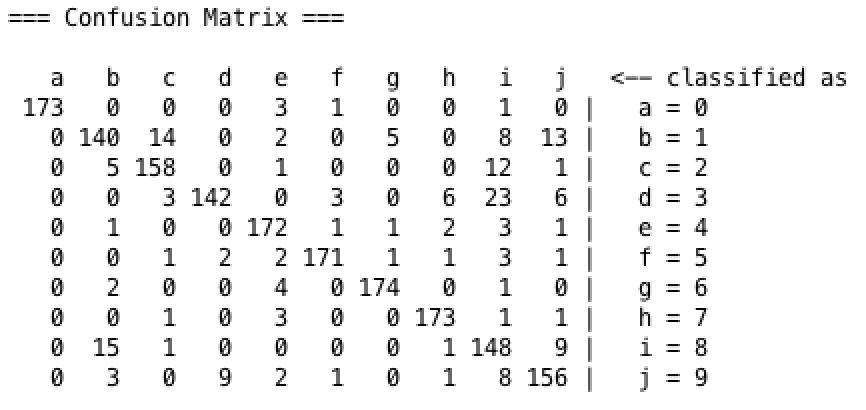
**#1 NaiveBayes**

The accuracy performance is 89.4268%



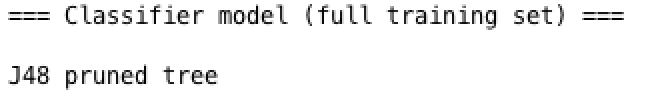


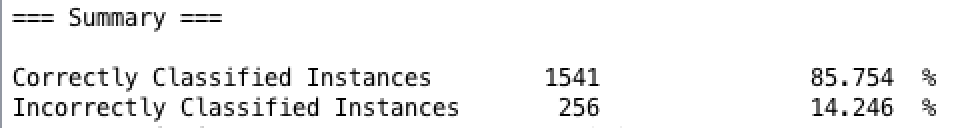
The corresponding confusion matrix is as following:

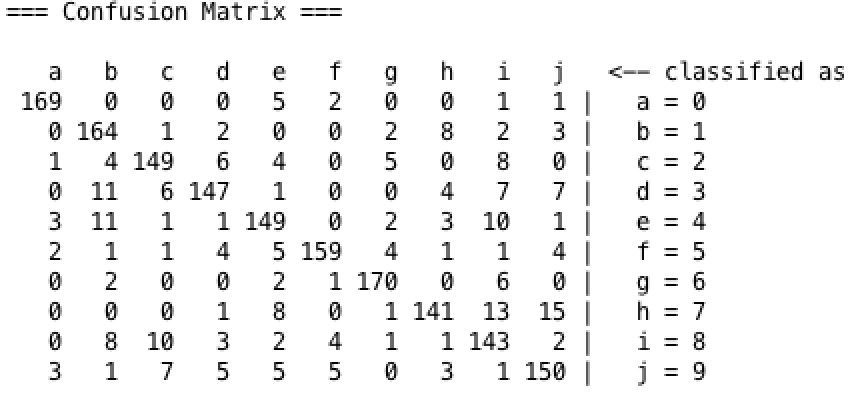


**J48 Tree**

The accuracy performance is 85.754%







In respect of accuracy performance, we can say that instance-based classifier （IBK, Kstar）outperforms the non-instance based classifier. In this problem domain, instance-based classifiers have better performance as Instance based classifier approximates the objective function locally for each query. And the model is trained using the training dataset, which is somehow similar to test set.

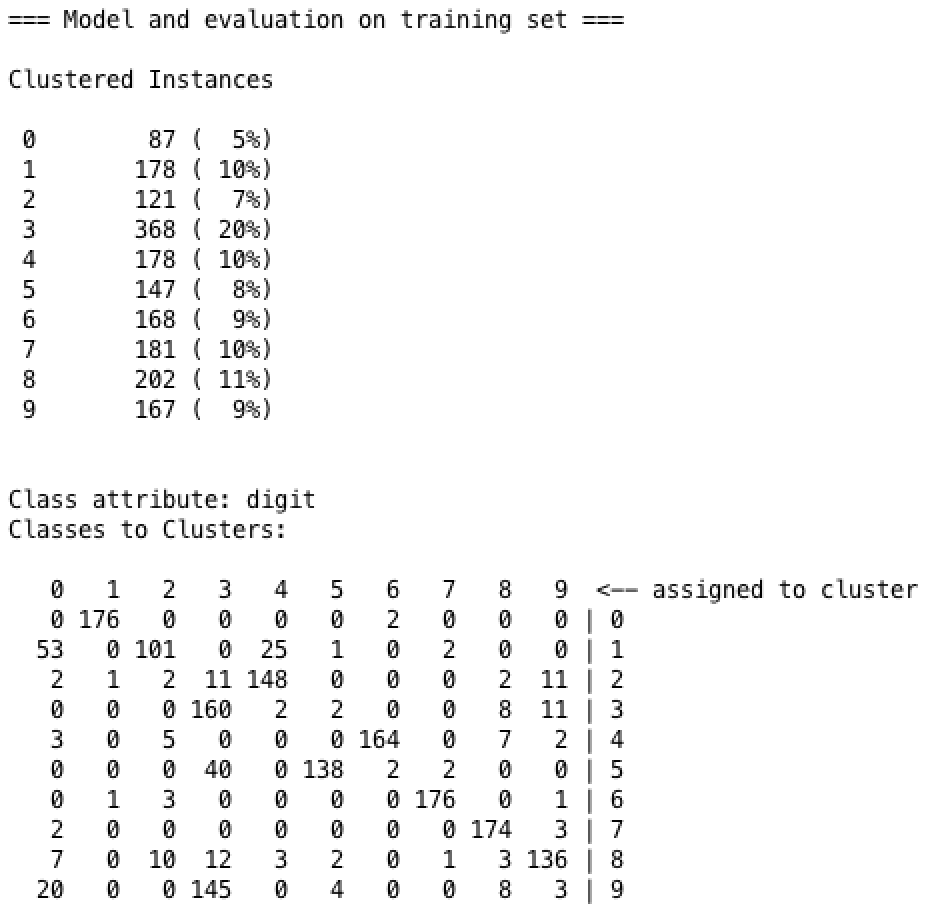
So if we use the model on test set, it can have better performance than those model trained using fixed rules.

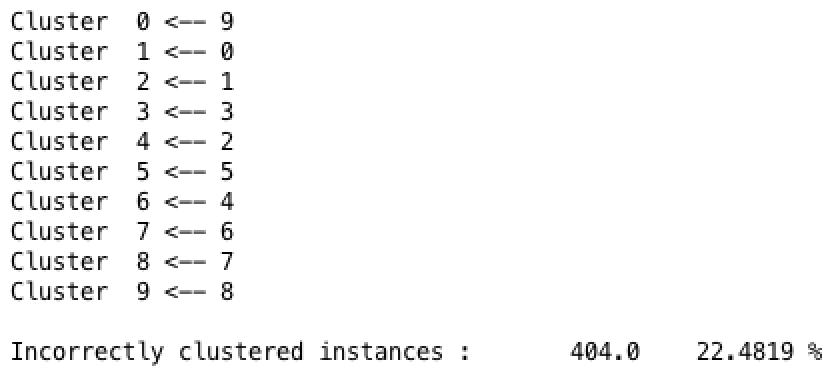
**e)**

Compare the accuracy performance of cluster and instance based classifier, we find that instance based classifier has better performance with higher accuracy. For classifier, we have already use training set to train a model and use such model as the prerequisite knowledge to test on test set. However, for the cluster, we know little about the data and the classification result is reached just from analyzing the features of data.

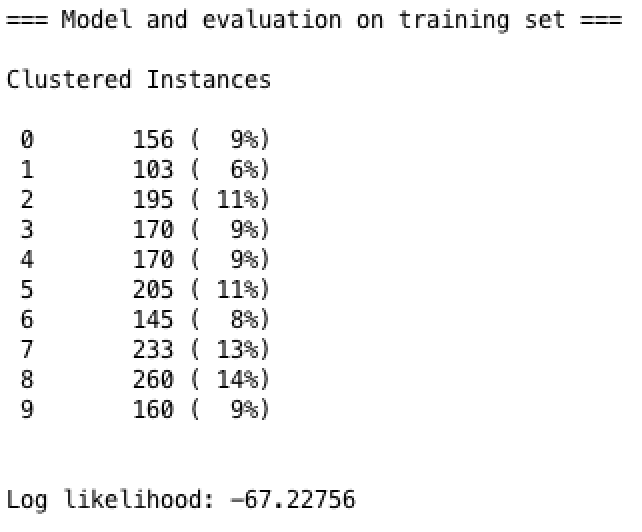
**SimpleKMeans** (with numClusters = 10, others are used with default setting)

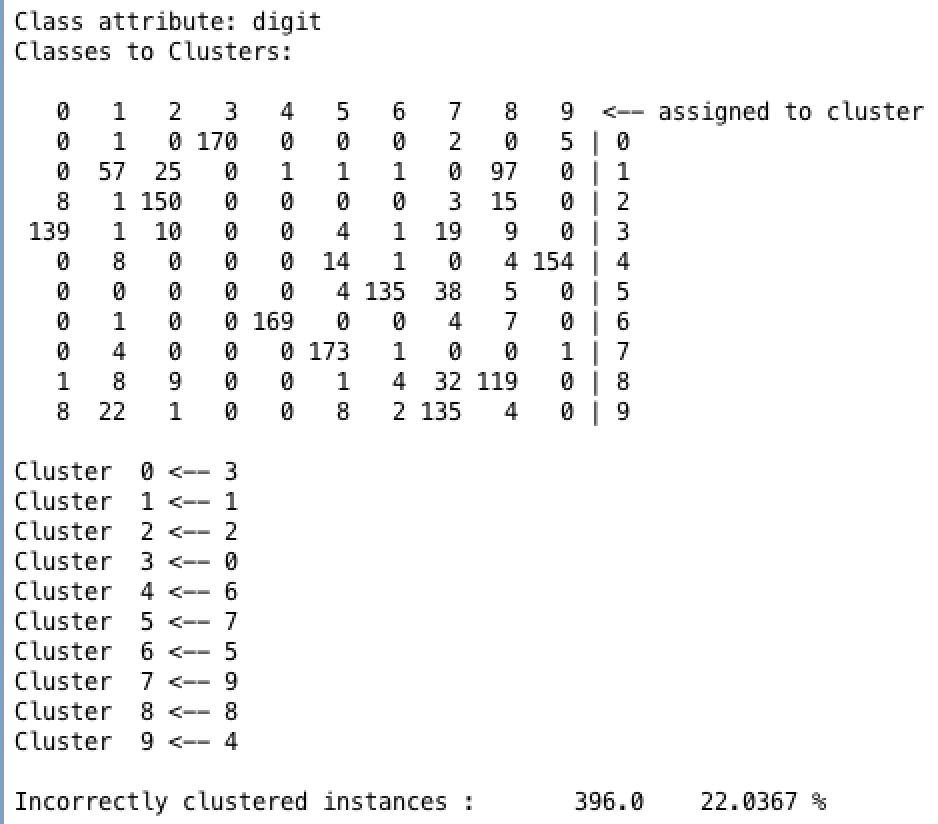
Set the cluster mode to “Classes to clusters evaluation with (Nom) digit”





**EM (numCluster = 10)**

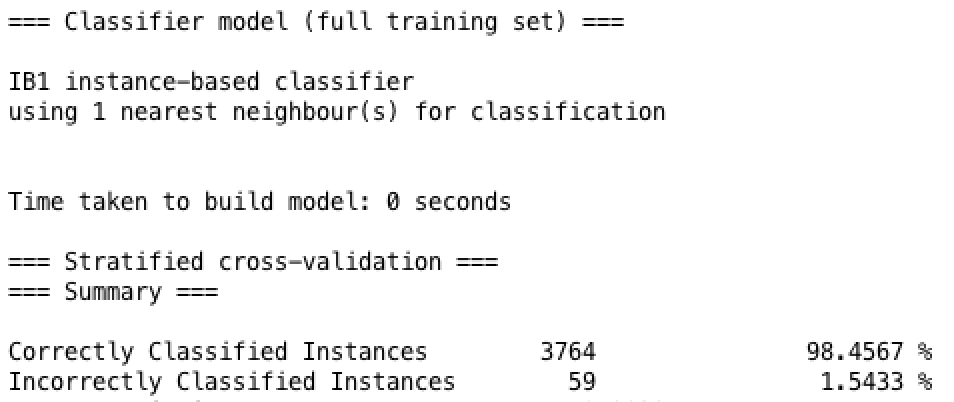


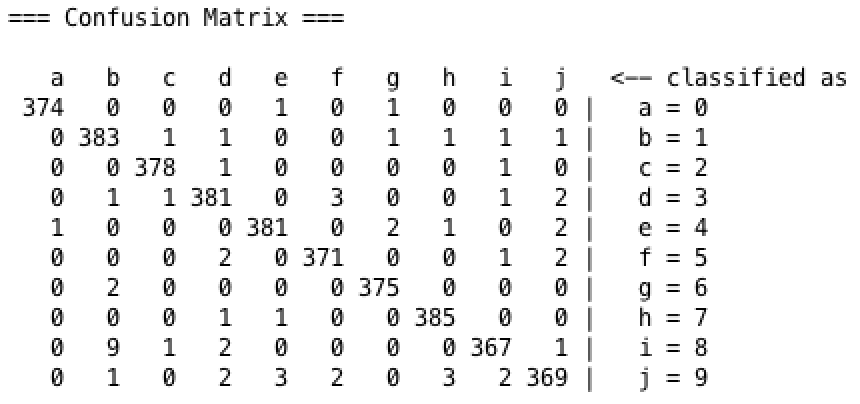


**f)**

The classifier I pick is **IBK K=1 with 10-fold cross validation**

**We get accuracy of 98.4567%**





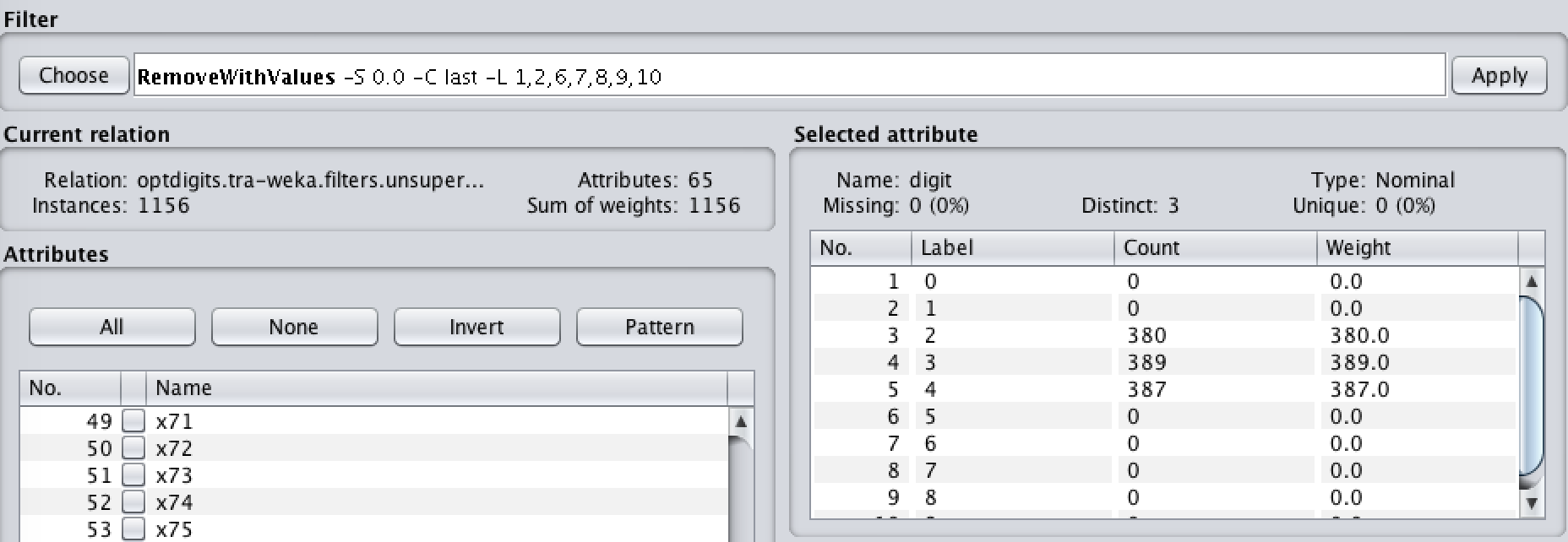
Obviously we find that the accuracy improved after applying 10-fold cross-validation.

Without cross validation we only have information on how does our model perform to in-sample data. Ideally, we would like to see how does the model perform when we have a new data in terms of accuracy and reliable performance of its predictions.

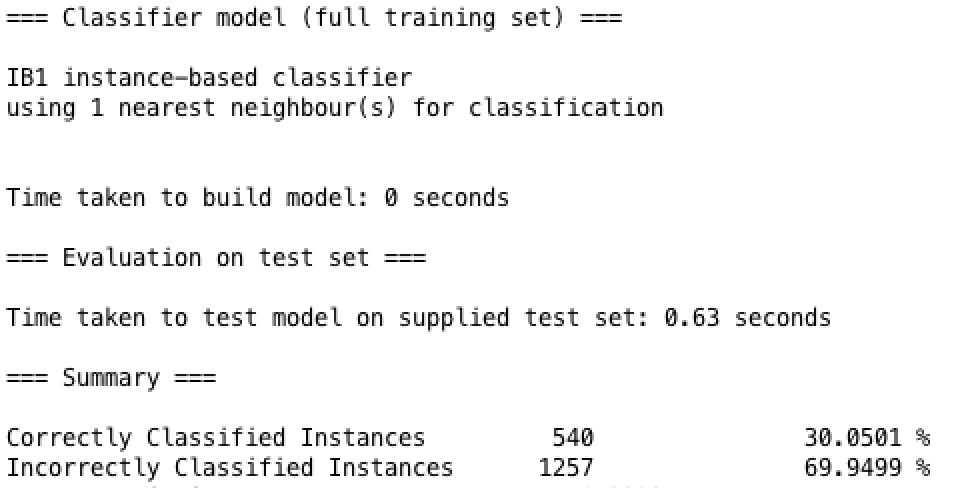
**g)**

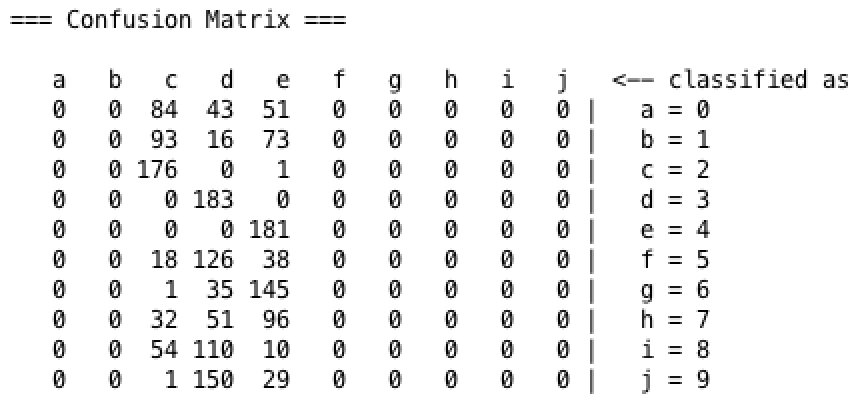
First we need to preprocess the training set by remove some instance using **RemoveWithValues** filter to just keep instance with tag 2, 3 and 4 like following.

**1’ (In other word, k = 3)**

****

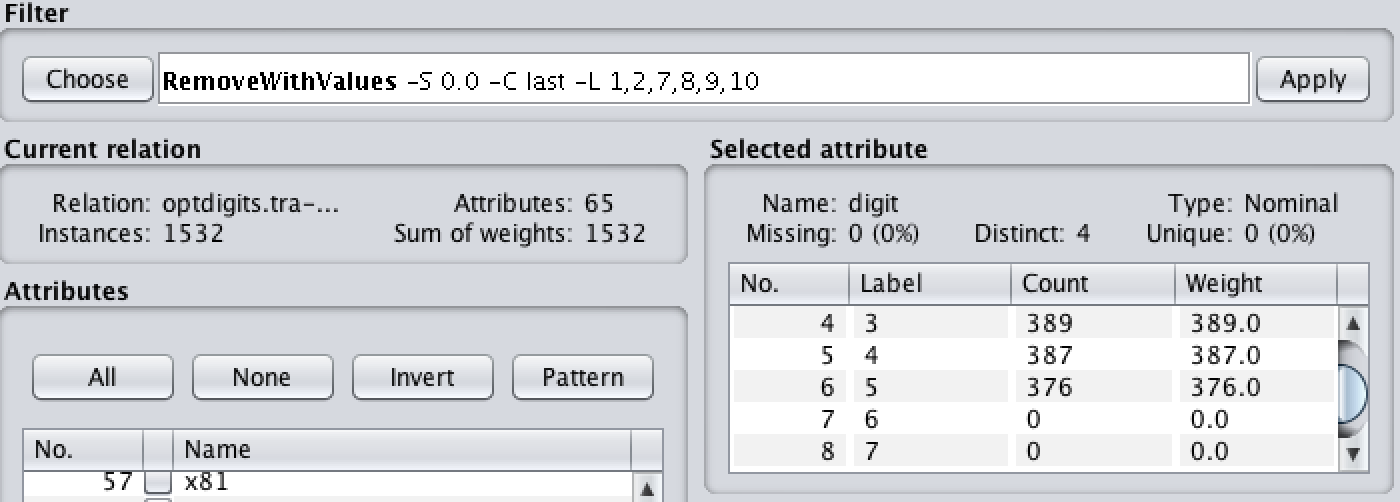
Then we apply IB1 classifier on test set:

****

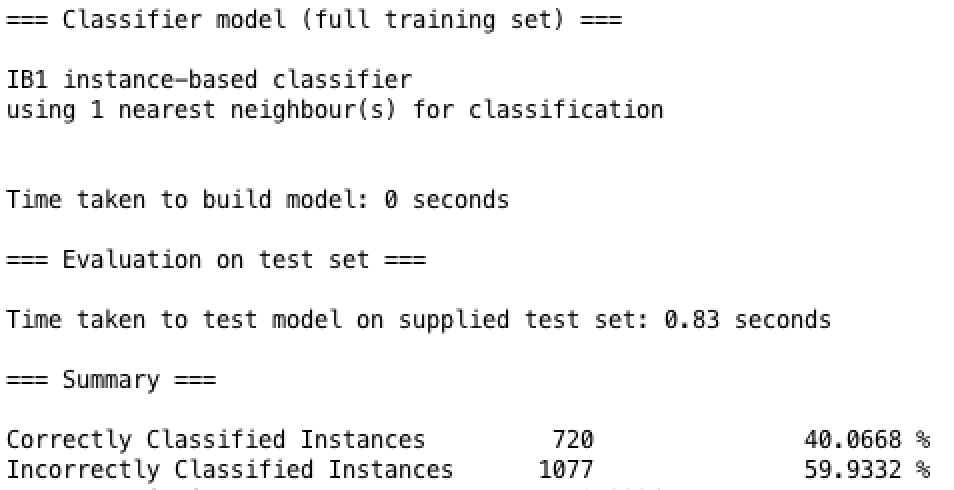
****

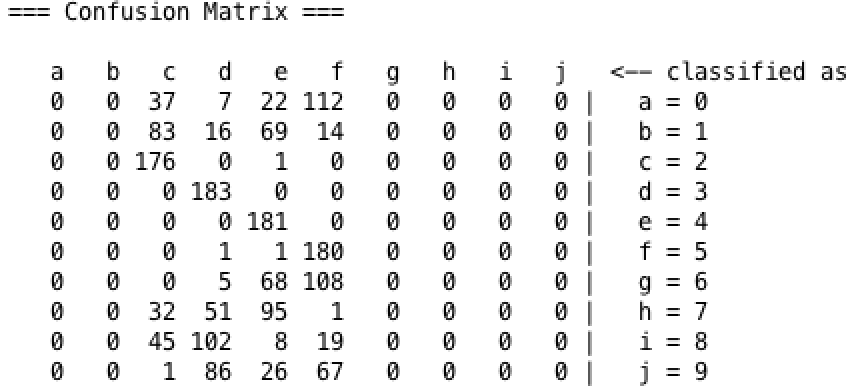
We find that when we have 3 classes prepared for the classifier training, the accuracy performance is not good with 30.0501%.

**2’ (k = 4)** Just keep instance with tag 2, 3, 4 and 5



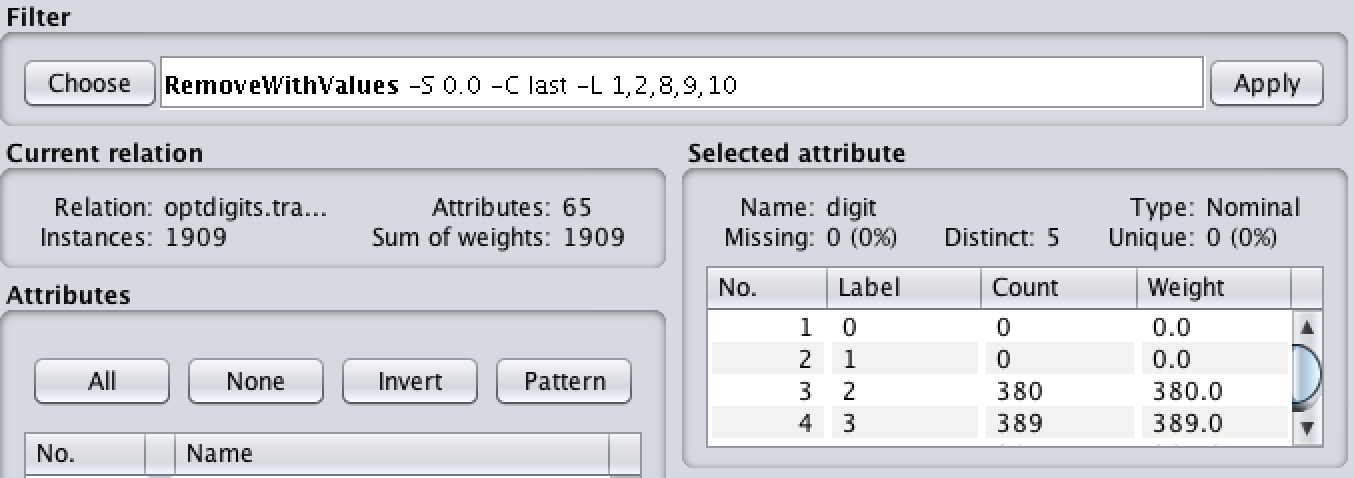
Then we apply IB1 classifier on test set:



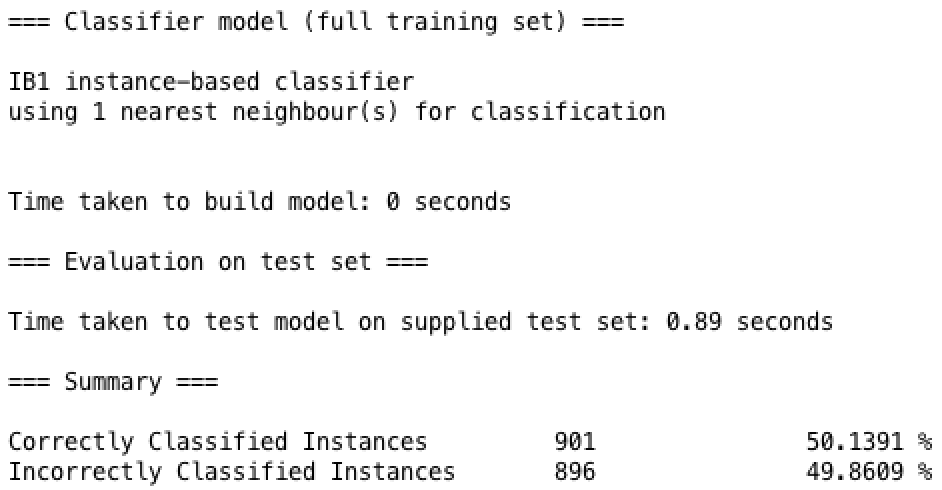


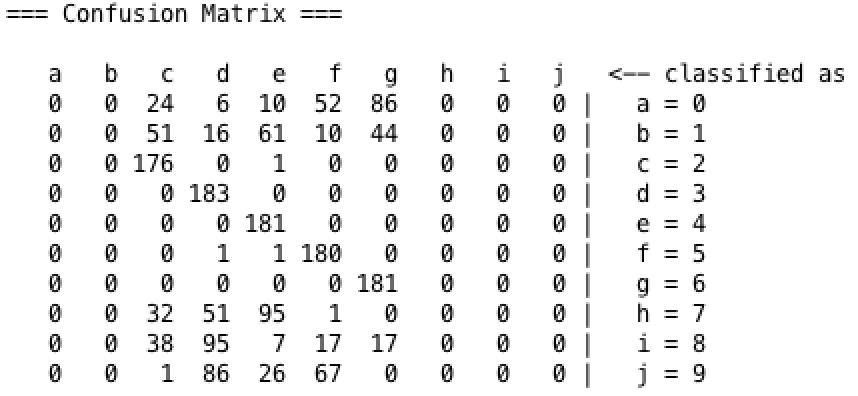
We find that when we have 4 classes prepared for the classifier training, the accuracy performance is improved with 40.0668%.

**3’ (k = 5)** Just keep instance with tag 2, 3, 4, 5 and 6



Then we apply IB1 classifier on test set:





We find that when we have 5 classes prepared for the classifier training, the accuracy performance is improved with 50.1391%.

|  |  |  |  |
| --- | --- | --- | --- |
| 3 classes | 4 classes | 5 classed | 10 classes |
| 30.0501% | 40.0668% | 50.1391% | 97.941% |

The following table shows that when we have more classes prepared for classifier training, the better classifying accuracy we can get. In other word, when we train our classifier, we are supposed to prepared training data as accurate and comprehensive as possible to feed the classifier. Just like the case showing in here, when training data is really rough (only identify 3 classes), the accuracy of classification result is really low. However, when we identify more classes in training data, the classification result improves a lot.

To put it into a real world situation, if the classifier we trained can only classify limited classes and we use such simple classifier to classify complicated data, we can expect a worse classification performance.

Although it is not reasonable to draw a function from, we find that every time we increase an identifiable class of the training data set, the accuracy improves by nearly 10%.

Besides, here is my personal idea --- we should not modify the test dataset in this experiment as in the real world, the test set is out of your control. All you can do is to observe your training data carefully and preprocess it appropriately for classifier feeding.

**h)**

1. Image enhancement

To improve the quality of images for human perception by removing noise, blurring, increase contrast and provide more details.

2. Noise removal

3. Skew detection/correction

4. Page/Character segmentation

5. Image size normalization