Classification Report

Phuong D. Pham, Oct 12

# Q1. Load the feature table

I use two kind data table to calculate in this project. The first table is color average feature table and the last one is histogram feature table. Each of them I use two data size, 400 rows data table (50 rows each category image) to test whether learning model is the best.

Color average feature table has 400 rows and 64 columns (include category column).

Histogram feature table has 400 rows and 1009 columns (include category column).

After I test with small data size, I have table of error:

Set.seed(4321)

Table : Error with data set size 400 rows

|  |  |  |
| --- | --- | --- |
|  | Color average | Color histogram |
| SVM | Training error: 0.325  Testing error: 0.5916667 | Training error: 0.3214286  Testing error: 0.7583333 |
| Random forest | Training error: 0  Testing error: 0.4583333 | Training error: 0  Testing error: 0.3916667 |
| Decision tree | Training error: 0.6892857  Testing error: 0.6833333 | Training error: 0.6607143  Testing error: 0.7166667 |
| Boost | Training error: 0  Testing error: 0.525 | Training error: 0  Testing error: 0.375 |

We can see the color histogram feature in Boost model is the best error. But the problem is the time running. It runs very slow, in data test demo just 400 rows, it runs about 40’. In this case, we should choose SVM, in my opinion, because it has normal error and time running is very fast. Don’t need more resource memory. If you have more time to wait, Boosting is the best choice. But in large data size It can run several days.

I choose Color average feature because it has less dimensions and the error is the same with Histogram, in addition, program can run fast than using histogram.

# Q2. Design a classifier to give a tag for each image.

As I mention above, I choose SVM to learn model. With all 2688 images among 806 images is test data and the rest is train data.

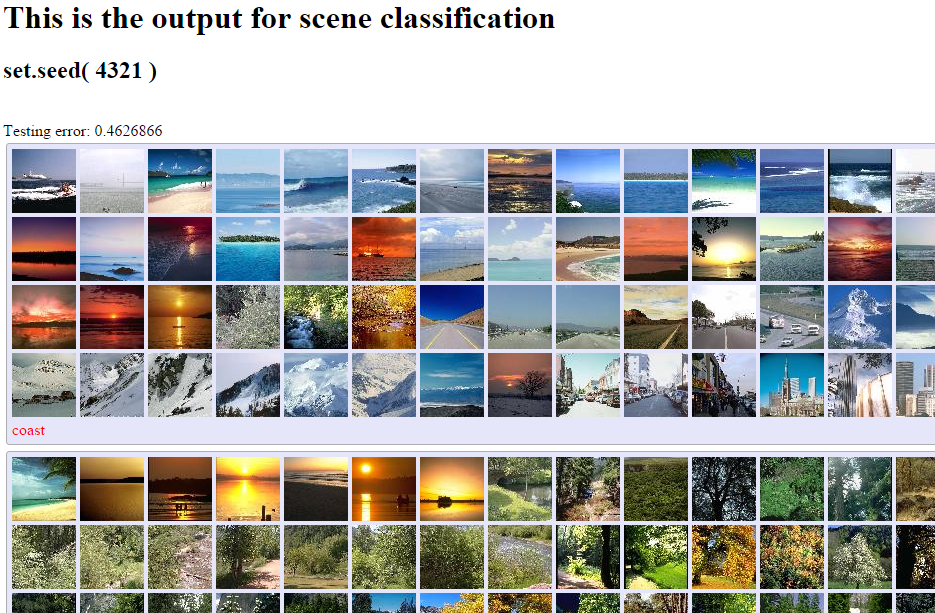
Table : Error full data

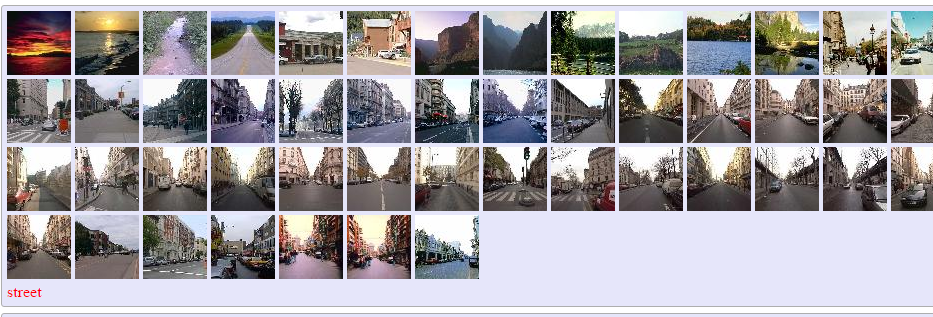
|  |  |  |
| --- | --- | --- |
|  | Train error | Test error |
| Set.seed(4321) | Number of Support Vectors: 1675  Training error: 0.3476645 | Testing error: 0.4626866 |
| Set.seed(432) | Number of Support Vectors: 1691  Training error: 0.3312102 | Testing error: 0.4776119 |
| Set.seed(125) | Number of Support Vectors: 1665  Training error: 0.3434183 | Testing error: 0.4975124 |
| Set.seed(5555) | Number of Support Vectors: 1679  Training error: 0.3221868 | Testing error: 0.5149254 |

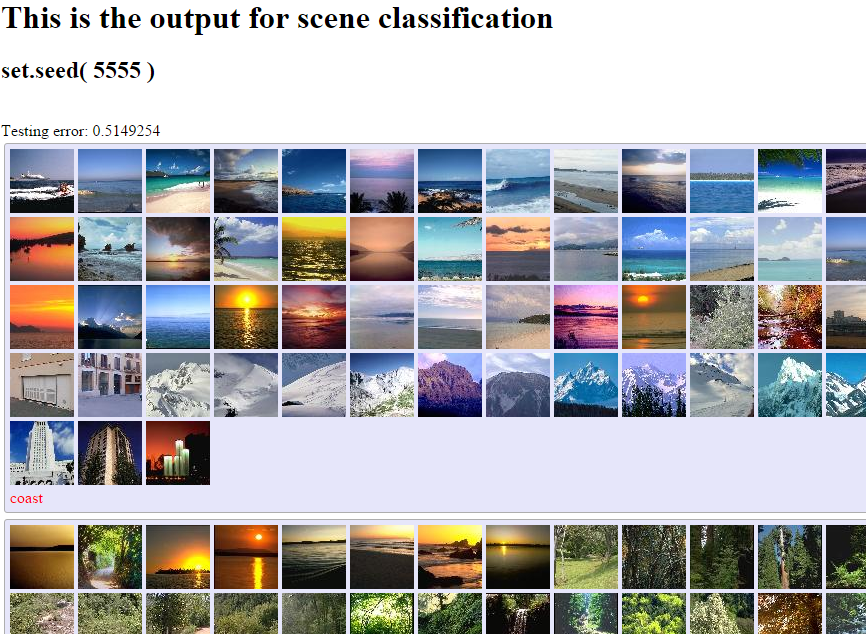
As we can see, in training when we set seed is 5555 there is the smallest error, but in testing the best is seed(4321). Every test is over fitting.

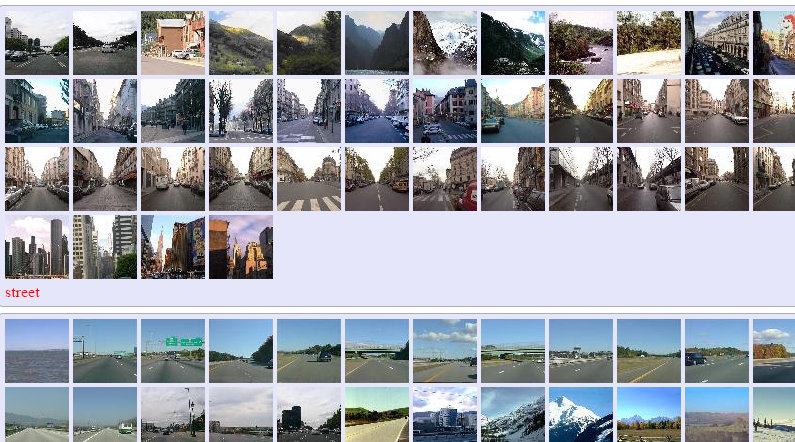
# Q3. Produce the output

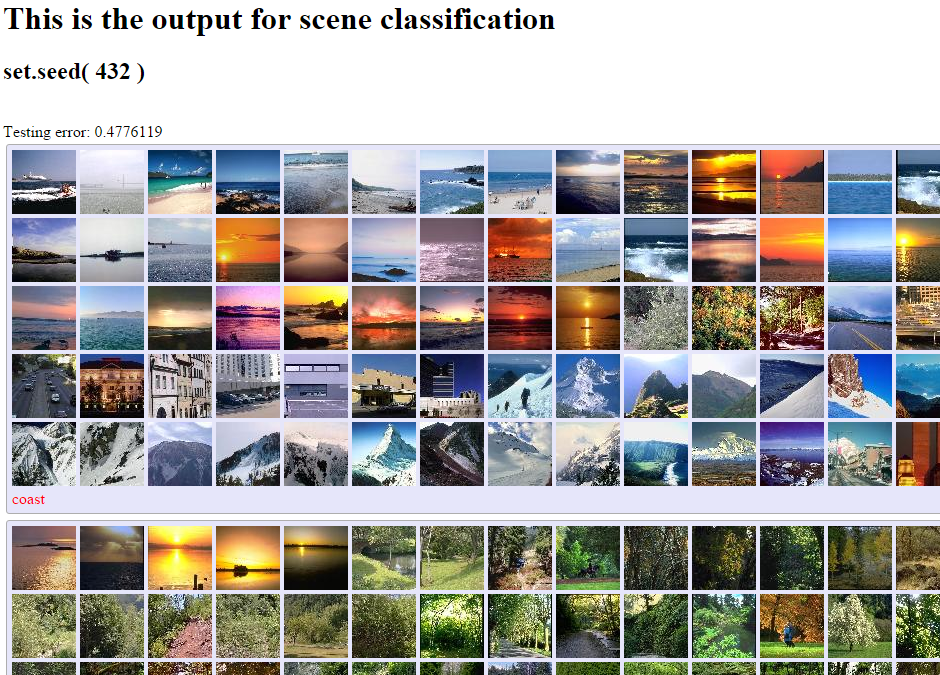
I render output to each HTML file, it look like these:

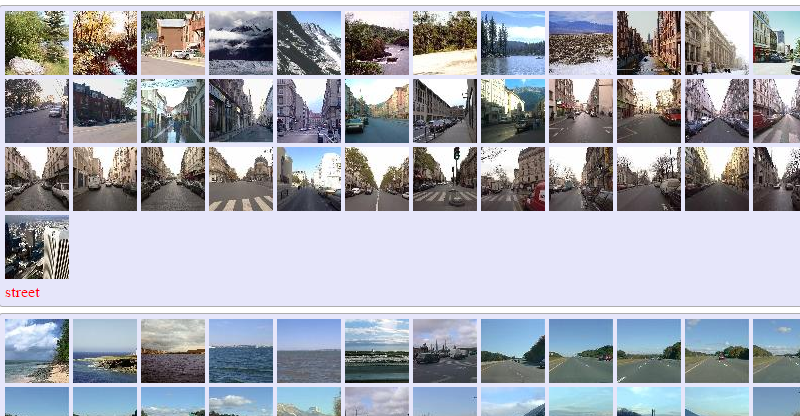


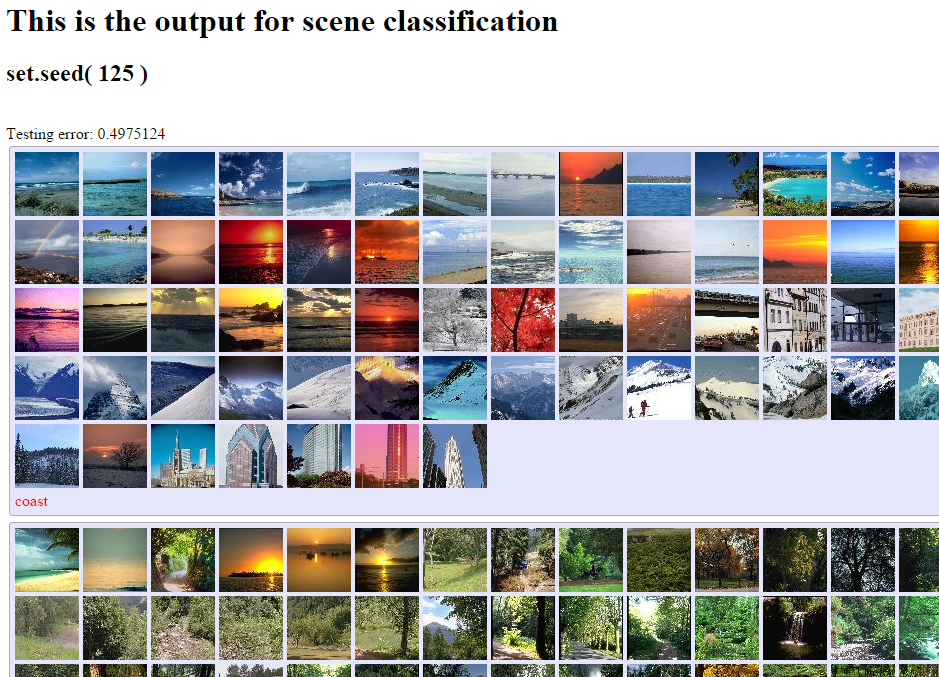














As you can see, it has more error in each classification. But look like so funny.