Natural Scenes Image Classification Project

Andrew Tiu, Kelly Wu, Sherry (Yixuan) Wu

May 1, 2020

Data Science and Statistical Learning 2020 Spring

**1. Background**

The goal of this project is to classify natural scenes image data, which is made available by Intel for an image classification challenge. Our project belongs to an important category of computer vision, which is image classification. With widespread applications in agriculture, e-commerce, and medical research, developing accurate algorithms for classifying image data is highly useful and can have diverse, lasting impacts.

Others have used deep learning methods, such as convolutional neural networks, to classify these natural scenes image data. However, while neural networks have become increasingly user-friendly in building and tuning, and many pre-trained networks are available, a plethora of other techniques are also capable of accurate and efficient classification. Therefore, we want to apply our knowledge in machine learning techniques ‒ k-nearest neighbors, random forest, and support vector machines ‒ to develop algorithms suitable for image classification.

**2. Data Overview**

Our data are obtained from the Intel Image Classification dataset, which was originally an online image classification challenge initiated by Intel. The data are available on Kaggle (<https://www.kaggle.com/puneet6060/intel-image-classification>). The train, test, and predict sets are already split before download. Within the downloaded image files, we used the training and test images, which are all in jpg format. The images are in six categories, which are buildings, forests, glaciers, mountains, sea, and streets. There are slightly over 2000 images within each category in the train file, and 400 to 600 images within each category in the test file. An example for each category is shown below.



Due to the time limitations and computer memory capacity, we decided to randomly sample 200 images from each category in both train and test data files to make the classification process more efficient. Since the standard size of each image is 150 x 150, we are able to convert each of these images to a vector of length 67501 (150 x 150 + 1 for the corresponding class label). During the selection process, we eliminate images that do not have the standard 150 x 150 size; therefore, our resulting train data contain 1195 observations and the test data contain 1197 observations. We obtain two data frames ready for PCA analysis, model fitting, and prediction.

**3. Analyses Results**

**3.1. Principal Component Analysis**

The vectors corresponding to each image are extremely large and do not lend themselves to quick, efficient modeling. To reduce the dimensionality of our data and drastically cut down on computation time, we conduct PCA on our training set of images.

From these resulting principal components (PCs), we examine the number of principal components that account for 75%, 80%, and 90% of the variance in our data, which are 99, 170, and 442 principal components, respectively. We choose to extract the PCs that account for 75% of the variance, because not only do the number of PCs increase dramatically to account for 80% and 90% of the variance, but using these latter two also does not improve the performance.

We will refer to the data frame constructed with these 99 principal components as our training data. We then project the result of the principal component analysis onto the test set of images. Therefore, the test set is dimension-reduced based on the training set and has a dimension of 1197 x 100 (one column for class).

**3.2 kNN**

The first supervised learning method examined here is k-nearest neighbors (kNN) on both the training and test set. All the principal components are min-max normalized so that all variables, regardless of their variability, are weighted equally in our decision. After scaling, the variables range between 0 and 1. Next, the optimal number of k is determined using 10-fold cross-validation. A plot of cross-validated accuracies across different neighbors is shown in Figure 1. 1 neighbor provides the highest accuracy. From the confusion matrix, accuracy is calculated to be 36.56%, with the misclassification error as 63.43%. The resulting confusion matrix is shown in Table 1.

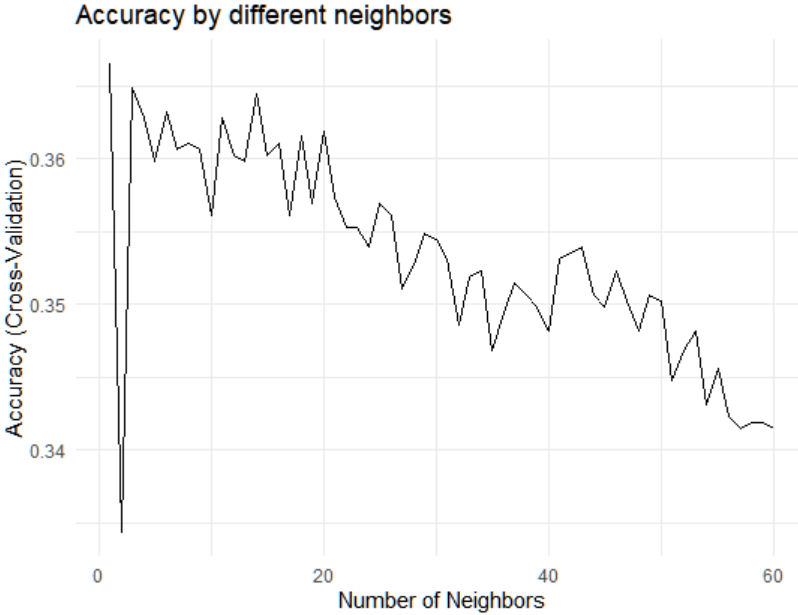


Figure 1. Accuracy by number of neighbors

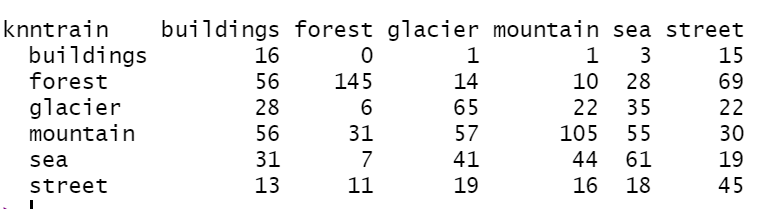


Table 1. kNN confusion matrix for test data

**3.3 Random Forest**

We also fit a random forest model, implemented in the *ranger* package for faster computation, on the training data. OOB error estimates are used to optimize for the parameters *mtry, min.node.size,* and *sample.fraction* via a grid search. These parameters correspond to the number of candidate variables considered at each node split, the minimum size of terminal nodes, and fraction of observations to sample, respectively. The final model uses *mtry* = 10, *min.node.size* = 7, and *sample.fraction* = 0.8. Additionally, 2000 trees are used.

The OOB estimate of our misclassification error rate is 42.76%. We see generally similar classification accuracy among classes, except for the building and sea classes. We also examine the mean decrease in impurity to assess variable importance. We find that removal of principal components 1 and 2 each yields a mean decrease in purity of at least 20%. A plot of variable importance can be found in our Appendix.

Finally, we predict classes for our test set of images using the fitted random forest model. The accuracy on the test data is 53.89%. A confusion matrix for the test data can be found below in Table 2.

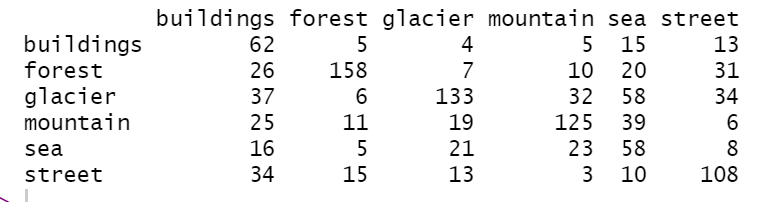


Table 2. Random forest confusion matrix for test data

**3.4 Support Vector Machines**

The last model that we adopt is the support vector machine (SVM). In order to find the optimal model fit, we first try different kernels in the fitting process. The radial basis kernel provides the best performance. Then, we conduct a 10-fold CV to find the optimal *cost* that will provide the highest accuracy for the model. We later use the SVM model to predict classification on the test set.

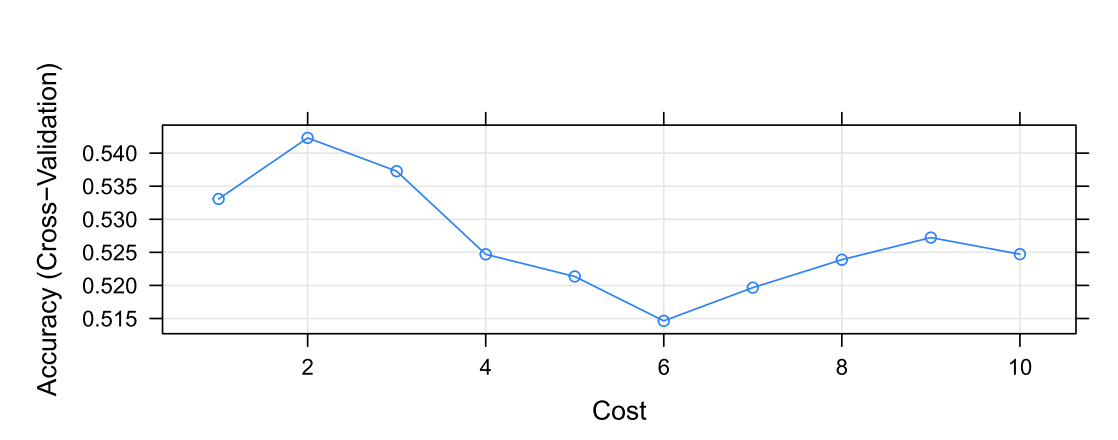


Figure 2. 10-fold CV for SVM model

Figure 2. demonstrates the result of the 10-fold CV performed on the training data in order to find the optimal *cost*. We can see that the accuracy level increases quickly when cost changes from 0 to 2 and reaches the peak when cost is 2. At the end, we predict the classes of the test set using our fitted multi-class SVM model. The confusion matrix is shown below in Table 3. The calculated accuracy level is around 53.05%.

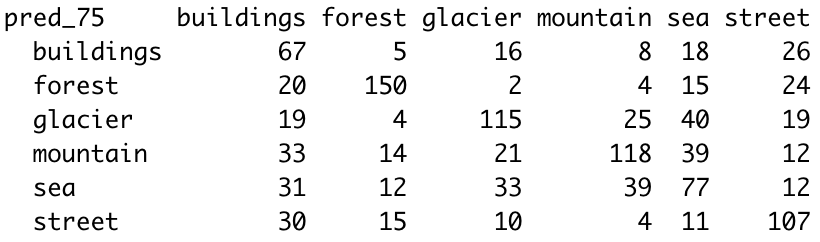


Table 3. Confusion Matrix from SVM prediction

**4. Performance Comparison**

Overall, random forest achieves the most accurate classification of 54%, followed by support vector machine with an accuracy of 53.05%. The lowest performing method is kNN, which only makes about 41.1% accurate classifications. All three methods, however, still provide much better results than if images are classified by chance, which would have an accuracy of about 1/6 = 16.67%. One prominent observation across the three methods is that the classification of sea and building images is not as accurate as the other four classes. One possible reason could be that some features in building images are similar to street images. The misclassified sea images are most likely assigned to the mountain or glacier classes due to similar pixel intensities.

**5. Conclusion**

We found that kNN, random forest, and SVM are capable of handling an image classification task. However, performance varied across methods and the best performing random forest model only achieved 54% accuracy.

We acknowledge the possible introduction of bias in our data, since the training and test set data are manually classified and labeled by humans. Such labels may lead to errors that decrease our models’ overall performance, especially when predicting on new data.

Computational time and local storage capacities limited our samples. Although each of our photos only has a dimension of 150 x 150, the resulting number of columns for one observation is 67500. Thus, we had to use a relatively small subsample of the original training and test sets, since our RAM is not able to handle larger data during PCA. With additional time and computing power, our results may improve.

In further studies, we would also like to explore more advanced feature selection and dimensionality reduction techniques besides PCA that may lead to better classification performance. Additionally, we are interested in techniques that allow specification of fields of interest within each image, as this may better reduce dimensionality and pinpoint useful features within the image data for classification.