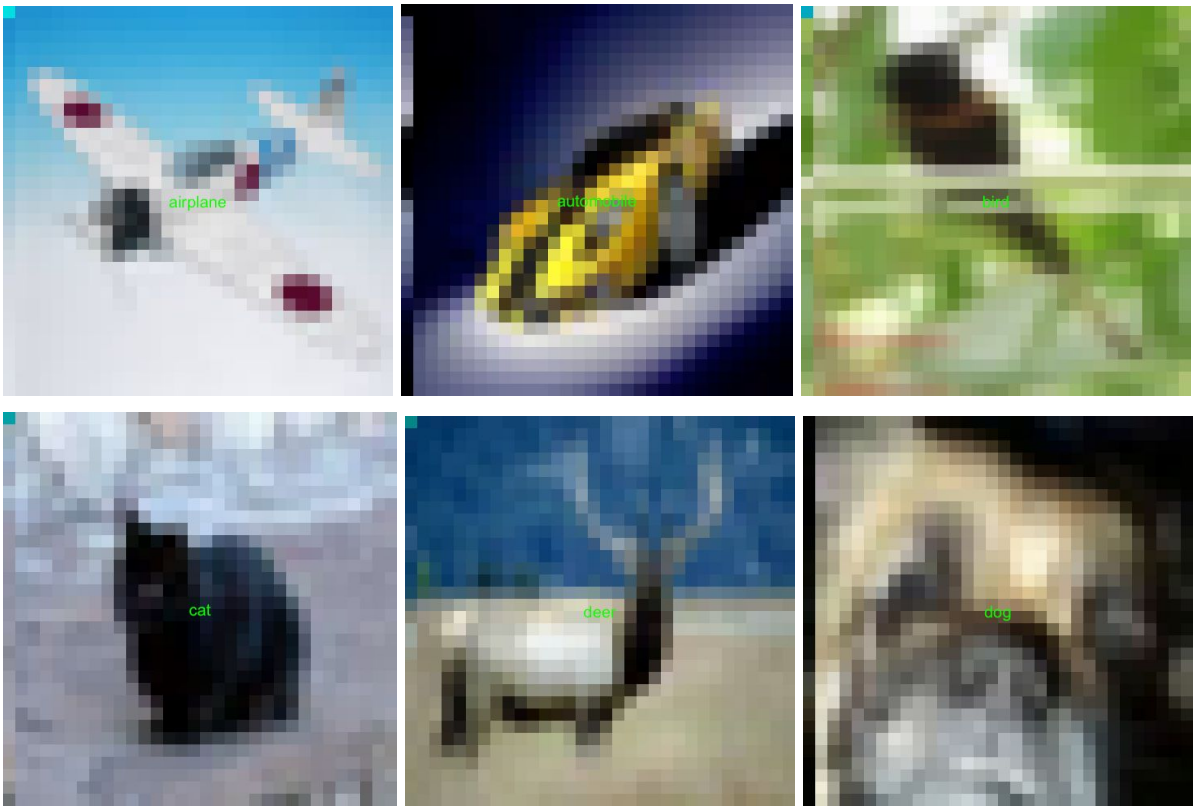
STA 141A Data Analysis Final Project Report

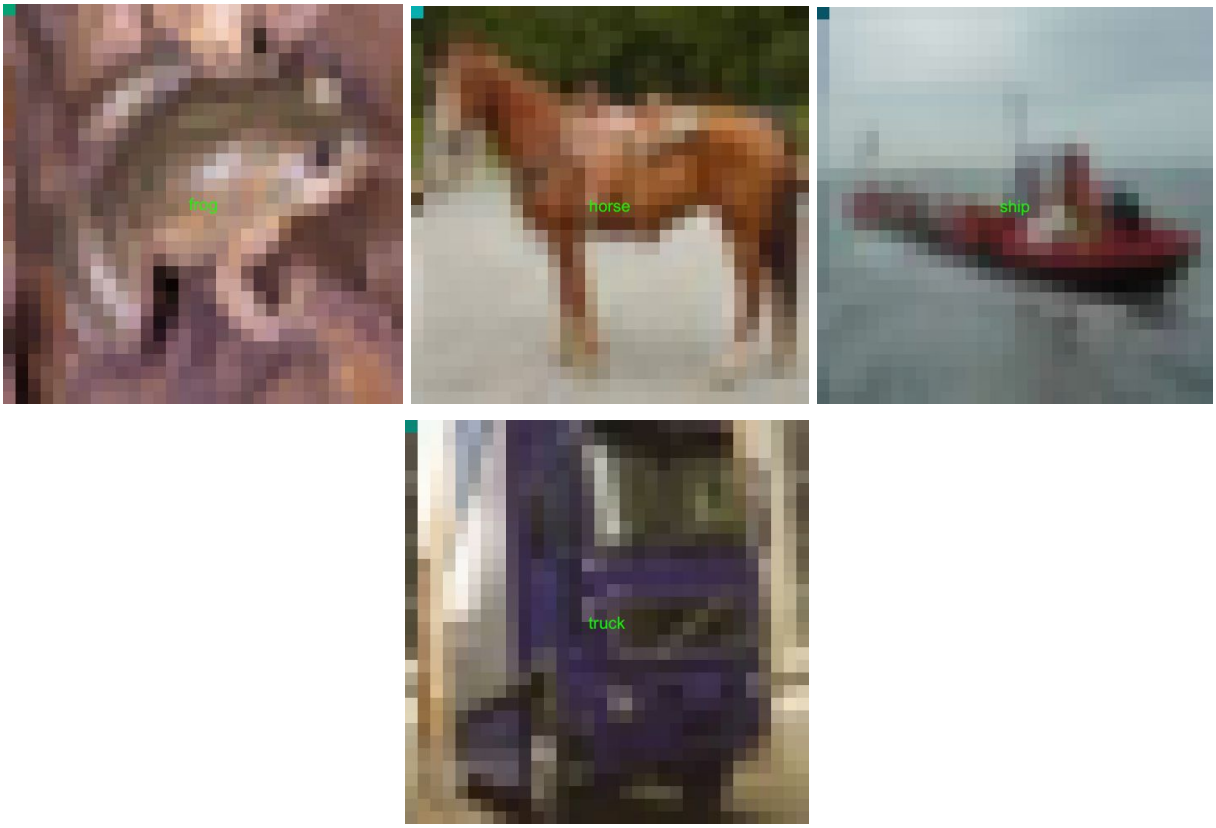
Team Members: Liya Li, Chengchen Luo, Sihui Li

Q1: The function **load_training_images()** which loads the training images and the corresponding labels, binds them and saves to an RDS file, and the function **load_testing_images()** which loads the testing images and the corresponding labels, binds them and saves to an RDS file are shown as R code in the appendix, under section Q1. Comments are included to explain codes and usages.

Q2: The function **view_images()** which displays one observation from the data set as a color image and its corresponding label is shown as R code in the appendix, under section Q2. Comments are included to explain codes and usages.

Q3: Randomly choose one image per class and display as follows:





If a pixel is located at the conners, its values may be more consistent among all picture. Therefore, this pixel may not differentiate images from different classes well. On the other hand, if the values of a pixel vary a lot (among all picture), it provides more information that those which don't vary much.

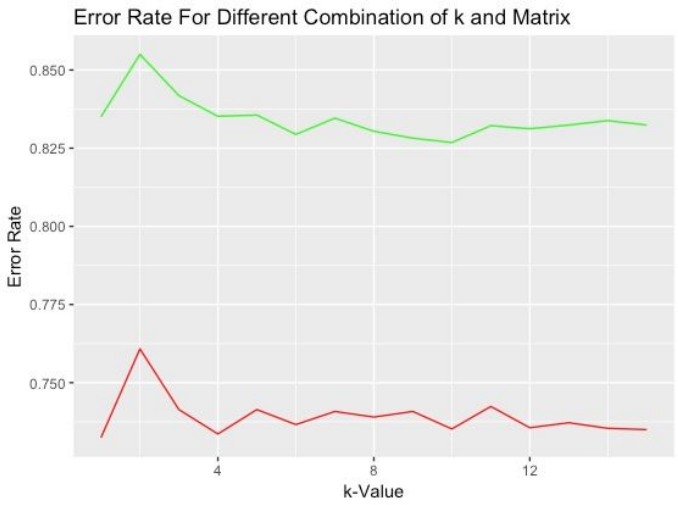
By calculating all pixels variances, we found that the pixel with larger variance values are likely to be useful for classification. In this dataset, the pixel of blue1 seems the most likely to be useful, the pixel of green367 seems the least likely to be useful for classification.

Q4: The function **predict_knn()** which uses k-nearest neighbor to predict the label for a point or collection of points is shown as R code in the appendix, under section Q4. Comments are included to explain codes and usages.

Q5: The function **cv_error_knn()** which uses 1--fold cross-validation to estimate the error rate for k-nearest neighbors is shown as R code in the appendix, under section Q5. Comments are included to explain codes and usages.

The strategy here is at the very beginning, out of our function, we obtain the 5000*5000 dimension distance_matrix from the training dataset and save the values, whose each cell represents the distance between two corresponding images. In this way, our function would run efficiently since we don't need to find the distance between each images each time in the for loop in function. We directly use distance values from the distance_matrix. Detailed strategy explanation are contained in R comments under section Q5.

Q6 : Display 10-fold CV error rates for k from 1 to 15 at distance metrics with Euclidean and Maximum methods as following:

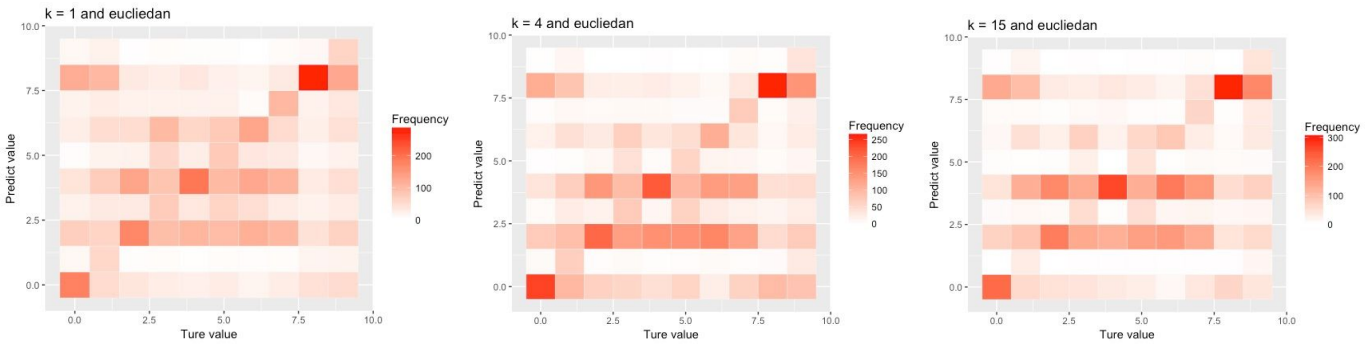


This line plot indicates that the Euclidean distance method(denoted with red line in the plot) has a overall much lower error rate than the Maximum distance method(denoted with green line in the plot). All 15 cases of k in Euclidean has lower error rate than the maximum, so we have enough evidence to conclude that the Euclidean metric is better than Maximum. In both methods, k=2 results the worst error rates.

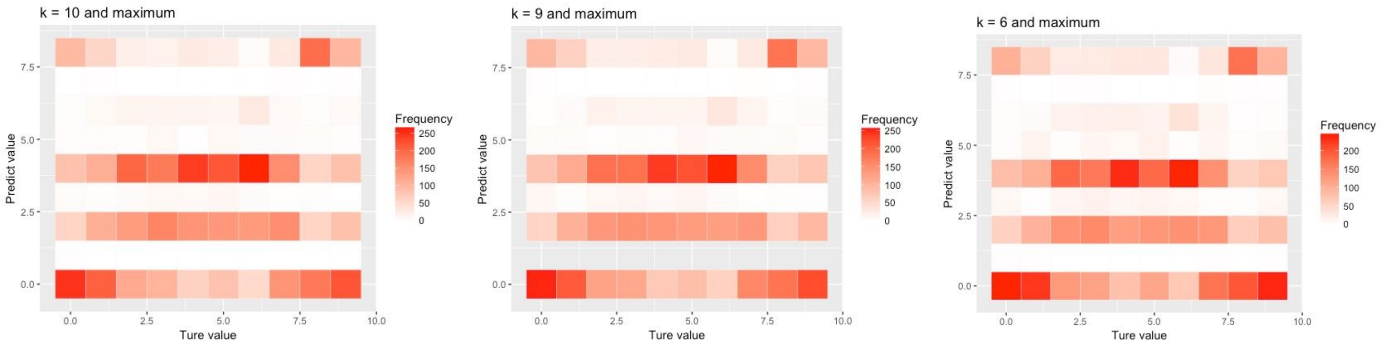
Among 30 combinations here, with the help of R output, we found that the combination of k=1 and Euclidean method yields the best, it has the lowest error rate. It may not be that useful to consider additional values of k since from the plot we notice that as k goes to larger values, the lines seem to stay within the same shape and pattern.

Q7: For each of the 3 best k and distance metric combinations, we use 10-fold cross-validation to estimate the confusion matrices.

Heatmaps for the 3 best k in euclidean method are as follows. We can see that at the diagonals tend to have deeper colors meaning that in that confusion matrix, knn predicts the right classes well, so more predictions match the real classes. For those blocks with very light colors, that indicates the comparisons don't result much matchings. This makes sense since if we are comparing the same class images, we should have the matching from it, but if we are comparing two different classes, there exists the mismatching.

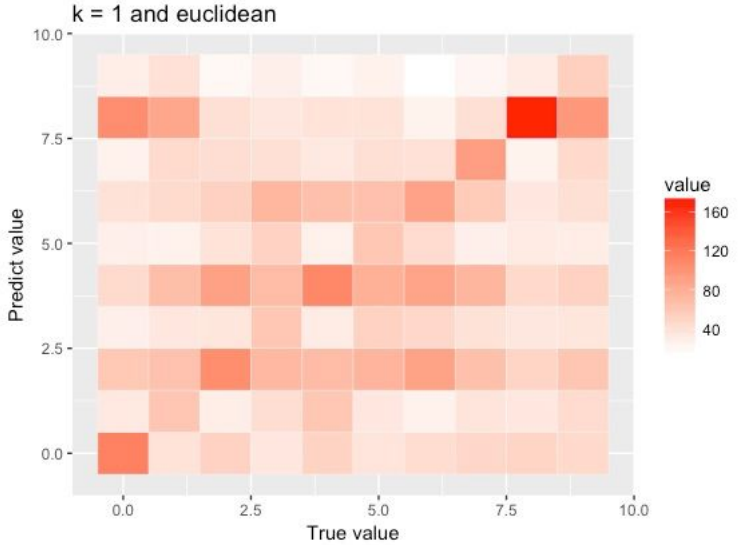


Heatmaps for the 3 best k in maximum method are as follows. We can see that at some predictions, eg. when predict value is 0, 2, 4, it tends to have deeper colors across all true values, meaning that in that confusion matrix, knn predicts all classes well with couple labels. This might make sense because 0-airplane, 2-bird, 4-cat sometime would have the similar shape and color as all other classes, it's possible that knn makes the confusion. At predict value 5,6,7 there are different situations. 5-deer, 6-dog, 7-frog sometimes can be very specifictly differentiable, and therefore, using maximum method to obtain the confusion matrix somehow may result low matchings between them to all the other classes except themselves.



This won't change much due to different combinations we choose as the best based on the heatmaps because we can easily notice that the color distributions and locations are approximately the same in different k with the same distance calculating method.

Q8: Heatmap for the best k and distance metric combination: k=1 and euclidean method:



From this heatmap, we can see that at the diagonals tend to have deeper colors meaning that in this confusion matrix, knn predicts the right classes well, so more predictions match the real classes. For those blocks with very light colors, that indicates the comparisons don't result much matchings. This makes sense since if we are comparing the same class images, we should

have the matching from it, but if we are comparing two different classes, there exists the mismatching. And we can see that the deepest color occurs at true label 9 and predict label 9, which indicates that this classifier works so well in differentiating images of ships. Somehow at predict 9 and 1 with true value 1, this classifier is confused because it predicts airplane with both airplane(which is right matching) and ship(which is mismatching) often. Also, if we look at true value 1,2,9,10 at predict value 9, we find that the colors in the boxes are also deep. We then conclude about this classifier again that, it uses ship to predict airplane, automobile, ship and truck a lot, and this makes sense because these are all types of transportations and they might have the same image color and shapes often.

Q9: Display test set error rates for k from 1 to 15 and in Euc and Max methods as follows:

Comparing the line plot in Q6 and this plot, we found that method Euclidean works better than Maximum in both using 10-fold CV error rates and not using, for its error lines indicate the lower error rates than maximum. However, not using 10-fold CV error rates results the lowest error rate at k=2 for Euclidean and k=3 for Maximum, while using 10-fold CV error rates results the highest error rate at both k=2.

By using R output, we verify this conclusion. And we can see that CV error rates are lower comparing the normal error rates, meaning that 10-fold cv cross-validation works well.

	Euclidean		Maximum	
k value	CV error rates	normal error rates	CV error rates	normal error rates
1	0.7324	0.761	0.835	0.924
2	0.7608	0.66	0.855	0.882
3	0.7414	0.689	0.8418	0.864
4	0.7336	0.692	0.8352	0.882
5	0.7414	0.713	0.8356	0.898
6	0.7366	0.726	0.8294	0.919
7	0.7408	0.718	0.8346	0.911
8	0.739	0.725	0.8304	0.924
9	0.7408	0.718	0.8282	0.918
10	0.7352	0.717	0.8268	0.925
11	0.7424	0.724	0.8322	0.921
12	0.7356	0.72	0.8312	0.918
13	0.7372	0.731	0.8324	0.919
14	0.7354	0.735	0.8338	0.913
15	0.735	0.729	0.8324	0.917

Q10: Project contribution summary:

Chengchen Luo	1,4,5,6,7,8,10
Sihui Li	1,3,6,7, R output providing
Liya Li	1,2,4,5,9, report editing