Introduction to Machine Learning *CMSC 678, Fall 2017*

HW1: Decision trees and KNN

Please note that only PDF submissions are accepted. We encourage using LaTeX to produce your writeups. You'll need *mydefs.sty* and *notes.sty* which can be downloaded from the course page.

1. (not graded): The following are true/false questions. You don't need to answer the questions. Just tell us which ones you can't answer confidently in less than one minute. (You won't be graded on this.) If you can't answer at least 8, you should probably spend some extra time outside of class beefing up on elementary math. I would strongly suggest going through this math tutorial by Hal Daume: http://www.umiacs.umd.edu/~hal/courses/2013S_ML/math4ml.pdf

   (a) $\log x + \log y = \log(xy)$

   (b) $\log[ab^c] = \log a + (\log b)(\log c)$

   (c) $\frac{\partial}{\partial x}\sigma(x) = \sigma(x) \times (1 - \sigma(x))$ where $\sigma(x) = 1/(1 + e^{-x})$ can't answer

   (d) The distance between the point (x_1, y_1) and line $ax + by + c$ is $(ax_1 + by_1 + c)/\sqrt{a^2 + b^2}$

   (e) $\frac{\partial}{\partial x} \log x = -\frac{1}{x}$

   (f) $p(a \mid b) = p(a, b)/p(b)$

   (g) $p(x \mid y, z) = p(x \mid y)p(x \mid z)$

   (h) $C(n, k) = C(n-1, k-1) + C(n-1, k)$, where $C(n, k)$ is the number of ways of choosing k objects from n can't answer

   (i) $||\alpha \boldsymbol{u} + \boldsymbol{v}||^2 = \alpha^2 ||\boldsymbol{u}||^2 + ||\boldsymbol{v}||^2$, where $||\cdot||$ denotes Euclidean norm, α is a scalar and $\boldsymbol{u}$ and $\boldsymbol{v}$ are vectors can't answer

   (j) $\left|\boldsymbol{u}^\top \boldsymbol{v}\right| \geq ||\boldsymbol{u}|| \times ||\boldsymbol{v}||$, where $|\cdot|$ denotes absolute value and $\boldsymbol{u}^\top \boldsymbol{v}$ is the dot product of $\boldsymbol{u}$ and $\boldsymbol{v}$ can't answer

   (k) $\int_{-\infty}^{\infty} \mathrm{d}x \exp[-(\pi/2)x^2] = \sqrt{2}$

2. (not graded): Go though this matlab tutorial by Stefan Roth:

   http://cs.brown.edu/courses/csci1950-g/docs/matlab/matlabtutorialcode.html

3. In class, we looked at an example where all the attributes were binary (i.e., yes/no valued). Consider an example where instead of the attribute "Morning?", we had an attribute "Time" which specifies when the class begins.

   (a) We can pick a threshold τ and use (Time $< \tau$)? as a criteria to split the data in two. Explain how you might pick the optimal value of τ.

   Try all the possible attributes Xj and threshold t and choose the one, j*, for which Information Gain(Y—Xj,t) is maximum, where Y is the class value. This will be the optimal value of T.

   (b) In the decision tree learning algorithm discussed in class, once a binary attribute is used, the subtrees do not need to consider it. Explain why when there are continuous attributes this may not be the case.

   In case of continuous attributes, they can be reused because at each split node it can be compared with different variables. For example, at the very first node code we compare attribute1 ¿ value1. Then the left hand child of that node could be split again to see if attribute1 ¿ value2. Here attribute1 is one such continuous attribute and can be used to compare with different attributes. Let's say that the value of attribute1 is 70, so in first comparison we can split the tree by 70¿60, and in the second level we can again split the tree using 70¿65. This helps to decide more precisely.

4. Why memorizing the training data and doing table lookups is a bad strategy for learning? How do we prevent that in decision trees?

   Because that will lead to no generalization at all. It will be an extreme form of overfitting a classification model to the training data. We would have fully replicated the data within the model. The model will work as a kind of 'lookup table' for the data. This will degrade the performance of the decision trees and hence it should be prevented. There are different methods to prevent this in decision trees: Using KNN and selecting larger value for K. By pruning the resulting tree based on performance on a validation set. Pre-pruning that stop growing the tree earlier, before it perfectly classifies the training set. Post-pruning that allows the tree to perfectly classify the training set, and then post prune the tree.

5. What does the decision boundary of 1-nearest neighbor classifier for 2 points (one positive, one negative) look like?

   The decision boundary of 1-nearest neighbour classifier for 2 points(one positive, one negative) will be in the center, equidistant from both the points.

6. Does the accuracy of a kNN classifier using the Euclidean distance change if you (a) translate the data (b) scale the data (i.e., multiply the all the points by a constant), or (c) rotate the data? Explain. Answer the same for a kNN classifier using Manhattan distance[1].

   In case of Euclidian distance for KNN classifier there will be no change in the accuracy even if we translate, scale or rotate the data. It is because even if the Euclidian distance changes in some cases, the overall change has the same effect it had earlier. Hence there will be no change in case of Euclidian distance for KNN classifier. Similarly for KNN using Manhattan distance, if we translate or scale the data it will have the same effect on the overall calculations it had earlier. So no change in accuracy if we translate or scale the data. But if we rotate the data, then the manhattan distance will change significantly as it is the intersection of the (x,y) coordinates. Hence rotating the data points will change the accuracy in case of KNN classifier for manhattan distance.

7. Implement kNN in Matlab or Python for handwritten digit classification and submit all codes and plots:

   (a) Download MNIST digit dataset (60,000 training and 10,000 testing data points) and the starter code from the course page. Each row in the matrix represents a handwritten digit image. The starter code shows how to visualize an example data point in matlab. The task is to predict the class (0 to 9) for a given test image, so it is a 10-way classification problem.

   (b) Write a Matlab or Python function that implements kNN for this task and reports the accuracy for each class (10 numbers) as well as the average accuracy (one number).

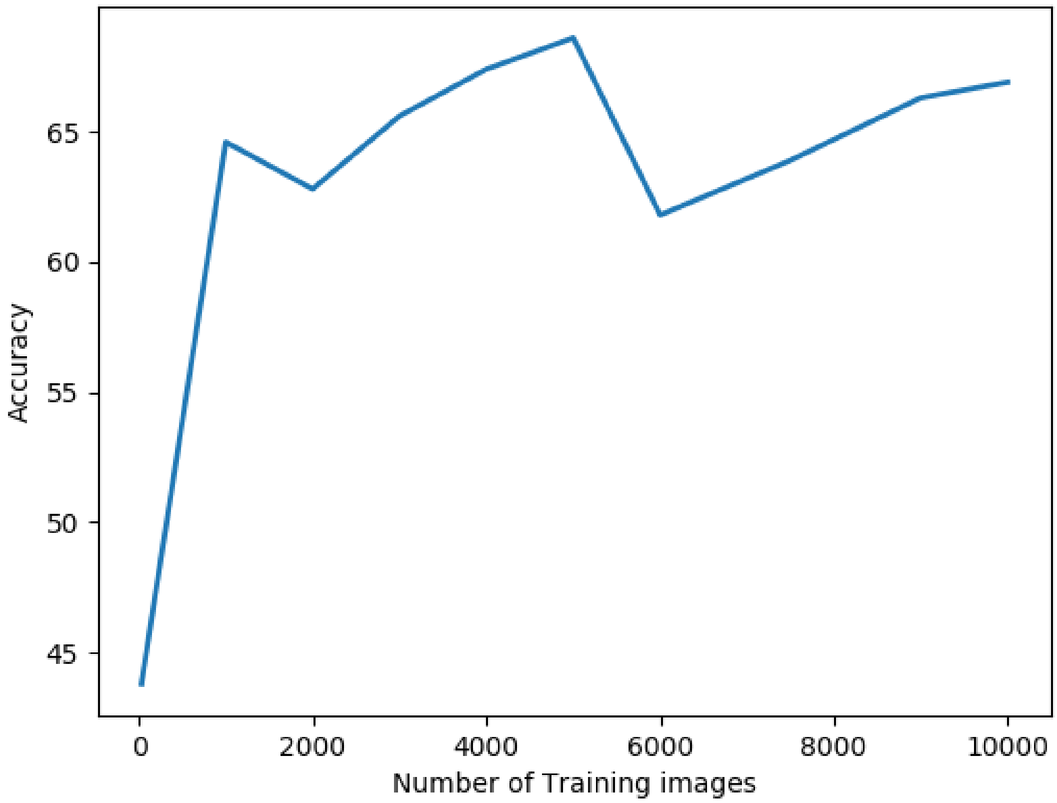
   [acc acc_av] = kNN(images_train, labels_train, images_test, labels_test, k)

   where *acc* is a vector of length 10 and *acc_av* is a scalar. Look at a few correct and wrong predictions to see if it makes sense. To speed it up, in all experiments, you may use only the first 1000 testing images.

   Answer: I obtained the following result from the KNN implementaion. Value of k is:5 Number of training images is 1000 Number of test images is 1000 Accuracy for 10 classes is as below: [100.0, 100.0, 62.5, 72.72727272727273, 64.28571428571429, 14.285714285714285, 80.0, 93.33333333333333, 100.0, 90.9090909090909] Average accuracy is: 79.0 [Finished in 1.1s]

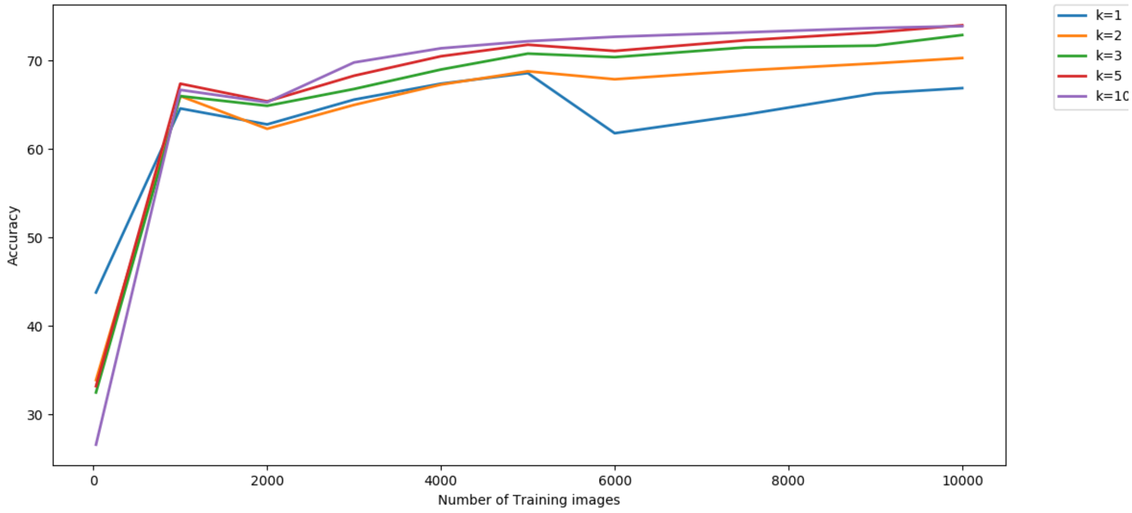
   (c) For $k = 1$, change the number of training data points (30 to 10,000) to see the change in performance. Plot the average accuracy for 10 different dataset sizes. You may use command *logspace* in matlab. In the plot, x-axis is for the number of training data and y-axis is for the accuracy.

[1] http://en.wikipedia.org/wiki/Taxicab_geometry



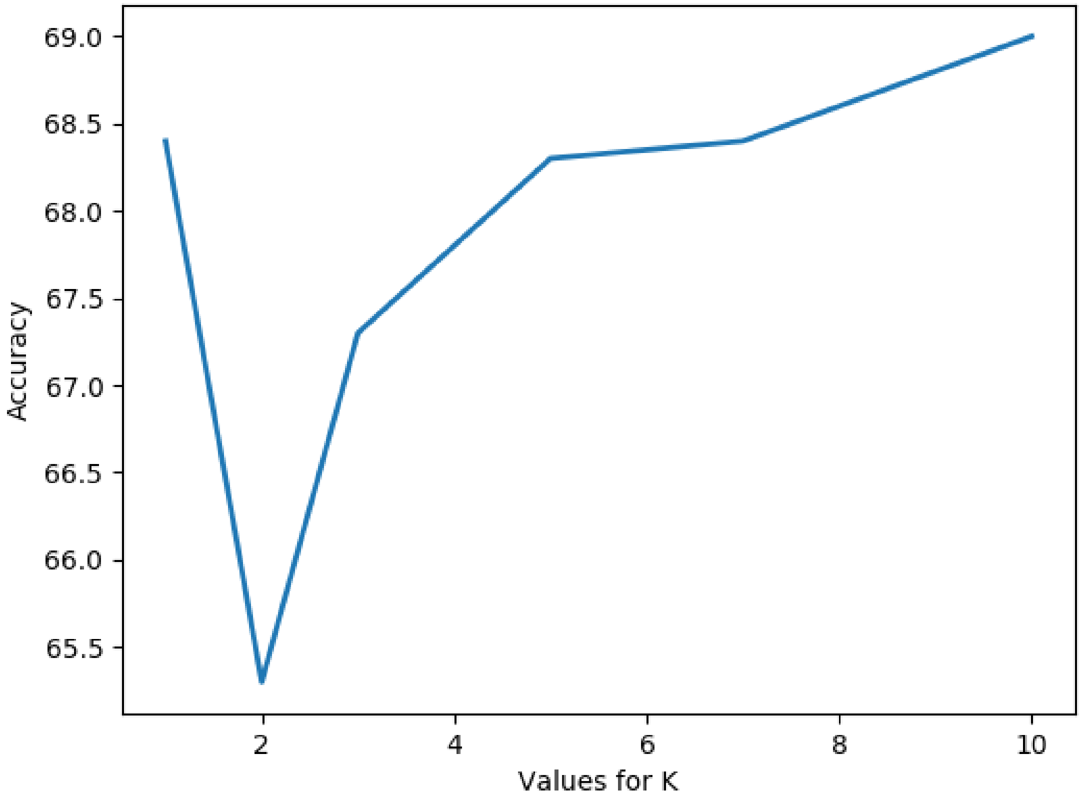
Answer: I took 10 different training dataset sizes. Highest accuracy was obtained for training dataset size of 5000, which decreased for 6000 and again increased as the training dataset size was increased.

(d) Show the effect of k on the accuracy. Make a plot similar to the above one with multiple colored curves on the top of each other (each for a particular k in [1 2 3 5 10].) You may use command *legend* in matlab to name different colors.



Answer: As seen from the above image, the average accuracy is improved, as the value for k is increased. For k=10 average accuracy is above 75, which is highest for the possible values of K.

(e) Choose the best k for 2,000 total training data by splitting the training data into two halves (the first for training and the second for validation). You may plot the average accuracy wrt k for this. Note that in this part, you should not use the test data. You may search for k in this list: [1 2 3 5 10].



Answer: As seen from the above image, the best value for K is 10 from a set of [1 2 3 5 10]. The average accuracy obtained for k=10 is 69.0