Introduction to Machine Learning *CMSC 678, Fall 2018*

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 $\sigma(x) = 1/(1 + e^{-x})$
   (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
   (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
   (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 $\boldsymbol{v}$
   (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.
   We can sort the time values. We pick a threshold between adjacent time values starting from least all the way to maximum value and compute the Information Gain for that threshold. Then, we pick the threshold which maximizes the information gain. So basic idea here is to pick a threshold that gives a pure split.

   (b) In the decision tree learning algorithm discussed in class, once a binary attribute is used, the sub trees do not need to consider it. Explain why when there are continuous attributes this may not be the case.
   For a binary attribute, dataset is divided into two parts with positive and negative values for the binary attribute. So, if we split the tree on same binary attribute again for the second time, dataset cannot be further split into two because it already has only those entries with positive or negative values for the binary attribute. But, for continuous attributes, we split the dataset on threshold. So, all the dataset entries with range of values below threshold go to one side and entries with range of values above threshold go to other side. Range of values for this continuous attribute on either side of the tree has variance. So, it may happen that we may need to consider this continuous attribute in future again which would give pure dataset split by splitting on optimal threshold grater or less than the earlier threshold we split on for this continuous attribute.

4. Why memorizing the training data and doing table lookups is a bad strategy for learning? How do we prevent that in decision trees?
   With memorization model will only work for the data it has seen during training time. For completely new but relative data, the model will perform poorly. It cannot generalize. This is called over-fitting. This can be prevented with pruning strategy. Pruning improves prediction accuracy by reducing over-fitting. It restricts or reduces the height of decision tree by removing subtrees that do not improve accuracy.

5. What does the decision boundary of 1-nearest neighbor classifier for 2 points (one positive, one negative) look like?
   For only two points for 1-nearest neighbour (one positive and one negative), we can draw a line segment joining these two points and perpendicular bisector to this line segment bisecting exactly at mid-point would give us decision boundary for this case. So, all points above the line would be for example labelled as positive and all points below would be negative.

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].
   (1) Euclidean Distance

   (a) Translate the data: Given (x,y) pairs of data points where x is N dimensional vector, data translation means to translate this data to a new structure/format. We apply a function to every pair of data to translate it. As the function is applied to every data point, Euclidean distance between two data points might change but nearest neighbours of a data point will be same. So, accuracy doesnt change if we translate the data.

   (b) Scale the data: Scaling is multiplying all the data points by a constant. So, Euclidean distance between two data points will be multiplied by the same constant. But again, nearest neighbours of a data point will remain same and accuracy doesnt change

   (c) Rotate of data: Here we rotate the data points about origin. So, data points distribution wont change. Euclidean distance also remains same. Hence, accuracy doesnt change.

   (2) Manhattan Distance

   (a) Translate the data: When we translate the data i.e. apply some function to all data points, Manhattan distance between two data point may change. But, nearest neighbours would remain same as function is applied on every data point. So, accuracy doesnt change.

   (b) Scale the data: After we multiply all the data points by a constant, Manhattan distance between data points changes. But, as scaling is applied to all the data points, nearest neighbours wont change. Thus, accuracy also remains same.

   (c) Rotate the data Manhattan distances are generally used in Taxicab geometry. So, let us consider one example from Taxicab geometry. A circle is represented by a radius and angle. Taxicab circles are squares with their sides oriented at 45 degree to co-ordinate axes. So, if we rotate the data, angle will change leading to a different Manhattan distance. So, accuracy is affected when we rotate the data.

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)

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

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.

(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.

(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.

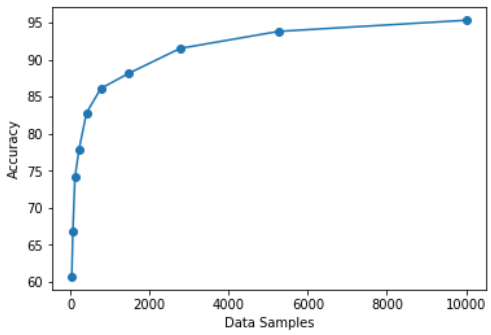
(e) Choose the best k for 2,000 total training data by splitting the training data into two halves (the first for training and the second for validation). You may plot the average accuracy wrt k for this. Note that in this part, you should not use the test data. You may search for k in this list: [1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

Below are graphs plotted for:

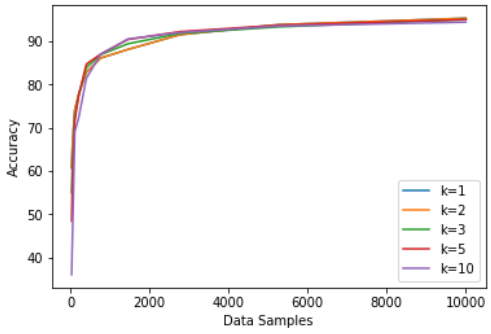
(1) Plot of accuracy vs data samples for differnt train sample sizes and k = 1

(2) Plot of accuracy vs data samples for differnt train sample sizes and different k values

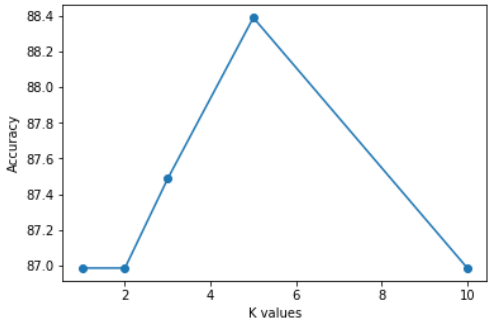
(3) Plot of accuracy vs k values for fixed train sample size and different k values



1 final.png



2 final.png



3 final.png