Introduction to Machine Learning *Intekhab Naser (ZC11577)*

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) $\log x + \log y = \log(xy)$: True

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

   (c) $\frac{\partial}{\partial x}\sigma(x) = \sigma(x) \times (1 - \sigma(x))$ where $\sigma(x) = 1/(1 + e^{-x})$: True

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

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

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

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

   (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: True

   (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) $|\boldsymbol{u}^\top \boldsymbol{v}| \geq ||\boldsymbol{u}|| \times ||\boldsymbol{v}||$, where $|\cdot|$ denotes absolute value and $\boldsymbol{u}^\top \boldsymbol{v}$ is the dot product of $\boldsymbol{u}$ and $\boldsymbol{v}$:

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

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.

      i. According to the description, we need to map a non binary attribute "Time" into a binary attribute τ, such that the classification can be done based on the time being less than or greater than τ. The optimal value of τ will split the data in two parts and give the most number of correct answers. We can calculate τ as follows:

      - We can represent time in hours (discarding minutes and seconds) with the 24 hour format.
      - List the hours for which student liked the subject and list the hours for which student didn't like the subject.
      - Then pick the maximum value from liked list but less than most elements in disliked list. Pick the minimum value from disliked list but greater than most elements in liked list.
      - The average of these two values can be used as the value of τ
      - Example: If the student liked the subject at 10, 12, 9, 11, 17 and disliked the subject at 15, 14, 16, 13, 6. The values we get are 12 and 13.
      - Now, we can generalize that for $\tau < 12.5$, the student likes the subject and dislikes otherwise.

(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.

   i. In continuous attributed features, we split the continuous values into specific ranges or windows, so that we can classify the labels accordingly:
      - Down the sub-tree, this splitting criteria can again narrow-down the window of range of values for further classification and optimum result.
      - An example for this can be fruit classification on the basis of width of fruit:
         - Let us say a fruit with width > 5 could be a watermelon or pine-apple as right sub-tree and if it is less than or equal to that, it could be an apple or banana as left sub-tree.
         - In the left sub-tree, if the width is less than 3, it is a banana and an apple otherwise. In the right sub-tree, if the width is greater than 8, it is a watermelon and a pine-apple otherwise.
         - This is not the case with binary attributes.

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

   (a) Memorizing the training data and doing table lookups will result in overfitting with no generalization for new or test data. In such a case, the classification for training data will be absolutely accurate as the labels have already been memorized. However, this model will perform poorly on test data as the classifier will fail to find the new data pattern in the lookup table and may predict wrong labels.

   (b) In decision trees, we can employ a number of measures to prevent memorizing the training data. In a method called Pre-pruning, it does not allow the decision tree to perfectly classify the training data i.e. it restricts levels or height of decision tree. In Post-pruning, it allows the decision tree to perfectly classify the training data and then post prunes the tree, as the name implies.

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

   (a) The simplest way to calculate the decision boundary for 1-nearest neighbor is to find the slope of the line joining the positive and negative points and draw a perpendicular bisector to this line. This perpendicular bisector can act as the decision boundary.

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].

   - For two points:
      - Euclidean Distance: $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$
      - Manhattan Distance: $|x_1 - x_2| + |y_1 - y_2|$

   (a) Translation of Data: Translation of data basically means to apply any function on data so as to shift the data points in a specific direction. As long as the same data translation is applied on train data and test data, the accuracy of kNN will not be affected.

      i. Euclidean Distance: Euclidean distance between data points may change after translation, but the relative distance-differences between test points and it's nearest neighbors will be same as before. Thus, accuracy of kNN classification will not be affected.

      ii. Manhattan Distance: Same is the case with Manhattan distance.

   (b) Scaling of Data: As described, scaling the data is multiplying each data point with a constant. So, assuming that the scaling factor is non-zero:

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

   i. Euclidean Distance: After scaling, the new Euclidean distance will be equal to the old Euclidean distance multiplied by the square-root of the constant. Thus, the relative distance differences remains same, which does not affect the accuracy of the kNN classifier.
   ii. Manhattan Distance: After scaling, the new Euclidean distance will be equal to the old Euclidean distance multiplied by the constant. Thus, the relative distance differences remains same, which does not affect the accuracy of the kNN classifier.

(c) Rotation of Data: This means rotating the data around the origin.

   i. Euclidean Distance: The Euclidean distance between each data point and training points remain the same after rotation, as Euclidean is the absolute distance between two points irrespective of their orientation to each other. Thus, kNN accuracy will not be affected.
   ii. Manhattan Distance: Manhattan distance will change based on the x,y intercepts which will be affected by the change in orientation of the points with respect to each other. Thus, the accuracy of kNN will be affected in this case.

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.

   (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].

      i. The best k in this case is 3 and 5.

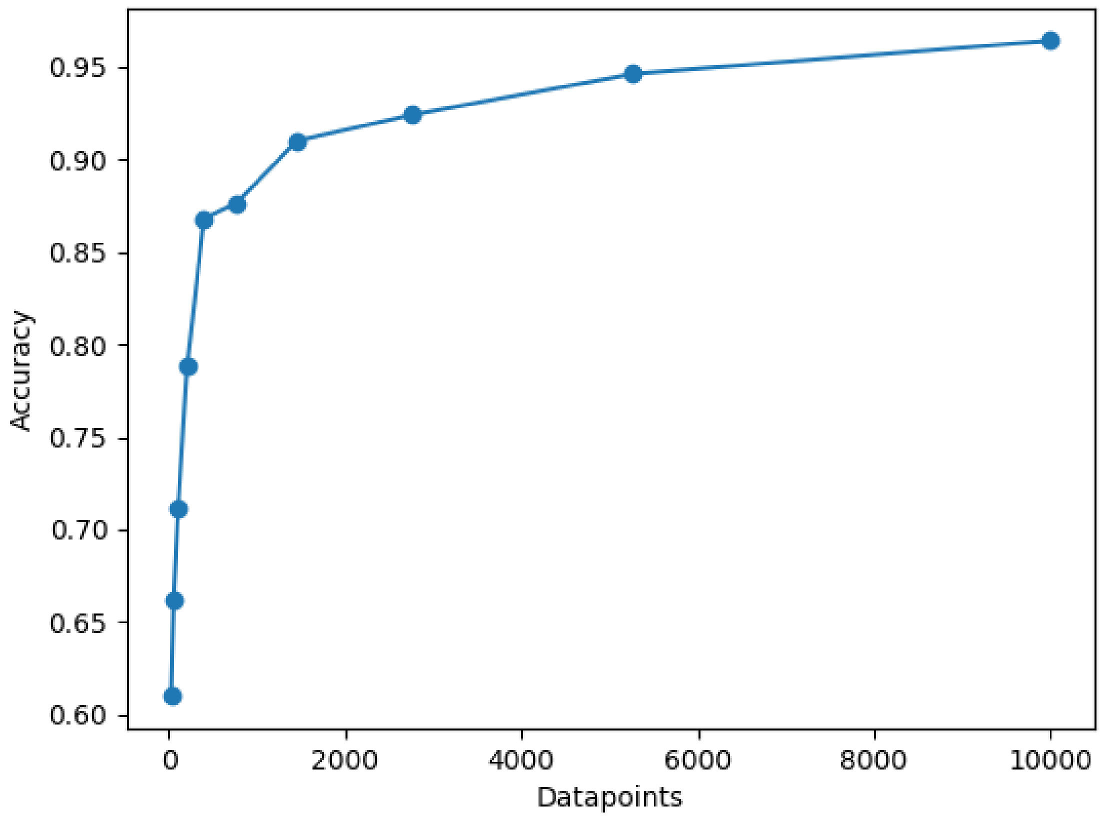

Figure 1: Accuracy measure along training data sizes ranging from 30 to 10,000

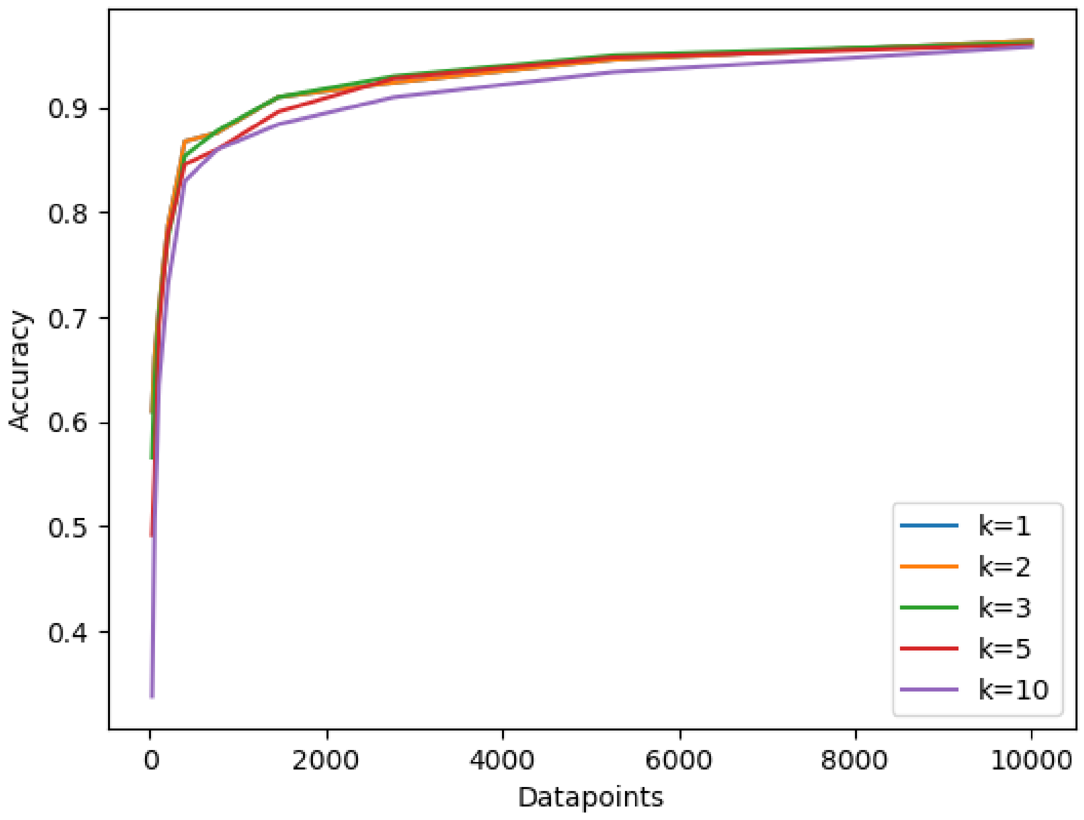

Figure 2: Accuracy measure along varying training data sizes and values of K

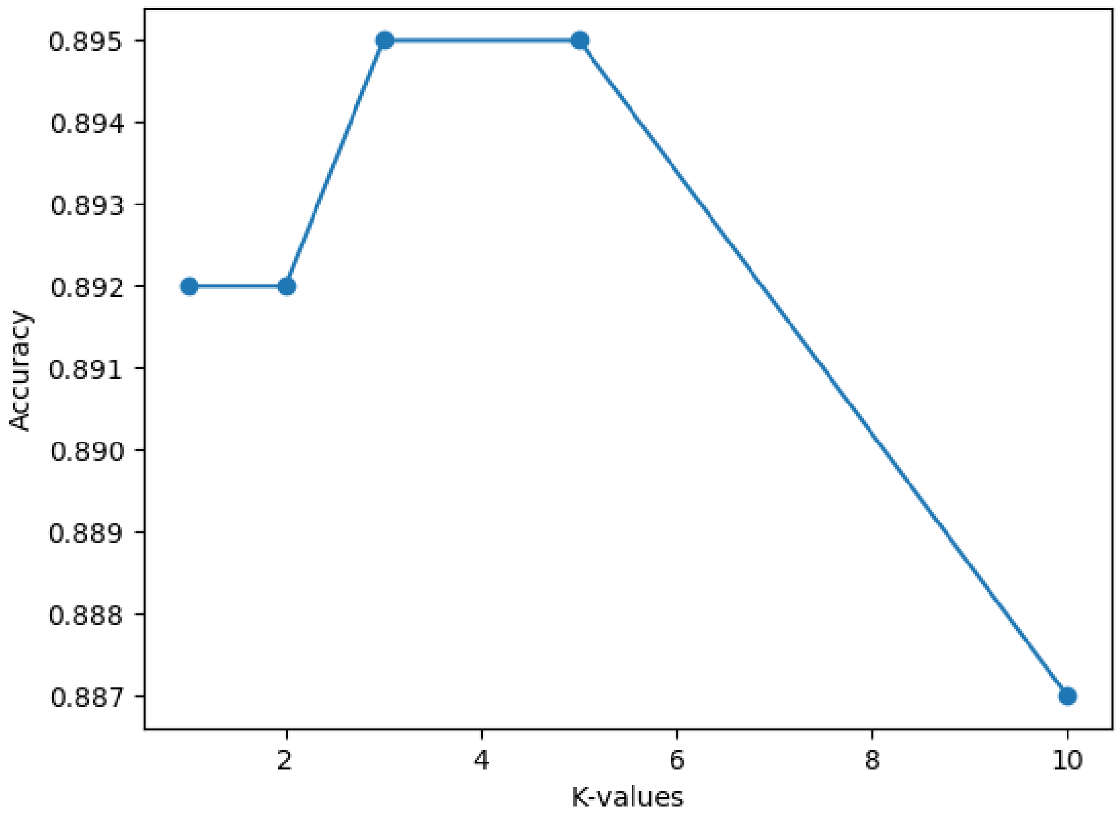

Figure 3: Accuracy measure for values of K in [1,2,3,5,10]