Machine Learning & Discovery *ECS 271, Fall 2025*

HW1: Decision trees and KNN

1 Math Practice (NOT GRADED)

1. $\frac{\partial}{\partial x} \log x = -\frac{1}{x}$
2. $p(a, b|c) = \frac{p(a|c)p(b|c)}{p(c)}$
3. $p(a \mid b) = p(a, b)/p(b)$
4. $\log x + \log y = \log(xy)$
5. $p(x \mid y, z) = p(x \mid y)p(x \mid z)$
6. $\int_{-\infty}^{\infty} \mathrm{d}x \exp[-(\pi/2)x^2] = \sqrt{2}$
7. $\log[ab^c] = \log a + (\log b)(\log c)$
8. $\frac{\partial}{\partial x}\sigma(x) = \sigma(x) \times (1 - \sigma(x))$ where $\sigma(x) = 1/(1 + e^{-x})$
9. The distance between the point (x_1, y_1) and line $ax + by + c$ is $(ax_1 + by_1 + c)/\sqrt{a^2 + b^2}$
10. $\left|\boldsymbol{u}^\top \boldsymbol{v}\right| \geq ||\boldsymbol{u}|| \times ||\boldsymbol{v}||$, where $|\cdot|$ denotes absolute value and $\boldsymbol{u}^\top \boldsymbol{v}$ is the dot product of $\boldsymbol{u}$ and $\boldsymbol{v}$
11. $C(n, k) = C(n - 1, k - 1) + C(n - 1, k)$, where $C(n, k)$ is the number of ways of choosing k objects from n
12. $||\alpha\boldsymbol{u} + \boldsymbol{v}||^2 = \alpha^2 ||\boldsymbol{u}||^2 + ||\boldsymbol{v}||^2$, where $||\cdot||$ denotes Euclidean norm, α is a scalar and $\boldsymbol{u}$ and $\boldsymbol{v}$ are vectors

2 Continuous Variable vs Discrete Variables

All the attributes we observed in the class so far for decision tree are binary like "Is it Morning?". Now lets assume that our attributes are numerical.

1. Let's say we want to consider the **Time** on which the class begins. One can argue that we can pick a threshold τ and use (Time $< \tau$)? as a criteria to split the data in two and make a binary tree. Explain how you might pick the optimal value of τ.

   For a continuous attribute like **Time**, we can consider all the unique values of **Time** in our training data. For example, the training data may have times 9:00, 11:00, 12:00, and 16:00. Unlike the yes/no case where we only have a single potential boundary, we now can consider splitting at any time between the times in our training data. A simple way to do this would be to take the midpoint between each

pair of consecutive unique values in the training data. In our example, this would mean our candidates for τ would be 10:00 (between 9:00 and 11:00), 11:30 (between 11:00 and 12:00), or 14:00 (between 12:00 and 16:00). We can then evaluate each candidate τ in a similar way we did for binary attributes, but this time across all candidate thresholds of a continuous attribute. In other words, we assign a score like misclassification error (what we did in lecture), information gain, or Gini impurity to each candidate τ and pick the one that optimizes our chosen score.

2. In the decision tree learning algorithm discussed in class, once a binary attribute is used, the subtrees do not need to consider it anymore. Explain why in using continuous attributes this may not be the case.

3 To Memorize or Not to Memorize:

Why using the training data as a dictionary or lookup table and referring to it is a bad strategy for learning? What is it called? How do we prevent it?

4 Visualize

What does the decision boundary of 1-nearest neighbor classifier for 2 classes (one positive, one negative) look like when the features are 1-D? How about 2-D and 3-D? Can you give a general form?

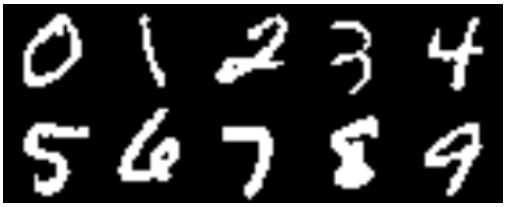
5 kNN and data manipulation

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 all the points by a constant), or (c) rotate the data? Explain. Answer the same for a kNN classifier using Manhattan distance[1] and Cosine Distance[2]. Make sure you provide mathematical proofs or at least some intuition that why your claim is correct.

6 Coding

Implement kNN in Python for handwritten digit classification and submit all codes and plots:

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. The task is to predict the class (0 to 9) for a given test image, so it is a 10-way classification problem.



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

1. Write a 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.

2. 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. In the plot, x-axis is for the number of training data and y-axis is for the accuracy.

3. 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 3 5 10].)

4. Choose the best k. First choose 2,000 training data randomly (to speed up the experiment). Then, split the training data randomly to two halves (the first for training and the second for cross-validation to choose the best k). Please plot the average accuracy wrt k on the validation set. You may search for k in this list: [1 3 5 10]. Finally, report the accuracy for the best k on the testing data.