HOMEWORK 1

CSC2515 Fall 2024

- **Deadline: Thursday, October 3, 2024 at 11:59**.
- **Submission:** You need to submit two files through MarkUs. One is a PDF file including all your answers and plots. The other is a source file (Python script or Jupyter Notebook) that reproduces your answers. You can produce the file however you like (e.g. LaTeX, Microsoft Word, etc) as long as it is readable. Points will be deducted if we have a hard time reading your solutions or understanding the structure of your code. If the code does not run, you may lose most/all of your points for that question. Your report must include all the relevant figures and graphs. We may not look at your code or Jupyter Notebook.
- **Late Submission:** 10% of the marks will be deducted for each day late, up to a maximum of 3 days. After that, no submissions will be accepted.
- **Collaboration:** You can discuss the assignment with up to two other students (group of three). You can work on the code together. But each of you need to **write your homework report individually**. You must mention the name of your collaborators clearly in the report and the source code.

1. Nearest Neighbours and the Curse of Dimensionality – 15 pts. In this question, we will verify the claim from lecture that "most" points in a high-dimensional space are far away from each other, and also have approximately the same distance.

(a) **[5 pts]** Consider two independent univariate random variables X and Y sampled uniformly from the unit interval $[0, 1]$. Determine the expectation and variance of the random variable $Z = |X - Y|^2$, i.e., the squared distance between X and Y.
Note: You can either compute the integrals yourself or use the properties of certain probability distributions. In the latter case, explicitly mention what properties you have used.

(b) **[5 pts]** Now suppose we draw two d-dimensional points X and Y from a d-dimensional unit cube with a uniform distribution, i.e., $X, Y \in [0, 1]^d$. Observe that each coordinate is sampled independently and uniformly from $[0, 1]$, that is, we can view this as drawing random variables $X_1, \ldots, X_d$ and $Y_1, \ldots, Y_d$ independently and uniformly from $[0, 1]$. The squared Euclidean distance $\|X - Y\|_2^2$ can be written as $R = Z_1 + \cdots + Z_d$, where $Z_i = |X_i - Y_i|^2$. Using the properties of expectation and variance, determine $\mathbb{E}\left[\|X - Y\|_2^2\right] = \mathbb{E}[R]$ and $\mathrm{Var}[\|X - Y\|_2^2] = \mathrm{Var}[R]$. You may give your answer in terms of the dimension d, and $\mathbb{E}[Z]$ and $\mathrm{Var}[Z]$ (the answers from part (a)).

(c) **[5 pts]** Based on your answer to part (b), compare the mean and standard deviation of $\|X - Y\|_2^2$ to the maximum possible squared Euclidean distance between two points within the d-dimensional unit cube (this would be the distance between opposite corners of the cube). Why does this support the claim that in high dimensions, "most points are far away, and approximately have the same distance"?

2. Limiting Properties of the Nearest Neighbour Algorithm – 10 pts. In this question, we will study the limiting properties of the k-nearest neighbour algorithm.

Suppose that we are given n data points $X_1, \ldots, X_n$, sampled uniformly and independently from the interval $[0, 2]$ and one test point, $y = 1$.

Let $Z_i = |X_i - y|$ denote the distance of y to the data point X_i and $Z_i = |X_i - y|$ denote the distance of y to its i-th nearest neighbour. Then, $Z_1 < \cdots < Z_n$.

(a) **[2 pts]** Show that the random variable Z_i is uniformly distributed between the unit interval $[0, 1]$.

(b) **[4 pts]** Show that $\mathbb{E}[Z_1] = \frac{1}{n+1}$, i.e., the expected distance to the 1st nearest neighbour is $\frac{1}{n+1}$.
Hint: It may help to first compute $\mathbb{P}\{Z_1 > t\}$. Note that Z_1 has the smallest value among $Z_1, \ldots, Z_n$, that is, $Z_1 = \min_{i=1,\ldots,n} Z_i$. So

$$\mathbb{P}\{Z_1 > t\} = \mathbb{P}\left\{\min_{i=1,\ldots,n} Z_i > t\right\}.$$

If $\min_{i=1,\ldots,n} Z_i > t$, what can you say about the value of $Z_1, \ldots, Z_n$ in relation to t? Do not forget to benefit from the independence of Z_is.

(c) **[2 pts]** Determine the expected value of the random variable Z_k, that is, the expected distance to the k-th nearest neighbour.
Hint: You can use the fact that the density function of Z_k is

$$f_{Z_k}(t) = \frac{n!}{(k-1)!(n-k)!} t^{k-1}(1-t)^{n-k}, \qquad t \in [0, 1].$$

(d) **[2 pts]** Based on your answer to part (c), what can you say about the expected distance to the k-th nearest neighbour as $n \to \infty$ and $\frac{k}{n} \to 0$.

3. Information Theory – 15 pts. The goal of this question is to help you become more familiar with the basic equalities and inequalities of information theory. They appear in many contexts in machine learning and elsewhere, so having some experience with them is helpful. We review some concepts from information theory, and ask you a few questions.

Recall the definition of the entropy of a discrete random variable X with probability mass function p:

$$H(X) = \sum_x p(x) \log_2 \left(\frac{1}{p(x)} \right).$$

Here the summation is over all possible values of $x \in \mathcal{X}$, which (for simplicity) we assume is finite. For example, $\mathcal{X}$ might be $\{1, 2, \ldots, N\}$.

(a) **[3pt]** Prove that the entropy $H(X)$ is non-negative.
(b) **[3pt]** If X and Y are independent random variables, show that $H(X, Y) = H(X) + H(Y)$
(c) **[3pt]** Prove the chain rule for entropy: $H(X, Y) = H(X) + H(Y|X)$.

An important concept in information theory is the relative entropy or the KL-divergence of two distributions p and q. It is defined as

$$\mathsf{KL}(p||q) = \sum_x p(x) \log_2 \frac{p(x)}{q(x)}.$$

The KL-divergence is one of the most commonly used measure of difference (or divergence) between two distributions, and it regularly appears in information theory, machine learning, and statistics. For this question, you may assume $p(x) > 0$ and $q(x) > 0$ for all x.

If two distributions are close to each other, their KL divergence is small. If they are exactly the same, their KL divergence is zero. KL divergence is not a true distance metric, as it isn't symmetric and doesn't satisfy the triangle inequality, but we often use it as a measure of dissimilarity between two probability distributions.

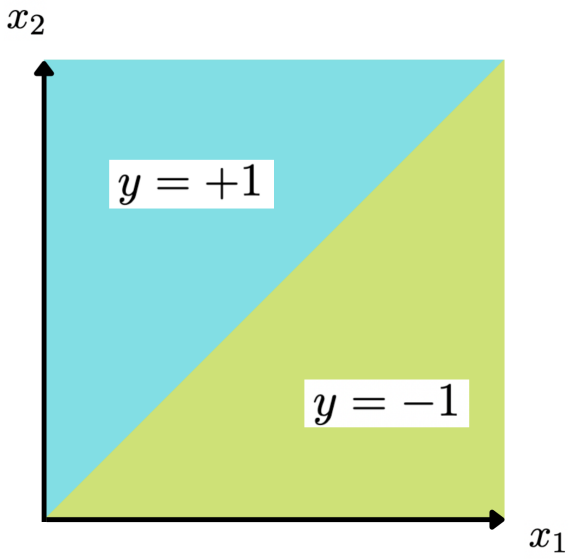
(d) **[3pt]** Prove that $\mathsf{KL}(p||q)$ is non-negative.
Hint: You may want to use Jensen's inequality, which is described in the Appendix.
(e) **[3pt]** The Information Gain or Mutual Information between X and Y is $I(Y; X) = H(Y) - H(Y|X)$. Show that

$$I(Y; X) = \mathsf{KL}(p(x, y)||p(x)p(y)),$$

where $p(x)$ is the marginal distribution of X and $p(y)$ is the marginal distribution of Y.

4. Approximation Error in Decision Trees – 10 pts. We will study how we can use a decision tree to approximate a function and how the quality of the approximation improves as the depth increases.

Consider the function $f^* : [0,1]^2 \to \{-1, +1\}$ as visualized below. This function takes the value of $+1$ on the upper left triangle and -1 on the lower right triangle.



We would like to approximate this function f^* using a decision tree with the maximum depth of d. We denote the best approximation with depth d as f_d.

(a) **[2 pts]** Explain why f_d with a finite d cannot represent f^* exactly.

(b) **[2 pts]** Show what f_4 is. You should draw the tree and include all the relevant information such as the attributes at each node, the splitting threshold, and the value of the leaves. You also need to show the regions that it induces.

Let us define the error the decision tree f_d makes in approximating f^* as

$$e_d = \int_{[0,1]^2} \mathbb{I}\{f_d(x) \neq f^*(x)\} \mathrm{d}x.$$

This is simply the area in the region $[0,1]^2$ where the function f_d is different from f^*.

(c) **[2 pts]** What is the value of e_2 and e_4?

(d) **[2 pts]** Provide the formula for e_d for any even d, that is, $d = 2k$ with $k \in \mathbb{N}$. You need to justify your formula, but your justification does not need to be detailed.

(e) **[2 pt]** What does this tell us about the quality of approximation as a function of depth d?

5. Decision Trees and K-Nearest Neighbour – 50 pts. In this question, you will use the `scikit-learn`'s decision tree and KNN classifiers to classify tweets that have been evaluated to determine whether they agree that climate change exists, or deny that it exists. The aim of this question is for you to read the `scikit-learn` API and get comfortable with training/validation splits.

We will use a dataset consisting of tweets scraped from Twitter, for which sentiment analysis has classified as either "climate change asserting" (meaning agreeing that climate change is real) or "climate change denying" (meaning disagreeing with climate change). We will now refer to these sets of tweets as **exists** and **DNE**, w.r.t. sentiment towards climate change. The **exists** dataset contains 3088 tweets, and the **DNE** dataset consists of 1099 tweets from https://data.world/xprizeai-env/sentiment-of-climate-change. The data were cleaned by removing special characters and removing all link artefacts ("[link]"). The cleaned data are available as `exists_climate.csv` and `DNE_climate.csv` on the course webpage. It is expected that you use these cleaned data sources for this assignment.

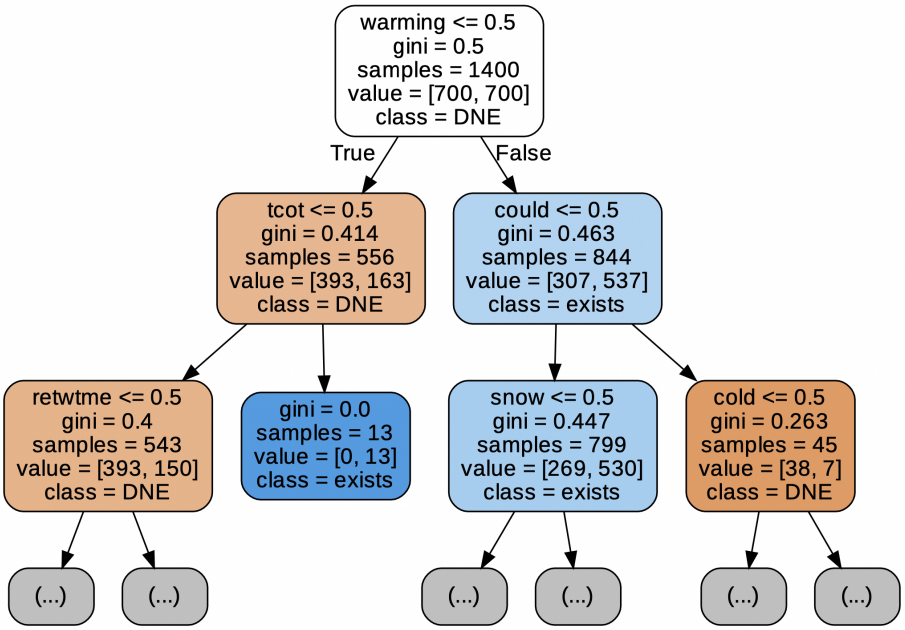
You will build a decision tree and KNN to classify "climate change asserting" or "climate change denying" tweets. Instead of coding these methods yourself, you will do what we normally do in practice: use an existing implementation. You should use the `DecisionTreeClassifier` and `KNeighborsClassifier` included in `scikit-learn`. Note that figuring out how to use this implementation, its corresponding attributes and methods is a part of the assignment.

All code should be submitted in `hw1_code.py`.

(a) **[10 pts]** Write a function `load_data` which loads the data, preprocesses it using a vectorizer (http://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text, we suggest you use `CountVectorizer` as it is the simplest in nature), and splits the entire dataset randomly into 70% training, 15% validation, and 15% test examples. You may use `train_test_split` function of `scikit-learn` within this function.

(b) **[10 pts]** (Decision Tree) Write a function `select_tree_model` that trains the decision tree classifier using at least 5 different *sensible* values of `max_depth`, as well as two different split criteria (Information Gain and Gini coefficient), evaluates the performance of each one on the validation set, and prints the resulting accuracies of each model.
You should use `DecisionTreeClassifier`, but you should write the validation code yourself. Include the output of this function in your solution.

(c) **[10 pts]** (Decision Tree) Now let's stick with the hyperparameters which achieved the highest validation accuracy. Report its accuracy on the test dataset. Moreover, extract and visualize the first two layers of the tree. Your visualization may look something like what is shown below, but it does not have to be an image; it is perfectly fine to display text. It may also be hand-drawn. Include your visualization in your solution pdf.



(d) **[10 pts]** (Decision Tree) Write a function `compute_information_gain` which computes the information gain of a split on the training data. That is, compute $I(Y, x_i)$, where Y is the random variable signifying whether the tweet is climate change asserting or denying, and x_i is the keyword chosen for the split. Your split should be based on whether the keyword x_i exists (True) or does not exist (False). You should ignore the number of times that the keyword appears in the sentence.
Report the outputs of this function for the topmost split from the previous part, and for several other keywords.

(e) **[10 pts]** (KNN) Write a function `select_knn_model` that uses a KNN classifier to classify between climate change asserting or denying tweets. Use a range of k values between 1 to 20 and compute both training and validation errors. You should generate a graph similar to the one on slide 44 of Lecture #1, which is Figure 2.4 of the Elements of Statistical Learning. You do not need to worry about the Bayes error or the Linear classifier in that figure. Report the generated graph in your report. Choose the model with the best validation accuracy and report its accuracy on the test data.

APPENDIX A: CONVEXITY AND JENSEN'S INEQUALITY

The briefly review the concept of convexity, which you may find useful for some of the questions in this assignment. You may assume anything given here.

Convexity is an important concept in mathematics with many uses in machine learning. We briefly define convex set and function and some of their properties here. Using these properties are useful in solving some of the questions in this homework. If you are interested to know more about convexity, refer to Boyd and Vandenberghe, Convex Optimization, 2004.

A set C is *convex* if the line segment between any two points in C lies within C, i.e., if for any $x_1, x_2 \in C$ and for any $0 \leq \lambda \leq 1$, we have

$$\lambda x_1 + (1 - \lambda)x_2 \in C.$$

For example, a cube or sphere in $\mathbb{R}^d$ are convex sets, but a cross (a shape like X) is not.

A function $f : \mathbb{R}^d \to \mathbb{R}$ is *convex* if its domain is a convex set and if for all x_1, x_2 in its domain, and for any $0 \leq \lambda \leq 1$, we have

$$f(\lambda x_1 + (1 - \lambda)x_2) \leq \lambda f(x_1) + (1 - \lambda)f(x_2).$$

This inequality means that the line segment between $(x_1, f(x_1))$ and $(x_2, f(x_2))$ lies above the graph of f. A convex function looks like $\smile$. We say that f is *concave* if $-f$ is convex. A concave function looks like $\frown$.

Some examples of convex and concave functions are (you do not need to use most of them in your homework, but knowing them is useful):

- Powers: x^p is convex on the set of positive real numbers when $p \geq 1$ or $p \leq 0$. It is concave for $0 \leq p \leq 1$.
- Exponential: e^{ax} is convex on $\mathbb{R}$, for any $a \in \mathbb{R}$.
- Logarithm: $\log(x)$ is concave on the set of positive real numbers.
- Norms: Every norm on $\mathbb{R}^d$ is convex.
- Max function: $f(x) = \max\{x_1, x_2, \ldots, x_d\}$ is convex on $\mathbb{R}^d$.
- Log-sum-exp: The function $f(x) = \log(e^{x_1} + \ldots + e^{x_d})$ is convex on $\mathbb{R}^d$.

An important property of convex and concave functions, which you may need to use in your homework, is *Jensen's inequality*. Jensen's inequality states that if $\phi(x)$ is a convex function of x, we have

$$\phi(\mathbb{E}[X]) \leq \mathbb{E}[\phi(X)].$$

In words, if we apply a convex function to the expectation of a random variable, it is less than or equal to the expected value of that convex function when its argument is the random variable. If the function is concave, the direction of the inequality is reversed.

Jensen's inequality has a physical interpretation: Consider a set $\mathcal{X} = \{x_1, \ldots, x_N\}$ of points on $\mathbb{R}$. Corresponding to each point, we have a probability $p(x_i)$. If we interpret the probability as mass, and we put an object with mass $p(x_i)$ at location $(x_i, \phi(x_i))$, then the centre of gravity of these objects, which is in $\mathbb{R}^2$, is located at the point $(\mathbb{E}[X], \mathbb{E}[\phi(X)])$. If ϕ is convex $\smile$, the centre of gravity lies above the curve $x \mapsto \phi(x)$, and vice versa for a concave function $\frown$.