EECS 553 Machine Learning
Homework 1

Jackson Chen

Wednesday 5th February, 2025

U M I C H
R O B O T
T E N E T
T O B O R
H C I M U

Problem I

Consider a synthetic dataset where training and test data are generated with n samples in d dimension according to X_{train} = `np.random.randn(n,d)/np.sqrt(d)`. Here $1/np.sqrt(d)$ normalization ensures that the points approximately lie on the unit Euclidean ball. Set $k = 1$ i.e. we are investigating vanilla nearest neighbor classifier. Let x be a test example similarly drawn from `np.random.randn(d)`. Let $d(x)$ be the distance to the nearest neighbor i.e.

$$d(x) = \min_{x' \in X_{\text{train}}} \|x - x'\|_2 .$$

We will plot the expected distance $\mathbb{E}[d(x)]$ as a function of d and n. Concretely, we vary $d \in \{2, 4, 6, 8, 10\}$ and $n = \{100, 200, 500, 1000, 2000, 5000\}$. In order to estimate the expectation, average over sufficiently many test points x (e.g. 100 realizations).

- Plot $\mathbb{E}[d(x)]$ as a function of n and create a separate curve for each choice of d. Create the plot in semilogx for better visualization.

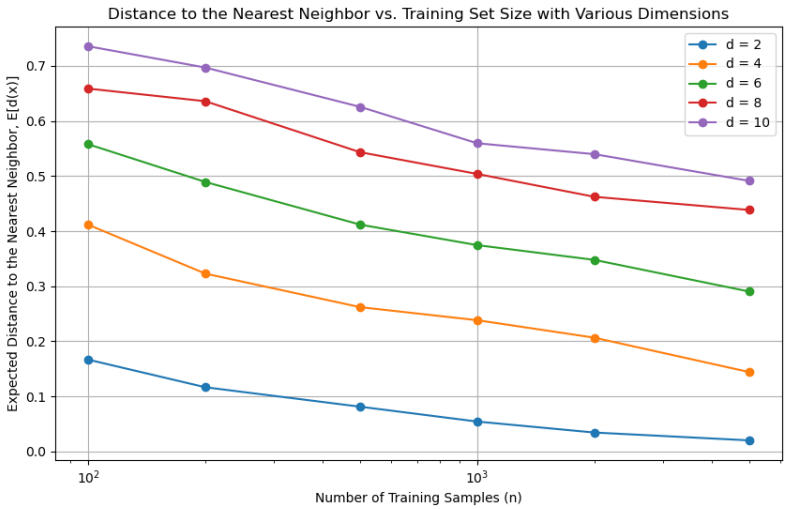

Figure 1: Expected Distance to the Nearest Neighbor vs. Training Set Size.

- Comment on the influence of d on the distance to the nearest neighbor.
  With the increase of d, the expected distance to the nearest neighbor becomes larger. The higher dimension the space is, the sparser the points in the space would become, regardless of how informative the data points are. In another word, you will need exponentially more points in higher dimension to achieve the same level of cluster as in lower dimensions.

- Suppose we wish to ensure $\mathbb{E}[d(x)] \leq \epsilon$ for some $\epsilon > 0$. Based on these experiments, how do you expect n should grow as a function of d and ϵ (your best guess e.g. exponential vs polynomial dependence)?

I would say it depends on r in a polynomial way while on d exponentially. So I did a little bit research online and found this "scaling law":

$$\mathbb{E}[d(x)] \sim n^{-1/d}$$

It comes from the cumulative distribution function of $d(x)$, which essentially is the probability of the nearest neighbor distance is less or equal to r. With this property, we can set

$$n^{-1/d} \leq \epsilon.$$

and thus we get

$$n \geq \epsilon^{-d}.$$

Problem II

Consider a synthetic dataset where training and test data are generated with n samples in $d = 2$ dimension according to `X_train=np.random.randn(n,2)`, `y_train=sign(np.random.randn(n))` (same for test). We will evaluate the computational performance of different kNN implementations on this dataset. In `sklearn.KNeighborsClassifier`, you can set `algorithm` to one of `algo='ball tree', 'kd tree', 'brute'`. Set k=5 neighbors i.e. set `clf = neighbors.KNeighborsClassifier(5, algorithm=algo)`. Use $n_{test} = 5000$ test examples for your evaluations.

(a) Verify that 'brute' is the fastest algorithm for training i.e. for running clf.fit(X_train, y_train). Try n = 1000, n = 10000, n = 100000 and report the time using time.time().

n	Ball Tree	KD Tree	Brute
1 000	0.000580 s	0.000496 s	**0.000457** s
10 000	0.002526 s	0.002678 s	**0.000727** s
100 000	0.034814 s	0.029738 s	**0.005301** s

Table 1: Training time for `KNeighborsClassifier.fit()` with different algorithms.

`Brute` is indeed the fastest algorithm for training with $k = 5$ with a 2 dimension dataset, especially when the scale of the dataset increases.

(b) Verify that 'brute' becomes slower at inference for sufficiently large n. By inference, we mean the time it takes to run clf.predict(X_test). Specifically, evaluate on the grid $n \in [1000, 2000, 5000, 10000, 20000, 50000, 100000, 200000, 500000]$ and plot the inference time of the three kNN methods as a function of n. For which n choice, 'brute' becomes the slowest option for the first time?

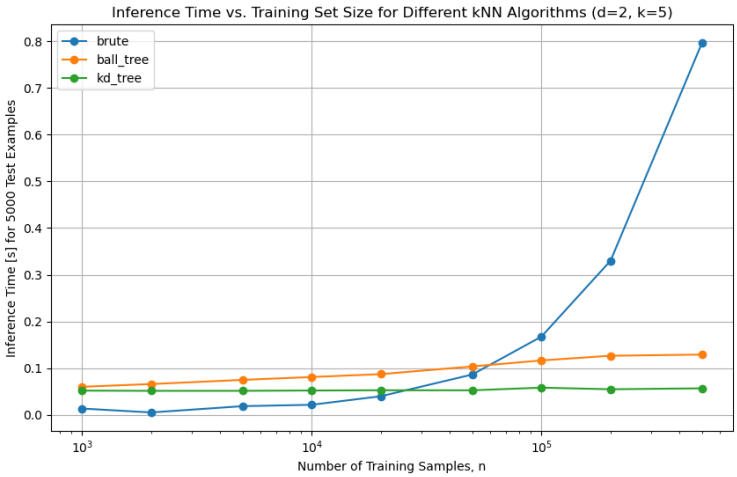

Figure 2: Inference Time vs. Training Set Size with Different kNN Algorithms.

We notice that for $n \geq 10^5$, `brute` becomes the slowest option.

Problem III

PSD matrices

(a) Prove that if all eigenvalues of a symmetric matrix A are positive, then $\boldsymbol{x}^T A\boldsymbol{x} > 0$ for all $\boldsymbol{x} \neq 0$ (and hence $\boldsymbol{A}$ is positive definite).

Proof. Since A is symmetric, by the Spectral Theorem there exists an orthogonal matrix Q (i.e., $Q^TQ = I$) and a diagonal matrix

$$\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$$

such that

$$A = Q\Lambda Q^T.$$

Here, the eigenvalues λ_i are all positive by assumption.

Let $\boldsymbol{x}$ be an arbitrary nonzero vector in $\mathbb{R}^n$ and define

$$\boldsymbol{y} = Q^T\boldsymbol{x}.$$

Since Q is orthogonal, the transformation is invertible, and hence $\boldsymbol{y} \neq \boldsymbol{0}$.

Now, consider the quadratic form:

$$\boldsymbol{x}^T A\boldsymbol{x} = \boldsymbol{x}^T Q\Lambda Q^T\boldsymbol{x} = \boldsymbol{y}^T\Lambda\boldsymbol{y}.$$

Because Λ is diagonal, this expression simplifies to

$$\boldsymbol{y}^T\Lambda\boldsymbol{y} = \sum_{i=1}^{n} \lambda_i y_i^2.$$

Since each $\lambda_i > 0$ and at least one $y_i \neq 0$ (because $\boldsymbol{y} \neq \boldsymbol{0}$), we have

$$\sum_{i=1}^{n} \lambda_i y_i^2 > 0.$$

Therefore, for all nonzero $\boldsymbol{x}$,

$$\boldsymbol{x}^T A\boldsymbol{x} > 0,$$

which proves that A is positive definite. □

(b) A Gram matrix is any $d \times d$ matrix whose $(i, j)^{th}$ entry is $\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$ for some vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_d$. Show that Gram matrices are positive semi-definite.

Proof. We can write a Gram matrix in the following form (d = 3):

$$\begin{bmatrix} x_1^T x_1 & x_1^T x_2 & x_1^T x_3 \\ x_2^T x_1 & x_2^T x_2 & x_2^T x_3 \\ x_3^T x_1 & x_3^T x_2 & x_3^T x_3 \end{bmatrix}$$

Since the inner product of two vectors is commutative, we can tell that the matrix is symmetric no matter the value of d. Using the conclusion from part (a), it is obviously that a Gram matrix is PSD. □

Problem IV

Unconstrained Optimization

1. Let $\boldsymbol{A}$ be an $m \times n$ matrix and $\boldsymbol{b} \in \mathbb{R}^m$. Consider a convex function $f : \mathbb{R}^m \to \mathbb{R}$. Using the definition of convexity, prove that $g(\boldsymbol{x}) = f(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b})$ is convex.

   Proof. We know from the definition that a function $f : \mathbb{R}^m \to \mathbb{R}$ is convex if for any $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^m$ and for all $\lambda \in [0, 1]$, the following inequality holds:
   $$f\big(\lambda\boldsymbol{u} + (1-\lambda)\boldsymbol{v}\big) \leq \lambda f(\boldsymbol{u}) + (1-\lambda) f(\boldsymbol{v}).$$
   Let $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ and let $\lambda \in [0, 1]$ be arbitrary. Again, let
   $$\boldsymbol{u} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b} \quad \text{and} \quad \boldsymbol{v} = \boldsymbol{A}\boldsymbol{y} + \boldsymbol{b}.$$
   Thus, we have
   $$f\big(\lambda\boldsymbol{u} + (1-\lambda)\boldsymbol{v}\big) \leq \lambda f(\boldsymbol{u}) + (1-\lambda) f(\boldsymbol{v}).$$
   Notice that the affine transformation distributes over the convex combination:
   $$\lambda\boldsymbol{u} + (1-\lambda)\boldsymbol{v} = \lambda(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}) + (1-\lambda)(\boldsymbol{A}\boldsymbol{y} + \boldsymbol{b}) = \boldsymbol{A}(\lambda\boldsymbol{x} + (1-\lambda)\boldsymbol{y}) + \boldsymbol{b}.$$
   Thus, the inequality becomes
   $$f\Big(\boldsymbol{A}(\lambda\boldsymbol{x} + (1-\lambda)\boldsymbol{y}) + \boldsymbol{b}\Big) \leq \lambda f(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}) + (1-\lambda) f(\boldsymbol{A}\boldsymbol{y} + \boldsymbol{b}).$$
   And it cam be rewrite the inequality as
   $$g(\lambda\boldsymbol{x} + (1-\lambda)\boldsymbol{y}) \leq \lambda g(\boldsymbol{x}) + (1-\lambda) g(\boldsymbol{y}),$$
   which proves that the function g is convex. □

2. Prove that if f is strictly convex, f has at most one minimizer (recall that for convex functions, all local minima are also global minima).

   Proof. We prove by **contradiction**:
   Suppose that f has two distinct minimizers $\boldsymbol{x}^*$ and $\boldsymbol{y}^*$ with
   $$f(\boldsymbol{x}^*) = f(\boldsymbol{y}^*) = \inf_{\boldsymbol{x} \in \mathbb{R}^n} f(\boldsymbol{x}).$$
   Since f is strictly convex, for any λ with $0 < \lambda < 1$ we have
   $$f\big(\lambda\boldsymbol{x}^* + (1-\lambda)\boldsymbol{y}^*\big) < \lambda f(\boldsymbol{x}^*) + (1-\lambda) f(\boldsymbol{y}^*).$$
   Because $\boldsymbol{x}^*$ and $\boldsymbol{y}^*$ are both minimizers, it follows that
   $$\lambda f(\boldsymbol{x}^*) + (1-\lambda) f(\boldsymbol{y}^*) = f(\boldsymbol{x}^*).$$
   Thus,
   $$f\big(\lambda\boldsymbol{x}^* + (1-\lambda)\boldsymbol{y}^*\big) < f(\boldsymbol{x}^*).$$
   This contradicts what we supposed at the first place. Thus, f must have at most one minimizer. □

3. Consider the function $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}^T\boldsymbol{x} + c$, where $\boldsymbol{A}$ is a symmetric $d \times d$ matrix. Derive the Hessian of f. Under what conditions on $\boldsymbol{A}$, f is convex? Strictly convex?
The Hessian is the derivative of the gradient with respect to $\boldsymbol{x}$. Since

$$\nabla f(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b},$$

taking the derivative with respect to $\boldsymbol{x}$ yields

$$\nabla^2 f(\boldsymbol{x}) = \boldsymbol{A}.$$

A twice differentiable function f is convex if its Hessian is PSD for all $\boldsymbol{x}$, and strictly convex if its Hessian is positive definite (PD) for all $\boldsymbol{x}$.

Since

$$\nabla^2 f(\boldsymbol{x}) = \boldsymbol{A},$$

the conditions for convexity and strict convexity are:

- f is **convex** if and only if $\boldsymbol{A}$ is positive semidefinite, i.e.,

$$\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \geq 0 \quad \text{for all } \boldsymbol{x} \in \mathbb{R}^d.$$

- f is **strictly convex** if and only if $\boldsymbol{A}$ is positive definite, i.e.,

$$\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} > 0 \quad \text{for all nonzero } \boldsymbol{x} \in \mathbb{R}^d.$$

Problem V

Consider the 0-1 loss. In class, the Bayes classifier was defined and discussed for multiclass classification. The Bayes classifier h^* achieves the lowest risk $R(h^*) = R^*$, which is called the Bayes Risk. This means that the quantity $R(h) - R^*$, which we call the *excess risk*, is non-negative for any classifier h.

In this problem, we consider binary classification with label $Y \in \{1, -1\}$, and define $\eta(\boldsymbol{x}) := \Pr(Y = 1 \mid X = x)$, the probability that $Y = 1$ given than $\boldsymbol{X}$ takes on the value x. For any classifier h, prove that the excess risk is given by

$$R(h) - R^* = \mathbb{E}_{\boldsymbol{X}}\left[|2\eta(\boldsymbol{X}) - 1|\mathbf{1}_{\{h(\boldsymbol{X}) \neq \text{sign}(2\eta(\boldsymbol{X})-1)\}}\right]$$

Here

$$\text{sign}(t) := \begin{cases} 1 & t \geq 0 \\ -1 & t < 0 \end{cases}$$

The convention $\text{sign}(0) = 1$ does not affect the problem. The results says that the excess risk depends on how much (on average) $\eta(\boldsymbol{X})$ deviates from $\frac{1}{2}$ at points where h disagrees with the Bayes classifier.

Hint: Refer to the proof for the Bayes classifier in the lecture notes, and for the binary case rewrite the proof in terms of η.

The Bayes classifier is given by

$$h^*(x) = \text{sign}(2\eta(x) - 1),$$

with the convention $\text{sign}(0) = 1$. Its risk is

$$R^* = R(h^*) = \mathbb{E}_X\left[\min\{\eta(x), 1 - \eta(x)\}\right].$$

For any classifier h, the risk is

$$R(h) = \mathbb{E}_X\left[\eta(x)\mathbf{1}_{\{h(x)\neq 1\}} + (1 - \eta(x))\mathbf{1}_{\{h(x)\neq -1\}}\right].$$

Proof. We pick an arbitrary $x \in \mathbb{R}^d$ and have

$$\eta := \eta(x).$$

Then, the conditional risk at x for any classifier h is

$$R_x(h) = \eta\mathbf{1}_{\{h(x)\neq 1\}} + (1 - \eta)\mathbf{1}_{\{h(x)\neq -1\}}.$$

The Bayes classifier at x is

$$h^*(x) = \text{sign}(2\eta - 1),$$

with its conditional risk as

$$R_x(h^*) = \min\{\eta, 1 - \eta\}.$$

If $h(x) = h^*(x)$.
Then,

$$R_x(h) = R_x(h^*) = \min\{\eta, 1 - \eta\},$$

so that

$$R_x(h) - R_x(h^*) = 0.$$

Note that in this case, $\mathbf{1}_{\{h(x)\neq h^*(x)\}} = 0$, and hence

$$|2\eta - 1|\mathbf{1}_{\{h(x)\neq h^*(x)\}} = 0.$$

If $h(x) \neq h^*(x)$.

When $h^*(x) = 1$, which happens if $2\eta - 1 \geq 0$, i.e. $\eta \geq \frac{1}{2}$ and $h(x) = -1$. Then,

$$R_x(h) = \eta \quad (\text{since } h(x) = -1 \text{ makes the error on the event } Y = 1),$$

and

$$R_x(h^*) = 1 - \eta.$$

Thus we have,

$$R_x(h) - R_x(h^*) = \eta - (1 - \eta) = 2\eta - 1 = |2\eta - 1|,$$

since $2\eta - 1 \geq 0$.
When $h^*(x) = -1$, which happens if $2\eta - 1 < 0$, i.e. $\eta < \frac{1}{2}$ and $h(x) = 1$. Then,

$$R_x(h) = 1 - \eta \quad (\text{since } h(x) = 1 \text{ makes the error on the event } Y = -1),$$

and

$$R_x(h^*) = \eta.$$

Thus we have,

$$R_x(h) - R_x(h^*) = (1 - \eta) - \eta = 1 - 2\eta = |2\eta - 1|,$$

since now $1 - 2\eta > 0$.

In conclusion, when $h(x) \neq h*(x)$, we have

$$R_x(h) - R_x(h^*) = |2\eta - 1|.$$

Essentially, we have

$$R_x(h) - R_x(h^*) = |2\eta - 1|\mathbf{1}_{\{h(x)\neq h^*(x)\}}.$$

Taking the expectation over X, we obtain

$$R(h) - R^* = \mathbb{E}_X\left[R_x(h) - R_x(h^*)\right] = \mathbb{E}_X\left[|2\eta(X) - 1|\mathbf{1}_{\{h(X)\neq \text{sign}(2\eta(X)-1)\}}\right].$$

□