Introduction to Machine Learning (0368-3235) - Assignment 1
Lior Erenreich

Homework 1:

Linear Algebra:

1.

1. **(15 pts)** A symmetric matrix A over $\mathbb{R}$ is called *positive semidefinite* (PSD) if for every vector $\mathbf{v}$, $\mathbf{v}^T A\mathbf{v} \geq 0$.

   (a) Show that a symmetric matrix A is PSD if and only if it can be written as $A = XX^T$, if and only if all of its eigenvalues are non-negative.

   Hint: Recall that a real symmetric matrix A can be decomposed as $A = QDQ^T$, where Q is an orthogonal matrix whose columns are eigenvectors of A and D is a diagonal matrix with eigenvalues of A as its diagonal elements.

   (b) Show that for all $\alpha, \beta \geq 0$ and PSD matrices $A, B \in \mathbb{R}^{n\times n}$, the matrix $\alpha A + \beta B$ is also PSD. Does this mean that the set of all $n \times n$ PSD matrices over $\mathbb{R}$ is a vector space over $\mathbb{R}$?

a. Proof:
Suppose A is a symmetric matrix that can be written as $A = XX^T$ for some matrix X. Let v be any vector. Then we have:

$$v^T Av = v^T XX^T v = (X^T v)^T (X^T v) \geq 0$$

since the square of any real number is non-negative. Therefore, matrix A is PSD.

Conversely, suppose A is a symmetric PSD matrix. Then by the spectral theorem, we can write A as $A = QDQ^T$, where Q is an orthogonal matrix whose columns are eigenvectors of A and D is a diagonal matrix with eigenvalues of A as its diagonal elements.
Let $X = Q\sqrt{D}$, where $\sqrt{D}$ is the diagonal matrix with the square roots of the eigenvalues of A as its diagonal elements. Then we have:

$$XTX = (\sqrt{D}QT)(Q\sqrt{D}) = \sqrt{D}QQT\sqrt{D} = \sqrt{D}D = D$$

since Q is orthogonal. Therefore, $A = XX^T$, and all of its eigenvalues are non-negative since they are the diagonal elements of D. ■

$$q_i^T QDQ^T q_i = e_i^T De_i = e_i^T d_i e_i = d_i ||e_i||^2 = d_i$$

b. To show that $\alpha \cdot A + \beta \cdot B$ is PSD for all $\alpha, \beta \geq 0$ and PSD matrices $A, B \in \mathbb{R}^{n\times n}$, we need to show that for any vector v,

$$v^T(\alpha A + \beta B)v \geq 0$$

We have:

$$v^T(\alpha A + \beta B)v = \alpha v^T Av + \beta v^T Bv$$

Since A and B are PSD, we have $v^T Av \geq 0$ and $v^T Bv \geq 0$ for any v.
Therefore, $\alpha v^T Av + \beta v^T Bv \geq 0$ for all $\alpha, \beta \geq 0$ and PSD matrices $A, B \in \mathbb{R}^{n\times n}$.
Hence, $\alpha A + \beta B$ is PSD.

Introduction to Machine Learning (0368-3235) - Assignment 1
Lior Erenreich

However, the set of all $n \times n$ PSD matrices over $\mathbb{R}$ is not a vector space over $\mathbb{R}$, because it is not closed under scalar multiplication by negative numbers. That is, for a PSD matrix A, and a negative scalar $-\alpha$, the matrix $-\alpha A$ is not necessarily PSD, since for some vector v, we may have $v^T(-\alpha A)v = -\alpha(v^T A v)$ which may not be non-negative. Therefore, the set of all $n \times n$ PSD matrices over $\mathbb{R}$ is not a vector space over $\mathbb{R}$.

Calculus and Probability:

1.

> 1. (**15 pts**) Let $X_1, ..., X_n$ be i.i.d $U([0,1])$ (uniform) continuous random variables. Let $Y = \max(X_1, ..., X_n)$.
>
> (a) What is the PDF of Y? Write the mathematical formula and plot the PDF as well. Compute $\mathbb{E}[Y]$ and $\mathrm{Var}[Y]$ - how do they behave as a function of n as n grows large?
>
> (b) (**No need to submit**) Verify your answer empirically using Python.

a. To find the PDF of Y, we can first find the CDF of Y:

$$P(Y \le y) = P(\max(X_1, \dots, X_n) \le y) = P(X_1 \le y, \dots, X_n \le y)$$
$$= P(X_1 \le y) \cdots P(X_n \le y)$$

since the events $X_1 \le y, \dots, X_n \le y$ are independent.
Since $X_1, \dots, X_n$ are uniformly distributed on $[0,1]$, we have $P(X_i \le y) = y$ for $0 \le y \le 1$.
Therefore, we have: $P(Y \le y) = y^n$ for $0 \le y \le 1$. Taking the derivative with respect to y, we get the PDF of Y:

$$f_Y(x) = nx^{n-1} \quad for \quad 0 < y < 1, \qquad otherwise\ 0.$$

To compute $\mathbb{E}[Y]$, we have:

$$\mathbb{E}[Y] = \int_0^1 y n y^{n-1}\, dy = \frac{n}{n+1}$$

To compute $Var[Y]$, we can use the formula $Var[Y] = \mathbb{E}[Y^2] - \mathbb{E}[Y]^2$. We have:

$$\mathbb{E}[Y^2] = \int_0^1 y n y^{n-1}\, dy = \frac{n}{n+2}$$

Therefore,

$$Var[Y] = \mathbb{E}[Y^2] - \mathbb{E}[Y]^2 = \frac{n}{n+2} - \left(\frac{n}{n+1}\right)^2$$

As n grows large, we have $\mathbb{E}[Y \approx 1$ and $Var[Y] \approx \frac{1}{n+2}$, which shows that Y becomes more and more concentrated around its mean as n grows large.

Optimal Classifiers and Decision Rules:

1.

Introduction to Machine Learning (0368-3235) - Assignment 1
Lior Erenreich

1. **(15 pts)**

(a) Let X and Y be random variables where Y can take values in $\mathcal{Y} = \{1, \ldots, L\}$. Let ℓ_{0-1} be the 0-1 loss function defined in class. Show that $h = \arg \min_{f:\mathcal{X}\to\mathcal{Y}} \mathbb{E}\left[\ell_{0-1}(Y, f(X))\right]$ is given by

$$h(x) = \arg\max_{i\in\mathcal{Y}} \mathbb{P}[Y = i|X = x]$$

(b) Let X and Y be random variables where Y can take values in $\mathcal{Y} = \{0, 1\}$. Let Δ be the following asymmetric loss function:

$$\Delta(y, \hat{y}) = \begin{cases} 0 & y = \hat{y} \\ a & y = 0, \hat{y} = 1 \\ b & y = 1, \hat{y} = 0, \end{cases}$$

where $a, b \in (0, 1]$ (note that this loss function generalizes the 0-1 loss defined in class). Compute the optimal decision rule h for the loss function Δ, i.e. the decision rule which satisfies:

$$h = \arg \min_{f:\mathcal{X}\to\mathcal{Y}} \mathbb{E}\left[\Delta(Y, f(X))\right]$$

a. The goal is to show that the hypothesis function that minimizes the expected ℓ_{0-1} loss function is given by:

$$h(x) = \arg max_i\, P[Y = i|X = x]$$

I start by expanding the definition of the expected ℓ_{0-1} loss function:

$$\mathbb{E}\big[\ell_{0-1}\big(Y, f(X)\big)\big] = \sum_{i=1}^{L} \ell_{0-1}\big(i, h(x)\big)P[X = x, Y = i]$$

Simplify according to Bayes law:

$$\mathbb{E}\big[\ell_{0-1}\big(Y, f(X)\big)\big] = \sum_{i=1}^{L} \ell_{0-1}\big(i, h(x)\big)P[X = x]P[Y = i|X = x]$$

Expanding the definition of the expected ℓ_{0-1} loss function:

$$\mathbb{E}\big[\ell_{0-1}\big(Y, f(X)\big)\big] = \sum_{i=1,\ i\neq h(x)}^{L} P[X = x]P[Y = h(x)|X = x]$$
$$= P[X = x](1 - P[Y = h(x)|X = x])$$

The optimal $f(x)$ is the one that minimizes the expected loss, that is:

$$h(x) = i = \arg max_i P[Y = i|X = x]$$

Since the summands are non-negative, minimizing the sum is equivalent to minimizing each term. Therefore:

$$f(x) = \arg min_u P[Y \neq u|X = x]$$

b. The optimal decision rule h for the loss function Δ is the one that minimizes the expected loss, i.e.

$$h = arg\min \mathbb{E}\big[\Delta\big(Y, f(X)\big)\big]$$

Let's consider the case where $f(X) = 1$. Then, we have:

$$\mathbb{E}\big[\Delta\big(Y, f(X)\big)\big] = \mathbb{E}_X\Big[\mathbb{E}\big[\Delta\big(Y, f(X)\big)|X\big]\Big]$$
$$= \mathbb{E}_X\big[P(Y=0|X)\Delta\big(0, h(X)\big) + P(Y=1|X)\Delta\big(1, h(X)\big)\big]$$
$$h(x) = \begin{cases} 1, & aP(Y=0|X=x) < bP(Y=1|X=x) \\ 0, & otherwise \end{cases}$$

2.

2. (**15 pts**) Let X and Y be random variables where X can take values in some set $\mathcal{X}$ and Y can take values in $\mathcal{Y} = \{0, 1\}$ (i.e. binary label space). Assume we wish to find a predictor $h : \mathcal{X} \to [0, 1]$ (note that the hypothesis can output any number between 0 and 1) which minimizes $\mathbb{E}[\Delta_{log}(Y, h(X))]$, where Δ_{log} is the following loss function known as the *log-loss*:

$$\Delta_{log}(y, \hat{y}) = -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y}).$$

Find the predictor $h : \mathcal{X} \to [0, 1]$ which minimizes $\mathbb{E}[\Delta_{log}(Y, h(X))]$.

Note: This loss function may seem odd at first, but it is very important and we'll discuss it further in the future.

To find the predictor h that minimizes the expected log-loss, we need to minimize the following expression:

$$\mathbb{E}\big[\Delta_{\log}\big(Y, h(X)\big)\big] = \mathbb{E}\big[-Ylog\big(h(X)\big) - (1-Y)log\big(1 - h(X)\big)\big]$$

To find the optimal predictor h, we can take the derivative of this expression with respect to $h(X)$ and set it equal to zero:

$$\frac{d}{dh(X)}\big(\mathbb{E}\big[\Delta_{\log}\big(Y, h(X)\big)\big]\big) = \mathbb{E}\left[-\frac{Y}{h(X)} + \frac{1-Y}{1-h(X)}\right] = 0$$

Solving for $h(X)$, we get:

$$\frac{\mathbb{E}[Y]}{\mathbb{E}[h(X)]} = \frac{\mathbb{E}[1] - \mathbb{E}[Y]}{\mathbb{E}[1 - h(X)]}$$

Simplifying, we get:

$$h(X) = \mathbb{E}[Y]$$

Therefore, the optimal predictor h is simply the expected value of Y.

Introduction to Machine Learning (0368-3235) - Assignment 1
Lior Erenreich

3. **(10 pts)**

Let X and Y be random variables taking values in $\mathcal{X} = \mathbb{R}$ and $\mathcal{Y} = \{0, 1\}$ respectively, and assume that given $Y = 0$, X is distributed normally with mean μ and variance σ_0^2, i.e. $X \sim \mathcal{N}(\mu, \sigma_0^2)$, and similarly, given $Y = 1$, $X \sim \mathcal{N}(\mu, \sigma_1^2)$, where $\sigma_0 \neq \sigma_1$. Also assume $\Pr[Y = 1] = p_1$.

Find the optimal decision rule for this distribution and the zero-one loss, i.e. find $h : \mathbb{R} \to \{0, 1\}$ which minimizes $\mathbb{E}[\ell_{0-1}(Y, h(X))]$ where ℓ_{0-1} is the zero-one loss defined in class (write the decision rule only in terms of $x, \mu, \sigma_0, \sigma_1$ and p_1).

To find the optimal decision rule h, we need to minimize the expected ℓ_{0-1}, in class we have seen that the decision rule that minimizes the expected ℓ_{0-1} loss in the binary case is:

$$h(x) = \begin{cases} 1, & P[Y = 1|X = x] > P[Y = 0|X = x] \\ 0, & otherwsie \end{cases}$$

So our goal is to understand for which value of x, the following restriction holds:

$$P[Y = 1|X = x] > P[Y = 0|X = x]$$

Bayes Rule:

$$P[Y = y|X = x] = \frac{\Pr(Y = y)\, f_{X|Y}(x)}{f_x(x)}$$

$$\frac{P[Y = 1] f_X(x|Y = 1)}{f_X(x)} > \frac{P[Y = 0] f_X(x|Y = 0)}{f_X(x)}$$

$$\frac{f_X(x|Y = 1)}{f_X(x|Y = 0)} > \frac{P[Y = 0]}{P[Y = 1]}$$

The values are:

$$f_X(x|Y = 1) = \frac{1}{\sqrt{2\pi\sigma_1^2}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma_1^2}}$$

$$f_X(x|Y = 0) = \frac{1}{\sqrt{2\pi\sigma_0^2}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma_0^2}}$$

$$P[Y = 0] = 1 - p_1$$

$$P[Y = 1] = p_1$$

So we get:

$$\frac{\sqrt{2\pi\sigma_0^2} \cdot e^{-\frac{(x-\mu)^2}{2\sigma_1^2}}}{\sqrt{2\pi\sigma_1^2} \cdot e^{-\frac{(x-\mu)^2}{2\sigma_0^2}}} > \frac{1 - p_1}{p_1}$$

Introduction to Machine Learning (0368-3235) - Assignment 1

Lior Erenreich

$$e^{-\frac{(x-\mu)^2}{2\sigma_1^2}+\frac{(x-\mu)^2}{2\sigma_0^2}} > \frac{1-p_1}{p_1}\cdot\frac{\sigma_1}{\sigma_0}$$

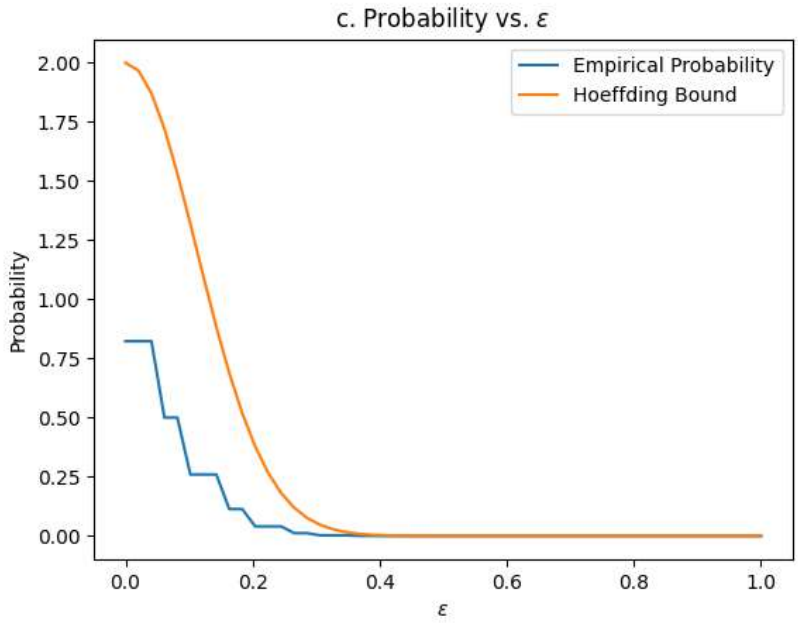
$$\frac{(x-\mu)^2}{2}\left(\frac{1}{\sigma_0^2}-\frac{1}{\sigma_1^2}\right) > \log\left(\frac{1-p_1}{p_1}\cdot\frac{\sigma_1}{\sigma_0}\right)$$

After simplification:

$$x > \mu + \sqrt{2\frac{\sigma_0^2\sigma_1^2}{\sigma_1^2-\sigma_0^2}\cdot\log\left(\frac{1-p_1}{p_1}\cdot\frac{\sigma_1}{\sigma_0}\right)}$$

Programming Assignment

1. Visualizing the Hoeffding bound:
   a. Plot:

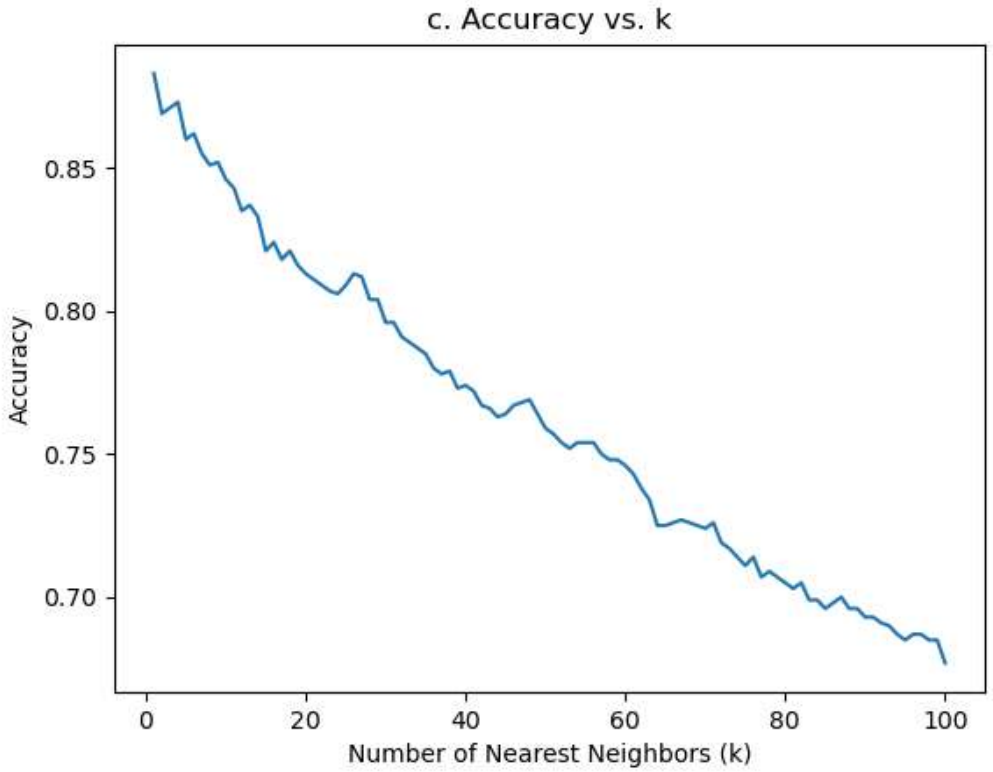


c. Nearest Neighbor:
   a. In the submitted code.
   b. The accuracy of the prediction I got for $k = 10$, $n = 1000$, is 84.3% (as you can see by running the code, the value of accuracy is printed).

```
b. The accuracy for k=10, n=1000 is  84.3 %.
```

   I would expect from a completely random predictor to have an accuracy of about 10%, that's because it would guess for each image a digit out of 10 digits, so assuming that the images are uniformly distributed, the probability for a true guess for each is 10%.
   c. Plot:

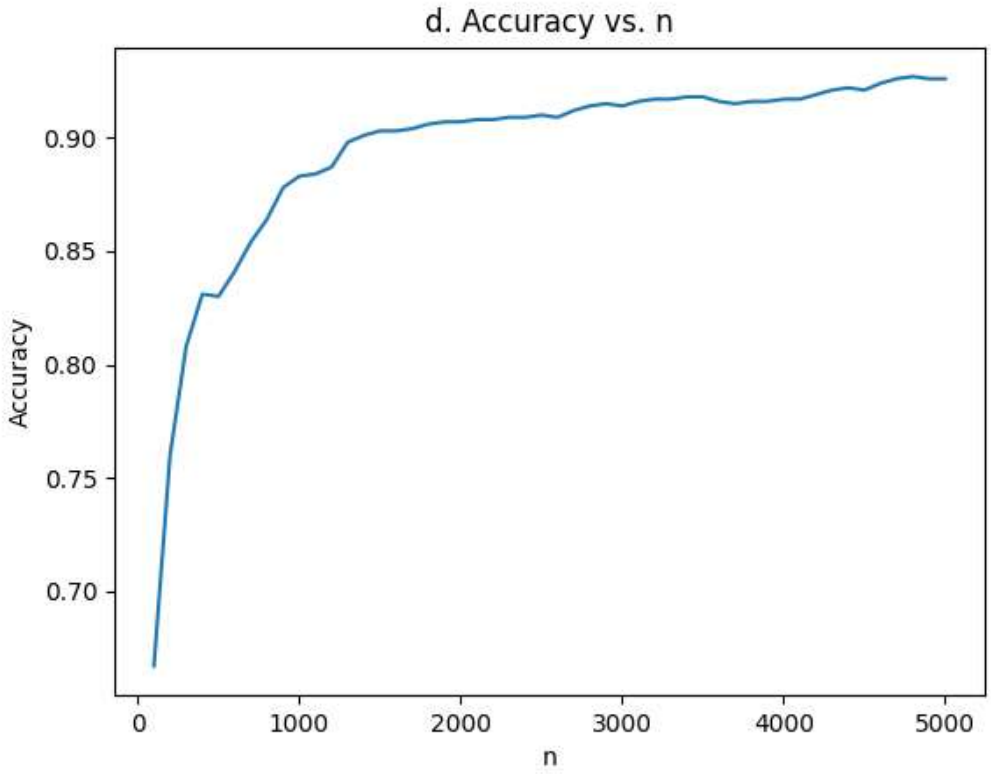
Introduction to Machine Learning (0368-3235) - Assignment 1
Lior Erenreich



From viewing the graph, we see that the accuracy decreases as the number k increases, so the value of k which we got the best accuracy for is $k = 1$. I assume that the explanation for this behavior is that the more samples of images labels we get for a predicting a specific image labeling, we get more images which are long distanced from the current image, this way we get higher chances to label it wrong.

d. Plot:

Introduction to Machine Learning (0368-3235) - Assignment 1
Lior Erenreich



The results I got shows that the accuracy increases as the number of images we train the model on, increases. This behavior makes sense since as we train the model on more images, the chances to encounter a similar image to the current image, increase.