Introduction to Machine Learning-Exercise 1

Linear Algebra

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

   (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.

Proof: Fix a symmetric matrix A. The sequence of the proof will be like that:

A is a PSD → all its eigenvalues are non-negative → A can be written as $A = XX^T$ → A is a PSD.

Which gives us the wanted claim.

A is a PSD → all its eigenvalues are non-negative:

Lets assume that A is PSD. Let $\{v_1, v_2, \dots, v_n\}$ be the eigenvectors of A and $\{\alpha_1, \alpha_2, \dots, \alpha_n\}$ be its eigenvalues, by definition we know that:

$$\forall i: Av_i = \alpha_i v_i$$

And since A is a PSD so we conclude that:

$\forall i: v_i^T Av_i \geq 0 \rightarrow \forall i: v_i^T \alpha_i v_i \geq 0 \rightarrow \forall i: \alpha_i v_i^T v_i \geq 0$

And since $v_i^T v_i \geq 0$ for every vector so we conclude that $\forall i: \alpha_i \geq 0$ as wanted.

A’s eigenvalues are all non-negative → A can be written as $A = XX^T$:

Lets assume that all of A’s eigenvalues are non-negative. A can be decomposed as $A = QDQ^T$ as $D = diag(\alpha_1, \alpha_2, \dots, \alpha_n)$. Lets pay attention to the fact that if we define a matrix called D’ as $D' = diag(\sqrt{\alpha_1}, \sqrt{\alpha_2}, \dots, \sqrt{\alpha_n})$ then $D = D'D' \overset{since\ D'\ is\ symmetric}{=} D'D'^T$. And then we can write:

$A = \ QDQ^T = Q\,D'D'^T Q^T = \ QD'(QD')^T$ as wanted, cause we can take X=QD’.

A can be written as $A = XX^T$ → A is a PSD:

Lets assume A can be written as $A = XX^T$. Then for every vector v:

$$\forall v: v^T Av = v^T XX^T v = (X^T v)^T X^T v$$

And lets pay attention to that $(X^T v)^T X^T v$ is equal to the Euclidean norm of the matrix $X^T v$ powered by 2, and since the Euclidean norm is non-negative then its power by 2 is also non-negative.

So, we conclude that:

$$\forall v: v^T Av = (X^T v)^T X^T v \geq 0 \rightarrow A\ is\ PSD\ as\ wanted \qquad \blacksquare$$

   (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 $\mathbb{R}$?

Proof:

Let $\alpha, \beta \geq 0$ and A,B PSD matrices. Then for every vector v:

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

A is PSD so $v^T A v \geq 0$, and $\alpha \geq 0$ hence $\alpha v^T A v \geq 0$.

B is PSD so $v^T B v \geq 0$, and $\beta \geq 0$ hence $\beta v^T B v \geq 0$.

And thus: $v^T(\alpha A + \beta B)v = \ \alpha v^T A v + \beta v^T B v \geq 0$, so the matrix $(\alpha A + \beta B)$ is PSD.

But this doesn't mean that the the set of all n x n PSD matrices over R is a vector space over R because the claim relies on that $\alpha\ and\ \beta$ are non-negative, meaning if we take a negative scalar then multiplying it by a PSD matrix A will not give us a PSD matrix, hence it is not closed under multiplication by a negative scalar.

Calculus and Probability

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.

Answer:

We know that the PDF of Y is the derivative of it's CDF, so we calculate the CDF first:

first, lets notice that for y between 0 and 1: $F_Y(y) = P(Y \leq y) = P(X_1 \leq Y)\ and\ P(X_2 \leq Y)\ and\ \dots\ P(X_n \leq Y) = \prod_{i=1}^{n} y = y^n$

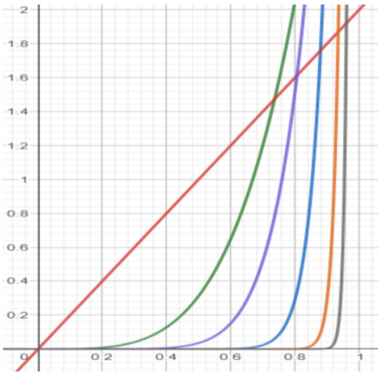
so, we conclude that:

$$F_Y(y) = \begin{cases} 0\ if\ y = 0 \\ y^n\ if\ 0 \leq y \leq 1 \\ 1\ if\ y > 1 \end{cases}$$

Thus:

$$f_Y(y) = \begin{cases} 0\ if\ y = 0 \\ n * y^{n-1}\ if\ 0 \leq y \leq 1 \\ 0\ if\ y > 1 \end{cases}$$

The plot of the CDF will look something like this:



Now, let’s compute E[Y]:

$$E[Y] = \int_{-\infty}^{\infty} y * f_Y(y)\, dy = \int_0^1 y * f_Y(y)\, dy = \int_0^1 y * n * y^{n-1}\, dy = \int_0^1 n * y^n\, dy = \frac{n * y^{n+1}}{n+1} /_0^1$$
$$= \frac{n}{n+1}$$

Notice that E[Y] as the n grows large it gets much closer to 1: $\lim_{n\to\infty} E[Y] = 1$

Now, let’s compute Var[Y]:

$$Var[Y] = E[Y^2] - (E[Y])^2 = \int_0^1 y^2 * n * y^{n-1} - (\frac{n}{n+1})^2 = \int_0^1 n * y^{n+1} - (\frac{n}{n+1})^2$$
$$= \frac{n * y^{n+2}}{n+2} /_0^1 - (\frac{n}{n+1})^2 = \frac{n}{n+2} - (\frac{n}{n+1})^2 = \cdots = \frac{n}{(n+2)(n+1)^2}$$

And notice that Var[Y] as the n grows it gets much closer to 0: $\lim_{n\to\infty} Var[Y] = 0$

Optimal Classifiers and Decision Rules

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

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

(In class you proved this for binary classification, i.e. $\mathcal{Y} = \{0, 1\}$, here you are asked to generalize the result to L labels.)

Answer:

Denote $g = \arg min_{f:X\to Y} E\left[l_{0-1}\left(Y, f(x)\right)\right], h(x) = \arg max_{i\in Y} P[Y = i|X = x]$. We need to prove g=h.

It holds that:

$$E\left[l_{0-1}\left(Y, f(x)\right)\right] = \sum_{x\in X} \sum_{i=1}^{L} P[X = x, Y = i] * l\left(i, f(x)\right)$$
$$= \sum_{x\in X} \left(P[X = x] \sum_{i=1}^{L} P(Y = i|X = x) * l\left(i, f(x)\right) \right)$$

That is because when Y=f(x) then l(I,f(x)=0 and thus its Pr it’s not calculated in the sum operator.

$$\Big\downarrow$$
$$= \sum_{x\in X} \left(P[X = x] * \left[1 - P[Y = f(x)|X = x]\right] \right)$$
$$\geq_{by\ the\ choice\ of\ h} \sum_{x\in X} \left(P[X = x] * \left[1 - P[Y = h(x)|X = x]\right] \right) = E\left[l_{0-1}\left(Y, h(x)\right)\right]$$

And thus, we conclude that h satisfies that $E\left[l_{0-1}\left(Y, h(x)\right)\right]$ is minimal the same as g, so g=h as wanted.

Notice that intuitively the second line in the equation(which is above the black arrow) is minimized when the maximum $P(Y = i|X = x)$ is not calculated in the sum operator, which happens when the loss function outputs 0 on these probabilities, which means when we choose f to be h(h by definition gives the maximum among those probabilities).

(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:

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

Answer:

For any $f: X \to Y$ it holds that:

$$E\big[\Delta\big(Y, f(x)\big)\big] = \sum_{x \in X} \sum_{y \in Y = \{0,1\}} P[X = x, Y = y] * \Delta\big(y, f(x)\big)$$

$$= \sum_{x \in X} \left(P[X = x] \sum_{y \in Y = \{0,1\}} P[Y = y|X = x] * \Delta\big(y, f(x)\big) \right)$$

$$= P[X = x] * \big[P[Y = 0|X = x] * \Delta\big(0, f(x)\big) + P[Y = 1|X = x] * \Delta\big(1, f(x)\big)\big]$$

We can see from the equation above, which we want to minimize its result, that if we predict that f(x)=0 then we "pay" $P[X = x] * P[Y = 1|X = x] * \Delta\big(1, f(x)\big) = P[X = x] * P[Y = 1|X = x] * b$, as $\Delta\big(0, f(x)\big) = 0$, and if we predict that f(x)=1 then we "pay" $P[X = x] * P[Y = 0|X = x] * \Delta\big(0, f(x)\big) = P[X = x] * P[Y = 0|X = x] * a$.

Thus, we conclude that the optimal decision rule h which satisfies the expected loss is minimal will predict that h(x)=0 when $b * P[Y = 1|X = x] < a * P[Y = 0|X = x]$, else it should predict that h(x)=1, more formally:

$$h(x) = \begin{cases} 0 \; if \; b * P[Y = 1|X = x] < a * P[Y = 0|X = x] \\ 1 \; else: b * P[Y = 1|X = x] \geq a * P[Y = 0|X = x] \end{cases} = \begin{cases} 0 \; if \; \dfrac{P[Y = 1|X = x]}{P[Y = 0|X = x]} < \dfrac{a}{b} \\ 1 \qquad\qquad otherwise \end{cases}$$

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 $\mathbb{E}[\Delta_{log}(Y, h(X))]$ (the final answer should be very intuitive).

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

Answer:

Note: we were told in the recitation, as I remember, to consider log as ln, so I solved the question with this consideration.

For any $f: X \to Y$ it holds that:

$$E\left[\Delta\left(Y, f(x)\right)\right] = \sum_{x \in X} \sum_{y \in Y = \{0,1\}} P[X = x, Y = y] * \Delta_{log}\left(y, f(x)\right)$$

$$= \sum_{x \in X} \left(P[X = x] \sum_{y \in Y = \{0,1\}} P[Y = y | X = x] * \Delta_{log}\left(y, f(x)\right) \right)$$

This transition is simply by compensating in Δ_{log} y=0, and then y=1…

$$= P[X = x] * \left[P[Y = 0 | X = x] * \Delta_{log}\left(0, f(x)\right) + P[Y = 1 | X = x] * \Delta_{log}\left(1, f(x)\right)\right]$$

$$= P[X = x] * \left[P[Y = 0 | X = x] * -\log\left(1 - f(x)\right) + P[Y = 1 | X = x] * -\log\left(f(x)\right)\right]$$

Denote $g(z) = -P[Y = 0 | X = x] * \log(1 - z) - P[Y = 1 | X = x] * \log(z)$ as z=f(x) which is our prediction for some x, and we want to find the minimum of this function g, meaning which z=f(x) should we predict so it gives us the minimum value.

notice that our variable is z=f(x) and that $P[Y = 0 | X = x], P[Y = 1 | X = x]$ are parameters.

To calculate the minimum, we first find the derivative of g:

$$g'(z) = -P[Y = 0 | X = x] * \frac{-1}{1 - z} - P[Y = 1 | X = x] * \frac{1}{z} = \frac{P[Y = 0 | X = x]}{1 - z} - \frac{P[Y = 1 | X = x]}{z}$$

$$= \frac{z * P[Y = 0 | X = x] - P[Y = 1 | X = x] + z * P[Y = 1 | X = x]}{(1 - z)z}$$

$$= \frac{z(P[Y = 0 | X = x] + P[Y = 1 | X = x]) - P[Y = 1 | X = x]}{z(1 - z)}$$

P[Y=0|X=x] +P[Y=1|X=x] =1

$$= \frac{z - P[Y = 1 | X = x]}{z(1 - z)}$$

And thus, we get the minimum when:

$$z - P[Y = 1 | X = x] = 0 \to z = P[Y = 1 | X = x]$$

Thus, we conclude that the wanted predictor h is given by:

$$h(x) = P[Y = 1 | X = x]$$

intuitively, when we predict h(x) to be $P[Y = 1 | X = x]$ then if this probability is “high“(closer to 1) then our prediction is indeed closer to 1 which is more likely the real value of x and if this probability is “small” then our prediction is indeed closer to 0 which is more likely the real value of of x, and thus we minimize the error.

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).

Answer:

In class we saw that the decision rule h that minimizes the expected 0-1 loss in the binary case is given by:

$$h(x) = \begin{cases} 1 \; if \; P(Y=1|X=x) > P(Y=0|X=x) \\ 0 \qquad\qquad\qquad else \end{cases}$$

now we want to find the values of x that hold:

$$P(Y=1|X=x) > P(Y=0|X=x)$$

Using Bayes theorem for continuous variables we get that the upper statement is equivalent to:

$$\frac{f_x(x|Y=1) * P(Y=1)}{f_x(x)} > \frac{f_x(x|Y=0) * P(Y=0)}{f_x(x)}$$

$$\frac{f_x(x|Y=1) * p_1}{f_x(x)} > \frac{f_x(x|Y=0) * (1-p_1)}{f_x(x)}$$

The equation above holds when:

$$f_x(x|Y=1) * p_1 > f_x(x|Y=0) * (1-p_1)$$

$$\frac{p_1}{\sqrt{2\pi{\sigma_1}^2}} * e^{-\frac{(x-\mu)^2}{2{\sigma_1}^2}} > \frac{(1-p_1)}{\sqrt{2\pi{\sigma_2}^2}} * e^{-\frac{(x-\mu)^2}{2{\sigma_2}^2}}$$

$$e^{-\frac{(x-\mu)^2}{2{\sigma_1}^2}+\frac{(x-\mu)^2}{2{\sigma_2}^2}} > \frac{(1-p_1) * \sqrt{2\pi{\sigma_1}^2}}{p_1 * \sqrt{2\pi{\sigma_2}^2}}$$

$$-\frac{(x-\mu)^2}{2{\sigma_1}^2} + \frac{(x-\mu)^2}{2{\sigma_2}^2} > \ln\left(\frac{(1-p_1) * \sigma_1}{p_1 * \sigma_2}\right)$$

$$(x-\mu)^2({\sigma_1}^2 - {\sigma_2}^2) > 2{\sigma_1}^2{\sigma_2}^2 * \ln\left(\frac{(1-p_1) * \sigma_1}{p_1 * \sigma_2}\right)$$

$$(x-\mu)^2 > \frac{2{\sigma_1}^2{\sigma_2}^2 * \ln\left(\frac{(1-p_1) * \sigma_1}{p_1 * \sigma_2}\right)}{({\sigma_1}^2 - {\sigma_2}^2)}$$

$$x-\mu > \sqrt{\frac{2{\sigma_1}^2{\sigma_2}^2 * \ln\left(\frac{(1-p_1)*\sigma_1}{p_1*\sigma_2}\right)}{({\sigma_1}^2-{\sigma_2}^2)}} \ \boldsymbol{or}\ x-\mu < -\sqrt{\frac{2{\sigma_1}^2{\sigma_2}^2 * \ln\left(\frac{(1-p_1)*\sigma_1}{p_1*\sigma_2}\right)}{({\sigma_1}^2-{\sigma_2}^2)}}$$

$$x > \mu + \sqrt{\frac{2{\sigma_1}^2{\sigma_2}^2 * \ln\left(\frac{(1-p_1)*\sigma_1}{p_1*\sigma_2}\right)}{({\sigma_1}^2-{\sigma_2}^2)}} \ \boldsymbol{or}\ x < \mu - \sqrt{\frac{2{\sigma_1}^2{\sigma_2}^2 * \ln\left(\frac{(1-p_1)*\sigma_1}{p_1*\sigma_2}\right)}{({\sigma_1}^2-{\sigma_2}^2)}}$$

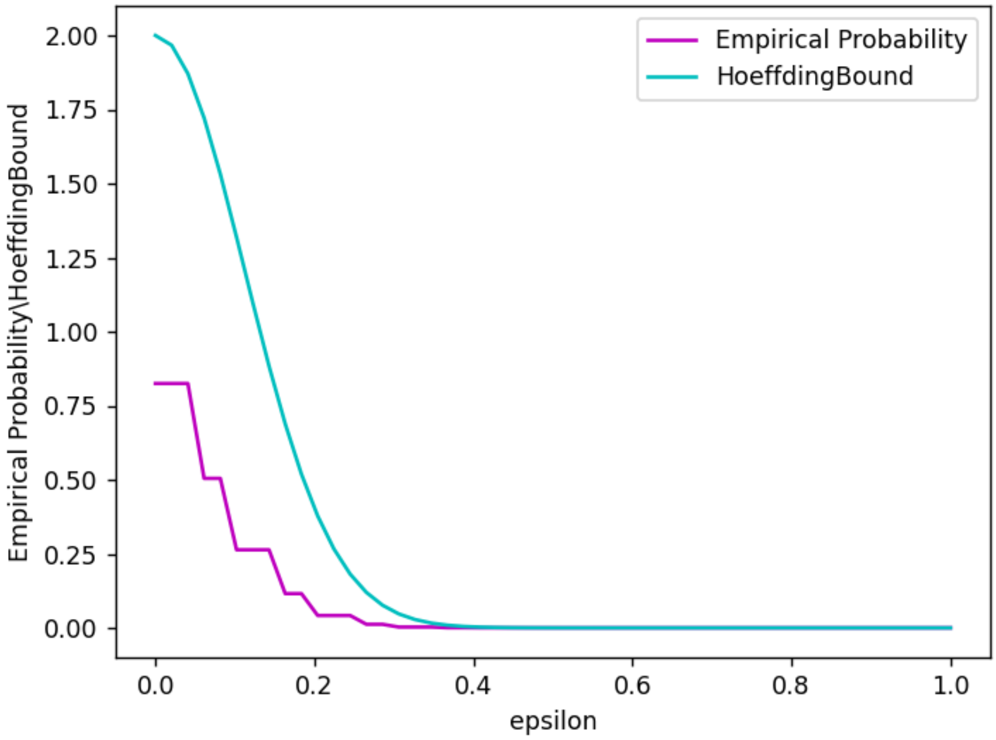
In these cases we predict that h(x)=1, otherwise [meaning that x is between these values] we predict that h(x)=0.

So, to sum up, our optimal decision rule h is:

$$h(x) = \begin{cases} 0\ if\ \mu - \sqrt{\frac{2{\sigma_1}^2{\sigma_2}^2 * \ln\left(\frac{(1-p_1)*\sigma_1}{p_1*\sigma_2}\right)}{({\sigma_1}^2-{\sigma_2}^2)}} < x < \mu + \sqrt{\frac{2{\sigma_1}^2{\sigma_2}^2 * \ln\left(\frac{(1-p_1)*\sigma_1}{p_1*\sigma_2}\right)}{({\sigma_1}^2-{\sigma_2}^2)}} \\ 1 \qquad\qquad\qquad\qquad otherwise \end{cases}$$

Programming Assignment

Question 1:



Question 2:

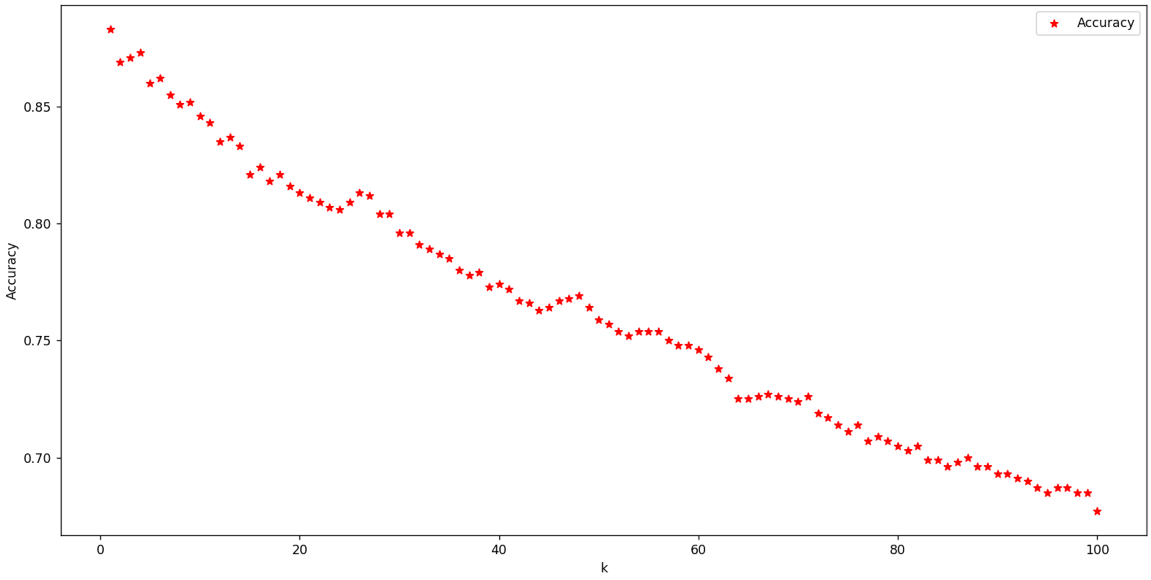
2b:

The accuracy of the prediction is: 0.846(84.6%).

In a completely random predictor, I would have expected to get an accuracy of 0.1(10%), since there are ten possibilities for the label: {0,1, 2, … ,9}, assuming that in the test set contains equal images of each label.

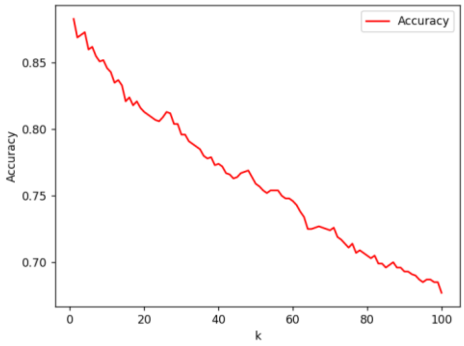
Note: if the test set is not uniformly distributed as said above, then the accuracy the prediction would change depending on the distribution.

2c:



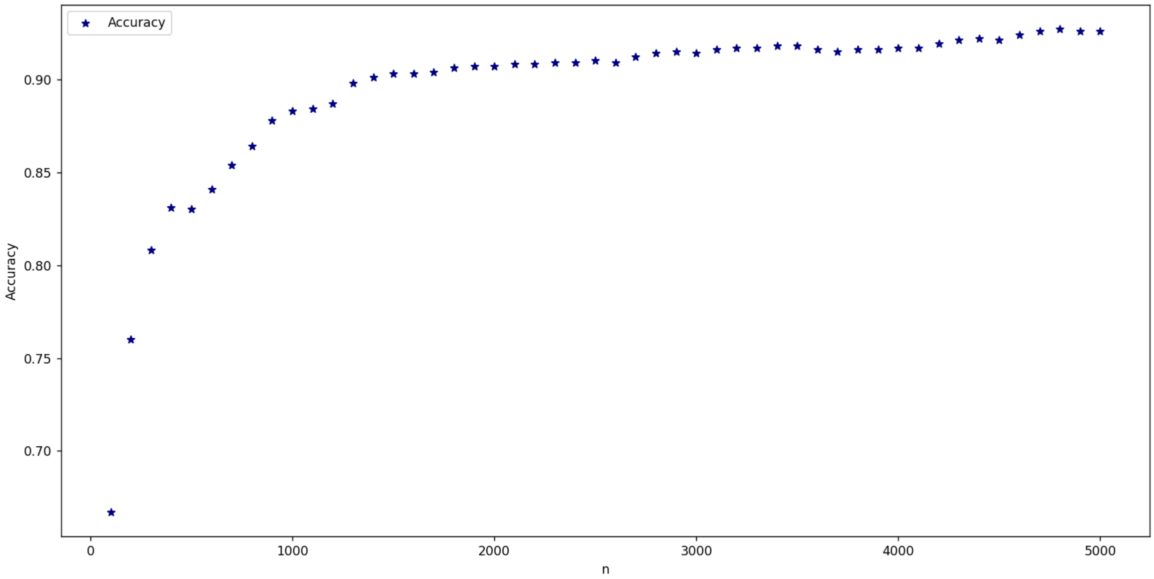
We can see from the plot above that k=1 is the best k, because it gives the highest accuracy.

Also, we can see that when k=2 the accuracy is worse than k=1 but then it rises up again when k=3 and then rises up again when k=4(still worse than k=1), meaning that k=4 is the second best k. this thing, meaning that the accuracy goes down and then rises up again, happens also for higher k values but I think that it happens less, meaning that after k=4 in general we can say that there is decline in the accuracy. Maybe the explanation of this might be that when k is big enough then in the k nearest images there are more images with different labels of the query image, and such there is more mistakes and less accuracy.



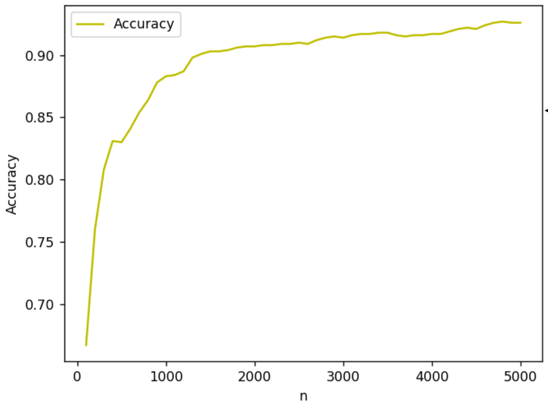
This is the same graph but not scattered, I think the one above is better cause we can see more obvious the points.

2d:



As we can see from the plot above, when k=1(which is the best k according to the previous question) as n increases the accuracy of our prediction gets better. And that makes sense cause intuitively when we train the machine on a large amount of images, training set, mistakes are made less meaning that accuracy gets better.

Note: the difference in the accuracies is more noticeable among the smallest values of n, meaning that when n is small as long as we increase it the accuracy gets better significantly, but after certain value of n, say n~2000, the accuracy gets better but not that much. That makes sense also because it must be a kind of a maximum accuracy which we can get to, mistakes will be made. I also think about this positively, cause this means that after a certain number of training data the usefulness of more and more data is negligible, and thus no need to get more training data…



This is the same graph but not scattered, I think the one above is better cause we can see more obvious the points.

עודי אגבאריה