Question 3.

Question 3. Let $x \in \mathbb{R}$, $\mathcal{Y} \in \{0, 1\}$. Given a text string c which describes a mathematical condition, define $h_c : x \to \mathcal{Y}$ as follows: $h_c(x) := \mathbb{I}[x \text{ satisfies } c]$. For instance, if c is the condition "x is a natural number", then $h_c(x) = \mathbb{I}[x \in \mathbb{N}]$. For any $n \in \mathbb{N}$, define the hypothesis class $\mathcal{H}_n \subseteq \mathcal{Y}^x$ as follows:

$$\mathcal{H}_n := \{h_c \mid c \text{ is a condition which can be described using at most } n \text{ characters}\}$$

For instance, the condition c given above is in $\mathcal{H}_n$ for all $n \geq 21$. The text of the conditions can include only ASCII characters (there are 128 ASCII characters).

Suppose that the examples x are temperature values, and the labels y indicate whether a doctor gives medicine to a person with this temperature. Suppose that we know that the rule that the doctors use to decide whether to give the medicine has at most 10 characters and that the doctors follow this rule exactly. We get a random independent sample from the distribution D over $x \times \mathcal{Y}$ over temperatures and doctor decisions. We would like to run an ERM algorithm with the hypothesis class $\mathcal{H}_{10}$ to find a condition with an error of at most 0.1. We have unlimited computational resources.

(a) Explain why running the ERM algorithm with the hypothesis class $\mathcal{H}_n$ for $n < 10$ or for $n > 10$ may be a worse idea than running it with $\mathcal{H}_{10}$.

(b) Use PAC bounds to calculate an upper bound on the size of a sample size that would be sufficient to find, with a probability at least 0.99, a condition with an error at most 0.1 on the distribution, using an ERM algorithm $\mathcal{H}_{10}$. Give a formal description of the guarantee that you are providing.

(c) Suppose we run ERM with $\mathcal{H}_n$ on the training sample for some $n < 10$. Calculate an upper bound on the size of a sample size that would be sufficient to find, with a probability at least 0.99, a condition with an excess error at most 0.1 on the distribution, using an ERM algorithm with $\mathcal{H}_n$. Give a formal description of the guarantee that you are providing.

Solution for question 3:

(a) The rule that the doctors use is in $\mathcal{H}_{10}$, but not necessarily in $\mathcal{H}_n$ for $n < 10$. If indeed the rule is not in $\mathcal{H}_n$ for $n < 10$, it is not possible to guarantee an error of at most 0.1, since we would have to consider the error of the best hypothesis in $\mathcal{H}_n$. The doctors rule is in $\mathcal{H}_n$ for $n > 10$, but using such a class would require more samples to avoid overfitting.

(b) Since the doctors' rule is in $\mathcal{H}_{10}$, this is the realizable case, and we can use the PAC theorem for this case:

$$m \geq \frac{\log(|\mathcal{H}_{10}|) + \log(1/\delta)}{\epsilon}.$$

Substituting $\delta = 0.01, \epsilon = 0.1$, $\log(|\mathcal{H}_{10}|) = 10\log(128)$, we get that if $m = 532$, then $Pr[\text{err}(\hat{h}, \mathcal{D}) \leq 0.1] \geq 0.99$, when $\hat{h}$ is the hypothesis that was outputted by the ERM algorithm.

(c) Since we do not know if the doctors' rule is in $\mathcal{H}_n$ for $n < 10$, we are in the agnostic case, so we need no more than

$$\frac{2\log(|\mathcal{H}_n|) + 2\log(2/\delta)}{\epsilon^2} \leq \frac{2\log(|\mathcal{H}_9|) + 2\log(2/\delta)}{\epsilon^2}.$$

Substituting $\delta = 0.01, \epsilon = 0.1, \log(|\mathcal{H}_9|) = 9\log(128)$, we get that with no more than 9,794 we can gurantee that $Pr[\text{err}(\hat{h}, \mathcal{D}) - \inf_{h \in \mathcal{H}_n} \text{err}(h, \mathcal{D}) \leq 0.1] \geq 0.99$.

Question 4:

Consider the hypothesis class of homogeneous linear predictors $\mathcal{H}_L^d := \{h_w \mid w \in \mathbb{R}^d\}$ where $h_w(x) = \text{sign}(\langle w, x\rangle)$, $\forall x \in \mathbb{R}^d$.
Prove that any linearly independent set of d vectors, $u_1, \ldots, u_d \in \mathbb{R}^d$, are shattered by $\mathcal{H}_L^d$.

Reminder: A set of input vectors is said to be shattered by an hypothesis class $\mathcal{H}$ if it can get all the possible labelings by predictors from $\mathcal{H}$.

Solution:

A set of $u_1, \ldots, u_d \in \mathbb{R}^d$ linearly independent vectors is given. Note that while the entire set of d is linearly independent, pairs of vectors from this set are not necessarily linearly independent.

Define the $d \times d$ matrix $U = [u_1, \ldots, u_d]$.

The matrix U has a linear independent set of columns and therefore has an inverse $U^{-1} \in \mathbb{R}^{d\times d}$. Hence,

$$U^{-1}U = I_d$$

which implies that, for $i \in \{1, \ldots . d\}$,

$$U^{-1}u_i = \mathbf{e}_i$$

where $\mathbf{e}_i$ is the ith standard basis vector in $\mathbb{R}^d$.

Let us denote the kth row of U^{-1} as r_k^T (the transpose denotes it is a row vector), then $U^{-1}u_i = \mathbf{e}_i$ implies that

$$\langle r_k, u_i\rangle = r_k^T u_i = \begin{cases} 1 & \text{if } k = i \\ 0 & \text{if } k \neq i \end{cases}$$

For labels $y_1, \ldots, y_d \in \{-1, +1\}$, choose a linear predictor that corresponds to

$$w = \sum_{k=1}^{d} y_k\, r_k$$

Then, for u_i for any $i \in \{1, \ldots, d\}$

$$h_w(u_i) = \text{sign}(\langle u_i, w\rangle) = \text{sign}\left(\langle u_i, \sum_{k=1}^{d} y_k\, r_k\rangle\right) = \text{sign}\left(\sum_{k=1}^{d} y_k\, \langle u_i, r_k\rangle\right) = \text{sign}(y_i) = y_i$$

Namely, we proved that $h_w(u_i) = y_i$.

To conclude, we showed that $\mathcal{H}_L^d$ has a predictor for any labeling of any given set of linearly independent d vectors in $\mathbb{R}^d$.

Hence, any given set of linearly independent d vectors in $\mathbb{R}^d$ is shattered by $\mathcal{H}_L^d$.

Question 6

Question 6. In this exercise, we will see that without margin assumptions, the Perceptron algorithm might run for a long time, exponential in the dimension d. We define a special sample $S = ((x_1, y_1), \ldots, (x_m, y_m))$, where $m = d$, $x_i \in \mathbb{R}^d$, and $y_i \in \{-1, 1\}$, where $y_i := (-1)^{i+1}$, and $x_i \in \mathbb{R}^d$ is defined by:

$$x_i(j) = \begin{cases} (-1)^i, & j < i \\ (-1)^{i+1}, & j = i \\ 0, & j > i. \end{cases}$$

In the following steps, you will prove that for the sample above, the Perceptron performs a number of updates at least exponential in d:

(a) Prove that in any iteration t of the Perceptron on the given sample S, and for any $i \leq d$, $|w^{(t+1)}(i)| \leq t$.
Hint: use induction on the Perceptron update $w^{(t+1)} \leftarrow w^{(t)} + y_i x_i$.

(b) Let $w^{(T)}$ be the separator that the Perceptron outputs. Prove that for every coordinate i, $|w^{(T)}(i)| \geq 2^{i-1}$. Hint: prove this by induction on i, using the fact that the separator defined by $w^{(T)}$ labels correctly all the examples in the given sample S.

(c) Conclude from the two previous items that the number of updates of the Perceptron until it stops and outputs $w^{(T)}$ is exponential in d.

To help with intuition, you may want to implement the Perceptron algorithm defined in class (no need to submit the code), and check its behavior on the sample S for $d = 1, 2, \ldots, 10$.

Solution for question 6:

(a) For $t = 1$, we have that $w^{(2)} \leftarrow w^{(1)} + y_i x_i = y_i x_i$, since $w^{(1)}$ is the zero vector. Then, $w^{(2)}(j) = y_i x_i(j) \in \{-1, 0, 1\}$. Therefore, $|w^{(2)}(j)| \leq 1 = t$.
Asumme that for every t, $|w^{(t+1)}(j)| \leq t$. Then,
$|w^{(t+2)}(j) = |w^{(t+1)}(j) + y_{i'} x_{i'}(j)| \leq |w^{(t+1)}(j)| + |y_{i'} x_{i'}(j)| \leq t + 1$.

(b) Since the algorithm stops when all exmaples are classified correctly, we have that for every i, $y_i \langle w, x_i \rangle > 0$. Also, for every $i \leq m, j \leq d$, we have that $y_i x_i(j) \in \{-1, 0, 1\}$. Then, since $w(j) = \sum y_i x_i(j)$ for every i that participated in w's update, $w(j)$ is a natural number. For $i = 1$, $y_1 \langle w, x_1 \rangle = y_1 w(1) x_1(1) = w(1) > 0$. Hence, $w(1) \geq 1 = 2^{i-1}$.
Assume for every $1 \leq k < i$, $w(k) \geq 2^{k-1}$.
Then,

$$\begin{aligned} y_i \langle w, x_i \rangle &= y_i \Big[\sum_{k<i} w(k) x_i(k) + w(i) x_i(i) \Big] \\ &= (-1)^{i+1} \Big[\sum_{k<i} w(k)(-1)^i + w(i)(-1)^{i+1} \Big] \\ &= -\sum_{k<i} w(k) + w(i) > 0. \end{aligned}$$

Then, we get that

$$w(i) > \sum_{k<i} w(k) \geq \frac{1}{2} \sum_{k<i} 2^k = 2^{i-1} - 1.$$

Since every coordinate of w is a natural number, $w(i) > 2^{i-1} - 1 \Rightarrow w(i) \geq 2^{i-1}$.

(c) From (a) we have that $|w^{(T)(d)}| \leq T - 1$. From (b) we have that $|w^{(T)}(d)| \geq 2^{d-1}$. Thus, $T \geq 2^{d-1} + 1 = O(2^d)$.

Question 5. For a given training sample $S = ((x_1, y_1), \ldots, (x_m, y_m))$, consider the following **modified version** of the soft-SVM optimization problem:

$$\lambda \|\mathbf{w}\|_1^2 + \sum_{i=1}^{m} \ell_h(w, (x_i, y_i)),$$

where $\ell_h(w, (x_i, y_i)) = \max\{0, 1 - y_i \langle w, x_i \rangle\}$ is the *hinge loss* defined in the lecture. Recall that $\|\mathbf{w}\|_1$ is the ℓ_1 norm defined as $\|\mathbf{w}\|_1 := \sum_{i=1}^{d} |w(i)|$.

Express this optimization problem as a quadratic program in standard form, as we showed in class.

(a) Write a quadratic minimization problem with constraints that is equivalent to the problem above, using auxiliary variables similar to the ξ_i in the soft-SVM implementation.

(b) Write what H,u,A,v in the definition of a Quadratic Program should be set to so as to solve the minimization problem you wrote.

a) For every $i \leq d$ we will define auxilary varibal ϕ_i s.t:

i. $w(i) \leq \phi_i \Rightarrow 0 \leq \phi_i - w(i)$

ii. $-w(i) \leq \phi_i \Rightarrow 0 \leq \phi_i + w(i)$

We define ϕ s.t

$$\phi_1 + \ldots + \phi_d = \sum_{i=1}^{d} \phi_i \leq \phi \Rightarrow 0 \leq \phi - \sum_{i=1}^{d} \phi_i$$

For every $j \leq m$ we will define the Auxilary Varibal ξ_j s.t:

i. $\xi_j \geq 0$

ii. $\xi_j \geq 1 - y_j \langle w, x_j \rangle \Rightarrow 1 \leq \xi_j + \langle w, x_j \rangle y_j$

and the objective will be:

$$\lambda \phi^2 + \sum_{i=1}^{m} \xi_i$$

b) Lets start with the Z vector

Let $z = (\phi, \phi_1, \dots, \phi_d, \xi_1, \dots, \xi_m, w_1, \dots, w_d)$

Let $H \in \mathbb{R}^{(2d+m+1) \times (2d+m+1)}$ be

$$H = \begin{pmatrix} 2\lambda & 0 & \cdots & 0 \\ 0 & \ddots & & \vdots \\ \vdots & & 0 & \\ 0 & \cdots & & 0 \end{pmatrix}$$

Let $u \in \mathbb{R}^{2d+m+1}$ be

$$u = (\underbrace{0}_{1}, \underbrace{0, \dots, 0}_{d}, \underbrace{1, \dots, 1}_{m}, \underbrace{0, \dots, 0}_{d})$$

and let $A \in \mathbb{R}^{(2m+2d+1)\times(2d+m+1)}$

with respect to

$$0 \leq \phi_i - w(i) \qquad \xi_j \geq 0 \qquad 0 \leq \phi - \sum_{i=1}^{d} \phi_i$$

$$0 \leq \phi_i + w(i) \qquad 1 \leq \xi_j + \langle w, x_j \rangle y_j$$

$$z = (\phi, \phi_1, \dots, \phi_d, \xi_1, \dots, \xi_m, w_1, \dots, w_d)$$

We will define $B \in \mathbb{R}^{m \times d}$

$$B = \begin{pmatrix} y_1 x_1(1) & \cdots & y_1 x_1(d) \\ \vdots & & \vdots \\ y_m x_m(1) & \cdots & y_m x_m(d) \end{pmatrix}$$

$$
A = \begin{array}{c}
\begin{array}{ccc} \overbrace{\qquad\qquad}^{d} & \overbrace{\qquad\qquad}^{m} & \overbrace{\qquad\qquad}^{d} \end{array} \\
\begin{pmatrix}
1 & -1 \;\cdots\; -1 & 0 \;\cdots\; 0 & 0 \;\cdots\; 0 \\
0 & & & \\
\vdots & I_d & O_{d\times m} & -I_d \\
0 & & & \\
0 & & & \\
\vdots & I_d & O_{d\times m} & I_d \\
0 & & & \\
0 & & & \\
\vdots & O_{m\times d} & I_m & O_{m\times d} \\
0 & & & \\
0 & & & \\
\vdots & O_{m\times d} & I_m & B \\
0 & & &
\end{pmatrix}
\begin{array}{l}
\left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} d \\
\left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} d \\
\left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} m \\
\left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} m
\end{array}
\end{array}
$$

$2d+2m+1$

and finally $v \in \mathbb{R}^{2d+2m+1}$

$$
v = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ \vdots \\ 1 \end{pmatrix}
\begin{array}{l}
\left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} 2d+m+1 \\
\left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} m
\end{array}
$$