Indian Institute of Technology Kanpur
CS771 Introduction to Machine Learning

Instructor: Purushottam Kar
Date: September 19, 2017
Total: 100 marks

ASSIGNMENT 2

Problem 2.1 (Oh holy ID3, show me how to grow a tree!). Consider the following toy dataset that you can use to learn how to choose an advisor.

S.No.	Name	Size of research group	Like the research area?	Average workload?	# of meetings per week	Good advisor?
1	Prof. R. Lupin	small	yes	average	$2-3$	no
2	Prof. P. Sprout	large	no	average	$0-1$	yes
3	Prof. R. Hagrid	small	no	average	> 3	no
4	Prof. C. Binns	medium	no	heavy	$0-1$	no
5	Prof. A. Sinistra	large	no	heavy	$0-1$	no
6	Prof. S. Snape	medium	no	heavy	$0-1$	yes
7	Prof. S. Kettleburn	small	no	light	> 3	no
8	Prof. Firenze	small	no	average	$2-3$	no
9	Prof. S. Vector	small	yes	average	$0-1$	yes
10	Prof. R. Hooch	medium	no	light	$0-1$	no
11	Prof. C. Burbage	large	no	average	$0-1$	yes
12	Prof. A. P. W. B. Dumbledore	medium	yes	light	$0-1$	yes
13	Prof. S. Trelawney	small	no	light	$2-3$	no
14	Prof. H. Slughorn	medium	no	heavy	$0-1$	no
15	Prof. Q. Quirrell	large	yes	heavy	$0-1$	no

1. Is the first attribute (name) useful in learning a binary classifier from this data? Why?
2. Is it possible to perfectly classify this data (no matter how complicated the classification algorithm)? Why?
3. Read about the ID3 decision tree learning algorithm. There are many nice resources available for this online. The following website does a nice job with examples `https://www.cise.ufl.edu/~ddd/cap6635/Fall-97/Short-papers/2.htm`. Build a decision tree using the ID3 algorithm on the data given above. You may stop splitting nodes that are at depth 2 or more (the root of the tree is depth 0). For any impure leaves, use the majority label at that leaf to take a decision for data points in that leaf. Break any ties arbitrarily.
4. *Fake Bonus*: Can you guess which research areas do I like?

(2+3+10+0=15 marks)

Solution. We will give the solution in parts below. The final and intermediate trees are depicted in Figure 2

1. The first attribute is not useful in general for two reasons. Firstly it is usually a unique attribute i.e. a certain value, for example "Prof. C Binns" will appear only once, either appear in the training set or the test set but not both. Thus it is no inductive power.

However, more importantly, we do not expect the name of a person to have any influence on whether the person is a good advisor or not so the inductive bias should itself throw such an attribute out even if we could engineer low-level features from the name so that the uniqueness problem can be overcome.

2. No, the dataset is not perfectly classifiable. Look at Prof. Binns and Prof. Snape. Apart from their names, which the earlier discussion shows that we should not use as an attribute, both faculty members have identical profiles, yet one is a good adviser but another one isn't. So no algorithm using this data alone (and excluding the name attribute), can perfectly classify the data. This shows that the set of features here is incomplete. There are missing features which reveal more about who is a good advisor and who isn't.

3. As there are 5 good and 10 bad advisors, the entropy of the data set at the root is around 0.92. All logarithms are in base 2 although the tree will turn out to be the same no matter which base is used.

$$-5/15 \cdot \log(5/15) - 10/15 \cdot \log(10/15) \approx 0.9183$$

Notice a rather high level of entropy. Now let us calculate the information gain for each attribute

   (a) Size of group? If we split according to this, we get three nodes
      i. Small: 1 good 5 bad i.e. entropy is 0.65
      ii. Medium: 2 good 3 bad i.e. entropy is 0.971
      iii. Large: 2 good 2 bad i.e entropy is 1 (highest level of entropy – total confusion)

      . There are 6 small groups, 5 medium groups and 4 large groups. Hence the new entropy of the system after this 3-way split is

$$6/15 \cdot 0.65 + 5/15 \cdot 0.971 + 4/15 \cdot 1 = 0.8503$$

      and the information gain is 0.9183 - 0.8503 = 0.0680. Notice that the average level of entropy across nodes did go down even though there are individual nodes where entropy went up (the "Large" node)!

   (b) Like Area? If we split according to this criteria, we can similarly calculate the information gain to be 0.032. This means that this attribute does not lower the entropy as much as the previous attribute (Size of Research Group) did.
   (c) Workload. Using this attribute gives an information gain of 0.062
   (d) Num Meeting. Using this attribute gives an information gain of 0.251

It is clear that we should split along the number of meetings since that provides the largest information gain. This gives the first tree depicted in Figure 2. Now performing this split results in 2 nodes that are absolutely pure and one node that is absolutely confused. Pure nodes need not be split further so we need only focus on the remaining one node now. Also note that number of meetings will no longer be used to split any nodes since it has been used once (to split the root node).

Of the remaining attributes, it can be seen that in the totally confused child node, Size of Research group offers an information gain of 0.1145, Like Area offers a gain of 0.0349, and Average Workload offers a gain of 0.439. This makes the choice of Average workload clear for splitting this child node. The tree obtained after this split is also depicted in Figure 2.

After this split, we get three children, one of which is pure, one if which is mostly pure (but cannot be further purified since it contains both Prof. Binns and Prof. Snape) and one node that is totally confused but contains only 2 data points. Usually nodes with such low number of data points are made into a leaf since no useful learning can happen with so less data. So there are no more splits and leaves are accorded decisions according to majority rule or else by random toss of a coin.

This gives us the final tree which is also depicted in Figure 2. This tree has entropy 0.3740. If

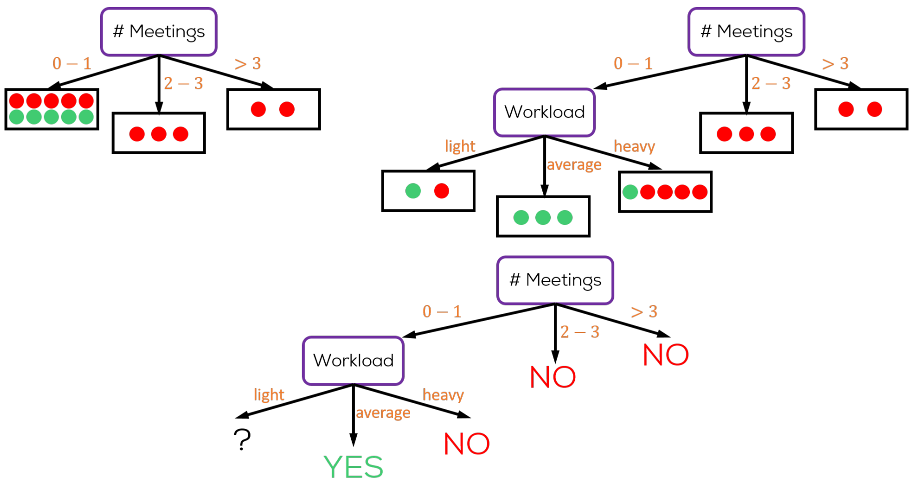

Figure 1: A tree generated by the ID3 algorithm for the given data.

we do wish to split the confused node further, we would find that Research Area completely purifies the node. The entropy of this tree is only 0.2406, (we had entropy 0.9183 at the root).

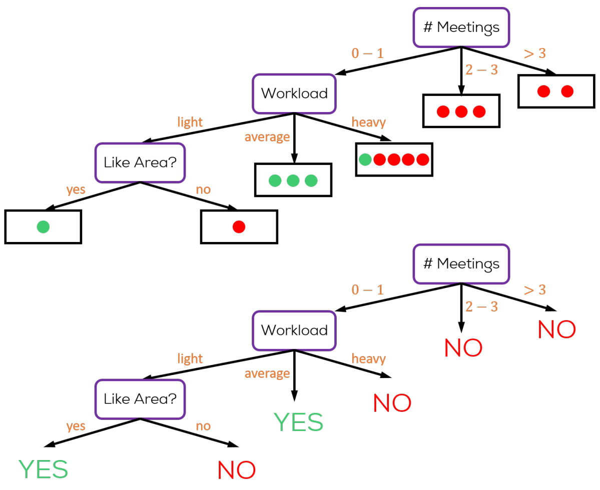

Figure 2: A deeper tree generated by the ID3 algorithm for the given data.

These decision trees give out a very wrong message but unfortunately, they are often a lot closer to what is used in real life by many of us, than we would like to admit. Please do give your interests a chance to inform your decision tree when choosing an advisor/academic program/job/career.

Problem 2.2 (Guess my Grocery List). I run a supermarket called ML-mart that stores L items. The j^{th} item has cost $c_j \in \mathbb{R}$. The i^{th} customer in ML-mart is represented as a d-dimensional real vector $\mathbf{x}^i$. Customer i always buys the same subset of items $S^i \subset [L]$ and pays the bill $b^i = \sum_{j \in S^i} c_j + \epsilon^i$ where $\epsilon^i \sim \mathcal{N}(0, \sigma^2)$ (where σ is the same for all users but not known to you) is some noise in the bill amount. The noise is to account for the fact that maybe the person bought some items, say *mirchi* and *dhaniya* – chillies and cilantro, which are not in the list of L items, in addition to items in the set S^i or else had a coupon for a small discount.

However, something went wrong with my billing machine and it erased all records of which of the L items did the customer buy! The only thing the billing machine stored was the customer ID $\mathbf{x} \in \mathbb{R}^d$ and the bill that customer paid $b \in \mathbb{R}$. I have such incomplete records for n customers $(\mathbf{x}^i, b^i)_{i=1,\dots,n}$. Can you help me complete my bills by figuring out which items the customers bought?

1. Cast the above problem as a latent variable learning problem and develop a sound generative story for the same
2. Write down the complete likelihood expression for the observed data
3. Design a hard-assignment alternating optimization algorithm to solve the problem
4. *Bonus 1*: Design a soft-assignment alternating optimization algorithm for the problem
5. *Bonus 2*: Can you design a hard-assignment alternating optimization algorithm that works in $\mathcal{O}(ndL)$ time?

(5+5+10=20+bonus marks)

Solution. We will give the solution in parts below

1. To develop the generative story, we introduce, for each customer i, a latent variable $\mathbf{z}^i \in \{-1, 1\}^L$ to denote which items were bought by the user: $+1$ indicating purchased, -1 indicating not purchased. Also, let the model be $\mathbf{\Theta} = (W, \sigma)$ where $W = [\mathbf{w}^1, \dots, \mathbf{w}^L] \in \mathbb{R}^{d \times L}$ (since $\mathbf{x}^i \in \mathbb{R}^d$). We can then model the observed data likelihood as

$$\mathbb{P}\left[b^i \,|\, \mathbf{x}^i, \mathbf{\Theta}\right] = \sum_{\mathbf{z} \in \{-1,1\}^L} \mathbb{P}\left[b^i, \mathbf{z} \,|\, \mathbf{x}^i, \mathbf{\Theta}\right]$$

   Note that this is a discriminative expression since it does not model $\mathbb{P}\left[\mathbf{x}^i \,|\, \mathbf{\Theta}\right]$. For any $\mathbf{z} \in \{-1, 1\}^L$, we can model the complete data likelihood as

$$\mathbb{P}\left[b^i, \mathbf{z} \,|\, \mathbf{x}^i, \mathbf{\Theta}\right] = \mathbb{P}\left[b^i \,|\, \mathbf{z}, \mathbf{x}^i, \mathbf{\Theta}\right] \cdot \mathbb{P}\left[\mathbf{z} \,|\, \mathbf{x}^i, \mathbf{\Theta}\right]$$

   To further model these distributions without clutter, let us introduce some notation. Let $\mathbf{c} = [c_1, \dots, c_L] \in \mathbb{R}^L$ denote the vector of item prices. For any $\mathbf{z} \in \{-1, 1\}^L$, let $c_{\mathbf{z}} := \sum_{j : \mathbf{z}_j = 1} c_j$ denote the sum of prices of the items indexed by $\mathbf{z}$ as "purchased". Given this, we can model

$$\mathbb{P}\left[b^i \,|\, \mathbf{z}, \mathbf{x}^i, \mathbf{\Theta}\right] = \mathbb{P}\left[b^i \,|\, \mathbf{z}, \mathbf{\Theta}\right] = \mathcal{N}(b^i; c_{\mathbf{z}}, \sigma)$$

   Note that this is exactly what was given in the problem statement: if we know which items were bought, then the bill amount b^i is simply the sum of the costs of those items plus a bit of Gaussian noise. Once $\mathbf{z}^i$ is known, $\mathbf{x}^i$ stops influencing the bill amount b^i directly. To model $\mathbb{P}\left[\mathbf{z} \,|\, \mathbf{x}^i, \mathbf{\Theta}\right]$ we make use of an assumption that items are bought independently of each other. This was given as a hint in the problem statement itself. We will make use of a logistic model to model user purchase behavior. Recall that $W = [\mathbf{w}^1, \ldots, \mathbf{w}^L] \in \mathbb{R}^{d\times L}$

$$\mathbb{P}\left[\mathbf{z}\,|\,\mathbf{x}^i, \boldsymbol{\Theta}\right] = \prod_{j=1}^{L} \mathbb{P}\left[\mathbf{z}_j\,|\,\mathbf{x}^i, \boldsymbol{\Theta}\right] = \prod_{j=1}^{L} \varsigma(\mathbf{z}_j \left\langle \mathbf{w}^j, \mathbf{x}^i \right\rangle)$$

To avoid confusion with the variance parameter σ, we have denoted the sigmoid function using ς instead. We have $\varsigma(t) = \frac{1}{1+\exp(-t)}$.

2. The likelihood expression, given the above is

$$\mathbb{P}\left[b^i\,|\,\mathbf{x}^i, \boldsymbol{\Theta}\right] = \sum_{\mathbf{z}\in\{-1,1\}^L} \mathcal{N}(b^i; c_{\mathbf{z}}, \sigma) \cdot \prod_{j=1}^{L} \varsigma(\mathbf{z}_j \left\langle \mathbf{w}^j, \mathbf{x}^i \right\rangle)$$

3. A hard assignment algorithm can be constructed given the above calculations. At every time step, using the current model $\boldsymbol{\Theta}^t$, we first make assignments to the latent variables. To do so, first use the Bayes rule to get

$$\mathbb{P}\left[\mathbf{z}\,|\,b^i, \mathbf{x}^i, \boldsymbol{\Theta}^t\right] = \frac{\mathbb{P}\left[b^i\,|\,\mathbf{z}, \mathbf{x}^i, \boldsymbol{\Theta}^t\right] \cdot \mathbb{P}\left[\mathbf{z}\,|\,\mathbf{x}^i, \boldsymbol{\Theta}^t\right]}{\mathbb{P}\left[b^i\,|\,\mathbf{x}^i, \boldsymbol{\Theta}^t\right]}$$

and use an MLE step $\hat{\mathbf{z}}^{i,t} = \arg\max_{\mathbf{z}\in\{-1,1\}^L} \mathbb{P}\left[\mathbf{z}\,|\,b^i, \mathbf{x}^i, \boldsymbol{\Theta}^t\right]$ to make the latent variable assignments. Then, using these latent variable assignments, we update the model as

$$\begin{aligned}
\boldsymbol{\Theta}^{t+1} &= \arg\max_{\boldsymbol{\Theta}} \sum_{i=1}^{n} \log \mathbb{P}\left[b^i, \hat{\mathbf{z}}^{i,t}\,|\,\mathbf{x}^i, \boldsymbol{\Theta}\right] \\
&= \arg\max_{\boldsymbol{\Theta}} \sum_{i=1}^{n} \log \mathcal{N}(b^i; c_{\hat{\mathbf{z}}^{i,t}}, \sigma) + \sum_{i=1}^{n}\sum_{j=1}^{L} \log \varsigma(\hat{\mathbf{z}}^{i,t}_j \left\langle \mathbf{w}^j, \mathbf{x}^i \right\rangle) \\
&= \arg\max_{\sigma} \sum_{i=1}^{n} \log \mathcal{N}(b^i; c_{\hat{\mathbf{z}}^{i,t}}, \sigma) + \arg\max_{W} \sum_{j=1}^{L}\sum_{i=1}^{n} \log \varsigma(\hat{\mathbf{z}}^{i,t}_j \left\langle \mathbf{w}^j, \mathbf{x}^i \right\rangle)
\end{aligned}$$

Note that model updates are made using the total likelihood expression and not the observed data likelihood since it is very difficult to perform optimizations with the observed data likelihood terms. The second part of the last expression can be solved as

$$W^{t+1} = \left[\arg\max_{\mathbf{w}^j} \sum_{i=1}^{n} \log \varsigma(\hat{\mathbf{z}}^{i,t}_j \left\langle \mathbf{w}^j, \mathbf{x}^i \right\rangle)\right]_{j=1,\ldots,L}$$

Thus, the problem turns into L independent logistic regression problems. The first part has a closed form solution. Since $\log \mathcal{N}(b^i; c_{\hat{\mathbf{z}}^{i,t}}, \sigma) = -\log\sigma - \frac{(b^i - c_{\hat{\mathbf{z}}^{i,t}})^2}{2\sigma^2} + C$, where C is a universal constant that does not depend on σ, applying first order optimality gives us

$$\arg\max_{\sigma} \sum_{i=1}^{n} \log \mathcal{N}(b^i; c_{\hat{\mathbf{z}}^{i,t}}, \sigma) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(b^i - c_{\hat{\mathbf{z}}^{i,t}})^2}$$

The final algorithm is detailed in Algorithm 1. Note that latent variable assignment (step 6) takes $O(2^L)$ time per user. The total time complexity of latent variable assignment (steps 4-7) is $\mathcal{O}\left(dn + n2^L\right)$. Note that naively done, this could have taken $\mathcal{O}\left(dn2^L\right)$ time which is much worse. However, the model update step can be performed in polynomial time. Step 9 takes only $\mathcal{O}\left(nL\right)$ time and steps 10-13 take $O(dnL)$ time since there are L logistic regression problems to be solved and the gradient step for each takes $O(dn)$ time.

```
Algorithm 1: Grocery List Generator

Input: User data $\left\{(\mathbf{x}^i, b^i)\right\}_{i=1}^n$
 1: $\hat{\mathbf{\Theta}}^0 \leftarrow \left\{\hat{W}^0, \hat{\sigma}^0\right\}$                    //Initialize
 2: for $t = 0, 1, 2, \ldots,$ do
 3:                                            //Update the latent vectors
 4:     for $i = 1, 2, \ldots, n$ do
 5:         $\mathbf{v}^{i,t} = (\hat{W}^t)^\top \mathbf{x}^i$
 6:         $\hat{\mathbf{z}}^{i,t} = \underset{\mathbf{z}\in\{-1,1\}^L}{\arg\max}\ \frac{(b^i - c_{\mathbf{z}})^2}{2(\hat{\sigma}^t)^2} + \sum_{j=1}^L \log\varsigma(\mathbf{z}_j \cdot \mathbf{v}_j^{i,t})$
 7:     end for
 8:                                            //Update the model
 9:     $\hat{\sigma}^{t+1} \leftarrow \sqrt{\frac{1}{n}\sum_{i=1}^n (b^i - c_{\hat{\mathbf{z}}^{i,t}})^2}$
10:     for $j = 1, 2, \ldots, L$ do
11:         $\hat{\mathbf{w}}^{j,t+1} \leftarrow \underset{\mathbf{w}}{\arg\max}\ \sum_{i=1}^n \log\varsigma(\hat{\mathbf{z}}_j^{i,t}\left\langle \mathbf{w}, \mathbf{x}^i\right\rangle)$
12:     end for
13:     $\hat{W}^{t+1} = \left[\hat{\mathbf{w}}^{1,t+1}, \ldots, \hat{\mathbf{w}}^{L,t+1}\right]$
14: end for
```

4. Apply the EM algorithm to this problem to get the "soft assignment" algorithm. It would not be efficient since the weighted MLE problem will have to be solved explicitly and will have $\mathcal{O}\left(2^L\right)$ terms. Nevertheless, the weighted MLE problem will still be a convex problem and gradient descent can be applied to it to obtain the solution.

Problem 2.3 (The Daft Dual). Consider the following problem formulation for n binary labelled data points $(\mathbf{x}^i, y^i)_{i=1,\ldots,n}$ where $\mathbf{x}^i \in \mathbb{R}^d$ and $y^i \in \{-1, +1\}$

$$\begin{aligned} \underset{\mathbf{w},\{\xi_i\}}{\arg\min}\ & \|\mathbf{w}\|_2^2 + \sum_{i=1}^n \xi_i^2 \\ \text{s.t. } & y^i\left\langle \mathbf{w}, \mathbf{x}^i\right\rangle \geq 1 - \xi_i, \text{ for all } i \in [n] \\ & \xi_i \geq 0, \text{ for all } i \in [n] \end{aligned} \qquad (P1)$$

This is a variant of the SVM formulation. Using a technique similar to the one you used for the "Break Free from Constraints" problem in Assignment 1, you can show that $(P1)$ is equivalent to minimizing the regularized squared hinge loss (see lecture 7). However, this question is not about that. We are more interested in solving this optimization problem now.

1. Show that the constraints $\xi_i \geq 0$ are vacuous i.e. the optimization problem does not change even if we remove the all constraints $\xi_i \geq 0$ for all $i \in [n]$.

2. Write the Lagrangian for $(P1)$.

3. Give the derivation for the Lagrangian dual problem for $(P1)$.

4. What are the differences between the dual problem for $(P1)$ and the dual problem for the original SVM problem? The dual for the original SVM problem may be referred to, for example, in [**DAU**] equation 11.38.

5. *Bonus*: Are the positivity constraints $\xi_i \geq 0$ vacuous for the original SVM problem as well? Recall that the original SVM has the term $\sum_{i=1}^n \xi_i$ in its objective instead of $\sum_{i=1}^n \xi_i^2$. Why can/cannot we remove the positivity constraints from the original SVM problem without changing the solution?

(10+2+10+3=25+bonus marks)

Solution. We will give the solution in parts below

1. Consider the optimization problem $(P2)$ below

$$\begin{aligned} \underset{\mathbf{w},\{\xi_i\}}{\arg\min}\ & \|\mathbf{w}\|_2^2 + \sum_{i=1}^n \xi_i^2 \\ \text{s.t. } & y^i \left\langle \mathbf{w}, \mathbf{x}^i \right\rangle \geq 1 - \xi_i, \text{ for all } i \in [n] \qquad (P2) \end{aligned}$$

Suppose the optimal solution to the above problem $(P2)$ is $(\mathbf{w}^*, \{\xi_i^*\})$. We will show that $\xi_i^* \geq 0$ for all $i \in [n]$. Thus, $(\mathbf{w}^*, \{\xi_i^*\})$ will be a feasible solution to $(P1)$ as well. Since the two problems $(P1)$ and $(P2)$ have identical objectives and constraints (except for the positivity constraint which $(\mathbf{w}^*, \{\xi_i^*\})$ already satisfies, as we will show), $(\mathbf{w}^*, \{\xi_i^*\})$ will be an optimal solution to $(P1)$ as well. This will establish that the positivity constraints in $(P1)$ are not required at all and that positivity is assured by the objective and the other set of constraints.

To show positivity, assume the contrapositive, i.e. that for some $i_0 \in [n]$, we have $\xi_{i_0}^* < 0$. If this is the case, consider the alternate solution $(\tilde{\mathbf{w}}, \left\{\tilde{\xi}_i\right\})$ where $\tilde{\mathbf{w}} = \mathbf{w}^*$, $\tilde{\xi}_i = \xi_i^*$ for $i \neq i_0$ and $\tilde{\xi}_{i_0} = 0$. Note that for data points $i \neq i_0$, we trivially have $y^i \left\langle \tilde{\mathbf{w}}, \mathbf{x}^i \right\rangle \geq 1 - \tilde{\xi}_i$ since $(\mathbf{w}^*, \{\xi_i^*\})$ is an optimal, hence feasible solution to $(P2)$. However, for data point i_0 we have

$$y^{i_0} \left\langle \tilde{\mathbf{w}}, \mathbf{x}^{i_0} \right\rangle = y^{i_0} \left\langle \mathbf{w}^*, \mathbf{x}^{i_0} \right\rangle \geq 1 - \xi_{i_0}^* > 1 = 1 - \tilde{\xi}_{i_0},$$

where we have used the fact that $\xi_{i_0}^* < 0$. So $(\tilde{\mathbf{w}}, \left\{\tilde{\xi}_i\right\})$ is clearly feasible for $(P2)$. However, comparing the objectives, we get

$$\|\mathbf{w}^*\|_2^2 + \sum_{i=1}^n (\xi_i^*)^2 = \|\mathbf{w}^*\|_2^2 + \sum_{i \neq i_0} (\xi_i^*)^2 + (\xi_{i_0}^*)^2 = \|\tilde{\mathbf{w}}\|_2^2 + \sum_{i \neq i_0} \tilde{\xi}_i^2 + (\xi_{i_0}^*)^2 < \|\tilde{\mathbf{w}}\|_2^2 + \sum_{i=1}^n \tilde{\xi}_i^2,$$

where we have used the fact that since $\xi_{i_0}^* < 0$, we must have $(\xi_{i_0}^*) > 0$. But this means that $(\tilde{\mathbf{w}}, \left\{\tilde{\xi}_i\right\})$ offers a better objective value than $(\mathbf{w}^*, \{\xi_i^*\})$ which contadicts the fact that $(\mathbf{w}^*, \{\xi_i^*\})$ was an optimal solution. This means that we must have had $\xi_i^* \geq 0$ for all $i \in [n]$. This concludes the proof.

2. For dual variables $\{\alpha_i, \beta_i\}$, the Lagrangian of the problem $(P1)$ is the following

$$\mathcal{L}(\mathbf{w}, \{\xi_i\}, \{\alpha_i\}, \{\beta_i\}) = \|\mathbf{w}\|_2^2 + \sum_{i=1}^n \xi_i^2 + \sum_{i=1}^n \alpha_i (1 - \xi_i - y^i \left\langle \mathbf{w}, \mathbf{x}^i \right\rangle) - \sum_{i=1}^n \beta_i \xi_i$$

3. The dual problem for $(P1)$ is

$$\max_{\substack{\alpha_i \geq 0 \\ \beta_i \geq 0}} \min_{\mathbf{w},\{\xi_i\}} \|\mathbf{w}\|_2^2 + \sum_{i=1}^n \xi_i^2 + \sum_{i=1}^n \alpha_i (1 - \xi_i - y^i \left\langle \mathbf{w}, \mathbf{x}^i \right\rangle) - \sum_{i=1}^n \beta_i \xi_i$$

Applying first order optimality to the inner problem gives us

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = 0, \text{ i.e., } \mathbf{w} = \frac{1}{2}\sum_{i=1}^{n} \alpha_i y^i \cdot \mathbf{x}^i,$$

and

$$\frac{\partial \mathcal{L}}{\partial \xi_i} = 0, \text{ i.e., } \xi_i = \frac{\alpha_i + \beta_i}{2}.$$

Putting these back into the dual expression gives us

$$\max_{\substack{\alpha_i \geq 0 \\ \beta_i \geq 0}} \sum_{i=1}^{n} \alpha_i - \frac{1}{4}\sum_{i,j=1}^{n} \alpha_i\alpha_j y^i y^j \left\langle \mathbf{x}^i, \mathbf{x}^j \right\rangle - \frac{1}{4}\sum_{i=1}^{n}(\alpha_i + \beta_i)^2$$

Since $\alpha_i, \beta_i \geq 0$, we always have $(\alpha_i + \beta_i)^2 \geq \alpha_i^2$. Thus, the optimal value for every β_i in the above optimization problem is $\beta_i = 0$. This gives us the dual optimization problem as

$$\begin{aligned} \arg\max_{\{\alpha_i\}} \; & \sum_{i=1}^{n} \alpha_i - \frac{1}{4}\sum_{i,j=1}^{n} \alpha_i\alpha_j y^i y^j \left\langle \mathbf{x}^i, \mathbf{x}^j \right\rangle - \frac{1}{4}\sum_{i=1}^{n}\alpha_i^2 \\ \text{s.t. } & \alpha_i \geq 0, \text{ for all } i \in [n] \end{aligned} \tag{D1}$$

A nicer way to write the above is to use notation $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_n]^\top \in \mathbb{R}^n$, $Q \in \mathbb{R}^{n\times n}$ with $Q_{ij} = y^i y^j \left\langle \mathbf{x}^i, \mathbf{x}^j \right\rangle$, and I as the $n \times n$ identity matrix, to get

$$\begin{aligned} \arg\max_{\boldsymbol{\alpha}\in\mathbb{R}^n} \; & \sum_{i=1}^{n} \boldsymbol{\alpha}_i - \frac{1}{4}\boldsymbol{\alpha}^\top(Q + I)\boldsymbol{\alpha} \\ \text{s.t. } & \boldsymbol{\alpha}_i \geq 0, \text{ for all } i \in [n] \end{aligned} \tag{D1}$$

4. Both the original hinge loss SVM as well as the squared hinge loss SVM have the same number of variables in the dual. However, compared to the dual for the original SVM problem, the dual $(D2)$ has less stringent constraints but more regularization. The hinge loss SVM had a constraint $\boldsymbol{\alpha}_i \in [0, C]$ for a constant C specified in the primal. The squared hinge loss SVM has no upper bound on the variable at all. However, the hinge loss SVM had a dual objective

$$\sum_{i=1}^{n} \boldsymbol{\alpha}_i - \frac{1}{4}\boldsymbol{\alpha}^\top Q\boldsymbol{\alpha}$$

whereas the squared hinge loss has an additional regularizer $-\left\|\boldsymbol{\alpha}\right\|_2^2$. The regularizer is subtracted since it is a maximization problem. So the squared hinge loss dual is less constrained but more regularized.

5. No, the positivity constraint in hinge loss SVM is not vacuous at all. Suppose we have the following optimization problem

$$\begin{aligned} \arg\min_{\mathbf{w},\{\xi_i\}} \; & \left\|\mathbf{w}\right\|_2^2 + \sum_{i=1}^{n}\xi_i \\ \text{s.t. } & y^i \left\langle \mathbf{w}, \mathbf{x}^i \right\rangle \geq 1 - \xi_i, \text{ for all } i \in [n] \end{aligned} \tag{$P3$}$$

Note that this is simply the regular SVM problem with the positivity constraint removed. Using a technique similar to the one used for Assignment 1 problem 1.4, we can show the above problem to be the same as

$$\arg\min_{\mathbf{w}} \ \|\mathbf{w}\|_2^2 - \sum_{i=1}^{n} y^i \left\langle \mathbf{w}, \mathbf{x}^i \right\rangle,$$

which has a closed form solution, by applying first order optimality, as

$$\mathbf{w}^* = \frac{1}{2}\sum_{i=1}^{n} y^i \cdot \mathbf{x}^i$$

Note that if we assume that the number of positive and negative data points is the same, this is nothing but the classifier we would get if we did classification with prototypes if the two prototypes were the averages of the vectors corresponding to the positive and negative points! The above classifier has no margin guarantees. Thus, the positivity constraint is very crucial for the hinge loss SVM.