Instructions:

- Print out this final, write your solutions on it, and scan your pdf for submission to Gradescope, maintaining the original ordering of pages. If you prefer to electronically superimpose text onto the PDF (instead of writing by hand), that's fine.
- Due 9pm, March 21st.
- No extensions, and you cannot use late hours.
- No collaboration, you must answer these questions all on your own.
- You are free to ask clarification questions on Piazza.
- This exam is designed to take up to 3 hours to do (plus any time previously spent on the Kaggle competition).
- All questions have short answers. If your answer is very long, you're probably over-thinking it.

1 Multiple Choice Questions (40 points)

Specify the correct answer within the box provided. No need to justify.

Lasso (2 points)

Question A: (True or False) Every L1-constrained regression problem is equivalent to some L1-regularized regression problem.

Recall that an L1-constrained regression problem is:

$$\arg\min_{w,b} \sum_{(x,y)\in S} \left(y - (w^T x + b)\right)^2, \quad \text{s.t.} \quad \|w\|_1 \le C,$$

and that an L1-regularized regression problem is:

$$\arg\min_{w,b} \lambda\|w\|_1 + \sum_{(x,y)\in S} \left(y - (w^T x + b)\right)^2.$$

Solution A: True

Bagging & Bootstrap Sampling (3 points)

Question B: (Multiple Choice) Suppose we generate M bootstraps $S_1', \ldots, S_M'$ of a training set $S = \{(x_i, y_i)\}_{i=1}^N$. For any specific $(x, y) \in S$, what is the probability that (x, y) does not appear in ANY of $S_1', \ldots, S_M'$?

(Recall that a bootstrap is a dataset with the same size as S generated by sampling uniformly with replacement from S.)

Option A:

$$\left(1 - \frac{1}{N}\right)^N M$$

Option B:

$$\left(1 - \frac{1}{N}\right)^{NM}$$

Option C:

$$\left(1 - \frac{1}{NM}\right)^{NM}$$

Option D:

$$\left(1 - \left(1 - \frac{1}{NM}\right)^N\right)^M$$

Solution B: B

Convolutional Filters (4 points)

Consider the three convolutional filters below:

$$K_1 = \begin{bmatrix} 0.0043 & 0.0144 & 0.0214 & 0.0144 & 0.0043 \\ 0.0144 & 0.0478 & 0.0712 & 0.0478 & 0.0144 \\ 0.0214 & 0.0712 & 0.1062 & 0.0712 & 0.0214 \\ 0.0144 & 0.0478 & 0.0712 & 0.0478 & 0.0144 \\ 0.0043 & 0.0144 & 0.0214 & 0.0144 & 0.0043 \end{bmatrix}$$

$$K_2 = \begin{bmatrix} 0.0068 & 0.0225 & 0.0335 & 0.0225 & 0.0068 \\ 0.0225 & 0.0746 & 0.1112 & 0.0746 & 0.0225 \\ 0.0335 & 0.1112 & 0.1660 & 0.1112 & 0.0335 \\ 0.0225 & 0.0746 & 0.1112 & 0.0746 & 0.0225 \\ 0.0068 & 0.0225 & 0.0335 & 0.0225 & 0.0068 \end{bmatrix}$$

$$K_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.8 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Question C: (Multiple Choice) If we apply K_1, K_2, and K_3 to Figure 1, which filter corresponds to which image in Figure 2?

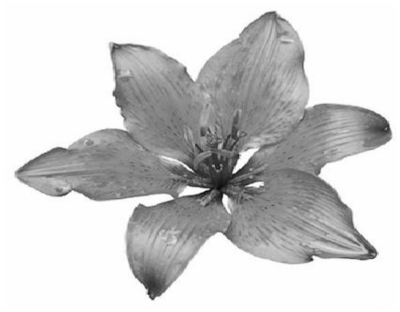


Figure 1: Original Image. All pixel intensities are clipped between 0 and 1 (black=0 and white=1).

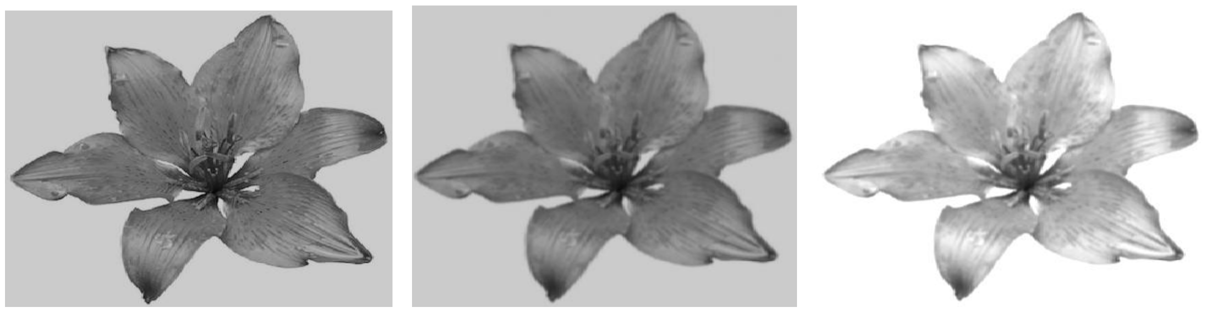


Figure 2: Convolved Images (hint: the grey background is intentional)

Option A: K_1 = Right, K_2 = Middle, K_3 = Left

Option B: K_1 = Middle, K_2 = Left, K_3 = Right

Option C: K_1 = Left, K_2 = Right, K_3 = Middle

Option D: K_1 = Middle, K_2 = Right, K_3 = Left

Solution C: B

Multiclass SVMs (3 points)

Consider the following 4-class multiclass SVM objective:

$$\arg\min_{w,\xi} \frac{1}{2}\|w\|^2 + \frac{C}{N}\sum_{i=1}^{N} \xi_i$$

s.t.

$$\forall i, \forall y' \in \{1,2,3,4\}: \; w_{y_i}^T x_i - w_{y'}^T x_i \geq \mathbf{1}_{[y_i \neq y']} - \xi_i$$

where:

$$w = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix},$$

and predictions are made via $\arg\max_y w_y^T x$. In other words, each ξ_i can be defined as:

$$\xi_i = \max_{y' \in \{1,2,3,4\}} \left\{ \mathbf{1}_{[y_i \neq y']} - \left(w_{y_i}^T x_i - w_{y'}^T x_i \right) \right\}. \quad (1)$$

Suppose for training data (x_i, y_i) we have $y_i = 1$ and:

$$w_1^T x_i = 1.0,$$

$$w_2^T x_i = 1.5,$$

$$w_3^T x_i = 0.1,$$

$$w_4^T x_i = -0.8.$$

Question D: (Multiple Choice) Which $\hat{y} \in \{1,2,3,4\}$ is the maximizer of (1)?

Solution D: 2

Feature Maps (3 points)

Consider the following 3-class multiclass feature map:

$$\phi(x,y) = \begin{bmatrix} \mathbf{1}_{[y=1]}x \\ \mathbf{1}_{[y=2]}x \\ \mathbf{1}_{[y=3]}x \\ -\mathbf{1}_{[y\in\{1,2\}]}x \end{bmatrix}, \tag{2}$$

where predictions are made via $\arg\max_{y\in\{1,2,3\}} w^T\phi(x,y)$.

Question E: (Multiple Choice) What is the effect of using (2) versus a "conventional" multiclass feature map of:

$$\phi(x,y) = \begin{bmatrix} \mathbf{1}_{[y=1]}x \\ \mathbf{1}_{[y=2]}x \\ \mathbf{1}_{[y=3]}x \end{bmatrix}. \tag{3}$$

Option A: Compared to (3), in (2) the model scores for Class 1 & 2 are encouraged to be correlated.

Option B: Compared to (3), in (2) the model scores for Class 1 & 2 are encouraged to be anti-correlated.

Option C: Either A or B are possible depending on the specific training data.

(Recall that the model score for a class y' is just $w^T\phi(x,y')$.)

Solution E: B

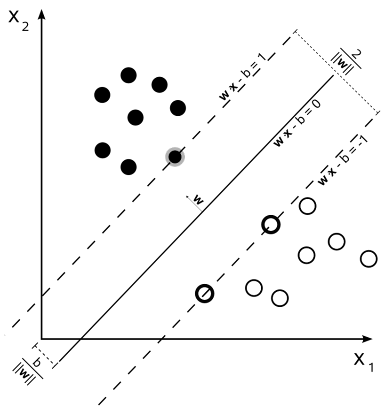
Hard-Margin SVMs (5 points)

A hard margin SVM can be written as:

$$\arg\min_{w,b} \frac{1}{2}\|w\|^2$$

s.t.

$$\forall(x,y)\in S : y(w^Tx-b)\geq 1$$



The figure to the right depicts a visualization of the hard-margin SVM for a 2-dimensional example (image source Wikipedia). The data points that are on the margin (for which the inequality constraint in the training optimization problem is a tight equality) are known as "support vectors".

Question F: (True or False) If the data is not linearly separable, then a hard-margin SVM returns $w = 0$.

Solution F: False

Question G: (Multiple Choice) Assume the data is linearly separable. What happens when we remove a data point that is NOT a support vector from the training set, and then retrain the model?

Option A: The solution w and b do not change from before.

Option B: The solution changes, and the training error is reduced.

Option C: The solution changes, and the training error is increased.

Solution G: A

AdaBoost (2 points)

Question H: (True or False) Each iteration of AdaBoost is guaranteed to reduce the training 0/1 Loss of the aggregate model.

Recall that the aggregate model of AdaBoost at iteration T is:

$$f_T(x) = \sum_{t=1}^{T} \alpha_t h_t(x) \in \Re,$$

and predictions are made via the sign of $\;_T(x)$: $h_T(x) = \texttt{sign}(f_T(x))$.

Solution H: False

Tensor Model Training (2 points)

Question I: (True or False) Every L2-regularized tensor latent-factor regression problem can be optimized via alternating closed-form optimization (i.e., converges to a local optimum).

Recall that a L2-regularized tensor regression problem can be written as:

$$\arg\min_{U,V,W} \frac{1}{2}\left(\|U\|_F^2 + \|V\|_F^2 + \|W\|_F^2\right) + \frac{1}{2}\sum_{(a,b,c)\in S}\left(Y_{a,b,c} - \langle u_a, v_b, w_c\rangle\right)^2,$$

where u_a, v_b, and w_c correspond to the a-th column of U, the b-th column of V, and the c-th column of W, respectively. The three-way dot product is defined as:

$$\langle u_a, v_b, w_c \rangle = \sum_{k=1}^{K} u_{a,k} v_{b,k}, w_{c,k}.$$

Alternating closed-form optimization implies that U (likewise V or W) can be solved optimally in closed-form if one holds the other two matrices V and W fixed.

Solution I: True

Bias-Variance Decomposition (2 points)

Let $f_S \in F_S$ denote a set of linear regression models trained on training set S using different learning algorithms (e.g., each f_S is trained using a different regularization strength). Let $\hat{f}_S$ denote the $f_S \in F_S$ that has the lowest bias in the bias-variance decomposition.

Question J: (True or False) The minimum bias learning algorithm is guaranteed to have the smallest test squared loss. In other words, the test error of $\hat{f}_S$ is guaranteed to be lower than the test error of any other f_S:

$$\forall f_S \in F_S: \ E_{(x,y)\sim P(x,y)} E_S \left[\left(y - \hat{f}_S(x) \right)^2 \right] \leq E_{(x,y)\sim P(x,y)} E_S \left[(y - f_S(x))^2 \right],$$

where $P(x, y)$ denotes the test distribution.

Solution J: False

HMM EM Learning (3 points)

Question K: (Multiple Choice) During EM training in the unsupervised setting, what best describes what happens when you initialize $P(y^j|y^{j-1})$ and $P(x^j|y^j)$ to both be the uniform distribution?

Option A: Nothing special, you just converge to some local optimum.

Option B: You converge to a degenerate local optimum where all the hidden states learn the same thing.

Option C: You get a divide-by-0 error and training crashes.

Recall that an HMM model is specified by:

$$P(x, y) = P(End|y^M) \prod_{j=1}^{M} P(x^j|y^j) P(y^j|y^{j-1}),$$

the unsupervised learning problem is specified by

$$\arg\max_\Theta \prod_{x \in S} P(x) = \arg\max_\Theta \prod_{x \in S} \sum_{y'} P(x, y'),$$

where S denotes the training set and Θ denotes all the HMM model parameters. The EM algorithm alternates between inferring the distribution of sequences y' for each training example x and using that inferred distribution to estimate better model parameters Θ.

Solution K: B

Non-Negative Matrix Factorization (2 points)

Consider the matrix factorization problem of minimizing squared reconstruction error:

$$\arg\min_{U\in\Re^{N\times K},V\in\Re^{M\times K}} \|Y - UV^T\|^2_{Fro},$$

where $Y \in \Re^{N\times M}$ is the matrix we wish to encode in a low-rank model, and U and V are our model components.

Question L: (True or False) Suppose Y is non-negative. If we enforce U and V to also be non-negative, then we typically need a larger latent dimension K to achieve the same squared reconstruction error compared to allowing U and V to take negative values.

Solution L: True

Decision Trees (2 points)

Question M: (True or False) Given infinite training data (sampled i.i.d. from the test distribution), decision trees can essentially learn arbitrary classifier functions. (Ignore the distinction between countably vs uncountably infinite, this is a high-level conceptual question.)

Solution M: True

Overfitting (2 points)

Question N: (True or False) Given infinite training data (sampled i.i.d. from the test distribution), it is essentially impossible to overfit. (Ignore the distinction between countably vs uncountably infinite, this is a high-level conceptual question.)

Solution N: True

Vanishing Gradients (5 points)

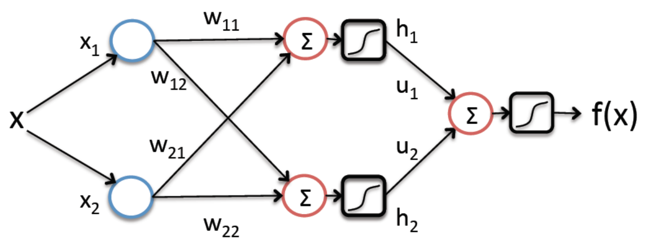


Figure 3: Illustration of Simple Neural Network.

In this question, we will consider the neural network depicted in Figure 3. There are six parameters in the model, u_1, u_2, w_{11}, w_{21}, w_{12}, and w_{22}.

The network takes in a 2-dimensional input x and outputs a real value $f(x) \in [0, 1]$. The final output $f(x)$ is a weighted combination of two hidden node activations with a sigmoid transfer function:

$$f(x) = \sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right), \tag{4}$$

where:

$$\sigma(s) = \frac{e^s}{1+e^s}, \qquad \text{and} \qquad h_i(x) = \sigma\left(\sum_{j=1}^{2} w_{ji} x_j\right).$$

The following definition of the derivative of $\sigma(s)$ is used in the derivation below:

$$\frac{\partial}{\partial s}\sigma(s) = \sigma(s)(1-\sigma(s)).$$

For a given training data point (x, y), the stochastic gradient of the squared-loss of (x, y) w.r.t. w_{11}: is derived as follows:

$$\begin{aligned}
\frac{\partial}{\partial w_{11}} L(y, f(x)) &\equiv \frac{\partial}{\partial w_{11}} (y - f(x))^2 \\
&= 2\,(y - f(x))\left(\frac{\partial}{\partial w_{11}} y - \frac{\partial}{\partial w_{11}} f(x)\right) \\
&= -2\,(y - f(x))\left(\frac{\partial}{\partial w_{11}} f(x)\right) \\
&= -2\,(y - f(x))\left(\frac{\partial}{\partial w_{11}} \sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\right) \\
&= -2\,(y - f(x))\,\sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\left(1 - \sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\right)\left(\frac{\partial}{\partial w_{11}} \sum_{i=1}^{2} u_i h_i(x)\right) \\
&= -2\,(y - f(x))\,\sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\left(1 - \sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\right)(u_1)\left(\frac{\partial}{\partial w_{11}} h_1(x)\right)
\end{aligned}$$

$$= -2\left(y - f(x)\right)\sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\left(1 - \sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\right)(u_1)\left(\frac{\partial}{\partial w_{11}}\sigma\left(\sum_{j=1}^{2} w_{j1}x_j\right)\right)$$

$$= -2\left(y - f(x)\right)\sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\left(1 - \sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\right)(u_1)\,\sigma\left(\sum_{j=1}^{2} w_{j1}x_j\right)\left(1 - \sigma\left(\sum_{j=1}^{2} w_{j1}x_j\right)\right)\left(\frac{\partial}{\partial w_{11}}\sum_{j=1}^{2} w_{j1}x_j\right)$$

$$= -2\left(y - f(x)\right)\sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\left(1 - \sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\right)(u_1)\,\sigma\left(\sum_{j=1}^{2} w_{j1}x_j\right)\left(1 - \sigma\left(\sum_{j=1}^{2} w_{j1}x_j\right)\right)\left(\frac{\partial}{\partial w_{11}}w_{11}x_1\right)$$

$$= -2\left(y - f(x)\right)\sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\left(1 - \sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\right)(u_1)\,\sigma\left(\sum_{j=1}^{2} w_{j1}x_j\right)\left(1 - \sigma\left(\sum_{j=1}^{2} w_{j1}x_j\right)\right)(x_1)$$

Question O: (Multiple Choice) In the final line of the derivation above, which term results in the vanishing gradient problem?

Option A: $(y - f(x))$, which shows why the vanishing gradient problem gets worse when you increase the depth of your network.

Option B: $\sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\left(1 - \sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\right)$, which shows why the vanishing gradient problem gets worse when you increase the depth of your network.

Option C: $(y - f(x))$, which shows why the vanishing gradient problem is an issue no matter how deep your network is.

Option D: $\sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\left(1 - \sigma\left(\sum_{i=1}^{2} u_i h_i(x)\right)\right)$, which shows why the vanishing gradient problem is an issue no matter how deep your network is.

Hint: It may help to find $\frac{\partial}{\partial w_{11}}L(y, f(x))$ where:

$$(x, y) = ((0.1, 0.5), 0.75)$$

$$(u_1, u_2) = (0.5, -0.1)$$

$$(w_{11}, w_{12}, w_{21}, w_{22}) = (0.25, 0.1, 0.05, -0.25)$$

Solution O: B

2 Naive Bayes (20 points)

We consider the following Naive Bayes model:

$$P(Happy?, Grade, Year) = P(Happy?)P(Grade|Happy?)P(Year|Happy?).$$

In other words, the y is the Happy? variable, and the two x's are the Grade and Year variables. We assume that all variables take two values, $Happy? \in \{Yes, No\}$, $Grade \in \{A, C\}$, and $Year \in \{Freshman, Senior\}$.

Consider the following training data:

Grade	Year	Happy?
A	Senior	Yes
A	Senior	Yes
A	Senior	No
A	Freshman	Yes
C	Freshman	No
C	Freshman	No
C	Senior	No
C	Senior	Yes

Question 1: (8 points) Fit the model parameters of the Naive Bayes using maximum likelihood with uniform unit pseudocounts. For instance, the maximum likelihood estimate for $P(Grade|Happy?)$ is:

$$P(Grade = A|Happy? = Yes) = \frac{1 + \sum_{(x,y)} 1_{[x_{Grade}=A \wedge y=Yes]}}{2 + \sum_{(x,y)} 1_{[y=Yes]}}.$$

Write out the final probability tables (in the box on the next page)

Solution :

	P(happy)
Happy? = yes	1/2
Happy? = no	1/2

	year = freshman	grade = A
Happy? = yes	1/3	2/3
Happy? = no	1/2	1/3

(Please show any work below, and then enter your answer in the box above)

$$P(Grade = A|Happy? = yes) = \frac{1+3}{2+4} = \frac{2}{3}$$

$$P(Grade = C|Happy? = yes) = \frac{1+1}{2+4} = \frac{1}{3}$$

$$P(Year = senior|Happy? = yes) = \frac{1+3}{2+4} = \frac{2}{3}$$

$$P(Year = frosh|Happy? = yes) = \frac{1+1}{2+4} = \frac{1}{3}$$

$$P(Grade = A|Happy? = no) = \frac{1+1}{2+4} = \frac{1}{3}$$

$$P(Grade = C|Happy? = no) = \frac{1+3}{2+4} = \frac{2}{3}$$

$$P(Year = senior|Happy? = no) = \frac{1+2}{2+4} = \frac{1}{2}$$

$$P(Year = frosh|Happy? = no) = \frac{1+2}{2+4} = \frac{1}{2}$$

Question 2: (5 points) Compute $P(Year = Freshman, Grade = C, Happy? = No)$ using the trained model from Question 1.

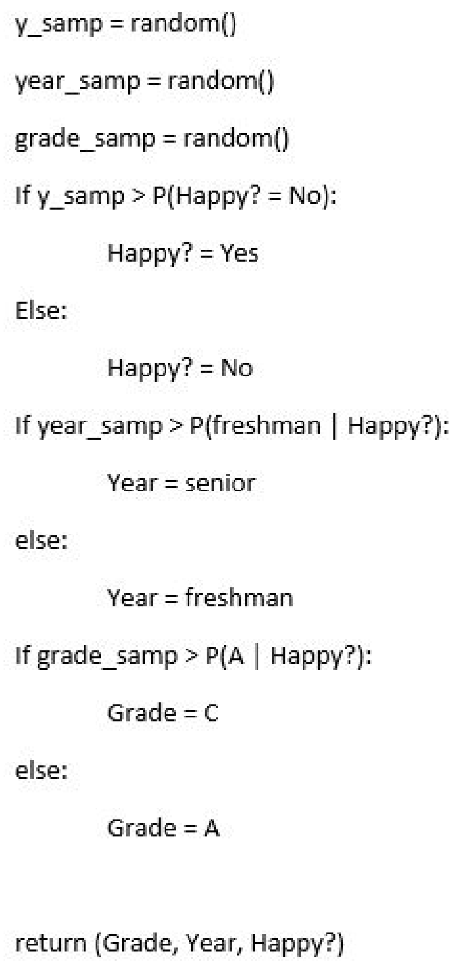
Solution :
$P(Year = Freshman, Grade = C, Happy? = No)$ = 1/6

(Please show any work below, and then enter your answer in the box above)

$$P(year = freshman, grade = C, Happy? = No)$$
$$= P(Happy? = No)P(Grade = C|Happy = No)P(Year = freshman|Happy? = No)$$
$$= \frac{1}{2} * \frac{2}{3} * \frac{1}{2} = \frac{2}{12} = 1/6$$

Question 3: (7 points) Write out the pseudocode for drawing a sample from any trained model. Assume you have repeated access to a function `random`() that returns a uniform random number in $[0, 1]$.

Solution :



```
y_samp = random()
year_samp = random()
grade_samp = random()
If y_samp > P(Happy? = No):
    Happy? = Yes
Else:
    Happy? = No
If year_samp > P(freshman | Happy?):
    Year = senior
else:
    Year = freshman
If grade_samp > P(A | Happy?):
    Grade = C
else:
    Grade = A

return (Grade, Year, Happy?)
```

3 Data Transformations (15 points)

We consider the problem of linear regression, where we predict real values given input features x via $w^T x$ using a linear model w (ignoring the bias term). Suppose we want to transform the data points x to a new representation via a transformation matrix: $\tilde{x} = Ax$. For instance, A can be a rescaling of the dimensions of x:

$$A = \begin{bmatrix} a_1 & 0 & \dots & 0 \\ 0 & a_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_D \end{bmatrix}, \tag{5}$$

where each $a_d > 0$ scales each dimension.

Question 1: (5 points) What is the relationship between w and $\tilde{w}$? In other words, write w as a function of $\tilde{w}$ and A such that $w^T x = \tilde{w}^T \tilde{x}$ holds for all x. Assume that A is a square, invertible matrix.

Solution :
$w = A^T\overline{w}$

(Please show any work below, and then enter your answer in the box above)

We have

$$w^T x = \overline{w}^T \overline{x}$$

$$\overline{x} = Ax$$

So,

$$w^T x = \overline{w}^T A x$$

$$w^T = \overline{w}^T A$$

$$w = A^T \overline{w}$$

Consider the ridge regression learning objective on the transformed data:

$$\arg\min_{\tilde{w}} \frac{\lambda}{2}\|\tilde{w}\|^2 + \sum_i (y_i - \tilde{w}^T\tilde{x}_i)^2, \tag{6}$$

Question 2: (5 points) Rewrite (6) using w, x, and A. In other words, what is the optimization problem that yields the w that corresponds to the $\tilde{w}$ learned in (6) (with the correspondence established in Question 1)? Assume that A is a square, invertible matrix.

Solution :

$$\arg\min_{w} \frac{\lambda}{2}\left\|(A^T)^{-1}w\right\|^2 + \sum_i (y_i - w^T x_i)^2$$

(Please show any work below, and then enter your answer in the box above)

$$\arg\min_{\overline{w}} \frac{\lambda}{2}\left\|\overline{w}\right\|^2 + \sum_i (y_i - \overline{w}^T \overline{x}_i)^2$$

With

$$\overline{w} = (A^T)^{-1}w$$

$$\overline{x} = Ax$$

$$A^T w^T = \overline{w}^T$$

So

$$\arg\min_{w} \frac{\lambda}{2}\left\|(A^T)^{-1}w\right\|^2 + \sum_i (y_i - (A^T w^T)(Ax_i))^2$$

$$=$$

$$\arg\min_{w} \frac{\lambda}{2}\left\|(A^T)^{-1}w\right\|^2 + \sum_i (y_i - w^T x_i)^2$$

Question 3: (5 points) Interpret your answers to Question 1 and Question 2 when A is a rescaling transformation such as (5). In other words, how is your answer to Question 2 different from standard ridge regression for w:

$$\arg\min_w \frac{\lambda}{2}\|w\|^2 + \sum_i (y_i - w^T x_i)^2.$$

Solution :

Compared to the standard ridge regression, we have the $(A^T)^{-1}$ coefficient to consider. Because we have scaled x, our w will also scale since the summation part of the equation is minimized. This is because y will stay the same even with the other scaling. Therefore, the main difference with the rescaling transformation is the $(A^T)^{-1}$ term which deals with the rescaling.

(Please show any work below, and then enter your answer in the box above)

4 Qless Kaggle Report (25 points)

Question 1: Please state your Kaggle name/ID, and your performance on the private leaderboard:

Solution : Name: Big Bus Escolar Place: 80th

Question 2: On the next page, you should include a ”report” on your work for the Kaggle competition. This is meant to serve as **concise** documentation of your work via the guiding questions below. **Please restrict your responses to the space within the box on the next page.**

- What steps did you take for preprocessing?
- What models did you try?
- What measures did you take to validate your models?
- Which model did you commit to the private leaderboard, and why did you choose that one?
- Did you use any other notable procedures that didn’t fit into the questions above? (don’t overthink this last question, ”No” is an acceptable response)

Solution :

Preprocessing:

I made a couple changes to the data during the preprocessing stage. I eliminated columns that had zero variance, like the first date columns that were all the same (Year/Day/Hour). I tried to normalize the columns that had extreme values. For example, the entering_host column had some extreme values along with a lot of missing values. So, I first filled the missing values with mean of the column values that were present, and I then normalized the entire column. Therefore, I dealt with columns with zero variance, the missing values, and extreme values.

Models:

I tried a couple of different models before coming to XGBoost. I initially tried random forests because I thought it would deal well with the large number of variables. I tested different parameters using MSE as validation, but I was not getting any good results. I decided to move on and try an out-of-the-box method: elastic net. I honestly have never used this before, but it popped up on the sklearn website. I gave it a couple tries, but the MSE was way too high. I finally came to my final model of XGBoost. Since this model performs well on Kaggle competitions, I thought I should use it. I tested parameters and tried to run a random grid, but my computer is too slow/I had other finals I had to run stuff for. This ultimately became my best model, which achieved my highest score.

Validation:

I used MSE as my main source of validation. I split the training data and ran different tests on each part of the training data. This gave me a better feel of how the models would perform. I used a random grid that split the data up many different times and tested different combinations of parameters on these different splits.

Committed Model:

When I first started using XGBoost, I achieved a good score on Kaggle, but I realized that I didn't use the data that had the preprocessing done to it. It was my highest score overall, so I used that as one of my submissions. I was a little skeptical about the situation, since I didn't touch the data, so I used the highest score XGBoost with preprocessing that I had as my second submission. Therefore, if there were any large changes to the private leaderboard, I would have that aspect covered.

Other Notable Procedures:

Nothing else, pretty much all I did was described above.