Homework 4

ESE 402/542

Due November 17, 2021 at 11:59pm

Type or scan your answers as a single PDF file and submit on Canvas.

Problem 1. Suppose we are given n data points $\{(x_i, y_i)\}_{i=1}^n$, which we assume to be generated

$$y_i = \beta_0 + \beta_1 x_i + \epsilon$$

where $\epsilon \sim \text{dist}(0, \sigma^2)$. Now suppose the data was fit using linear regression by minimizing RSS (least squares), resulting in the two estimated parameters $\hat{\beta}_0, \hat{\beta}_1$. We now want to estimate the y-variable at a new point, x_0. Denoting its true value on the line by $\mu_0 = \beta_0 + \beta_1 x_0$, the estimate is:

$$\hat{\mu}_0 = \hat{\beta}_0 + \hat{\beta}_1 x_0$$

(a) Find variance of $\hat{\mu}_0$.

(b) The standard deviation of $\hat{\mu}_0$ can be expressed as a function of $(x_0 - \bar{x})$. Find this function and briefly explain its shape.

(c) Find 95% confidence interval for $\mu_0 = \beta_0 + \beta_1 x_0$ under assumption of normality. (n is large enough)

Problem 2. Assume that $X \sim N(0,1)$, $E \sim N(0,1)$, X and E are independent, and $Y = X + \beta E$. Show that:

$$r_{XY} = \frac{1}{\sqrt{\beta^2 + 1}}$$

Note that r_{XY} is defined as $r_{XY} = \frac{\text{Cov}(X,Y)}{\sigma_X \sigma_Y}$.

Problem 3. Suppose there are n data points $\{x_i \in \mathbb{R}, y_i \in \mathbb{R}\}_{i=1}^n$. We fit a line $y = a + bx$ with least squares, and another line $x = c + dy$ with least squares. Show that $bd \leq 1$, and briefly explain when $bd = 1$ and what it means.
Hint: Cauchy-Schwarz Inequality. $|\text{Cov}(X,Y)|^2 \leq \text{Var}(X) \cdot \text{Var}(Y)$.

Problem 4. (Extra Credit) A student wants to predict a variable, $Y \in \mathbb{R}^n$, from two other variables, $X_1 \in \mathbb{R}^n$ and $X_2 \in \mathbb{R}^n$, using multiple regression. The student defines a new variable $X_3 = X_1 + X_2$ and uses multiple regression to predict Y from X_1, X_2, X_3. Show why this method is problematic.
Hint 1: $A_{n \times n}$ is invertible $\Leftrightarrow$ Rank$(A) = n$.
Hint 2: Rank$(AB) \leq \min(\text{Rank}(A), \text{Rank}(B))$.

Problem 5. See the Jupyter notebook file for problem 5.

Homework 4

ESE 402/542

Due November 12, 2020 at 11:59pm

Type or scan your answers as a single PDF file and submit on Canvas.

Problem 1. Suppose we fit n data points with a line by minimizing RSS (least squares), and that we want to estimate the line at a new point, x_0. Denoting its value on the line by μ_0, the estimate is:

$$\hat{\mu}_0 = \hat{\beta}_0 + \hat{\beta}_1 x_0$$

(a) Find variance of $\hat{\mu}_0$.

(b) The standard deviation of $\hat{\mu}_0$ can be expressed as a function of $(x_0 - \bar{x})$. Find this function and briefly explain its shape.

(c) Find 95% confidence interval for $\mu_0 = \beta_0 + \beta_1 x_0$ under assumption of normality.

Sloution:

(a) The line we are observing is given by:

$$y = \beta_0 + \beta_1 x$$

where β_0 and β_1 are:

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

$$\beta_1 = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2}$$

We know that the line goes through point $(x_0, \hat{\mu}_0)$.

$$\begin{aligned} \hat{\mu}_0 &= \beta_0 + \beta_1 x_0 \\ &= \bar{y} - \beta_1 \bar{x} + \beta_1 x_0 \\ &= \bar{y} + \beta_1 (x_0 - \bar{x}) \end{aligned}$$

Now we can find the variance of $\hat{\mu}_0$:

$$\begin{aligned}
\operatorname{Var}\left(\hat{\mu}_0\right) &= \operatorname{Var}\left(\bar{y} + \beta_1\left(x_0 - \bar{x}\right)\right) \\
&= \operatorname{Var}(\bar{y}) + \left(x_0 - \bar{x}\right)^2 \operatorname{Var}\left(\beta_1\right) \\
&= \frac{\sigma^2}{n} + \left(x_0 - \bar{x}\right)^2 \frac{\sigma^2}{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2} \\
&= \sigma^2\left(\frac{1}{n} + \left(x_0 - \bar{x}\right)^2 \frac{1}{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}\right)
\end{aligned}$$

And according to theorem:

$$\operatorname{Var}\left(\hat{\beta}_1\right) = \frac{n\sigma^2}{n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2}$$

We have:

$$\operatorname{Var}\left(\beta_1\right) = \frac{\sigma^2}{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}$$

(b) Variance of $\hat{\mu}_0$ is given by:

$$\operatorname{Var}\left(\hat{\mu}_0\right) = \frac{\sigma^2}{n} + \left(x_0 - \bar{x}\right)^2 \frac{\sigma^2}{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}$$

Standard deviation of $\hat{\mu}_0$ is given by:

$$s\left(\hat{\mu}_0\right) = \sqrt{\frac{\sigma^2}{n} + \left(x_0 - \bar{x}\right)^2 \frac{\sigma^2}{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}}$$

Writing standard deviation as a function of $x_0 - \bar{x}$. Now we have:

$$f\left(x_0 - \bar{x}\right) = \sqrt{\frac{\sigma^2}{n} + \left(x_0 - \bar{x}\right)^2 \frac{\sigma^2}{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}}$$

Or simply:

$$f(z) = \sqrt{\frac{\sigma^2}{n} + z^2 \frac{\sigma^2}{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}}$$

Since n, σ^2 and $\frac{\sigma^2}{\sum_{i=1}^{n}(x_i-\bar{x})^2}$ are constants, we can define them as:

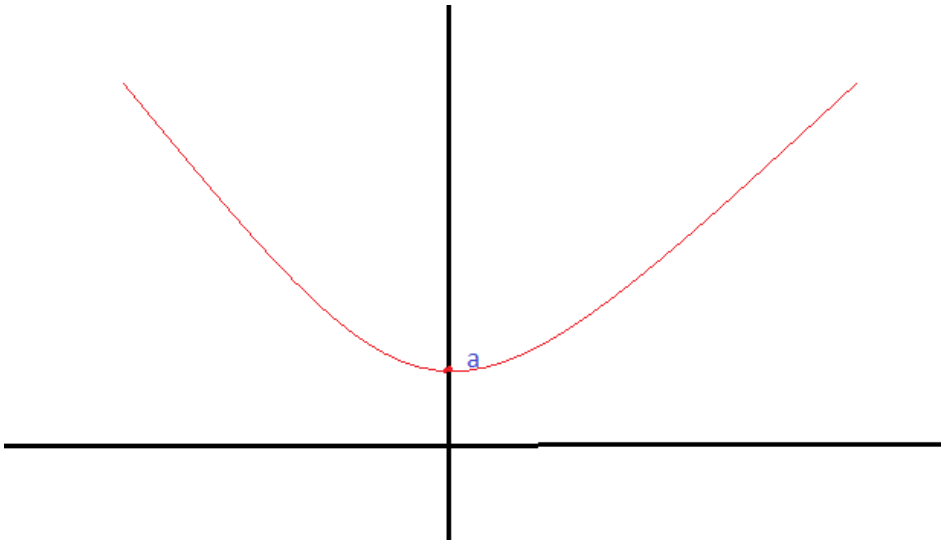
$$\frac{\sigma^2}{n} := a$$

$$\frac{\sigma^2}{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2} := b$$

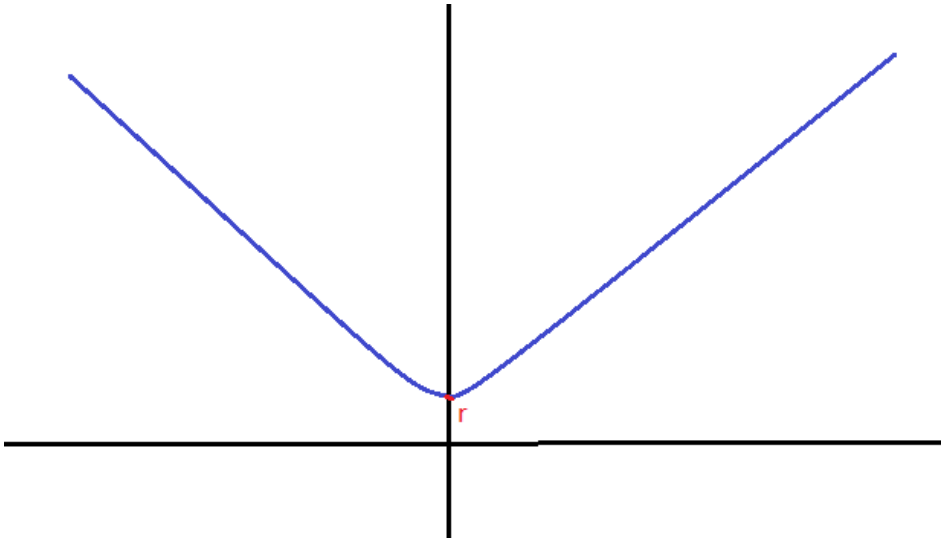
Notice that both a and b are positive constants. Now, function f can be written as:

$$f(z) = \sqrt{a + z^2 b}$$

First, we can draw function $g(z) = a + z^2 b$. The result is parabola:



Now we can draw function f, the square root of function g. Point r represents the square root of a:



(c) Standard deviation of $\hat{\mu}_0$ is given by:

$$s\left(\hat{\mu}_0\right) = \sqrt{\frac{\sigma^2}{n} + \left(x_0 - \bar{x}\right)^2 \frac{\sigma^2}{\sum_{i=1}^{n} \left(x_i - \bar{x}\right)^2}}$$

95% confidence interval is given by:

$$\hat{\mu}_0 \pm s_{\hat{\mu}_0} z\left(\frac{\alpha}{2}\right)$$

Problem 2. $X \sim N(0,1)$, $E \sim N(0,1)$, X and E are independent, and $Y = X + \beta E$. Show that:

$$r_{XY} = \frac{1}{\sqrt{\beta^2 + 1}}$$

Note that r_{XY} is defined as $r_{XY} = \frac{\sigma_{XY}}{\sigma_X \sigma_Y}$

Sloution:

Variables X and E have a standard normal distribution and variable Y is defined as $X + \beta E$. Variables X and E are independent, thus the covariance between these two is 0.
We know that correlation between two variables A and B is:

$$\rho_{A,B} = \frac{\operatorname{Cov}(A,B)}{\sigma_A \sigma_B}$$

Applying this formula to variables X and Y, we get:

$$\begin{aligned}
\rho_{X,Y} &= \frac{\operatorname{Cov}(X,Y)}{\sigma_X \sigma_Y} \\
&= \frac{\operatorname{Cov}(X, X+\beta E)}{\sigma_X \sigma_{X+\beta E}} \\
&= \frac{\operatorname{Cov}(X,X) + \operatorname{Cov}(X, \beta E)}{\sigma_X \sqrt{\operatorname{Var}(X+\beta E)}} \\
&= \frac{\operatorname{Cov}(X,X) + \beta \operatorname{Cov}(X,E)}{1 \cdot \sqrt{\operatorname{Var}(X+\beta E)}} \\
&= \frac{\operatorname{Var}(X)}{\sqrt{1 + \beta^2 \cdot 1}} \\
&= \frac{1}{\sqrt{1+\beta^2}}
\end{aligned}$$

Problem 3. Suppose there are n data points. We fit a line $y = a + bx$ with least squares, and fit a line $x = c + dy$ with least squares. Show that $bd \leq 1$, and briefly explain when $bd = 1$ and what it means.

Solution:
The first line $y = a + bx$ has:

$$b = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2}$$

The second line $x = c + dy$ has:

$$d = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (y_i - \bar{y})^2}$$

So we have:

$$\begin{aligned} bd &= \frac{\left(\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})\right)^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2 \sum_{i=1}^{n} (y_i - \bar{y})^2} \\ &= \frac{\left(\frac{1}{n}\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})\right)^2}{\frac{1}{n}\sum_{i=1}^{n} (x_i - \bar{x})^2 \frac{1}{n}\sum_{i=1}^{n} (y_i - \bar{y})^2} \\ &= \frac{(\mathrm{Cov}(X, Y))^2}{\mathrm{Var}(X)\,\mathrm{Var}(Y)} \end{aligned}$$

According to Cauchy-Schwarz inequality, $(\mathrm{Cov}(X, Y))^2 \leq \mathrm{Var}(X)\,\mathrm{Var}(Y)$, so $bd \leq 1$.

And two sides are equal iff $X - \bar{X} = (x_1 - \bar{x}, x_2 - \bar{x}, \cdots, x_n - \bar{x})$ and $Y - \bar{Y} = (y_1 - \bar{y}, y_2 - \bar{y}, \cdots, y_n - \bar{y})$ are linearly dependent, which means you can fit a perfect line on n data points with zero error.

Problem 4. A student wants to predict a variable, Y, from two other variables, $X1$ and $X2$, using multiple regression. He defines a new variable $X3 = X1 + X2$ and uses multiple regression to predict Y from $X1$, $X2$, $X3$. Show that this method is problematic.
Hint 1: $A_{n\times n}$ is invertible $\Leftrightarrow \mathrm{Rank}(A) = n$.
Hint 2: $\mathrm{Rank}(AB) \leq \min(\mathrm{Rank}(A), \mathrm{Rank}(B))$.

Solution:

We are given varaibles X_1, X_2 and Y. Also, the sum of variables X_1 and X_2 is variable X_3.
We want to predict Y from three X variables using multiple regression.
The model we will be analysing is

$$Y = \beta X + E$$

where matrices Y, X,β and E are

$$Y = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}$$

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix}$$

$$X = \begin{bmatrix} 1 & X_{11} & X_{12} & X_{13} \\ 1 & X_{21} & X_{22} & X_{23} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & X_{n1} & X_{n2} & X_{n3} \end{bmatrix}$$

Since $X_1 + X_2 = X_3$, columns of X are connceted and the column rank of matrix X is 3. Since matrix X doesn't have a full column rank, it doesn't have an inverse.
First, let's notice that the rank of matrix X is the same as the rank of matrix X^τ.
Now we can remember that if we have two matrices, A and B, the nex inequality is valid for the rank of AB:

$$r(AB) \leq min\{r(A), r(B)\}$$

Applying this to our matrices X and X^τ, we have:

$$r(X^\tau X) \leq min\{r(X), r(X^\tau)\} = min\{3, 3\} = 3$$

Since matrix $X^\tau X$ comes from the set of matrices with dimension 4×4 and its rank is 3 or less, we can conclude that it doesn't have an inverse.

Now we have a problem, because we have to find the least square estimator for β which is given by

$$\beta = (X^\tau X)^{-1} X^\tau Y$$

So, we cannot find β because we cannot find the inverse of $X^\tau X$.

Homework 5

ESE 402/542

Due on 12/01/2021 at 11:59pm

(For Problems 1 and 2, no other package except numpy and matplotlib should be used for the programming questions. For problem 3 you can use the packages of your choice.)

Problem 1.

(a) In this problem we will analyze logistic regression learned in class.
Sigmoid function can be written as $S(x) = \frac{1}{1+e^{-x}}$

- For a given variable X assume $P(Y = +1|X)$ is modeled as $P(Y = +1|X) = S(\beta_0 + \beta_1 X)$. Plot a 3d figure showing the relation between output and variable β_0 and β_1 when X = 1. Take values between [-2, 2] for both β_0 and β_1 with a step size of 0.1 to plot the 3d plot.

(b) In class, we have done binary classification with labels $Y = \{0, 1\}$. In this problem, we will be using the labels as $Y = \{-1, 1\}$ as it will be easier to derive the likelihood of the $P(Y|X)$.

- Show that if Y $\in \{-1, 1\}$ the probability of Y given X can be written as

$$P(Y|X) = \frac{1}{1 + e^{-y(\beta_0+\beta_1\mathbf{x})}}$$

- We have learned that the coefficients β_0 and β_1 can be found using MLE estimates. Show that the Log Likelihood function for m data points can be written as

$$\ln \mathcal{L}(\beta_0, \beta_1) = -\sum_{i=1}^{m} \ln\left(1 + e^{-y_i(\beta_0+\beta_1\mathbf{x}_i)}\right)$$

- Plot a 3d figure showing the relation between Log Likelihood function and variable β_0 , β_1 when $X = 1, Y = -1$ and $X = 1, Y = 1$. Take values between $[-2, 2]$ for both β_0 and β_1 with a step size of 0.1 to plot the 3d plot.

- Based on the graph, is it possible to maximize this function?

Problem 2. (Numerically Solving Logistic Regression) In general, there is no closed form expression for the optimal β_0, β_1 that maximize the above log-likelihood in Problem 1. Hence, we will use an iterative algorithm to solve for the coefficients.

Observe that in general, $\max_\alpha f(\alpha) = \min_\alpha -f(\alpha)$. Hence, we will minimize the negative-log-likelihood,

$$L(\beta_0, \boldsymbol{\beta}) = -\ln \boldsymbol{\mathcal{L}}(\beta_0, \boldsymbol{\beta}) = \frac{1}{m}\sum_{i=1}^{m} \ln\left(1 + e^{-y_i\left(\beta_0 + \boldsymbol{\beta}^\top \mathbf{x}_i\right)}\right)$$

which serves as our loss function. Note that our input data $\boldsymbol{x}_i \in \mathbb{R}^d$ are now d-dimensional vectors, and $\boldsymbol{\beta} \in \mathbb{R}^d$ is also a d-dimensional vector. Hence, our model has a total of $d+1$ parameters (including β_0).

Our objective is to iteratively find $\beta_0, \boldsymbol{\beta}$ that decrease this loss at each step, i.e. we want to find a sequence of iterates $\{\beta_0^{(k)}, \boldsymbol{\beta}^{(k)}\}_{k=1}^{K}$ such that $L(\beta_0^{(k+1)}, \boldsymbol{\beta}^{(k+1)}) \leq L(\beta_0^{(k)}, \boldsymbol{\beta}^{(k)})$. We will do this using gradient descent.

In this problem we will be working with real image data where the goal is to classify if the image is 0 or 1 using logistic regression.
The input $X \in \mathbb{R}^{m \times d}$, is a matrix, where a single data point $\mathbf{x}_i \in \mathbb{R}^d$ with $d = 784$. The labels are contained in a vector $y \in \mathbb{R}^m$, where each label $y_i \in \{0, 1\}$.

- Load the data and visualize one input data point as an image for each of label 0 and label 1. (The data should be reshaped back to [28 x 28] to be able to visualize it. Hint: use plt.imshow)

- The data is in between 0 to 255. Normalize the data to [0, 1].

- Set $y_i = 1$ for images labeled 0 and $y_i =$ -1 for images labeled 1. Split the data randomly into train and test with a ratio of 80:20.
  Why is random splitting better than sequential splitting in our case?

- Initialize the coefficients (iterates) $\beta_0^{(1)}, \boldsymbol{\beta}^{(1)}$ using a normal distribution of mean 0 and variance 1. (Hint: for $\boldsymbol{\beta}^{(1)}$, you can initialize all d entries to be $\mathcal{N}(0, 1)$).

- Write a function that computes the loss.
  Note:
  $$L(\beta_0, \boldsymbol{\beta}) = \frac{1}{m}\sum_{i=1}^{m} \ln\left(1 + e^{-y_i\left(\beta_0 + \sum_{j=1}^{d} \boldsymbol{\beta}_j \mathbf{x}_{i,j}\right)}\right)$$
  where $\boldsymbol{x}_{i,j}$ is the j-th entry of data point $\boldsymbol{x}_i$.

- To minimize the loss function, we will use gradient descent. We can write the gradients

of loss function as

$$\frac{\partial L}{\partial \beta_0} = -\frac{1}{m}\sum_{i=1}^{m}\frac{e^{-y_i\cdot(\beta_0+\boldsymbol{\beta}^\top \boldsymbol{x}_i)}}{1+e^{-(y_i\cdot(\beta_0+\boldsymbol{\beta}^\top \boldsymbol{x}_i))}}y_i = d\beta_0$$

$$\nabla_{\boldsymbol{\beta}} L = -\frac{1}{m}\sum_{i=1}^{m}\frac{e^{-y_i\cdot(\beta_0+\boldsymbol{\beta}^\top \boldsymbol{x}_i)}}{1+e^{-(y_i\cdot(\beta_0+\boldsymbol{\beta}^\top \boldsymbol{x}_i))}}y_i\boldsymbol{x}_i = d\boldsymbol{\beta}$$

Write a function to compute the gradients.

- Update the parameters as

$$\boldsymbol{\beta} = \boldsymbol{\beta} - 0.05 * d\boldsymbol{\beta}$$
$$\beta_0 = \beta_0 - 0.05 * d\beta_0$$

  (Gradient updates should be computed based on the train set)

- Repeat the process for 50 iterations and report the loss after the 50^{th} epoch.
- Plot the loss for each iteration for the train and test sets
- Logistic regression is a classification problem. We classify as $+1$ if $P(Y = 1|X) \geq 0.5$. Derive the classification rule for the threshold 0.5. (Not a programming question)
- For the classification rule derived compute the accuracy on the test set for each iteration and plot the accuracy

This format may help you write your code:

```
import numpy as np
from matplotlib import pyplot as plt

def compute_loss(data, labels, B, B_0):

    return logloss

def compute_gradients(data, labels, B, B_0):

    return dB, dB_0

if __name__ == '__main__':
    x = np.load(data)
    y = np.load(label)

    ## Split the data to train and test
    x_train, y_train, x_test, y_test = #split_data
```

```
    B = np.random.randn(1, x.shape[1])
    B_0 = np.random.randn(1)

    lr = 0.05

    for _ in range(50):

        ## Compute Loss
        loss = compute_loss(x_train, y_train, B, B_0)

        ## Compute Gradients
        dB, dB_0 = compute_gradients(x_train, y_train, B, B_0)

        ## Update Parameters
        B = B - lr*dB
        B_0 = B_0 - lr*dB_0

        ##Compute Accuracy and Loss on Test set (x_test, y_test)
        accuracy_test =
        loss_test =

    ##Plot Loss and Accuracy
```

It may help if you vectorize your code. Ideally 50 iterations should run in 10 seconds or less.

Problem 3. Recall that in classification we assume that each data point is an i.i.d. sample from a (unknown) distribution $P(X = x, Y = y)$. In this question, we are going to design the data distribution P and evaluate the performance of logistic regression on data generated using P. Keep in mind that we would like to make P as simple as we could. In the following, we assume $x \in \mathbb{R}$ and $y \in \{0, 1\}$, i.e. the data is one-dimensional and the label is binary. Write $P(X = x, Y = y) = P(X = x)P(Y = y|X = x)$. We will generate $X = x$ according to the uniform distribution on the interval $[0, 1]$ (thus $P(X = x)$ is just the pdf of the uniform distribution).

1. Design $P(Y = y|X = x)$ such that (i) $P(y = 0) = P(y = 1) = 0.5$; and (ii) the classification accuracy of any classifier is at most 0.9; and (iii) the accuracy of the Bayes optimal possible classifier is at least 0.8.

2. Using Python, generate $n = 100$ training data points according to the distribution you designed above and train a binary classifier using logistic regression on training data.

3. Generate and $n = 100$ test data points according to the distribution you designed in part 1 and compute the prediction accuracy (on the test data) of the classifier that you designed in part 2. Also, compute the accuracy of the Bayes optimal classifier on the test data. Why do you think Bayes optimal classifier is performing better?

4. Redo parts 2,3 with $n = 1000$. Are the results any different than part 3? Why?

Problem 4.

K-means clustering can be viewed as an optimization problem that attempts to minimize some objective function. For the given objectives, determine the update rule for the centroid, c_k of the k-th cluster C_k . In other word, find the optimal c_k that minimizes the objective function. The data x contains p features.

1. Show that setting the objective to the sum of the squared Euclidean distances of points from the center of their clusters,

$$\sum_{k=1}^{K} \sum_{x \in C_k} \sum_{i=1}^{p} (c_{ki} - x_i)^2$$

   results in an update rule where the optimal centroid is the mean of the points in the cluster.

2. Show that setting the objective to the sum of the Manhattan distances of points from the center of their clusters,

$$\sum_{k=1}^{K} \sum_{x \in C_k} \sum_{i=1}^{p} |c_{ki} - x_i|$$

   results in an update rule where the optimal centroid is the median of the points in the cluster.

Homework 5 Solutions

ESE 402/542

Due November 25, 2020 at 11:59pm

(For problems 1 and 2, no other package except numpy and matplotlib should be used for the programming questions. For problem 3, you can use the packages of your choice.)

Problem 1.

(a) In this problem we will analyze logistic regression learned in class.
Sigmoid function can be written as $S(x) = \frac{1}{1+e^{-x}}$

For a given variable X assume $P(Y = +1|X)$ is modeled as $P(Y = 1|X) = S(\beta_0+\beta_1 X)$. Plot a 3d figure showing the relation between output and variable β_0 and β_1 when X = 1. Take values between [-2,2] for both β_0 and β_1 with a step size of 0.1 to plot the 3d plot.

(b) In class, we have done binary classification with labels Y=$\{0, 1\}$. In this problem, we will be using the labels as Y=$\{-1, 1\}$ as it will be easier to derive the likelihood of the $P(Y|X)$.

- Show that if Y takes values $\{-1, 1\}$, the probability of Y given X can be written as,(Not programming)
$P(Y|X) = \frac{1}{1+e^{-y(\beta_0+\beta_1 \mathbf{x})}}$

- We have learned that the coefficients β_0 and β_1 can be found using MLE estimates. Show that the Log Likelihood function for m data points can be written as(Not Programming)

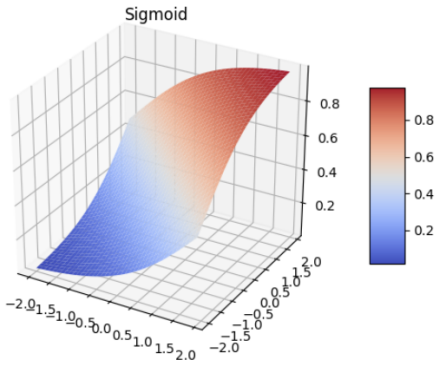
$$\ln L\left(\beta_0, \beta_1\right) = -\sum_{i=1}^{m} \ln\left(1 + e^{-y_i(\beta_0+\beta_1 \mathbf{x_i})}\right)$$

- Plot a 3d figure showing the relation between log likelihood function and variable β_0 , β_1 when X = 1, Y = -1 and X = 1, Y = 1. Take values between [-2, 2] for both β_0 and β_1 with a step size of 0.1 to plot the 3d plot

- Based on the graph, is it possible to maximize this function?

Solution

(a)



(b)

- Let $\eta(x) = \mathbf{P}(Y = +1|X = x)$ and $1 - \eta(x) = \mathbf{P}(Y = -1|X = x)$

$$\begin{aligned}
\eta(\mathbf{x}) &= \frac{1}{1 + e^{-(\beta_0 + \beta_1 \mathbf{x})}} \\
1 - \eta(\mathbf{x}) &= 1 - \frac{1}{1 + e^{-(\beta_0 + \beta_1 \mathbf{x})}} \\
&= \frac{e^{-(\beta_0 + \beta_1 \mathbf{x})}}{1 + e^{-(\beta_0 + \beta_1 \mathbf{x})}} \\
&= \frac{1}{(e^{(\beta_0 + \beta_1 \mathbf{x})}) \cdot (1 + e^{-(\beta_0 + \beta_1 \mathbf{x})})} \\
\mathbf{P}(Y = -1|X) &= \frac{1}{1 + e^{(\beta_0 + \beta_1 \mathbf{x})}}
\end{aligned}$$

Since η and $1-\eta$ differ by ± 1 sign in the power of the exponent in the denominator. We can write the combined probability as

$$p(Y|\mathbf{X}) = \frac{1}{1 + e^{-y(\beta_0 + \beta_1 \mathbf{x})}}$$

Since y $\in -1, 1$

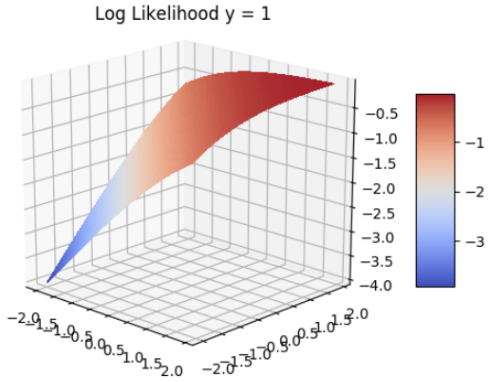
- Likelihood can be written as

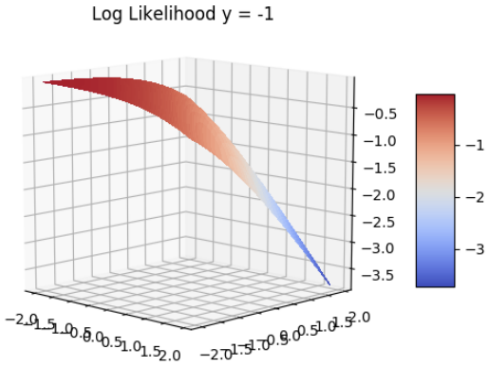
$$\begin{aligned}\mathcal{L}(\mathbf{w}) &= p\left(y_1, \ldots, y_m | \mathbf{x}_1, \ldots, \mathbf{x}_m; \beta_{\mathbf{0}}, \beta_{\mathbf{1}}\right) \\ &= \prod_{i=1}^{m} p\left(y_i | \mathbf{x}_i; \beta_{\mathbf{0}}, \beta_{\mathbf{1}}\right) \\ &= \prod_{i=1}^{m} \frac{1}{1 + e^{-y_i(\beta_0 + \beta_1 \mathbf{x})}}\end{aligned}$$

  Log Likelihood can be written as

$$\ln \mathcal{L}(\mathbf{w}) = -\sum_{i=1}^{m} \ln\left(1 + e^{-y_i \mathbf{w}^\top \mathbf{x}_i}\right)$$

•





- Looking at the graphs we can see that the function can be maximized

Problem 2.

1. While we can formalize the Likelihood estimate there is no close form expression for the coefficients β_0, β_1 maximising the above log likelihood. Hence, we will use an iterative algorithm to solve for the coefficients.
   We can see that

$$\max(-\sum_{i=1}^{m} \ln\left(1 + e^{-y_i(\beta_0+\beta_1 x_\mathbf{i})}\right)) = \min(\sum_{i=1}^{m} \ln\left(1 + e^{-y_i(\beta_0+\beta_1 x_\mathbf{i})}\right))$$

   We will describe our function loss as $L = \frac{1}{m}\sum_{i=1}^{m} \ln\left(1 + e^{-y_i(\beta_0+\beta_1 x_\mathbf{i})}\right)$. Our objective is to iterative decrease this loss as we keep computing the optimal coefficients. Here $x_i \in R$

   In this problem we will be working with real image data where the goal is to classify if the image is 0 or 1 using logistic regression.
   The input $X \in R^{m*d}$ where a single data point $x_i \in R^d$, d = 784. The matrix labels $Y \in R^m$, where each label $y_i \in \{0, 1\}$

   - Load the data into the memory and visualize one input as an image for each of label 0 and label 1. (The data should be reshaped back to [28 x 28] to be able to visualize it.)
   - The data is inbetween 0 to 255. Normalise the data to [0,1]
   - Set y_i = 1 for images labeled 0 and y_i = -1 for images labeled 1. Split the data randomly into train and test with a ratio of 80:20.
     Why is random splitting better than sequential splitting in our case?
   - Initialize the coefficients using a univariate "normal" (Gaussian) distribution of mean 0 and variance 1. (Remember that coefficients are a vector of = $[\beta_0, \beta_1 ... \beta_d]$, where d is the dimension of the input)
   - Compute the loss using the above mentioned Loss L.
     (The loss can be written as $L = \frac{1}{m}\sum_{i=1}^{m} \ln\left(1 + e^{-y_i\left(\beta_0+\sum_{j=0}^{d-1}\beta_{(j+1)}\cdot \mathbf{x}_{i,j}\right)}\right)$, where (i, j) represent the data point i for i $\in$ {1 ..m} and jth dimension of the data point x_i for j $\in$ {0 ...d-1})
   - To minimize the loss function a widely known algorithm is going in the direction opposite to the gradients of the loss function.
     (It's helpful to write the coefficients $[\beta_1 ... \beta_d]$ as a vector β and β_0 as a scalar.
     Now β is of size [d] $\in \mathrm{R}^d$ and β_0 is of size [1] $\in$ R)
     We can write the gradients of loss function as

$$\frac{\partial L}{\partial \beta_0} = -\frac{1}{m}\sum_{i=1}^{m} \frac{e^{-y_i\cdot(\beta_0+\beta\cdot x_i^T)}}{1 + e^{-(y_i\cdot(\beta_0+\beta\cdot x_i^T))}} y_i = d\beta_0$$

$$\frac{\partial L}{\partial \beta} = -\frac{1}{m}\sum_{i=1}^{m} \frac{e^{-y_i\cdot(\beta_0+\beta\cdot x_i^T)}}{1 + e^{-(y_i\cdot(\beta_0+\beta\cdot x_i^T))}} y_i x_i = d\beta$$

Write a function to compute the gradients

- Update the parameters as

$$\beta = \beta - 0.05 * d\beta$$
$$\beta_0 = \beta_0 - 0.05 * d\beta_0$$

(Gradient updates should be computed based on the train set)

- Repeat the process for 50 iterations and report the loss after the 50th epoch.
- plot the loss for each iteration for the train and test sets
- Logistic regression is a classification problem. We classify as $+1$ if $P(Y = 1|X) \geq 0.5$. Derive the classification rule for the threshold 0.5.(Not a programming question)
- for the classification rule derived compute the accuracy on the test set for each iteration and plot the accuracy.

The final code should be along this format

```
import numpy as np
from matplotlib import pyplot as plt

def compute_loss(data, labels, B, B_0):

    return logloss

def compute_gradients(data, labels, B, B_0):

    return dB, dB_0

if __name__ == '__main__':
    x = np.load(data)
    y = np.load(label)

    ## Split the data to train and test
    x_train, y_train, x_test, y_test = #split_data

    B = np.random.randn(1, x.shape[1])
    B_0 = np.random.randn(1)

    lr = 0.05

    for _ in range(50):

        ## Compute Loss
        loss = compute_loss(x_train, y_train, B, B_0)
```

```
        ## Compute Gradients
        dB, dB_0 = compute_gradients(x_train, y_train, B, B_0)

        ## Update Parameters
        B = B - lr*dB
        B_0 = B_0 - lr*dB_0

        ##Compute Accuracy and Loss on Test set (x_test, y_test)
        accuracy_test =
        loss_test =

    ##Plot Loss and Accuracy
```

Make sure to vectorize the code. Ideally 50 iterations should run in 15 seconds or less. If possible avoid using for loops, except for the 50 iterations of gradient updates given in the sample code.

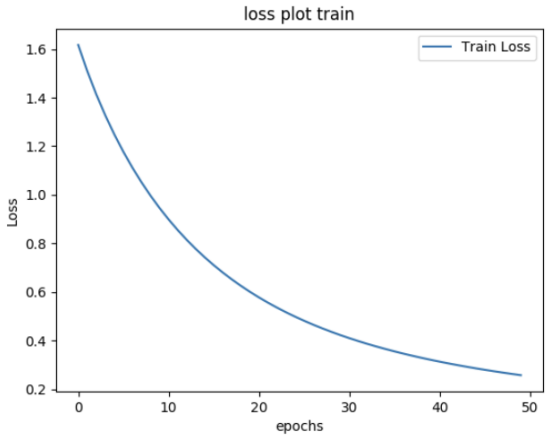
Solution

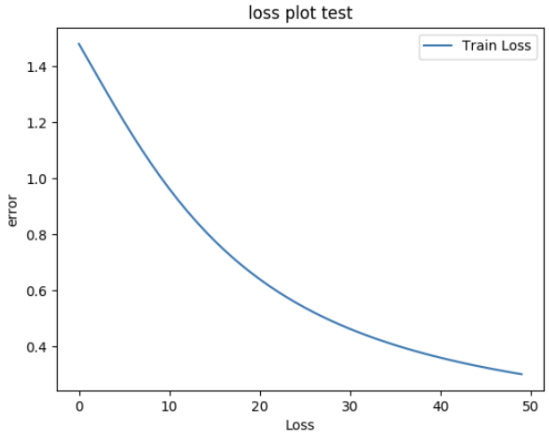
(a) The data is arranged in the order of 1's and 0's. So if we split sequentially, the testset will contain only images and labels belonging to 1's.

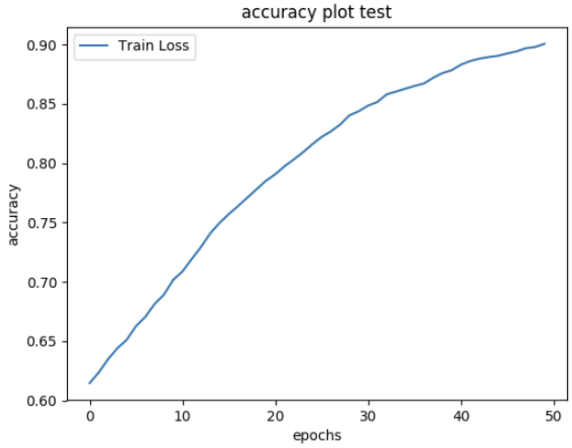
(b) Classification rule can be written as:

$$\begin{aligned} P(Y = +1|X) = \frac{1}{(1 + e^{-(\beta_0 + \beta_1 x_\mathbf{i})})} &>= 0.5 \\ = (\beta_0 + \beta_1 x_\mathbf{i}) &>= 0 \\ So, & \\ P(Y|X) = sign\,(\beta_0 + \beta_1 x_\mathbf{i}) & \end{aligned}$$

(c) The graphs are:


loss plot train
Train Loss
Loss
epochs
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0
10
20
30
40
50


loss plot test
Train Loss
error
Loss
1.4
1.2
1.0
0.8
0.6
0.4
0
10
20
30
40
50


accuracy plot test
Train Loss
accuracy
epochs
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0
10
20
30
40
50

Problem 3.

Recall that in classification we assume that each data point is an i.i.d. sample from a(n unknown) distribution $P(X = x, Y = y)$. In this question, we are going to design the data distribution P and evaluate the performance of logistic regression on data generated using P. Keep in mind that we would like to make P as simple as we could. In the following, we assume $x \in R$ and $y \in \{0, 1\}$, i.e. the data is one-dimensional and the label is binary. Write $P(X = x, Y = y) = P(X = x)P(Y = y|X = x)$. We will generate $X = x$ according to the uniform distribution on the interval [0,1] (thus $P(X = x)$ is just the pdf of the uniform distribution).

1. Design $P(Y = y|X = x)$ such that (i) $P(y = 0) = P(y = 1) = 0.5$; and (ii) the classification accuracy of any classifier is at most 0.9; and (iii) the accuracy of the Bayes optimal possible classifier is at least 0.8.

2. Using Python, generate $n = 100$ training data points according to the distribution you designed above and train a binary classifier using logistic regression on training data.

3. Generate $n = 100$ test data points according to the distribution you designed in part 1 and compute the prediction accuracy (on the test data) of the classifier that you designed in part 2. Also, compute the accuracy of the Bayes optimal classifier on the test data. Why do you think Bayes optimal classifier is performing better?

4. Redo parts 2,3 with $n = 1000$. Are the results any different than part 3? Why?

Solution

1. Let $h(X)$ be the predicted label and $\eta(X) = P(Y = 1|X)$. We define $\eta(X)$ as

$$\eta(X) = \mathbf{P}(Y = +1|X = \mathbf{x}) = \begin{cases} p & \text{if } x \geq 0.5 \\ q & \text{otherwise} \end{cases}$$

We can design such that $p > 0.5$ and $q < 0.5$. The error for the classification is:

$$\begin{aligned} \text{er}_D[h] &= \mathbf{P}_{(X,Y)\sim D}(h(X) \neq Y) \\ &= \mathbf{E}_{(X,Y)\sim D}[\mathbf{1}(h(X) \neq Y)] \\ &= \mathbf{E}_X\left[\mathbf{E}_{Y|X}[\mathbf{1}(h(X) \neq Y)]\right] \\ &= \mathbf{E}_X[\mathbf{P}(Y = +1|X) \cdot \mathbf{1}(h(X) \neq +1) + \mathbf{P}(Y = 0|X) \cdot \mathbf{1}(h(X) \neq 0)] \\ &= \mathbf{E}_X[\eta(X) \cdot \mathbf{1}(h(X) = 0) + (1 - \eta(X)) \cdot \mathbf{1}(h(X) = +1)] \end{aligned}$$

We can write above as the PDF of the X is given as $f(X) = 1$. The CDF for X will be $P(X \leq x) = x$. For Bayes classifier error, the minimum achievable error, which is given as:

$$\begin{aligned} \text{er}^*_D[h] = \mathbf{E}_X[\min(\eta(X), (1 - \eta(X)))] &\geq 0.1 \\ \Rightarrow 0.5 * q + 0.5(1 - p) &\geq 0.1 \\ p - q &\leq 0.8 \end{aligned}$$

Another equation according the other constraint of the maximum error achievable, is given as

$$\begin{aligned} \mathbf{E}_X[\min(\eta(X), (1 - \eta(X)))] &\leq 0.2 \\ \Rightarrow p - q &\geq 0.6 \end{aligned}$$

We can any value of p from 0.8 to 0.9. Let's take the values as $p = 0.85$ and $q = 0.15$, we can have the following model

$$\mathbf{P}(Y = +1|X = \mathbf{x}) = \begin{cases} 0.85 & \text{if } x \geq 0.5 \\ 0.15 & \text{otherwise} \end{cases}$$

From the above design $p(Y = +1) = 0.5 * 0.85 + 0.5 * 015 = 0.5$.

For part(b) and (c), the accuracy of logistic regression should be less than that of Bayes optimal classifier as the Bayes classifier accuracy is calculated using the true distribution.

For part (d), the accuracy of logistic regression with 1000 data points will be slightly better than 100 data points.

Problem 4.

K-means clustering can be viewed as an optimization problem that attempts to minimize some objective function. For the given objectives, determine the update rule for the centroid, c_k of the k-th cluster C_k . In other word, find the optimal c_k that minimizes the objective function. The data x contains p features.

1. Show that setting the objective to the sum of the squared Euclidean distances of points from the center of their clusters,

$$\sum_{k=1}^{K} \sum_{x \in C_k} \sum_{i=1}^{p} (c_{ki} - x_i)^2$$

   results in an update rule where the optimal centroid is the mean of the points in the cluster.

2. Show that setting the objective to the sum of the Manhattan distances of points from the center of their clusters,

$$\sum_{k=1}^{K} \sum_{x \in C_k} \sum_{i=1}^{p} |c_{ki} - x_i|$$

   results in an update rule where the optimal centroid is the median of the points in the cluster.

Solution

1. Without the loss of generality, for the k^{th} cluster for the i^{th} point, the gradient for the centroid value c_{ki} are:

$$\begin{aligned}
\frac{\partial}{\partial c_{ki}} \sum_{k=1}^{K} \sum_{x \in C_k} \sum_{i=1}^{p} (c_{ki} - x_i)^2 &= \sum_{k=1}^{K} \sum_{x \in C_k} \sum_{i=1}^{p} \frac{\partial}{\partial c_{ki}} (c_{ki} - x_i)^2 \\
&= \sum_{x_i \in C_{ki}} 2 (c_{ki} - x_i) \\
&= 0 \\
\Rightarrow |C_{ki}| \, c_{ki} &= \sum_{x_i \in C_{ki}} x_i \\
\Rightarrow c_{ki} &= \frac{1}{|C_{ki}|} \sum_{x_i \in C_{ki}} x_i
\end{aligned}$$

   Thus, c_k is the mean of the k^{th} cluster.

2. Without loss of generality, the first order condition for the k^{th} centroid at predictor i is

$$\begin{aligned}\frac{\partial}{\partial c_{ki}} \sum_{k=1}^{K} \sum_{x \in C_k} \sum_{i=1}^{p} |c_{ki} - x_i| &= \sum_{k=1}^{K} \sum_{x \in C_k} \sum_{i=1}^{p} \frac{\partial}{\partial c_{ki}} |c_{ki} - x_i| \\ &= \sum_{x_i \in C_{ki}} \operatorname{sgn}(x_i - c_{ki}) \\ \Rightarrow |\{x_i | x_i \leq c_{ki}\}| &= |\{x_i | x_i \geq c_{ki}\}|\end{aligned}$$

Thus c_k is the median of the k^{th} cluster.

Homework 6

ESE 402/542

Due December 10, 2021 at 11:59pm

Type or scan your answers as a single PDF file and submit on Gradescope.

Problem 1. *Principal Component Analysis.* Consider the following dataset:

x	y
0	1
1	1
2	1
2	3
3	2
3	3
4	5

(a) Standardize the data and derive the two principal components in sorted order. What is the new transformed dataset using the first principal component?

(b) Repeat the previous analysis but this time do not standardize the original data. Is Principal Component Analysis scale invariant?

Problem 2. *Polynomial Regression.* Load the dataset poly_data.csv. The first column is a vector of inputs x and the second column is a vector of responses y. Suppose we believe it was generated by some polynomial of the inputs with Gaussian error, i.e. for some (unknown) p,

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \cdots + \beta_p x_i^p + \epsilon_i$$

where $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$. We would like to recover the true coefficients of the underlying process. A polynomial regression can be estimated by including all powers of x as predictors in the model (see recitation 7 for details). However, the problem is that we don't know what the true value of p is. We will use k-fold cross validation to solve this problem.

(a) Pick a set of polynomial models, i.e. all polynomials of degree 1 to degree 40. Compute the k-fold cross validation error (mean squared error) for each of these models. Report the value of k that you use and plot the cross-validation error as a function of polynomial degree. Which polynomial degree fit the data the best?

(b) Choose the best polynomial model obtained from the previous part, and use to it regress the entire dataset. Report the polynomial coefficients and make a scatter plot of the x_i's and y_i's with your fitted polynomial.

Problem 3. *Bayes Optimal vs. Logistic Regression.* Recall that in classification, we assume that each data point (x_i, y_i) is drawn i.i.d. from a joint distribution P, i.e. $P(X = x, Y = y) = P(Y = y)P(X = x|Y = y))$. In this problem, we will examine a particular distribution on which Logistic Regression is optimal. Suppose that this distribution is supported on $x \in \mathbb{R}, y \in \{-1, +1\}$, and given by:

$$P(Y = +1) = P(Y = -1) = 1/2$$
$$P(X = x|Y = +1) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(x-5)^2}{2}}$$
$$P(X = x|Y = -1) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(x+5)^2}{2}}$$

(a) Show that the joint data distribution is given by

$$P(X = x, Y = y) = \frac{1}{2\sqrt{2\pi}} e^{-\frac{(x-5y)^2}{2}}$$

(b) Plot (either using code or hand-drawn neatly) the conditional distributions $P(X = x|Y = +1)$ and $P(X = x|Y = -1)$ in a single figure. Note: these are just Gaussian PDFs.

(c) Write the Bayes optimal classifier $h^*(x)$ given the above distribution P and simplify. Hint: you should get a classification rule that classifies x based on whether or not it is greater than a threshold.

(d) Compute the classification error rate of the Bayes optimal classifier, i.e.

$$\Pr_{(x,y)\sim P}(h^*(x) \neq y) = \mathop{\mathbb{E}}_{(x,y)\sim P}[\mathbb{1}_{h^*(x)\neq y}]$$

Hint: Your result should be of the form $1 - \Phi(c)$, where $\Phi(\cdot)$ is the Gaussian CDF.

(e) (Extra Credit) Recall that Logistic Regression assumes that the data distribution is of the form

$$P(Y = +1|X = x) = \frac{1}{1 + e^{-\beta_0 - \beta_1 x}}$$

Show that the distribution given above satisfies this assumption. What values of β_0, β_1 does this correspond to?

Problem 4. (Extra Credit) *k-means is suboptimal.* Recall that the k-means algorithm attempts to minimize the following objective:

$$\min_{c_1,\ldots,c_k} \sum_{i=1}^{n} \|x_i - c(x_i)\|_2^2 \tag{1}$$

where $c(x_i)$ is the closest center to x_i. Show that the k-means algorithm algorithm does not always find the optimal solution of the above objective.

Hint 1: Let OPT denote the optimal objective. For every $t > 1$, show there exists an instance of the above optimization problem for which the k-means algorithm *might* find a solution whose objective value at least $t \cdot$OPT. In other words, find a set of points $x_1, \ldots, x_n$ for which k-means, with some bad initialization of the centers, will output a set of centers that achieves an objective of $t \cdot$OPT.

Hint 2: Start with an example of 4 points in a 2-D plane, with 2 clusters. You can then generalize this example to arbitrary p dimensions, n data points, and k clusters.

Problem 5. (Extra Credit) Load the Labeled Faces in the Wild dataset from sklearn. You can load this data as follows:

```
from sklearn.datasets import fetch_lfw_people
faces = fetch_lfw_people(min_faces_per_person=60)
```

For this exercise, we will use PCA on image data, in particular pictures of faces, to extract features.

(a) Perform PCA on the dataset to find the first 150 components. Since this is a large dataset, you should use randomized PCA instead, which can also be found on sklearn. Show the eigenfaces associated with the first 1 through 25 principal components.

(b) Using the first 150 components you found, reconstruct a few faces of your choice and compare them with the original input images.

Homework 6

ESE 402/542

December 17, 2020

Solution 1. (Principal Component Analysis)

(a) Solution:

The Two PCA Components are:

[[0.70710678 0.70710678]

[0.70710678 -0.70710678]]

The Transformed Dataset using the first Principal Component is:

[[-1.87305312]

[-1.30527815]

[-0.73750317]

[0.28355177]

[0.34079927]

[0.85132674]

[2.44015666]]

(b) Solution:

```
The Two PCA Components are:

[[ 0.65908697  0.75206673]

 [ 0.75206673 -0.65908697]]


The Transformed Dataset is:

[[-2.37927216]

 [-1.72018519]

 [-1.06109821]

 [ 0.44303524]

 [ 0.35005549]

 [ 1.10212221]

 [ 3.26534264]]
```

The above results show that Principal Component Analysis is not scale invariant, that is, multiplying each feature vector by a non-zero number changes the Principal Components.
Therefore, it is essential to standardize the dataset before using PCA.

Solution 2. (k-means is suboptimal)

Consider an example with 4 data points in 2D plane: $(\sqrt{t}, 1), (-\sqrt{t}, 1), (\sqrt{t}, -1), (-\sqrt{t}, -1)$ where $t > 1$. If $k = 2$, we know that optimal partition is $C_1 = \{(\sqrt{t}, 1), (\sqrt{t}, -1)\}, C_2 = \{(-\sqrt{t}, 1), (-\sqrt{t}, -1)\}$, OPT $= \min_{c_1, c_2, \cdots, c_k} \sum_{i=1}^{n} ||x_i - c(x_i)||_2^2 = 4$ with centers $(\sqrt{t}, 0), (-\sqrt{t}, 0)$.

Now if we initialize on centers: $(0, 1), (0, -1)$, we can no longer update clusters, algorithm reaches end, and $\min_{c_1, c_2, \cdots, c_k} \sum_{i=1}^{n} ||x_i - c(x_i)||_2^2 = 4t = t \times \text{OPT}$.

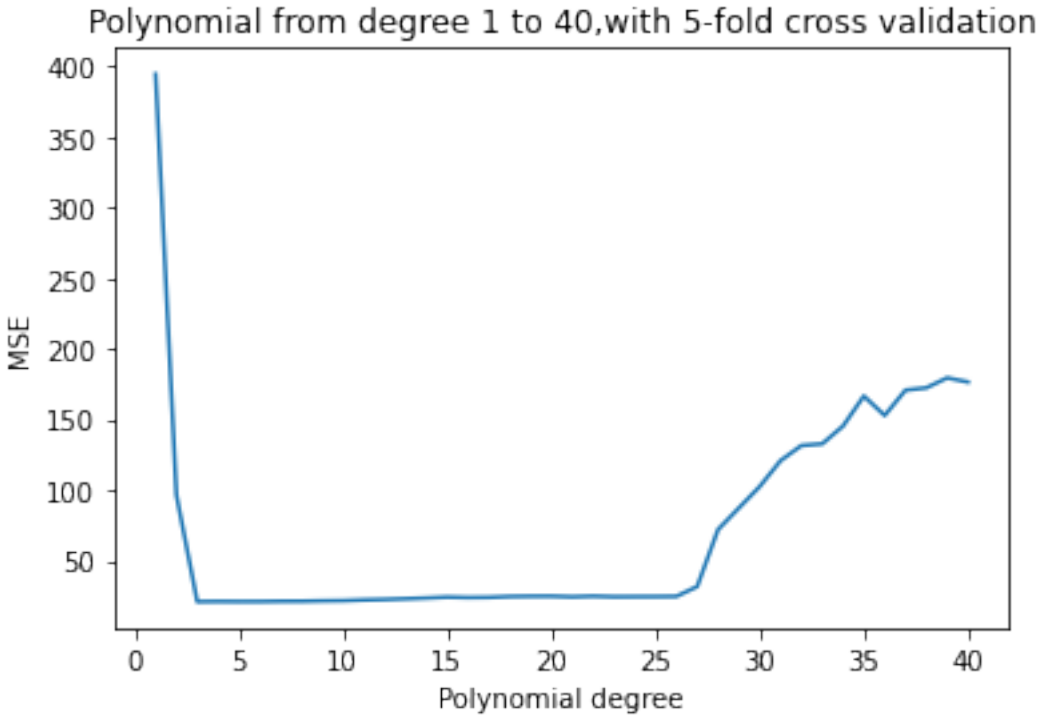
We can easily generalize this example for p-dimension:
$(\sqrt{t}, 1, 0, 0, \cdots, 0)$, $(-\sqrt{t}, 1, 0, 0, \cdots, 0)$, $(\sqrt{t}, -1, 0, 0, \cdots, 0)$, $(-\sqrt{t}, -1, 0, 0, \cdots, 0)$.

Or N data points:
$(\sqrt{t}, 0)$,$(-\sqrt{t}, 0)$,$(\sqrt{t}, 2)$,$(-\sqrt{t}, 2)$,$(\sqrt{t}, 4)$,$(-\sqrt{t}, 4) \cdots$,$(\sqrt{t}, 2n)$,$(-\sqrt{t}, 2n)$ with initialized centers $(0, 0), (0, 2), \cdots, (0, n)$.
There are $N = 4n$ data points and $k = 2n$ clusters. For $N = 4n + 1/4n + 2/4n + 3$, we can just add $1/2/3$ data points which are very far away from any other points, (Like $(10^8 t, 0), (-10^8 t, 0), (0, -10^8 n)$ etc.) and $k = 2n + 1/2n + 2/2n + 3$.

Or k clusters:
Above example, $k = 2n$ or $2n + 1$.

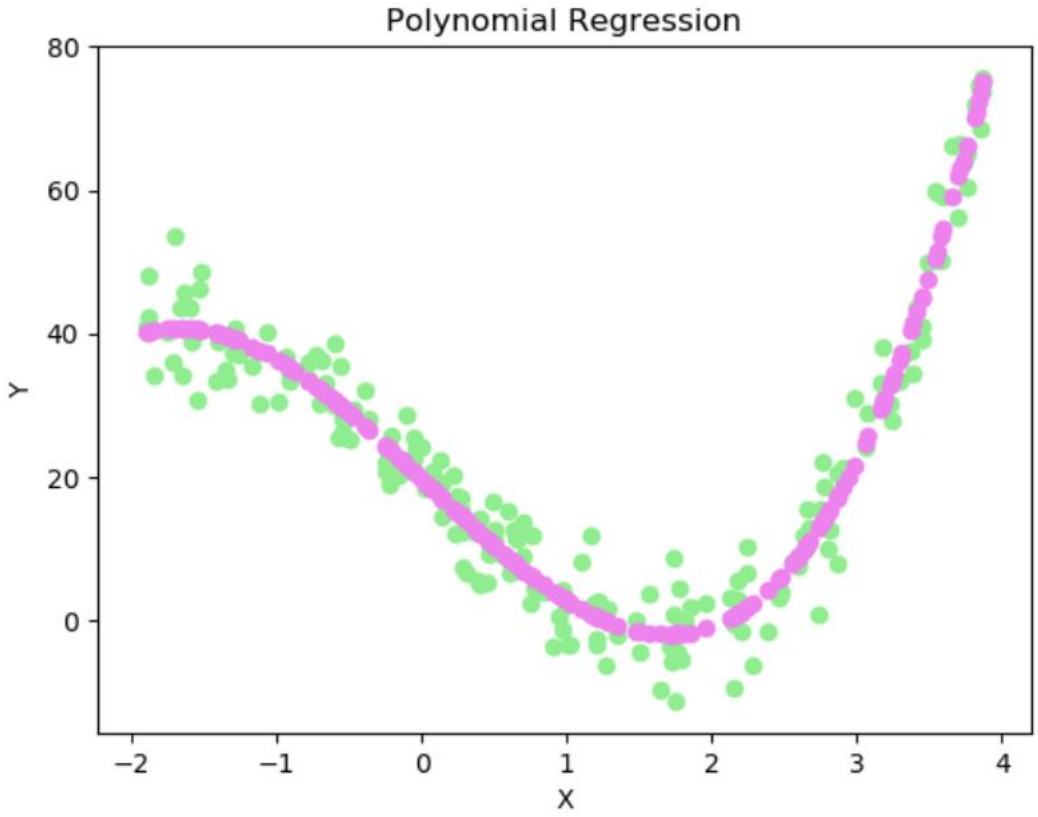
Solution 3. (Polynomial Regression)

(a) We examine the polynomials from 1 to 40 and choose k = 5.



It can be seen that the lowest MSE is when the degree is 3.

(b) Since the lowest classification error was when the degree of the polynomial was 3, I chose the cubic model for part b.
After running polynomial regression, the coefficients calculated are as follows:
0, -19.04905889, -0.09961775, 2.24871642.
There are 4 coefficients since it's a cubic polynomial. (B0, B1, B2 and B3)
The scatter plot is shown below:



The green scatter plot is of the original data. The violet scatter plot is of the cubic curve (that is, $y_{hat} = B_0 + B_1 * x + B_2 * x^2 + B_3 * x^3$).

Solution 4. (Classification)

1. Solution:

We can re-state given equations into:

$$P(X=x \mid Y=y) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x-5y)^2}{2}\right)$$

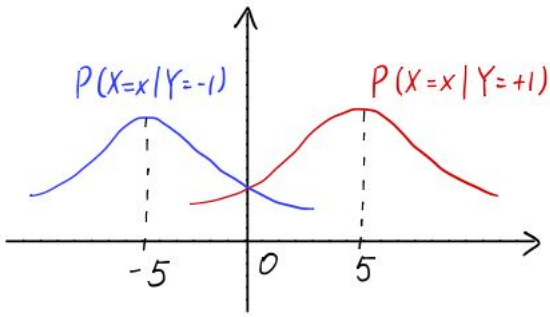
$$\therefore \quad P(X=x, Y=y)$$

$$= P(Y=y)\, P(X=x \mid Y=y)$$

$$= \frac{1}{2} \cdot \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x-5y)^2}{2}\right)$$

$$= \frac{1}{2\sqrt{2\pi}} \exp\left(-\frac{(x-5y)^2}{2}\right)$$

2. Solution:



3. Solution:

$$P(Y=y \mid X=x) = \frac{P(X=x, Y=y)}{P(X=x)}$$

$$= P(Y=y \mid X=x) \cdot \frac{P(Y=y)}{P(X=x)}$$

$$\therefore \quad \hat{y} = h(x) = \arg\max_y P(Y=y \mid X=x)$$

$$= \arg\max_y P(Y=y \mid X=x) \cdot \frac{P(Y=y)}{P(X=x)}$$

$$= \arg\max_y P(X=x \mid Y=y)$$

$$= \begin{cases} +1, & \text{if } x > 0 \\ -1, & \text{otherwise} \end{cases} \quad \text{by plot}$$

4. Solution:

$$
\begin{aligned}
&\text{error} \\
&= \mathbb{E}_{(X,Y)\sim P}\left[1\left(h(x) \neq Y\right)\right] \\
&= P_{(X,Y)\sim P}\left(h(x) \neq Y\right) \\
&= P(X>0)\,P(Y=-1 \mid X=x) + P(x<0)\,P(Y=+1 \mid X<0) \\
&= P(Y=-1, X>0) + P(Y=+1, X<0) \\
&= P(Y=-1)\,P(X>0 \mid Y=-1) + P(Y=+1)\,P(X<0 \mid Y=+1) \\
&= 2 \cdot \frac{1}{2}\left(1-\Phi\left(\frac{5}{1}\right)\right) \\
&= 1-\Phi(5)
\end{aligned}
$$

5. Solution:

Estimate: $P(Y=+1 \mid X=x)$, $P(Y=-1 \mid X=x)$

Model: $P(Y=+1 \mid X=x) = \dfrac{e^{\beta_0+\beta_1 x}}{1+e^{\beta_0+\beta_1 x}}$

$$P(Y=-1 \mid X=x) = \frac{1}{1+e^{\beta_0+\beta_1 x}}$$

Optimization:

$$
\begin{aligned}
L(\beta_0, \beta_1 \mid x) &= Pr(Y \mid X; \beta_0, \beta_1) \\
&= \prod_{i=1}^{n} P(Y=y_i \mid X=x_i; \beta_0, \beta_1)
\end{aligned}
$$

$$\max_{\beta_0, \beta_1} L(\beta_0, \beta_1 \mid x)$$

6. Solution:

$$\text{From: } P(X=x \mid Y=+1) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x-5)^2}{2}\right),\ P(Y=+1)=\frac{1}{2}$$

$$P(X=x \mid Y=-1) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x+5)^2}{2}\right),\ P(Y=-1)=\frac{1}{2}$$

$$P(Y=+1 \mid X=x)$$

$$= \frac{P(X=x,\ Y=+1)}{P(X=x,\ Y=+1) + P(X=x,\ Y=-1)}$$

$$= \frac{P(X=x \mid Y=+1)P(Y=+1)}{P(X=x \mid Y=+1)P(Y=+1) + P(X=x \mid Y=-1)P(Y=-1)}$$

$$= \frac{\frac{1}{2}\cdot\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x-5)^2}{2}\right)}{\frac{1}{2}\cdot\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x-5)^2}{2}\right) + \frac{1}{2}\cdot\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x+5)^2}{2}\right)}$$

$$= \frac{\exp\left(-\frac{(x-5)^2}{2}\right)}{\exp\left(-\frac{(x-5)^2}{2}\right) + \exp\left(-\frac{(x+5)^2}{2}\right)}$$

$$= \frac{1}{\exp\left(\frac{(x-5)^2}{2} - \frac{(x+5)^2}{2}\right) + 1}$$

$$= \frac{1}{\exp(-10x) + 1}$$

Note that logistic regression would try estimate

$$P(Y=+1 \mid X=x) = \frac{1}{1+\exp(-\beta_1 x - \beta_0)}$$

When n is large, which means there is enough data to estimate probability accurately, So $\beta_0 = 0$, $\beta_1 = 10$.

Solution 5. (Extra Credit)

(a) Some eigenfaces associated with the first 25 principal components:

(b) Reconstructed faces:

Original faces:

It can be seen that the reconstructed faces are very similar to the original faces. This is because the amount of capture of data energy/variation reduces as we go to the lower principal components. Here, we considered the first 150 components. So, these components capture the most energy (PC1 has the highest, followed by PC2, and so on).

Homework 7

ESE 402/542

Ungraded

Problem 1. Assume we have 1000 data points that are 50-dimensional, i.e. $\{x_i \in \mathbb{R}^{50}\}_{i=1}^{1000}$. Suppose the singular values of the data matrix are given by $\sigma_1 = 10, \sigma_2 = 9, \sigma_3 = 8, \ldots, \sigma_{10} = 1, \sigma_{11} = 0, \sigma_{12} = 0, \ldots$. We would like to find a low-dimensional representation of the data points in a way that at least 50% of the energy of the data is kept. What is the minimum value of the dimension that we can use for the low-dimensional representation of the data?

Problem 2. Let $\mathcal{H}$ be the class of functions of the form $h_{a,b} : \mathbb{R} \mapsto \{0, 1\}$, where

$$h_{a,b}(x) = \mathbb{1}_{\{x \in [a,b]\}}$$

for all $a, b \in \mathbb{R}$. Our goal will be to show that $\text{VCdim}(\mathcal{H}) = 2$.

(a) Let $C = \{c_1, \ldots, c_k\} \subset \mathbb{R}$. Recall that the restriction of $\mathcal{H}$ to C, denoted by $\mathcal{H}_C$, is the set of all binary k-tuples that can be derived from evaluating the functions in $\mathcal{H}$ on the set C. That is,

$$\mathcal{H}_C = \{(h(c_1), \ldots, h(c_k)) : h \in \mathcal{H}\}$$

Compute $\mathcal{H}_C$ for $C = \{1, 2\}$, and $C = \{1, 2, 3\}$.

(b) For any set C with $|C| = 2$, show that $|\mathcal{H}_C| = 4$. Also, can we say that for any set C with $|C| = 3$ we have $|\mathcal{H}_C| < 8$?

(c) Recall that a function class $\mathcal{H}$ shatters a set C if $|\mathcal{H}_C| = 2^{|C|}$. Given the specific choice of $\mathcal{H}$ as above, does $\mathcal{H}$ shatter any set C of size 2? Is there a set C of size 3 that is shattered by $\mathcal{H}$?

(d) Recall that the VC dimension of $\mathcal{H}$ is the maximal size of a set C that can be shattered by $\mathcal{H}$. What is the VC dimension of the class $\mathcal{H}$ considered in the question?

Problem 1. *Short answer:*

(a) In your own words, define what it means for a hypothesis class to be *PAC learnable*.

(b) Let $\mathcal{H}$ be the class of *all* functions from $\mathbb{R}$ to $\{0, 1\}$. What is the VC dimension of $\mathcal{H}$?

(c) For a finite function class $\mathcal{H}$ with m functions $h_1, \ldots, h_m$, such that $h_i : \mathcal{X} \to \{0, 1\}$, explain why $\text{VCdim}(\mathcal{H}) \leq \log_2 m$. Are there cases where the VC dimension is exactly $\log_2 m$?

Homework 7 Solutions

ESE 402/542

Due Dec 15th, 2020 11:59 pm

Problem 1. *Solution:*

$$E = \sum_{i=1}^{n} ||X_i||^2$$

$$E = \sum_{i=1}^{n} \sigma_j^2 = 385$$

We want to find a compression algorithm that retains at least 50% of the energy. Thus, the sum of the singular values must add up to at least $\frac{385}{2} = 192.5$.

If we select the first 3 components:

$$E = 10^2 + 9^2 + 8^2 = 245$$

Since $245 > 192.5$ the minimum value of the dimension is 3.

Problem 2. *Solution:*

1. For $c = \{1, 2\}$:
$$\mathcal{H}_c = \{(0,0), (0,1), (1,1), (1,0)\}$$
For $c = \{1, 2, 3\}$:
$$\mathcal{H}_c = \{(0,0,0), (1,0,0), (1,1,0), (1,1,1), (0,0,1), (0,1,1), (0,1,0)\}$$
$(1, 0, 1)$ is not possible.
2. Let $C_1 = \{1, 2\}$. $|\mathcal{H}_c| = 4$.
Let $C_2 = \{1, 2, 3\}$. $|\mathcal{H}_c| = 7 < 8$,
3. For $|C| = 2, \mathcal{H}_\mathcal{C}$ shatters C since $|\mathcal{H}_\mathcal{C}| = 4 = 2^{|C|}$.
For $|C| = 3, \mathcal{H}_\mathcal{C}$ does not shatter C since $|\mathcal{H}_\mathcal{C}| < 2^{|C|}$.
4. $VCdim(\mathcal{H}) = 2$

Problem 3. *Solution:*

(a) If a hypothesis class $\mathcal{H}$ is PAC learnable then it means there is a hypothesis h that yields a true error lower than ϵ with probability $1 - \delta$.

(b) $VCdim(\mathcal{H}) = \infty$.

(c) We show that $\text{VCdim}(\mathcal{H}) \leq \log_2 m$ as follows:

Let $\text{VCdim}(\mathcal{H}) = c$.

We know that $|\mathcal{H}| = m \geq |\mathcal{H}_\mathcal{C}|$ and that $|\mathcal{H}_\mathcal{C}| = 2^c$.

Thus:

$$m \geq |\mathcal{H}_\mathcal{C}|$$
$$m \geq 2^c$$
$$m \geq 2^{VCdim(\mathcal{H})}$$

And after taking log of both sides:

$$\log_2 m \geq VCdim(\mathcal{H})$$