Advanced ML assignment 1

Md Mahim Anjum Haque

February 2022

1 Problem 1

1.1 a

$$E[g(x)] = \sum_x p(x)g(x)$$

1.2 b

$$Var[X] = E[(X - E(X))^2]$$
$$Var[X] = E[(X^2 - 2XE(X) + E(X)^2]$$
$$Var[X] = E(X^2) - 2E(X)E(X) + E(X)^2$$
$$Var[X] = E(X^2) - E(X)^2$$

1.3 c

$$Cov[X, Y] = E[(X - E(X))(Y - E[Y])]$$
$$Cov[X, Y] = E[XY - YE[X] - XE[Y] + E(X)E(Y)]$$
$$Cov[X, Y] = E[XY] - E[Y]E[X] - E[X]E[Y] + E(X)E(Y)$$
$$Cov[X, Y] = E[XY] - E[X]E[Y]$$

Expectation of expectation is just an expectation because after the first expectation it's just a constant

1.4 d

We know that $Var[X] = E(X^2) - E(X)^2$ So,

$$Var[X + Y] = E[(X + Y)^2] - E(X + Y)^2$$
$$Var[X + Y] = E[X^2] + E[Y^2] + 2E[XY] - E[X + Y]^2$$
$$Var[X + Y] = E[X^2] + E[Y^2] + 2E[XY] - E[X]^2 - E[y]^2 - 2E[X]E[Y]$$
$$Var[X + Y] = E[X^2] - E[X]^2 + E[Y^2] - E[y]^2 + 2E[XY] - 2E[X]E[Y]$$
$$Var[X + Y] = Var[X] + Var[Y] + Cov[X, y]$$

2 2

2.1 a

$$P(Size = Small|Color = Green) = \frac{P(Size = Small, Color = Green)}{P(Color = Green)} = 2/3$$
$$P(Color = Red|Shape = Circle) = \frac{P(Color = Red, Shape = Circle)}{P(Shape = Circle)} = 2/4 = 1/2$$
$$P(Shape = Irregular|Size = Large) = \frac{P(Shape = Irregular, Size = Large)}{P(Size = Large)} = 1/3$$

2.2 b

$$P(Size = Small|Color = Red) = \frac{P(Color = Red|Size = Small) \times p(Size = Small)}{P(Color = Red)}$$
$$P(Color = Red|Size = Small) = 2/4 = 1/2$$
$$P(Size = Small) = 4/7$$
$$P(Color = Red) = 4/7$$
$$P(Size = Small|Color = Red) = \frac{(1/2) * (4/7)}{4/7} = 1/2$$

2.3 c

2.3.1 1

Mean of living area = (2104+1600+2400+1416+3000)/5 = 2104
Median of living area = 2104
Mean for price = (400+330+369+232+540)/5 = 374.2
Median for price = 369

2.3.2 2

Sample Variance of living area = 404448.0
Population Variance of living area = 323558.4
sample Variance of living price = 12589.2
Population Variance of living price = 10071.36

2.4 d

population covariance between living area and price = 53425.6

3 3

3.1 a

3.1.1 1

$$P(k|\lambda) = \frac{\lambda^k e^{-\lambda}}{k!} Likelihoodfunctionforthisis$$

$$L(\lambda; x_1, x_2, ..x_n) = log(\prod_{j=1}^{n} \frac{\lambda^{x_j} e^{-\lambda}}{x_j!})$$

$$= -n\lambda + log(\lambda) \sum_{j=1}^{n} x_j - \sum_{j=1}^{n} log(x_j!)$$

$$\frac{d}{d\lambda} L(\lambda; x_1, x_2, ..x_n) = -n + \frac{1}{\lambda} \sum_{j=1}^{n} x_j$$

$$\frac{d}{d\lambda} L(\lambda; x_1, x_2, ..x_n) = 0$$

$$-n + \frac{1}{\lambda} \sum_{j=1}^{n} x_j = 0$$

$$\lambda = \frac{\sum_{j=1}^{n} x_j}{n}$$

3.1.2 2

$$\sum_{k=1}^{n} \frac{\lambda^k}{k!} * k = e^{-\lambda} \sum_{k=1}^{n} \frac{\lambda^k}{k!} * k = e^{-\lambda} \sum_{k=1}^{n} \frac{\lambda^k}{(k-1)!}$$

$$= \lambda e^{-\lambda} \sum_{k=1}^{n} \frac{\lambda^{k-1}}{(k-1)!} = \lambda e^{-\lambda} * e^{\lambda} = \lambda$$

as n tends to infinity this is exactly as λ.

3.2 b

$$L(p, x_1, x_2, ...x_n) = \prod_{i=1}^{n} p(x_i)(1-p)^{(1-x_i)}$$

$$l(p) = log(p)\sum_{i=1}^{n} x_i + log(1-p)\sum_{i=1}^{n}(1-x_i)$$

$$\frac{d}{dp}(l(p)) = \frac{1}{p}\sum_{i=1}^{n} x_i - \frac{1}{1-p}\sum_{i=1}^{n}(1-x_i) = 0$$

$$\sum_{i=1}^{n} x_i - p\sum_{i=1}^{n} x_i = p\sum_{i=1}^{n}(1-x_i)$$

$$\sum_{i=1}^{n} x_i = pn$$

$$p = \frac{1}{n}\sum_{i=1}^{n} x_i$$

4 4

4.1 a

Sample Variance of living area = 404448.0
Sample STD of living area = $\sqrt{404448.0} = 635.96$
sample Variance of living price = 12589.2
sample STD of living price = $\sqrt{12589.2} = 112.20$
Standardized living area features = 0.,-0.79,0.47,-1.08, 1.41
Standardized living price features =0.23, -0.39,-0.05,-1.27, 1.48

4.2 b

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(h_\theta(x^i) - y^i)^2$$

4.3 c

The way I understand it the other choices we have for a loss function is simply the difference (between label and output) or the absolute difference or the cube of the difference.
The cube doesn't make sense that's unnecessarily complicated. as we can get the sampe thing with just the difference.
The difference and absolute difference as the loss function make sense but the problem is with just difference as The normal error it can be either positive or negative for each of the samples. If we sum up some positive and some negative

numbers, we may get 0. When the points are evenly distributed around the regression line, no matter how far away, the normal error may be 0. So this creates a problem. the error is zero when both side contribute same amount to the error.
Absolute value comes to the rescue here but the problems is multiple combination can contribute to the same absolute error value there is no fixed answer so this is is problem for both numerical and analytical methods and if we talk about gradient decent the differentiation of an absolute value funtion is tricky and we need to make many assumptions on the non continues points otherwise it's not even differentiable.
Squared value solves all these problems also guranteeing a single minima. It's not necessarily perfect(As we are guiding the model based on square of distance, one single outlier far away can disrupt the whole linear regression line which is a huge problem. The line will shift just for that outlier without caring for all other valid samples that it's not now a better fit) but it's the best we have based on complexity and computational feasibility.

4.4 d

$$h_\theta = \theta^T X$$

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(h_\theta(x^i) - y^i)^2$$

$$\frac{d}{d\theta}\sum_{i=1}^{n}(h_\theta(x^i) - y^i)^2 = \frac{2}{n}\sum_{i=1}^{n}(h_\theta(x^i) - y^i)\frac{d}{d\theta}h_\theta(x^i)$$

so for a perticular θ_j, $\frac{d}{d\theta_j}(\theta_j^T x_j) = x_j$ so we get

$$\frac{d}{d\theta_j}\sum_{i=1}^{n}(h_\theta(x^i) - y^i)^2 = \frac{2}{n}\sum_{i=1}^{n}(h_\theta(x^i) - y^i)x_j^i$$

4.5 e

so we got the gradient term for each of the weights. To minimize the loss we need to update the weights. We can check that if we subtract the gradient term we get to the minima.
New weight = old weight - α * gradient WRT that weight

$$\theta_j = \theta_j - \alpha * \frac{2}{n}\sum_{i=1}^{n}(h_\theta(x^i) - y^i)x_j^i$$

which is called batch gradient decent and computationally expensive as for each step we need to iterate over the whole training set. We can use a computational friendly version of this same formula by just sampling m samples and using the same formula averaged over m. This way the gradients are not going to be exactly as the true gradient values but after a few iterations this converges to the true values as well.

4.6 f

4.6.1 1

this is basically a line. Which seems won't fit the data good enough. This model has a very high bias and a very low variance. SO both poor performance on training and testing dataset will be observed.

4.6.2 2

This model seems very complicated with a lot of variables to tune which seems like an overkill for a simple problem. So this model willl have a very low bias and very high variance which can result in overfitting and leading to poor score in the testing dataset while doing very good on training dataset.