Homework 2 - Math 4803

Dennis Goldenberg

February 2023

1 Theoretical Questions

1. (Poisson Process and finding the Posterior)

a)

Proof. We use Bayes' rule first and note that:

$$\mathbb{P}(\lambda|y) = \frac{\mathbb{P}(y|\lambda)\mathbb{P}(\lambda)}{\mathbb{P}(y)} \propto \mathbb{P}(y|\lambda)\mathbb{P}(\lambda)$$

Note that we can confirm this proportionality from the fact that $\mathbb{P}(\lambda|y)$ is a function of λ and not y; therefore $\mathbb{P}(y)$ serves as a normalizing constant and doesn't really matter for our calculations. We can continue to use proportionality to our advantage. Note that:

$$\mathbb{P}(\lambda) = \frac{\lambda^{\alpha-1}e^{-\beta\lambda}\beta^{\alpha}}{\Gamma(\alpha)} \propto \lambda^{\alpha-1}e^{-\beta\lambda}$$

This is due to the fact that $\mathbb{P}(\lambda)$ is a function of λ and not α or β. So, our goal is to show that $\mathbb{P}(\lambda|y) \propto \lambda^{\alpha'-1}e^{-\beta'\lambda}$, and we will have shown that the Gamma distribution is indeed a conjugate prior for the Poisson distribution. We calculate:

$$\begin{aligned}
\mathbb{P}(\lambda|y) &\propto \mathbb{P}(y|\lambda)\mathbb{P}(\lambda) \\
&= \frac{\lambda^{y}e^{-\lambda}}{y!} * \frac{\lambda^{\alpha-1}e^{-\beta\lambda}\beta^{\alpha}}{\Gamma(\alpha)} \\
&\propto \frac{\lambda^{y}e^{-\lambda}}{y!} * \lambda^{\alpha-1}e^{-\beta\lambda} \\
&\propto \lambda^{y}e^{-\lambda} * \lambda^{\alpha-1}e^{-\beta\lambda} \\
&= \lambda^{y+\alpha-1}e^{-\lambda(\beta+1)} \\
&= \lambda^{\alpha'-1}e^{-\beta'\lambda}
\end{aligned}$$

Here, $\alpha' = \alpha + y$ and $\beta' = \beta + 1$. Thus, we have arrived at our original goal, as we have shown the posterior to be proportional to a Gamma distribution; the Gamma distribution is therefore a conjugate prior. □

b) We are predicting an outcome after observing y; therefore, we know y. However, since z is from the same Poisson distribution, it is dependent on λ ($\mathbb{P}(z|\lambda)$) but the true λ is unknown, so this must be taken into account. Fortunately, we computed the posterior distribution of λ once y is observed. So, we know the following equations:

$$\mathbb{P}(z|\lambda) \text{ and } \mathbb{P}(\lambda|y)$$

It is tempting to use an estimate for λ given y, call it $\hat{\lambda}$, such as a maximum likelihood estimate, to develop a probability guess for z where $\mathbb{P}(z|y) = \mathbb{P}(z|\hat{\lambda})$ and then plug that $\hat{\lambda}$ into the Poisson distribution to get our probability. However, this does not take into account the uncertainty of λ, and assumes that λ can be perfectly determined from the one observation y, a faulty assumption. Therefore, to get the true probability of z, we must account for all possible λ values (given that λ follows a Gamma distribution, this would be all real values from 0 to ∞) and, since we have the posterior distribution of λ given y, we can use that updated information and integrate over that range to find the total probability of z given y:

$$\begin{aligned}
\mathbb{P}(z|y) &= \int_0^\infty \mathbb{P}(z|\lambda, y)\mathbb{P}(\lambda|y)d\lambda \\
&= \int_0^\infty \mathbb{P}(z|\lambda)\mathbb{P}(\lambda|y)d\lambda \text{ (because z is conditionally independent of y)}
\end{aligned}$$

This is exactly the posterior distribution indicated. When we plug our known values to solve:

$$\begin{aligned}
\mathbb{P}(z|y) &= \int_0^\infty \mathbb{P}(z|\lambda)\mathbb{P}(\lambda|y)d\lambda \\
&= \int_0^\infty \frac{\lambda^z e^{-\lambda}}{z!} * \frac{\lambda^{\alpha'-1}e^{-\beta'\lambda}\beta'^{\alpha'}}{\Gamma(\alpha')}d\lambda \\
&= \frac{\beta'^{\alpha'}}{z!\Gamma(\alpha')}\int_0^\infty \lambda^z e^{-\lambda} * \lambda^{\alpha'-1}e^{-\beta'\lambda}d\lambda \\
&= \frac{\beta'^{\alpha'}}{z!\Gamma(\alpha')}\int_0^\infty \lambda^{z+\alpha'-1}e^{-(\beta'+1)\lambda}d\lambda \\
&= \frac{\beta'^{\alpha'}}{z!\Gamma(\alpha')}\int_0^\infty \frac{\Gamma(\alpha'+z)}{(\beta'+1)^{\alpha'+z}} * \frac{\lambda^{z+\alpha'-1}e^{-(\beta'+1)\lambda}(\beta'+1)^{\alpha'+z}}{\Gamma(\alpha'+z)}d\lambda \\
&= \frac{\beta'^{\alpha'}\Gamma(\alpha'+z)}{z!\Gamma(\alpha')(\beta'+1)^{\alpha'+z}}\int_0^\infty \frac{\lambda^{z+\alpha'-1}e^{-(\beta'+1)\lambda}(\beta'+1)^{\alpha'+z}}{\Gamma(\alpha'+z)}d\lambda \\
&= \frac{\beta'^{\alpha'}\Gamma(\alpha'+z)}{z!\Gamma(\alpha')(\beta'+1)^{\alpha'+z}} \text{ (because integrand was just gamma distribution)}
\end{aligned}$$

Note that, due to properties of the gamma function:

$$\frac{\Gamma(\alpha'+z)}{\Gamma(\alpha')} = \frac{(\alpha'+z-1)!}{(\alpha'-1)!}$$

So, we continue towards the solution:

$$
\begin{aligned}
\mathbb{P}(z|y) &= \frac{\beta'^{\alpha'}\Gamma(\alpha'+z)}{z!\Gamma(\alpha')(\beta'+1)^{\alpha'+z}} \\
&= \frac{(\alpha'+z-1)!}{(\alpha'-1)z!} * \frac{\beta'^{\alpha'}}{(\beta'+1)^{\alpha'+z}} \\
&= \binom{z+\alpha'-1}{z} * \frac{1}{(\beta'+1)^z} * \left(\frac{\beta}{\beta+1}\right)^{\alpha'} \\
&= \binom{z+\alpha'-1}{z} * \left(\frac{1}{\beta'+1}\right)^z * \left(\frac{\beta'}{\beta'+1}\right)^{\alpha'} \\
&= \text{Negative Binomial}\left(\alpha', \frac{\beta'}{\beta'+1}\right)
\end{aligned}
$$

2. Section 4.8 Exercise 6

a) Encode the response variable Y as follows:

$$Y = \begin{cases} 1 & \text{if student receives an A} \\ 0 & \text{otherwise} \end{cases}$$

Thus, given our generated estimates for the beta coefficients, we can generate the following model for logistic regression:

$$\mathbb{P}(Y_i = 1) = \frac{e^{-6+.05(\text{hours studied})_i+1(\text{undergrad GPA})_i}}{1 + e^{-6+.05(\text{hours studied})_i+1(\text{undergrad GPA})_i}}$$

So, for a student with 40 hours studied and a 3.5 GPA:

$$\mathbb{P}(Y_i = 1) = \frac{e^{-6+.05*40+3.5}}{1 + e^{-6+.05*40+3.5}} = .37754 = 37.754\%$$

That student has a 37.754% chance of obtaining an A according to the model.

b) For the student to have a 50% chance of getting an A in the class ($\mathbb{P}(Y_i = 1) = .5$) with the fixed GPA at 3.5, we can plug in these values to solve for $(\text{hours studied})_i$:

$$\begin{aligned}
.5 &= \frac{e^{-6+.05(\text{hours studied})_i+3.5}}{1 + e^{-6+.05(\text{hours studied})_i+3.5}} \\
&\rightarrow .5 + .5e^{-6+.05(\text{hours studied})_i+3.5} = e^{-6+.05(\text{hours studied})_i+3.5} \\
&\rightarrow .5 = .5e^{-6+.05(\text{hours studied})_i+3.5} \\
&\rightarrow 1 = e^{-6+.05(\text{hours studied})_i+3.5} \\
&\rightarrow \ln(1) = \ln(e^{-6+.05(\text{hours studied})_i+3.5}) \\
&\rightarrow 0 = -6 + .05(\text{hours studied})_i + 3.5 \\
&\rightarrow (\text{hours studied})_i = \frac{6 - 3.5}{.05} = 50
\end{aligned}$$

So such a student would have to study for **50** hours.

3. Section 4.8 Exercise 7

We start by noting our mean and variance assumptions. Note that Y, or the response variable on whether a stock issued a dividend or not, is dependent on one variable, X, or profit. The mean profit is different for those who issued versus those who didn't, but the variance is the same. Given equations (4.20 and 4.21) in the textbook "An Introduction to Statistical Learning", we can use the information in the question as estimates for mean and variance parameters, as well as the prior probability of issuance parameters:

$$\begin{aligned}
&\widehat{\mu_{\text{Issued}}} = \bar{X}_{\text{Issued}} = 10 \\
&\widehat{\mu_{\text{Not Issued}}} = \bar{X}_{\text{Not Issued}} = 0 \\
&\widehat{\sigma^2_{\text{Issued}}} = \widehat{\sigma^2_{\text{Not Issued}}} = 36 = \widehat{\sigma^2} \\
&\widehat{\pi_{\text{Issued}}} = .8 \\
&\widehat{\pi_{\text{Not Issued}}} = 1 - .8 = .2
\end{aligned}$$

Using Bayes' rule (equation 4.17) and assuming the X's are normally distributed in both cases, we find the probability:

$$\begin{aligned}
&\mathbb{P}(Y = \text{Issued} | X = 4) \\
&= \frac{\widehat{\pi_{\text{Issued}}} * \frac{1}{\sqrt{2\pi\widehat{\sigma^2}}} * e^{\frac{-1}{2} * \frac{(x - \widehat{\mu_{\text{Issued}}})^2}{\widehat{\sigma^2}}}}{\widehat{\pi_{\text{Issued}}} * \frac{1}{\sqrt{2\pi\widehat{\sigma^2}}} * e^{\frac{-1}{2} * \frac{(x - \widehat{\mu_{\text{Issued}}})^2}{\widehat{\sigma^2}}} + \widehat{\pi_{\text{Not Issued}}} * \frac{1}{\sqrt{2\pi\widehat{\sigma^2}}} * e^{\frac{-1}{2} * \frac{(x - \widehat{\mu_{\text{Not Issued}}})^2}{\widehat{\sigma^2}}}} \\
&= \frac{.8 * \frac{1}{\sqrt{2\pi * 36}} * e^{\frac{-1}{2} * \frac{(4-10)^2}{36}}}{.8 * \frac{1}{\sqrt{2\pi * 36}} * e^{\frac{-1}{2} * \frac{(4-10)^2}{36}} + .2 * \frac{1}{\sqrt{2\pi * 36}} * e^{\frac{-1}{2} * \frac{4^2}{36}}} \\
&= \frac{.8e^{\frac{-1}{2}}}{.8e^{\frac{-1}{2}} + .2e^{\frac{-16}{72}}} \\
&= \frac{.8e^{\frac{-1}{2}}}{.8e^{\frac{-1}{2}} + .2e^{\frac{-2}{9}}} \\
&= 0.752
\end{aligned}$$

So the dividend has roughly a **75.2%** chance of being issued given a percent profit of 4.

4. Section 4.8 Exercise 8

We should still use **logistic regression**, even though the average error between testing and training datasets is higher (25% as opposed to 18%). Notice that the individual values for error rate for training and testing data sets when 1-nearest neighbors were used was omitted. This is to hide the fact that, when K is very low, there is a very high likelihood of overfitting, or having way too flexible of a boundary, as each prediction only takes into account the closest neighbor causing erratic variations in prediction even within a small area. Thus, it is likely that the training error is extremely low, due to the flexibility of such a model, and the testing error is very high, as the testing data is different from the training data and the model is built to change prediction given just one datapoint seemingly out of place. Thus, logistic regression is likely to be more accurate given a significant number of random datapoints.

5. Section 4.8 Exercise 12

a) Using Equation (4.4), we find that the log odds are as follows for my model:

$$\log\left(\frac{p(Y=\text{orange}|x)}{p(Y=\text{apple}|x)}\right) = \log\left(\frac{p(Y=\text{orange}|x)}{1-p(Y=\text{orange}|x)}\right) = \hat{\beta}_0 + \hat{\beta}_1 x$$

b) Using Equation (4.14) in the textbook, we find:

$$\begin{aligned}\log\left(\frac{p(Y=\text{orange}|x)}{p(Y=\text{apple}|x)}\right) &= (\hat{\beta}_{\text{orange0}} - \hat{\beta}_{\text{apple0}}) + (\hat{\beta}_{\text{orange1}} - \hat{\beta}_{\text{apple1}})x \\ &= (\hat{\alpha}_{\text{orange0}} - \hat{\alpha}_{\text{apple0}}) + (\hat{\alpha}_{\text{orange1}} - \hat{\alpha}_{\text{apple1}})x\end{aligned}$$

c) Our estimates are $\hat{\beta}_0 = 2$ and $\hat{\beta}_1 = -1$. In softmax coding, we do not select for a baseline class; rather, we generate coefficients for all classes. For the softmax coding to be consistent with ours regarding the relationship between the orange and apple classes, then, what we had as $\hat{\beta}_1$, our friend has as $\hat{\alpha}_{\text{orange1}} - \hat{\alpha}_{\text{apple1}}$ because both classes now have coefficients; therefore:

$$\hat{\alpha}_{\text{orange1}} - \hat{\alpha}_{\text{apple1}} = -1$$

The intercept should work similarly, as this also applies for β_0:

$$\hat{\alpha}_{\text{orange0}} - \hat{\alpha}_{\text{apple0}} = 2$$

The order is so because "apple" was picked to be the baseline class.

d) The relationship stays the same, though now we have the $\hat{\alpha}$ values. So, we can calculate the β values from the α values:

$$\hat{\beta}_1 = \hat{\alpha}_{\text{orange1}} - \hat{\alpha}_{\text{apple1}} = -2 - 0.6 = -2.6$$

The relationship is similar for β_0:

$$\hat{\beta}_0 = \hat{\alpha}_{\text{orange0}} - \hat{\alpha}_{\text{apple0}} = 1.2 - 3 = -1.8$$

$\hat{\beta}_0 = -1.8$ and $\hat{\beta}_1 = -2.6$ would be our estimated parameters using logistic regression.

e) Notice what happends when we the formulas for $\hat{\beta}_0$ and $\hat{\beta}_1$ into our model:

$$\begin{aligned}\mathbb{P}(Y=\text{orange}|X=x) &= \frac{e^{\hat{\beta}_0+\hat{\beta}_1 x}}{1+e^{\hat{\beta}_0+\hat{\beta}_1 x}} \\ &= \frac{e^{\hat{\alpha}_{\text{orange0}}-\hat{\alpha}_{\text{apple0}}+(\hat{\alpha}_{\text{orange1}}-\hat{\alpha}_{\text{apple1}})x}}{1+e^{\hat{\alpha}_{\text{orange0}}-\hat{\alpha}_{\text{apple0}}+(\hat{\alpha}_{\text{orange1}}-\hat{\alpha}_{\text{apple1}})x}} \\ &= \frac{e^{\hat{\alpha}_{\text{orange0}}-\hat{\alpha}_{\text{apple0}}+(\hat{\alpha}_{\text{orange1}}-\hat{\alpha}_{\text{apple1}})x}}{1+e^{\hat{\alpha}_{\text{orange0}}-\hat{\alpha}_{\text{apple0}}+(\hat{\alpha}_{\text{orange1}}-\hat{\alpha}_{\text{apple1}})x}} * \frac{e^{\hat{\alpha}_{\text{apple0}}+\hat{\alpha}_{\text{apple1}}x}}{e^{\hat{\alpha}_{\text{apple0}}+\hat{\alpha}_{\text{apple1}}x}} \\ &= \frac{e^{\hat{\alpha}_{\text{orange0}}+\hat{\alpha}_{\text{orange1}}x}}{e^{\hat{\alpha}_{\text{apple0}}+\hat{\alpha}_{\text{apple1}}x}+e^{\hat{\alpha}_{\text{orange0}}+\hat{\alpha}_{\text{orange1}}x}}\end{aligned}$$

This model is exactly our friends model, so due to this relationship between the α's and β's, these models would predict the same exact probability. Thus, we expect them to have the same predicted label **100%** of the time.

6. Exercise 3.12 (From "Elements of Statistical Learning")

Proof. We start by showing what the least squares solution for the betas is to the optimization problem:

$$\hat{\beta}_{LSS} = (X^T X)^{-1} X^T y$$

Next, we lay out the solution to ridge regression:

$$\hat{\beta}_{Ridge} = (X^T X + \lambda I)^{-1} X^T y$$

Let's call the augmented X matrix X', and the augmented y matrix y'. Since X is an $n \times p$ matrix, X' will have a $n + p \times p$ dimensionality. Similarly, y' will have a $n + p \times 1$ dimensionality. Note that, since the extra p values added to the y are just of value 0:

$$\begin{aligned}
X'^T y' &= \begin{bmatrix} x_{1,1} & x_{2,1} & x_{3,1} & ... & x_{n,1} & \sqrt{\lambda} & 0 & 0 & ... & 0 \\ x_{1,2} & x_{2,2} & x_{3,2} & ... & x_{n,2} & 0 & \sqrt{\lambda} & 0 & ... & 0 \\ x_{1,3} & x_{2,3} & x_{3,3} & ... & x_{n,3} & 0 & 0 & \sqrt{\lambda} & ... & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{1,p} & x_{2,p} & x_{3,p} & ... & x_{n,p} & 0 & 0 & 0 & ... & \sqrt{\lambda} \end{bmatrix} * \begin{bmatrix} y_1 \\ \vdots \\ y_n \\ 0 \\ \vdots \end{bmatrix} \\
&= \begin{bmatrix} \sum_{i=1}^n y_i x_{1,i} + 0\sqrt{\lambda} \\ \sum_{i=1}^n y_i x_{2,i} + 0\sqrt{\lambda} \\ \sum_{i=1}^n y_i x_{3,i} + 0\sqrt{\lambda} \\ \vdots \\ \sum_{i=1}^n y_i x_{p,i} + 0\sqrt{\lambda} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n y_i x_{1,i} \\ \sum_{i=1}^n y_i x_{2,i} \\ \sum_{i=1}^n y_i x_{3,i} \\ \vdots \\ \sum_{i=1}^n y_i x_{p,i} \end{bmatrix} \\
&= X^T y
\end{aligned}$$

Next, we examine the term inside of the inverse. We calculate $X'^T X'$ (on next page):

$$\begin{bmatrix} x_{1,1} & x_{2,1} & x_{3,1} & ... & x_{n,1} & \sqrt{\lambda} & 0 & 0 & ... & 0 \\ x_{1,2} & x_{2,2} & x_{3,2} & ... & x_{n,2} & 0 & \sqrt{\lambda} & 0 & ... & 0 \\ x_{1,3} & x_{2,3} & x_{3,3} & ... & x_{n,3} & 0 & 0 & \sqrt{\lambda} & ... & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{1,p} & x_{2,p} & x_{3,p} & ... & x_{n,p} & 0 & 0 & 0 & ... & \sqrt{\lambda} \end{bmatrix} * \begin{bmatrix} x_{1,1} & x_{1,2} & x_{1,3} & ... & x_{1,p} \\ x_{2,1} & x_{2,2} & x_{2,3} & ... & x_{2,p} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{n,1} & x_{n,2} & x_{n,3} & .... & x_{n,p} \\ \sqrt{\lambda} & 0 & 0 & ... & 0 \\ 0 & \sqrt{\lambda} & 0 & ... & 0 \\ & & & & \\ 0 & 0 & \ddots & ... & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \sqrt{\lambda} \end{bmatrix}$$

$$= \begin{bmatrix} \sum_{i=1}^n x_{i,1}x_{i,1} + \lambda & \sum_{i=1}^n x_{i,1}x_{i,2} + 0 & ... & \sum_{i=1}^n x_{i,1}x_{i,p} + 0 \\ \sum_{i=1}^n x_{i,2}x_{i,1} + 0 & \sum_{i=1}^n x_{i,2}x_{i,2} + \lambda & ... & \sum_{i=1}^n x_{i,2}x_{i,p} + 0 \\ \vdots & \vdots & \vdots & \vdots \\ \sum_{i=1}^n x_{i,p}x_{i,1} + 0 & \sum_{i=1}^n x_{i,p}x_{i,2} + 0 & ... & \sum_{i=1}^n x_{i,p}x_{i,p} + \lambda \end{bmatrix}$$

$$= \begin{bmatrix} \sum_{i=1}^n x_{i,1}x_{i,1} & \sum_{i=1}^n x_{i,1}x_{i,2} & ... & \sum_{i=1}^n x_{i,1}x_{i,p} \\ \sum_{i=1}^n x_{i,2}x_{i,1} & \sum_{i=1}^n x_{i,2}x_{i,2} & ... & \sum_{i=1}^n x_{i,2}x_{i,p} \\ \vdots & \vdots & \vdots & \vdots \\ \sum_{i=1}^n x_{i,p}x_{i,1} & \sum_{i=1}^n x_{i,p}x_{i,2} & ... & \sum_{i=1}^n x_{i,p}x_{i,p} \end{bmatrix} + \begin{bmatrix} \lambda & 0 & ... & 0 \\ 0 & \lambda & ... & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & ... & \lambda \end{bmatrix}$$

$$= X^T X + \lambda I$$

Therefore, if we were to do the LSS of this augmented data:

$$\begin{aligned} \hat{\beta}_{\text{LSS Augemented}} &= (X'^T X')^{-1} X'^T y' \\ &= (X^T X + \lambda I)^{-1} X'^T y' \\ &= (X^T X + \lambda I) X^T y \\ &= \hat{\beta}_{Ridge} \end{aligned}$$

□

7. Exercise 3.28 (From "Elements of Statistical Learning")

Solution

Lasso Regression tends to shrink coefficients by providing harsh penalties for large coefficients, that being the $\lambda \sum_{j=1}^{p} |\beta_j|$ term. Here is the full expanded lasso regression minimization problem:

$$\text{argmin}_\beta \left\{ \frac{1}{2} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{l=1}^{p} x_{il}\beta_l \right)^2 + \lambda \sum_{l=1}^{p} |\beta_l| \right\}$$

The first part of this minimization is simply least squares. If we know that the estimate without a cloned β_j is a for parameter β_j and the penalty term contains an absolute value, with the absolute value function is minimized at 0. This means that two things are true:

- Both β_j terms, combined, have an a effect on the outcome variable via lasso regression.
- The penalty term will be minimized with the minimum total absolute value between the two β_j parameters.

Note that in each element in the least squares summation has a squared β_l term for each β_l so it has a β_j^2 and β_j^{*2} term. The minimum value for those combined terms, with the constraint that $\beta_j + \beta_j^* = a$, is where one of those values is 0, and the other is a.Therefore, it will deal with the exact collinearity by shrinking one of the coefficients to 0 and acting as if there is only one coefficient; ergo:

$$\hat{\beta}_j, \hat{\beta}_j^* \in \{0, a\}$$

Here, one parameter will be given the value 0, and the other the value a. This lines up with the behavior of lasso regression, which has a tendency of setting unneeded coefficients to 0 if they don't improve the model enough.

8. Exercise 3.29 (From "Elements of Statistical Learning")

Proof. Unlike Lasso Regression, Ridge Regression has a closed form solution:

$$(X^T X + \lambda I)^{-1}(X^T y) = \hat{\beta}_{ridge}$$

Since we only have one explanatory variable X, we only have one parameter we are estimating, as, according to the textbook, due to the centering of parameters, the "remaining coefficients get estimated by a ridge regression without intercept."(Hastie, Tibshirani, Friedman, pp. 64). So this problem turns 1-dimensional, and the new equation is:

$$\hat{\beta}_{ridge} = \left(\sum_{i=1}^{n} x_i^2 + \lambda\right)^{-1} \left(\sum_{i=1}^{n} x_i y_i\right) = \frac{(\sum_{i=1}^{n} x_i y_i)}{\sum_{i=1}^{n} x_i^2 + \lambda} = a$$

Now, let us augment another vector $X^* = X$ to make this regression 2-dimensional. Then:

$$X_{new} = \begin{bmatrix} x_1 & x_1 \\ x_2 & x_2 \\ \vdots & \vdots \\ x_n & x_n \end{bmatrix}$$

If we run ridge regression in the 2D case, setting $\sum_{i=1}^{n} x_i^2 = s$ and $\sum_{i=1}^{n} x_i y_i = xy$ for convenience:

$$\begin{aligned}
\hat{\beta}_{ridge} &= (X_{new}^T X_{new} + \lambda I)^{-1} X_{new}^T y \\
&= \left(\begin{bmatrix} x_1 & x_2 & \dots & x_n \\ x_1 & x_2 & \dots & x_n \end{bmatrix} * \begin{bmatrix} x_1 & x_1 \\ x_2 & x_2 \\ \vdots & \vdots \\ x_n & x_n \end{bmatrix} + \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} \right)^{-1} X_{new}^T y \\
&= \left(\begin{bmatrix} s+\lambda & s \\ s & s+\lambda \end{bmatrix} \right)^{-1} * \begin{bmatrix} x_1 & x_2 & \dots & x_n \\ x_1 & x_2 & \dots & x_n \end{bmatrix} * \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \\
&= \left(\begin{bmatrix} s+\lambda & s \\ s & s+\lambda \end{bmatrix} \right)^{-1} * \begin{bmatrix} xy \\ xy \end{bmatrix}
\end{aligned}$$

We take the inverse of a 2 by 2 matrix:

$$\begin{aligned}
\left(\begin{bmatrix} s+\lambda & s \\ s & s+\lambda \end{bmatrix} \right)^{-1} &= \frac{1}{(s+\lambda)^2 - s^2} \begin{bmatrix} s+\lambda & -s \\ -s & s+\lambda \end{bmatrix} \\
&= \frac{1}{2s\lambda + \lambda^2} \begin{bmatrix} s+\lambda & -s \\ -s & s+\lambda \end{bmatrix}
\end{aligned}$$

Therefore, we have that:

$$\begin{aligned}
\hat{\beta}_{ridge} &= \frac{1}{2s\lambda + \lambda^2} \begin{bmatrix} s+\lambda & -s \\ -s & s+\lambda \end{bmatrix} * \begin{bmatrix} xy \\ xy \end{bmatrix} \\
&= \begin{bmatrix} \frac{xy\lambda}{2s\lambda+\lambda^2} \\ \frac{xy\lambda}{2s\lambda+\lambda^2} \end{bmatrix} \\
&= \begin{bmatrix} \frac{xy}{2s+\lambda} \\ \frac{xy}{2s+\lambda} \end{bmatrix} \\
&= \frac{s+\lambda}{2s+\lambda} \begin{bmatrix} \frac{xy}{s+\lambda} \\ \frac{xy}{s+\lambda} \end{bmatrix} \\
&= \frac{s+\lambda}{2s+\lambda} \begin{bmatrix} a \\ a \end{bmatrix}
\end{aligned}$$

So the coefficients **are the same** and each $\hat{\beta}_j = \frac{(s+\lambda)a}{2s+\lambda}$. Note that, in general with m variables are m copies of X, you are minimizing the problem:

$$\text{argmin}_\beta \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{m} \beta_j x_{ij} \right)^2 + \lambda \sum_{i=1}^{n} \beta_j^2 \right\}$$

Ignore all but 2 elements for a second; call them β_j and β_j^*. Note that this changes the problem to:

$$\text{argmin}_\beta \left\{ \sum_{i=1}^{n} \left(y_i - x_i(\beta_j + \beta_j^*) \right)^2 + \lambda \left(\beta_j^2 + \beta_j^{*2} \right) \right\}$$

This is because all values of X on different columns are just copies, so $x_{ij} = x_{ij^*} = x_{i1}$. Define new variables as follows:

$$\begin{aligned}
c &= \beta_j + \beta_j^* \\
d &= \beta_j - \beta_j^*
\end{aligned}$$

This changes the equation to the following:

$$\text{argmin}_{c,d} \left\{ \sum_{i=1}^{n} \left(y_i - x_i c \right)^2 + \frac{\lambda}{2} \left(c^2 + d^2 \right) \right\}$$

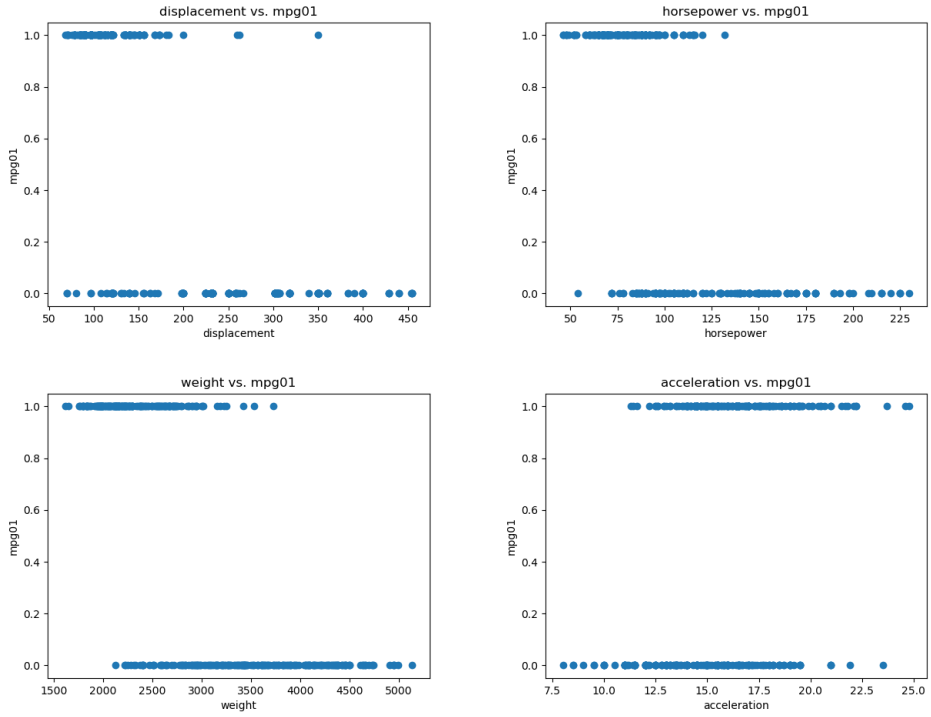
Note that the only term that contains d is the penalty term, and this can be minimized simply. As d^2 is always positive the minimum would be where $d = 0$. As a consequence $\beta_j - \beta_j^* = 0 \rightarrow \beta_j = \beta_j^*$. This can be easily extended to more than just two variables, as we know, for minimization, the difference between any two coefficients must be 0. Thus, the $\hat{\beta}$ values are all the same. □

2 Programming Problems

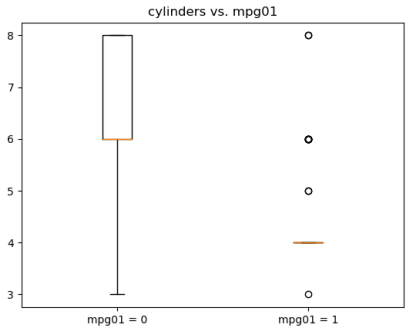
9. Section 4.8 Question 14
a) Done in `HW2_coding.py`
b) For this section, the columns of the data was broken down into two parts: Numeric and categorical. The breakdown was as follows:

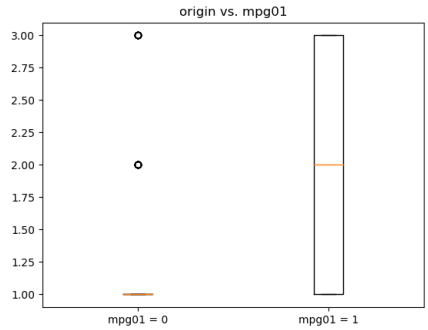
- Numeric: displacement, horsepower, weight, acceleration
- Categorical: cylinders, year, origin

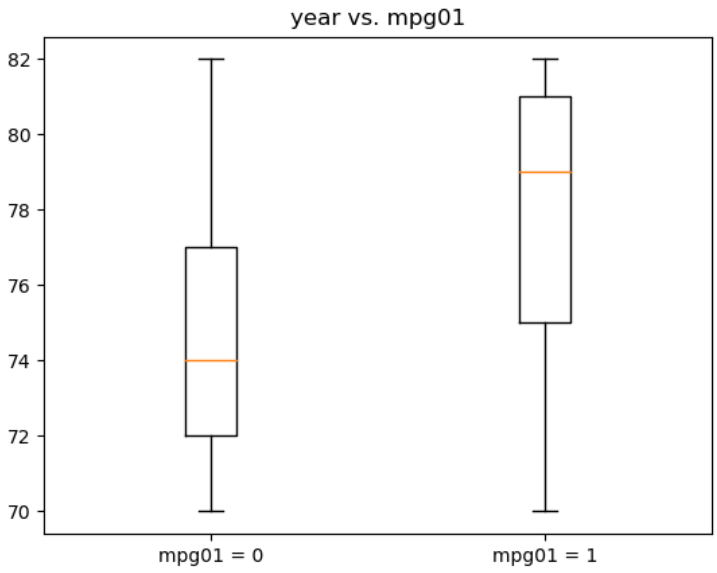
Note that the *name* and *mpg* columns were excluded as names are qualitative and mpg01 was based directly off of mpg. For the numeric data columns, scatterplots were run with the following results:



From first glance of the numeric values, it appears that **horsepower** and **weight** seem to have the strongest correlation, both being negative, though there is indication of a slight positive correlation with **acceleration**.
For the categorical data columns, we use box plots instead, arriving at the following box plots:







Based on the graphs, it seems that the strongest correlations with mpg01 are the **cylinders** and **year** variables.

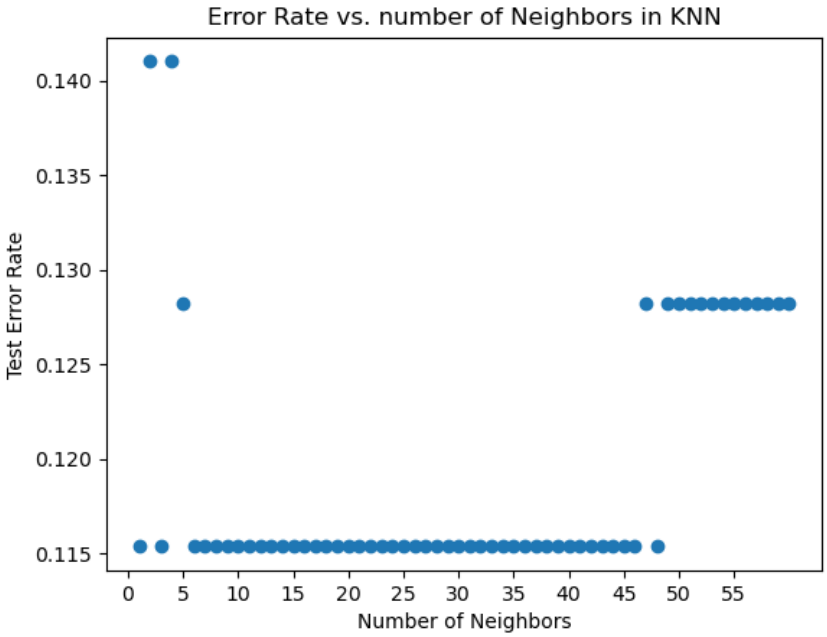
c) Done in `HW2_coding.py`; the split used was 80% training data, 20% testing data.

f) Logistic Regression was run using the variables *cylinder*, *horsepower*, *acceleration*, *weight*, and *year*. The test error rate returned was:

```
Error Rate for Logistic Regression:  0.08974358974358974
```

So the rate was roughly **0.09**.

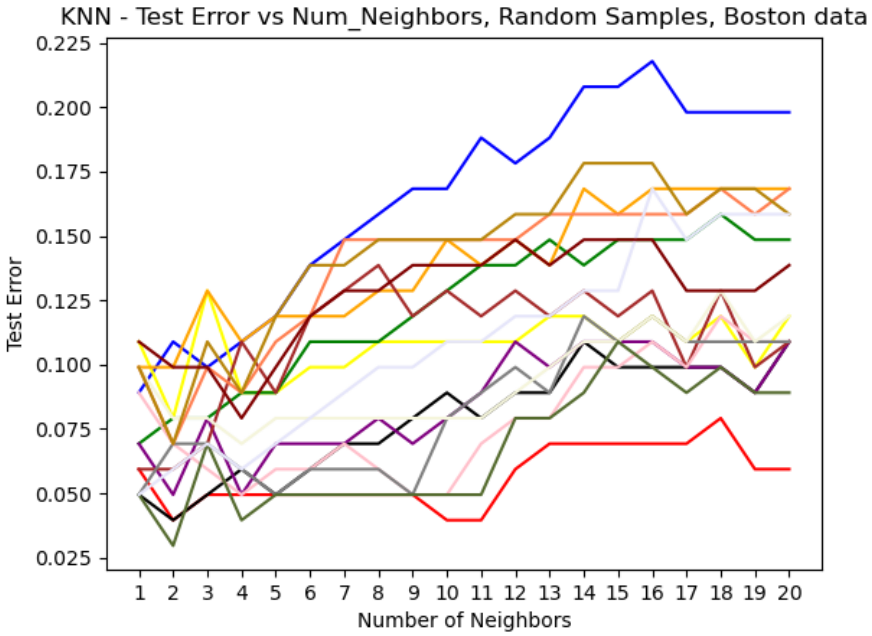
g) K-nearest neighbors was run using the variables *cylinder*, *horsepower*, *acceleration*, *weight*, and *year* for $K = 1$ to $K = 61$. The following graph comparing the value of the parameter k to the resultant test error rate was produced:



Test error rate seems to be minimized where k is **greater than 5 or less than or equal to 45**. The test error in general, even at its minimum, is slightly higher for KNN than for logistic regression.

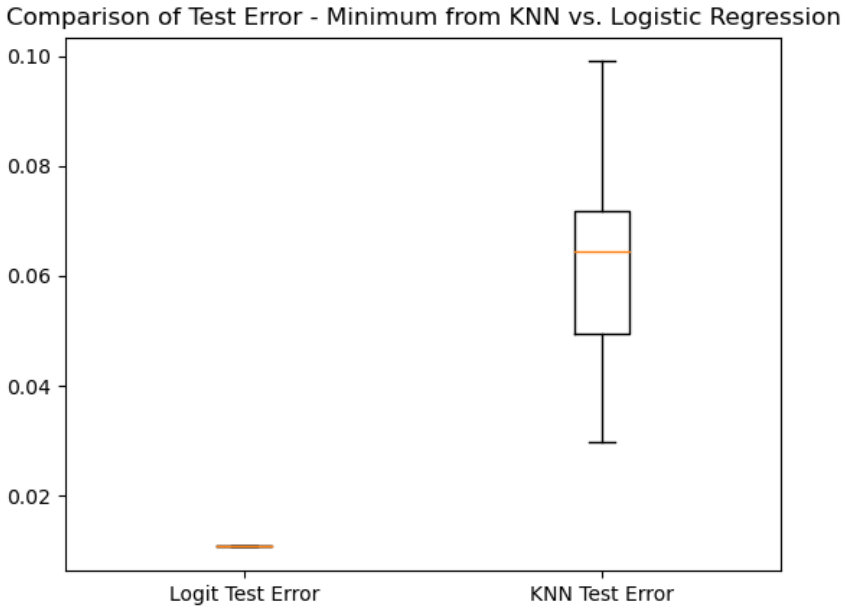
10. Section 4.8 Question 16

Similar to question 9, a variable *CRIM01* was created, which gave a value of 1 if the *CRIM* variable was above its median for that row and 0 if not. The same 80-20 train-test split was devised for this dataset and 16 random samples were genereated, with the predictor variables being all variables except for *CRIM* itself and the response being *CRIM01*. For each of the 16 random samples, KNN was run, where the number of neighbors parameter varied from 1 to 20 to see the performance, and the test error of these is shown below:



Each line represents a different one of these samples, and it is notable that the minimum test error for many sample is at $k = 2$ neighbors, with more neighbors worsening performance. This suggests non-linearity.

Logistic regression was also run, and the test error for each sample with logistic regression was compared to the minimum test error for KNN in the following boxplot:



Note that logistic regression was incredibly consistent and incredibly accurate, as it had the same error rate, hovering around 1%. However, the minimum for KNN both varied widely and on average performed worse. However, both methods seemed effective at predicting whether crime would be above or below the median.