HW2 STAT542

Wenke Huang

Question 1

1.1 Prepare the dataset

First, we consider seasonal patterns. We generate a new variable `weekend` which tells us whether a day is weekend. `weekend` and `holiday` together can give us the effect of weekend, holiday and their combo. What's more, we will also extract a new variable called `day` from variable `dteday`. We can find that variables `weekday` `workingday`, `instant` and `dteday` can be dropped because the other seasonal variables have contain the informations that these tell us.

Second, we discuss historical patterns. We notice that values of `casual` plus values of `registered` will be exactly the same as values of `cnt`, and the definition of `cnt` tells us the same story. In this case, we consider $log(cnt + 1)$ to be a continuous variable even though cnt is an discrete integer one. Thus, we treat $log(cnt + 1)$ as the response directly. At the same time, the LOG transformation can also solve our problem of multicollinearity.

Last, we convert some variables into factor ones. All of the predictors in this case and their corresponding variable type are shown below.

Predictors	season	yr	mnth	hr	holiday
Type	factor	factor	factor	factor	factor

Predictors	weathersit	temp	atemp	hum	windspeed
Type	factor	numeric	numeric	numeric	numeric

Predictors	day	weekend	casual	registered
Type	factor	factor	numeric	numeric

1.2 Models Fitting

In this part, we will use several methods to fit models to predict the total counts of bike rental. Some of them may contain tuning parameters, which means we may fit a bunch of models when trying one method. Here, we will only use the train dataset to fit the models, and only use the test dataset to calculate RMSE $= \sqrt{\frac{1}{n}\sum_{i=1}^{n}(log(\hat{y}_i - 1) - log(y_i - 1))^2}$. The

model with the smallest RMSE of one particular method will be considered the best model of this method in this case.

1.2.1 Linear regression

At the very beginning, we fit a linear regression model which contains all possible predictors we have chosen in the previous part. We get the coefficients $\hat{\beta}^{ols}$ from the function $\hat{\beta}^{ols} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$. By the Gauss-Markov Theorem, we know that this estimation is the best linear unbiased estimator (BLUE).

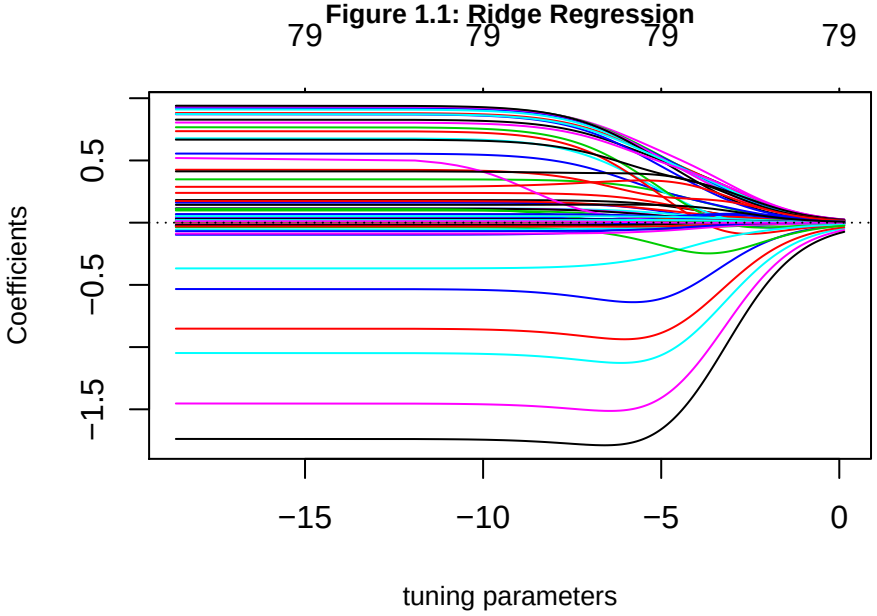
After calculating, we get the training and testing RMSE of linear regression are 0.47121 and 0.4721885, respectively.

1.2.2 RIDGE regression

Ridge regression penalize the coefficients $\hat{\beta}^{ridge}$ from the function $\hat{\beta}^{ridge} = \arg\min_{\beta}||\mathbf{y} - \mathbf{X}\beta||^2 + \lambda||\beta||^2$ while at the same time $\hat{\beta}^{ridge}$ is still a linear estimator just like ridge regression (Degrees of freedom $= \sum_{i=1}^{p} \frac{d_j^2}{d_j^2+\lambda}$, and we can find that degrees of freedom and λ are negative correlated).

Unlike linear regression, $\hat{\beta}^{ridge}$ is not an unbiased estimator. Based on the function of ridge regression, we can find that it will not zero out the coefficients, which means that this method will only do coefficients shrinkage.

Figure 1.1: Ridge Regression



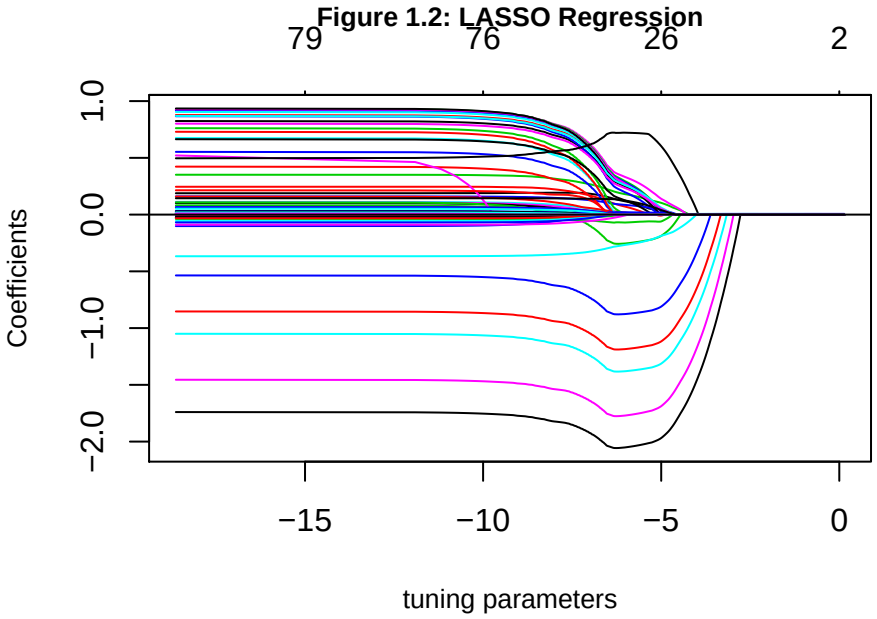
The graph shown above tells us the relationship among all of the coefficients and its corresponding dtuning parameter. The value of tuning parameter $= log(\lambda/8691)$. When λ is going larger, $\hat{\beta}^{ridge}$ goes towards 0 at the same time.

After calculating, we get the training and testing RMSEs are 0.4712394 and 0.4721781, respectively, while its corresonding λ is 0.6018311 (degrees of freedom is 78.2).

1.2.3 LASSO

LASSO penalize the coefficient $\hat{\beta}^{lasso}$ from the function $\hat{\beta}^{lasso} = \arg\min_{\beta} ||\mathbf{y} - \mathbf{X}\beta||^2 + \lambda||\beta||$ while at the same time $\hat{\beta}^{lasso}$ is still a linear estimator.

Like ridge regression, $\hat{\beta}^{lasso}$ is a biased estimator. Based on the function of LASSO, we can find that it may zero out the coefficients. It illustrates that this method will do coefficients shrinkage and variable selection at the same time, which is different from ridge regression.



The graph shown above tells us the relationship among all of the coefficients and their corresponding tuning parameter. The value of tuning parameter $= log(\lambda/8691)$. When λ is going larger, $\hat{\beta}^{lasso}$ goes towards 0 at the same time.

Based on the graph shown above, we can clearly see that just like ridge regression above all of the coefficients are shrunk as the degrees of freedom is going towards zero (λ is going larger at the same time). After calculating, we get the training and testing RMSEs are 0.4713212 and 0.4721329, respectively, while its corresponding λ is 0.5083409.

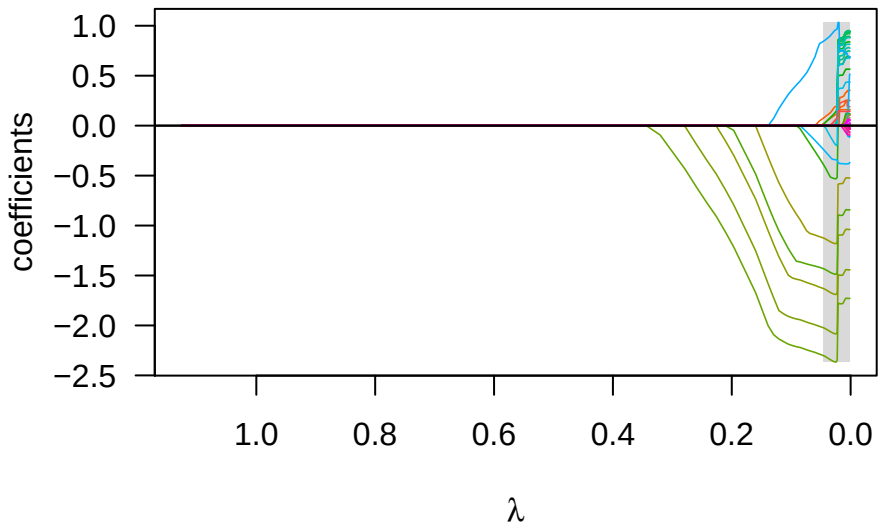
1.2.4 MCP

MCP stands for Minimax Concave Penality which is an unbiased penality. It has exactly the same formulation of the penalized loss funtion which is $\frac{1}{2n}||\mathbf{y} - \mathbf{X}\beta||^2 + \sum_{i=1}^{p} P(\beta_j)$. For some $\gamma > 1$, its penality term is defined as:

$$P(\beta) = \begin{cases} \lambda|\beta| - \dfrac{\beta^2}{2\gamma}, \text{ if } |\beta| \leq \gamma\lambda \\ \dfrac{1}{2}\gamma\lambda^2, \text{ if } |\beta| \geq \gamma\lambda \end{cases}$$

The maximum concavity of this penalty function is $1/\gamma$, which is exactly controlled.

Figure 1.3: MCP



The graph shown above tells us the relationship among all of the coefficients and their corresponding degrees of freedom. When λ is going towards zero, $\hat{\beta}^{mcp}$ goes greater. When λ is going greater, $\hat{\beta}^{lasso}$ goes towards 0.

Here, we will only use default γ and λ to fit `MCP` models. After calculating, we get the training and testing RMSEs are 0.471699 and 0.4719627, respectively, while its corresponding λ is 0.0056035.

1.3 Summary

Model	Linear regression	Ridge	LASSO	MCP
training RMSE	0.47121	0.4712394	0.4713212	0.471699
testing RMSE	0.4721885	0.4721781	0.4721329	0.4719627

In this case, we tried four methods to fit our models, `Linear regression`, `ridge regression`, `LASSO` and `MCP`. `Linear regression` and `MCP` offer us an unbiased estimator while the rest two can make biased estimator. But being unbiased or not is not the standard that we use to find the best model.

Here, we use testing RMSE to decide which method we used is the best. We summary the training and testing RMSE of the four methods that we have used in this part. This table is to make the results more clearly. We can find that all of the testing RMSE are pretty close

to each other while `MCP` which is 0.4719627 is the smallest of all. We can believe that this method is neither over nor under fitting. At the same time, what `MCP` provides us is also an unbiased estimator. Hence, we think `MCP` should be the best model of this question.

There are still some insufficient parts in this case. We can find that all of the tuning λ that we use are very small here. It illustrates that all of shrinkage coefficients are still prettly close to the OLS coefficients, which can also explain why these RMSEs are almost equal to each other. Further, we may consider using cross-validation RMSE to find the best tuning parameter instead of test RMSE in each method.

Question 2

2.1 Quadratic Discriminant Analysis

Quadratic Discriminant Function is based on the Bayes Theorem. We model each class density as multivariate Gaussian Distribution $f_k(x)$, and each covariance matrices are NOT the same accross all k. Here, μ_k is the group k means. And $\Sigma_i = (\mathbf{x}_k - \mu_k)(\mathbf{x}_k - \mu_k)^T$, where Σ is the covariance matrix of group k. $\pi_k = \frac{n_k}{n}$ is prior probability of the group k, where n_k and n are group observations numbers and total observation numbers, respectively.

The best prediction is picking the one that maximizes the posterior $\pi_i f_i(x)$. In order to make it more convinent, we fit the discriminant function: $\delta_k(x) = -\frac{1}{2}log|\Sigma_k| - \frac{1}{2}(\mathbf{x} - \mu_k)^T\Sigma_k^{-1}(\mathbf{x} - \mu_k) + log\pi_k$. The `y` will be classified in to group i if $\delta_i(x)$ is the largest of all of the discriminant values.

After calculating, we can obtain its training error is 0.0113636 which is extremely closed to 1. Just considering the training error is not enough because we just fit this model based on this training dataset. We also check the testing error which is 0.5281385.

2.2 Further Models Fitting

2.2.1 Linear Discriminant Function

Linear Discriminant Function is also based on the Bayes Theorem. We model each class density as multivariate Gaussian Distribution $f_k(x)$, and each covariance matrices are the same accross all k. Here, μ_k is the group k means. And $\Sigma = (\mathbf{x} - \mu)(\mathbf{x} - \mu)^T$, where Σ is the covariance matrix of every group. $\pi_k = \frac{n_k}{n}$ is prior probability of the group k, where n_k and n are group observations numbers and total observation numbers, respectively.

Thus, the linear discriminant function we fit is $\delta_k(x) = \mathbf{x}^T\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k + log\pi_k$. The `y` will be classified in to group i if $\delta_i(x)$ is the largest of all of the discriminant values, just like QDA.

The package `MASS` of R has already generated function `lda` for us to use, which can save us valuable time. After calculating, we can obtain its training error is 0.3162879 which is

extremely closed to 1. Like previous part, just considering the training error is not enough. We also check the testing error which is 0.5562771.

2.2.3 re-fit QDA

We have talked about how QDA works in classification in the previous part. Here, we only attempt to fit a QDA model and predict from the `MASS` package. After calculating, we will know that its QDA training and testing errors are 0.0113636 and 0.0113636, respectively. This gives us an opportunity to check the correctness of the previous part. However, we will not do it until the very end.

2.2.3 Logistic regression

We will also try the method of Logistic regression. This method looks like the linear regression but they are not exactly the same. The basic function we use here is $log\frac{y}{1-y} = x^T\beta$ where $log\frac{y}{1-y}$ is called log-odds here and we model it as a linear function of X. We can also re-write the function as $y = \frac{exp(x^T\beta)}{1+exp(x^T\beta)}$.

In this part, it is a multiomial logistic regression which has 11 possible outcomes. We can just running 10 independent binary logistic regression models. The ith funciton is like $log\frac{p(y=i)}{p(y=11)} = \mathbf{X}\beta$.

Luckily, R has already provided us a function `multinom` in the `nnet` package. After calculating, we can get that its training error is 0.2215909 and testing error is 0.512987.

2.3 Summary

Model	LDA	QDA	Logistic regression	QDA (Part 2.1)
Train Error	0.3162879	0.0113636	0.2215909	0.0113636
Test Error	0.5562771	0.5281385	0.512987	0.5281385

We summary the train and test errors (including the previous part) of the methods that we have tried. This table is to make the results more clearly. We can find that the test error of `Logistic regression` is the lowest of these four which is 0.512987. We can believe that this method is neither over nor under fitting in this case. Here we think that `Logistic regression` should be considered the best model of this question.

What's more, we can also find that the train and test errors of QDA without and with using package `MASS` are both 0.0113636 and 0.5281385, respectively. They match each other perfectly.