STA363 Project 2

Zishan Shao

2022-10-20

Abstract

Short-term lending of the house for vacation is becoming a trending business, and price evaluation of rented houses becomes increasingly important. The main purpose of this report was to construct a statistical model that could efficiently provide the price for renting services by VRBO. The data proceeded in this report was from the past rentals information of the VRBO and consisted of 1561 with 13 features (both categorical and numerical) and price as the response variable to be predicted. We applied the least-square linear regression model (LSLR), ridge regression model, lasso regression model, and the elastic net. Considering the size of the sample data, we applied the 10-fold cross-validation technique to train and test the model. We then find the test MSE of the series of models and explore the optimal solution that maximizes the likelihood. The model will be chosen based on simplicity and accuracy (measured by the MSE). The final results indicate that the ridge regression model provides the most accurate predictive result, while the lasso regression model provides comparable accuracy with the least complexity. Our final recommendation is the lasso regression model.

Contents

Section 1: Introduction

With the popularization of applications such as Airbnb, an increasing proportion of non-professionals post their houses/apartments online for short-term renting. However, the veracity of the renters and houses post challenges for the platform to determine the rental prices. It is unfeasible to set a general price because the houses/apartments vary in location, size, quality, etc. Therefore, it is essential to set an evaluation system to determine a relatively legitimate price.

In this report, we are working with the rental data from the VRBO. The main service of the company was allowing individuals to make short-term rentals of their apartments. However, the houses vary a lot in their qualities, locations, size, etc. Therefore, the main purpose of the report was to offer predictions of rental prices for VRBO's customers.

The data was consisted of 1561 past rentals information with 13 features (both categorical and numerical) and price as the response variable (as we aim to estimate the price). The model was based on two assumptions:

- The rental price was related to the basic conditions of the house and there are patterns to be traced.
- The sample data is the representative of the population set, which means the model could be applied to the actual data and should not be overly biased.

Because the house price is a numerical variable, so we are dealing with a regression problem. While the least-square linear regression model was the most basic and commonly used linear regression model, it has difficulties in dealing with correlated features, so penalized regression models such as ridge regression are also considered. The models will be evaluated with the mean-squared error (MSE) and simplicity.

Section 2: Data Cleaning

In this section, we explore the observations with missing data and evaluate the features with their traits. Based on the analysis, there are 138 observations containing missing information, which could lead to computational problems in model building.

Data Loss Table

Table 1: Data Loss Table (before & after cleaning)

Names	Count
Original	1561
Cleaned	1423
Total Removal	138
Percent Data Loss (%)	8.84

Removing observations with missing values is a commonly applied approach. However, it will inevitably cause a reduction in the size of training data. In this case, the observations to be removed consisted of around 8.84% of total observations, which is a considerable size of data loss. In the worst case, it could increase the difficulty of capturing the general pattern of the data and cause inaccuracy in prediction.

Table 2: Number of Observations (with/not with minstay)

Orignal	No.Minstay
1561	1561

Interestingly, the distribution of the missing information indicates that the feature *minimum nights (minstay)* contains all the missing values. After removing the *minimum nights*, the cleaned data has an identical number of observations with the original data. Therefore, it is possible to keep the size of the data by removing the *minimum nights*.

However, the second approach could only be applicable if the feature to be removed is loosely related to the response variable. If it is strongly related to the response variable, removing the feature could severely undermine the predictive accuracy of the model. Therefore, we construct plots to explore the relationship between the *minimum nights* and the response variable.

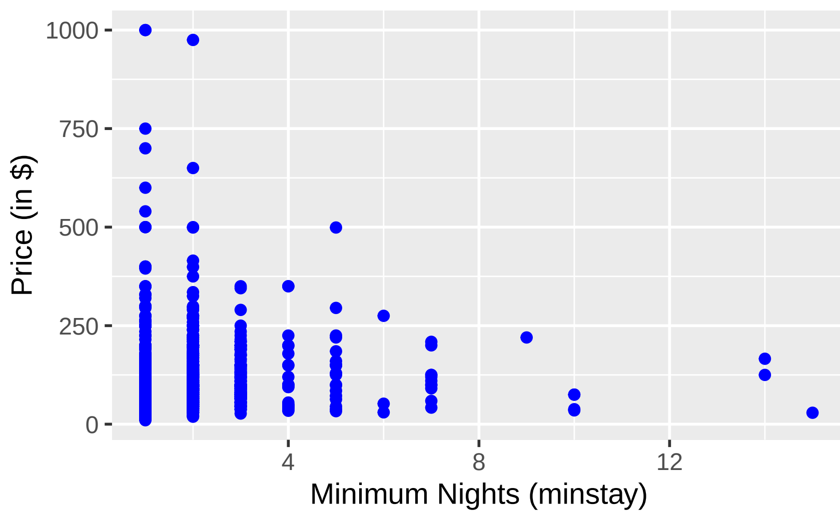


Figure 2.1: Minimum Nights vs Price (in $)

Figure 2.1 indicates there is not a strong linear relationship between the price and minimum nights. Hence, approach two is applicable in this case.

Noticing that the UnitNumber is a categorical feature that assigns an unique value to all observations, so it is impossible to find a pattern between the UnitNumber and the Price. Therefore, we should remove the UnitNumber.

Distribution of Response Variable

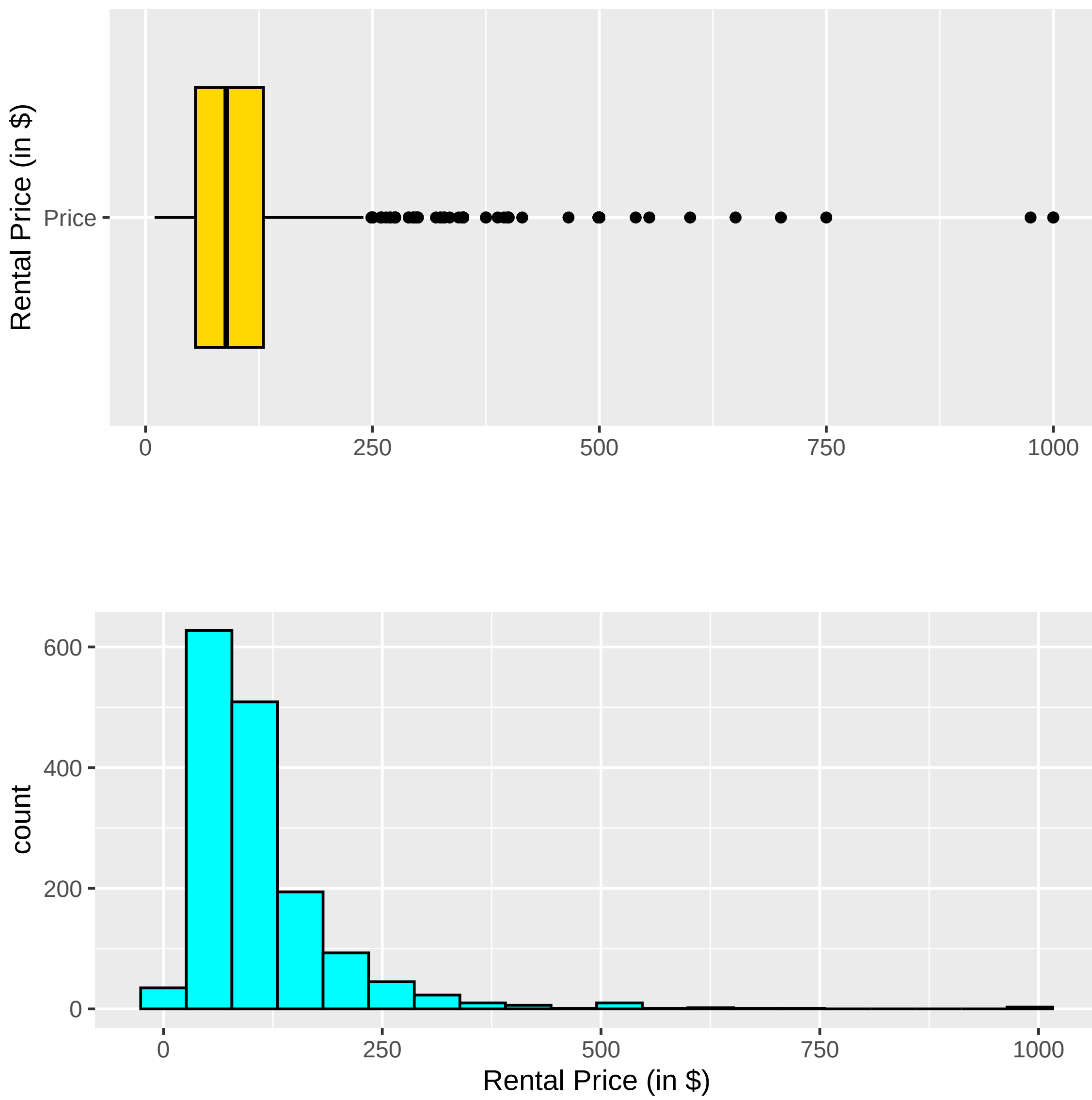


Figure 2.2: Distribution of Response Variable (*Price* in $)

Figure 2.2 indicates that the distribution of the price in the data, although skewed, is reasonable since the conditions of the houses vary significantly. It is a unimodal distribution and there are no obvious outliers in the data that could impact the predictive capacity of the model.

Section 3: LSLR & Ridge Regression

In this part, the least-square linear regression (LSLR) and the ridge regression will both be trained and evaluated. The difference between the LSLR and the ridge is that, while LSLR obtains the optimal $\hat{\beta}$ through minimizing the residual sum squared (RSS), which simultaneously maximized the likelihood of the model, the ridge considers both the MSE and the coefficients $\hat{\beta_{ridge}}$. The closed form of both are listed below:

$$LSLR : \hat{\boldsymbol{\beta_{LS}}} = (\boldsymbol{X_D^T X_D})^{-1} \boldsymbol{X_D^T Y}$$

$$Ridge : \hat{\boldsymbol{\beta_{Ridge}}} = (\boldsymbol{X_D^T X_D} + \boldsymbol{\lambda_{Ridge} I})^{-1} \boldsymbol{X_D^T Y}$$

The penalty factor $\lambda \hat{\beta}^T \hat{\beta}$ limits the size of the coefficients while minimizing the RSS. λ, as the tuning parameter of the ridge regression, controls the extent of shrinkage. Thus, ridge regression effectively constrains the variances of the model caused by correlated features (which lead to exploding variance due to a small determinant of the design matrix).

To determine the optimal value of the λ, we first select a range of possible values for the tuning parameter λ, plot them and choose the one with the lowest test MSE, the range of λ should include 0 as part of the choice, which is the case of LSLR.

Although not required, the LSLR was included to serve as the base case of the models. By comparison, it is more convenient to observe the effectiveness of the penalized regression models in shrinking the coefficients.

Correlation Plot

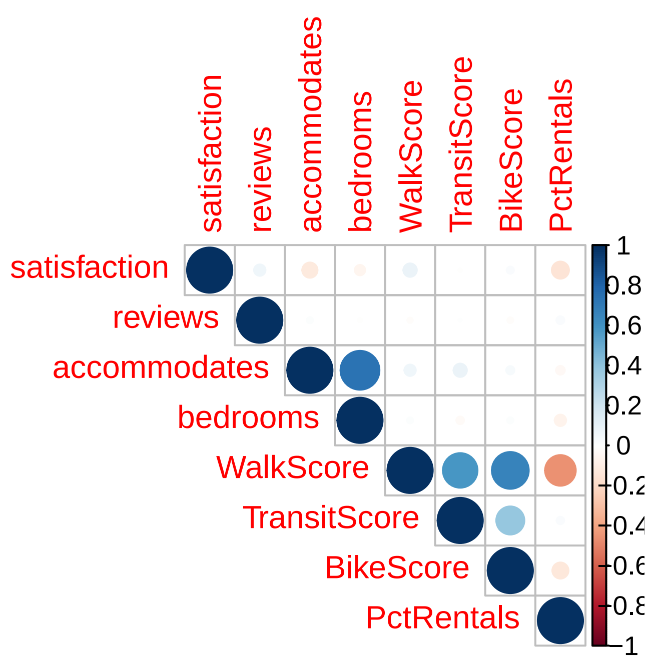


Figure 3.1: Correlation Plot between features

Based on the correlation plot, the *WalkScore - BikeScore, bedrooms - accommodates, and PctRentals - WalkScore*, are linearly correlated (with a correlation larger than or approximately equal to 0.5). This means that, in this case, a penalized regression technique might be necessary.

To apply the ridge regression, we need to create the designed matrix in form of $\boldsymbol{X_D} = [\mathbf{1} \quad \boldsymbol{X}]_{(\boldsymbol{n \times p})}$ that includes all features (both categorical and numerical) and intercept. The $\mathbf{X}$ is the features in matrix form.

Tuning Parameter (Ridge)

We explore the optimal tuning parameter λ with a range of 0 to 100, with 0.5 for each interval, the 10-fold cross-validation technique is applied to find the $\hat{\beta_{ridge}}$ that has the smallest test mean-squared error (MSE). The MSE is the mean of RSS after averaging through the number of observations. The minimum MSE indicates that, at this value of λ, the model performs the best because the estimates have the least difference from the actual value. For every λ there should be an optimal model with the least test MSE, and we can then choose the best λ by selecting through the set of optimal models. Via the cross-validation technique, we could also avoid choosing the λ in an overly large value and overshrink the coefficients.

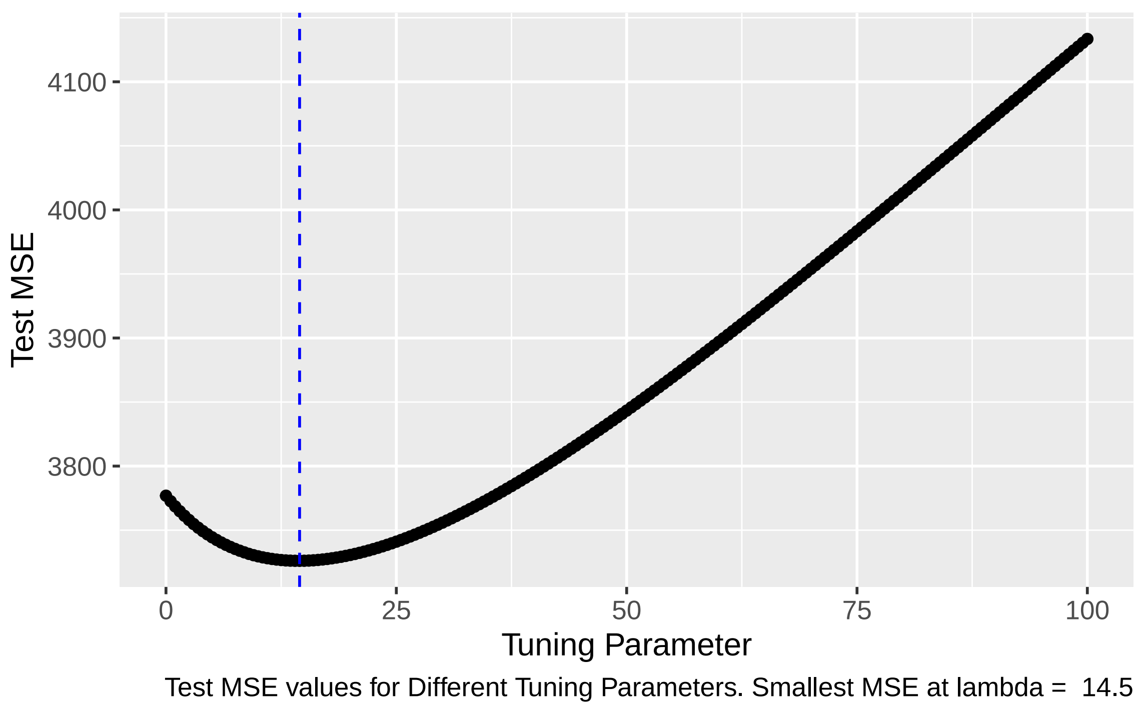


Figure 3.2: Test MSE vs Tuning Parameter of Ridge Regression model

Based on the result of the cross-validation, we could see that the $\hat{\beta_{ridge}}$ with the smallest test MSE is obtained at $\lambda = 14.5$. Therefore, we should select 14.5 as the value for training the ridge regression model.

Table 3: Metrics of LSLR vs Ridge

	Lambda	Test MSE	Test RMSE
Ridge	14.5	3726.107	61.04185
LSLR	0.0	3776.969	61.45705

Model Training & Evaluation

Table 4: Coefficients with Tuning Parameter = 0, 14.5

	LSLR	ridge
(Intercept)	-158.756	-237.574
overall_satisfaction	34.213	27.470
reviews	-0.121	-0.098
room_typePrivate room	-25.239	-25.796
room_typeShared room	-57.118	-49.615
accommodates	12.856	11.138
bedrooms	29.215	27.895
neighborhoodArcher Heights	-16.708	2.769
neighborhoodAvondale	-13.347	-4.378
neighborhoodBeverly	-7.548	19.510
neighborhoodBridgeport	-25.213	-11.567
neighborhoodBrighton Park	-42.259	-13.847
neighborhoodBurnside	-32.570	-2.299
neighborhoodCalumet Heights	-6.104	17.293
neighborhoodEast Garfield Park	-34.903	-14.024
neighborhoodEdgewater	4.703	-8.142
neighborhoodEdison Park	-8.374	14.670
neighborhoodEnglewood	3.770	25.027
neighborhoodGage Park	-10.355	3.515
neighborhoodHegewisch	-40.994	1.664
neighborhoodHermosa	-24.103	-13.750
neighborhoodHumboldt Park	-21.456	-11.252
neighborhoodHyde Park	-7.820	-2.096
neighborhoodIrving Park	35.494	31.587
neighborhoodJefferson Park	-9.141	-3.391
neighborhoodKenwood	-9.129	7.465
neighborhoodLincoln Park	38.331	25.989
neighborhoodLincoln Square	17.133	11.785
neighborhoodLogan Square	-13.751	-8.594
neighborhoodMcKinley Park	-16.382	-0.102
neighborhoodMontclare	-38.250	-16.658
neighborhoodMorgan Park	-32.915	-1.605
neighborhoodNear West Side	26.797	19.666
neighborhoodNorth Center	-5.527	1.440
neighborhoodNorth Park	11.986	18.493
neighborhoodO'Hare	11.202	33.065
neighborhoodPortage Park	-21.875	-11.114
neighborhoodPullman	-12.557	25.651
neighborhoodRogers Park	0.937	-8.402
neighborhoodSouth Chicago	-18.502	-2.263

	LSLR	ridge
neighborhoodSouth Shore	-31.585	-3.833
neighborhoodThe Loop	58.659	31.844
neighborhoodUptown	28.311	9.395
neighborhoodWashington Park	-2.702	21.163
neighborhoodWest Elsdon	-22.100	1.478
neighborhoodWest Englewood	-131.238	-72.480
neighborhoodWest Lawn	-65.419	-33.227
neighborhoodWest Town	14.048	5.742
neighborhoodWoodlawn	-32.325	-3.504
districtFar North	-16.310	-5.678
districtFar Southeast	1.146	6.822
districtFar Southwest	0.788	5.582
districtNorth	1.786	-3.039
districtNorthwest	2.006	7.489
districtSouth	0.287	-4.358
districtSouthwest	0.258	-3.924
districtWest	-0.806	-0.751
WalkScore	-0.150	0.567
TransitScore	0.382	0.983
BikeScore	0.268	0.467
PctRentals	9.676	-6.586

Table 5: Metrics of LSLR vs Ridge

	Lambda	MSE	Sum Coefficients (abs)
Ridge	14.5	3480.2	71.2
LSLR	0.0	3449.7	510.6

The final training result shows that the LSLR has a smaller MSE but the difference was minor compared to the size of both MSEs. However, the ridge regression has a significantly smaller sum of coefficients in absolute value (71.2) compared with 510.6 for LSLR. This means the ridge regression has attained comparable accuracy with prominently less variance and coefficients.

Although penalizing the coefficients introduces biases in the model, it could effectively limit the variance of the model (indicated by the sum of coefficients). In other words, the unbiasedness of the LSLR was sacrificed to limit the variance of the model.

Nonetheless, ridge regression has drawbacks. For the ridge regression, only shrinkage was performed, and the model had to retain all features in the design matrix. In this case, there are 60 features in total, which increases the complexity of the model. Moreover, the complexity of the model is proportional to the required size of data necessary to effectively capture the pattern of the data. Therefore, ridge regression could be inaccurate due to the limited size of the customer data.

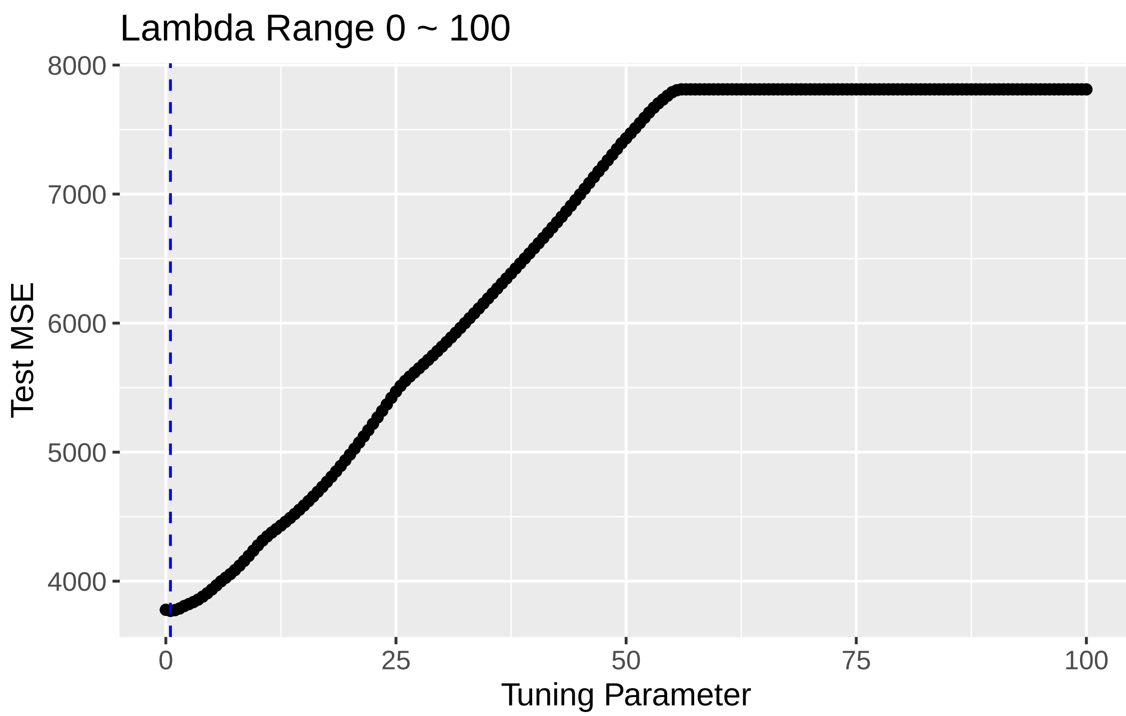
Section 4: Lasso

Considering the number of features included in the model, the model is expected to be difficult to interpret. Therefore, it is necessary to reduce the complexity by selecting the most influential features while limiting the coefficients. In other words, we are expected to conduct *shrinkage* and *selection*.

The lasso regression could fulfill our requirement. Lasso regression minimizes $RSS + \lambda_{lasso} \mid \hat{\beta} \mid$, which enables the system to set the coefficients of less important features to zero, and thus performed the selection along with shrinkage.

Tuning Parameter (Lasso)

Similar to ridge regression, we use the 10-fold cross-validation technique to train and test models with a range of λ. The range of λ is the same as the one for ridge regression and the best model, similarly, will be selected with λ that provides the minimal test MSE.



Test MSE values for Different Tuning Parameters. Smallest MSE at lambda = 0.5

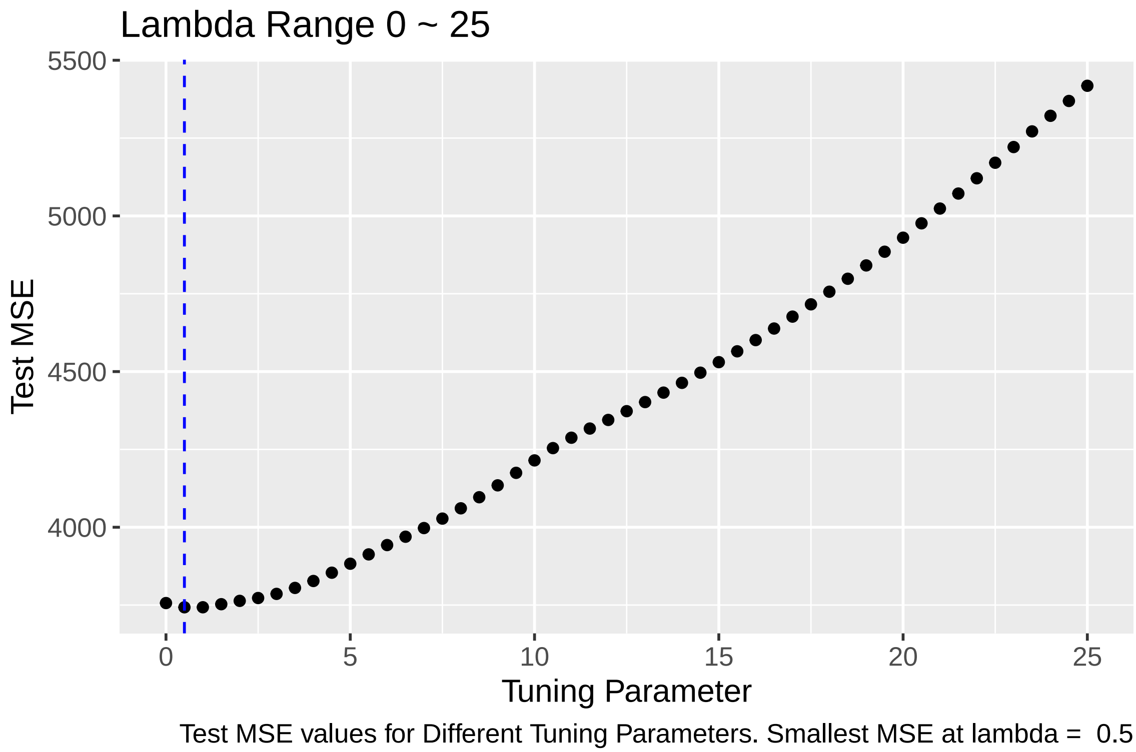


Figure 4.1: Test MSE vs Tuning Parameter of Lasso Regression model

Figure 4.1 indicates that the optimal λ, in this case, should be 0.5, so we should train the lasso regression model by setting the $\lambda = 0.5$. We obtain the betas of the lasso and add them to the coefficients table.

Evaluation

Table 6: Coefficients Table of LSLR, Ridge, Lasso

	LSLR	ridge	Lasso
(Intercept)	-158.756	-237.574	-248.006
overall_satisfaction	34.213	27.470	32.111
reviews	-0.121	-0.098	-0.109
room_typePrivate room	-25.239	-25.796	-24.627
room_typeShared room	-57.118	-49.615	-53.399
accommodates	12.856	11.138	12.638
bedrooms	29.215	27.895	29.064
neighborhoodArcher Heights	-16.708	2.769	0.000
neighborhoodAvondale	-13.347	-4.378	-0.431
neighborhoodBeverly	-7.548	19.510	12.950
neighborhoodBridgeport	-25.213	-11.567	-7.369
neighborhoodBrighton Park	-42.259	-13.847	-10.273
neighborhoodBurnside	-32.570	-2.299	0.000
neighborhoodCalumet Heights	-6.104	17.293	9.634
neighborhoodEast Garfield Park	-34.903	-14.024	-19.253
neighborhoodEdgewater	4.703	-8.142	-4.368

	LSLR	ridge	Lasso
neighborhoodEdison Park	-8.374	14.670	0.000
neighborhoodEnglewood	3.770	25.027	5.806
neighborhoodGage Park	-10.355	3.515	0.000
neighborhoodHegewisch	-40.994	1.664	0.000
neighborhoodHermosa	-24.103	-13.750	0.000
neighborhoodHumboldt Park	-21.456	-11.252	-8.127
neighborhoodHyde Park	-7.820	-2.096	0.000
neighborhoodIrving Park	35.494	31.587	45.435
neighborhoodJefferson Park	-9.141	-3.391	-5.651
neighborhoodKenwood	-9.129	7.465	0.000
neighborhoodLincoln Park	38.331	25.989	32.698
neighborhoodLincoln Square	17.133	11.785	11.235
neighborhoodLogan Square	-13.751	-8.594	-3.433
neighborhoodMcKinley Park	-16.382	-0.102	0.000
neighborhoodMontclare	-38.250	-16.658	-3.852
neighborhoodMorgan Park	-32.915	-1.605	0.000
neighborhoodNear West Side	26.797	19.666	18.639
neighborhoodNorth Center	-5.527	1.440	4.049
neighborhoodNorth Park	11.986	18.493	13.236
neighborhoodO'Hare	11.202	33.065	20.463
neighborhoodPortage Park	-21.875	-11.114	0.000
neighborhoodPullman	-12.557	25.651	8.646
neighborhoodRogers Park	0.937	-8.402	-8.965
neighborhoodSouth Chicago	-18.502	-2.263	0.000
neighborhoodSouth Shore	-31.585	-3.833	-5.984
neighborhoodThe Loop	58.659	31.844	33.654
neighborhoodUptown	28.311	9.395	7.647
neighborhoodWashington Park	-2.702	21.163	2.740
neighborhoodWest Elsdon	-22.100	1.478	0.000
neighborhoodWest Englewood	-131.238	-72.480	-82.608
neighborhoodWest Lawn	-65.419	-33.227	-29.951
neighborhoodWest Town	14.048	5.742	11.203
neighborhoodWoodlawn	-32.325	-3.504	-13.365
districtFar North	-16.310	-5.678	0.000
districtFar Southeast	1.146	6.822	2.469
districtFar Southwest	0.788	5.582	0.000
districtNorth	1.786	-3.039	0.000
districtNorthwest	2.006	7.489	0.000
districtSouth	0.287	-4.358	0.000
districtSouthwest	0.258	-3.924	0.000
districtWest	-0.806	-0.751	0.000
WalkScore	-0.150	0.567	0.000
TransitScore	0.382	0.983	1.365
BikeScore	0.268	0.467	0.422
PctRentals	9.676	-6.586	0.000

The trained Lasso regression obtained an MSE of 3459.3, which is slightly higher than the LSLR model. However, the lasso regression assigned 22 features as zero coefficients. This implies that the lasso regression attained similar accuracy as the ridge regression and LSLR with 22 fewer features. Moreover, the shrinkage by lasso regression is improved. The lasso regression achieved a sum of around 34.3 compared to the 71.2 obtained by the ridge regression. This means the lasso limits the variance of the model more effectively.

Regarding the purpose of this report, comparable accuracy after feature reduction provides us with a more interpretative, trainable model. For the predictive purpose, lasso regression outperforms ridge regression and LSLR.

Section 5: Elastic Net

An elastic net is a generalized form of ridge and Lasso regressions. It introduced the α as the tuning parameter, which it minimizes $RSS\ +\ \lambda\sum((1-\alpha)\hat{\beta}_j^{\ 2} + \alpha\mid\hat{\beta}_j\mid)$, where $\lambda \geq 0$ and $\alpha \geq 0$ are scalars. Elastic Net balances through ridge and Lasso regressions, taking advantage of both by tuning the α.

Tuning Parameters (Elastic Net)

To select the α and λ that minimize the *RSS + Penalty Factor*, we apply the 10-fold cross-validation technique. Provided that two tuning parameters are required to be chosen, iteratively matching each value of α and λ is necessary to select the optimal pair.

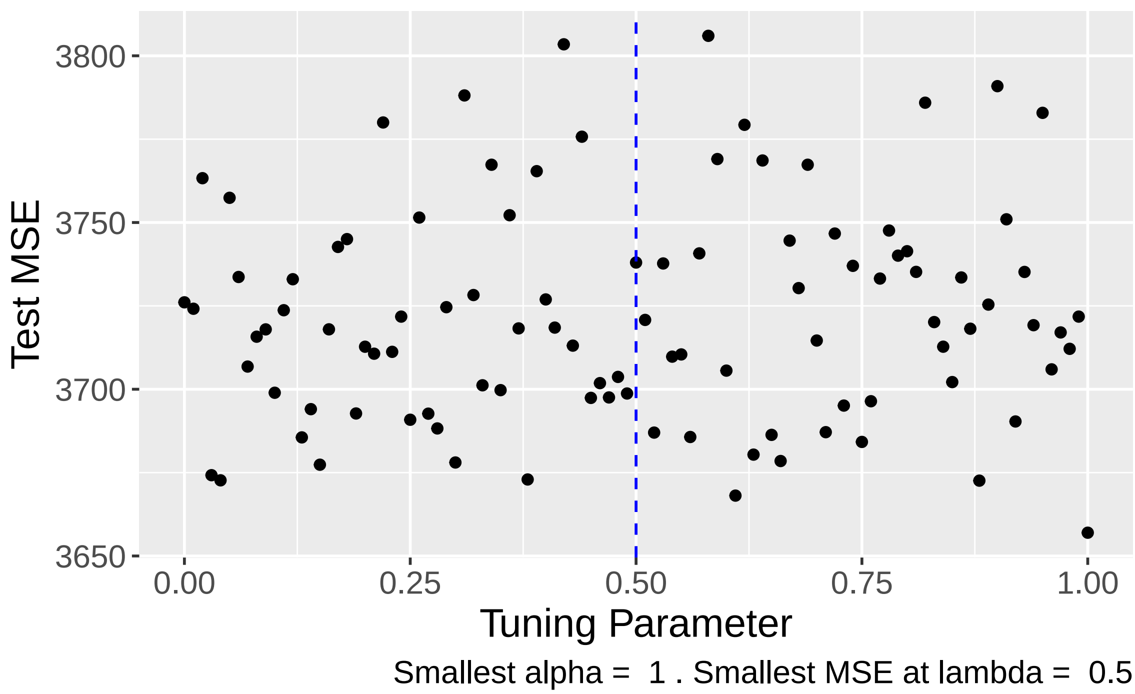


Figure 5.1: Test MSE vs Tuning Parameters of Elastic Net Model

Figure 5.1 indicates that the optimal pair of (α, λ) is (1.0, 0.5).

Training the Elastic Net

	LSLR	Ridge	Lasso	Elastic
(Intercept)	-158.7559107	-237.5743499	-248.0063689	-248.0063689
overall_satisfaction	34.2126288	27.4703267	32.1105121	32.1105121
reviews	-0.1214983	-0.0980393	-0.1086806	-0.1086806
room_typePrivate room	-25.2385430	-25.7963453	-24.6270072	-24.6270072
room_typeShared room	-57.1184039	-49.6152657	-53.3986469	-53.3986469
accommodates	12.8563825	11.1381340	12.6378240	12.6378240
bedrooms	29.2152240	27.8946223	29.0635846	29.0635846
neighborhoodArcher Heights	-16.7076837	2.7686696	0.0000000	0.0000000
neighborhoodAvondale	-13.3467345	-4.3783160	-0.4310673	-0.4310673
neighborhoodBeverly	-7.5477691	19.5098217	12.9504361	12.9504361
neighborhoodBridgeport	-25.2126026	-11.5674501	-7.3686229	-7.3686229
neighborhoodBrighton Park	-42.2592281	-13.8471415	-10.2726044	-10.2726044
neighborhoodBurnside	-32.5701439	-2.2989565	0.0000000	0.0000000
neighborhoodCalumet Heights	-6.1044236	17.2926880	9.6343909	9.6343909
neighborhoodEast Garfield Park	-34.9026220	-14.0235265	-19.2531936	-19.2531936
neighborhoodEdgewater	4.7032967	-8.1419616	-4.3676000	-4.3676000
neighborhoodEdison Park	-8.3738747	14.6702012	0.0000000	0.0000000
neighborhoodEnglewood	3.7698227	25.0269380	5.8063830	5.8063830
neighborhoodGage Park	-10.3553949	3.5154600	0.0000000	0.0000000
neighborhoodHegewisch	-40.9938448	1.6641986	0.0000000	0.0000000
neighborhoodHermosa	-24.1032886	-13.7502629	0.0000000	0.0000000
neighborhoodHumboldt Park	-21.4563639	-11.2517776	-8.1271431	-8.1271431
neighborhoodHyde Park	-7.8199131	-2.0959073	0.0000000	0.0000000
neighborhoodIrving Park	35.4941240	31.5865196	45.4352749	45.4352749
neighborhoodJefferson Park	-9.1408863	-3.3910216	-5.6508203	-5.6508203
neighborhoodKenwood	-9.1289557	7.4647839	0.0000000	0.0000000
neighborhoodLincoln Park	38.3310100	25.9886106	32.6981847	32.6981847
neighborhoodLincoln Square	17.1326480	11.7847507	11.2354512	11.2354512
neighborhoodLogan Square	-13.7508981	-8.5940124	-3.4327184	-3.4327184
neighborhoodMcKinley Park	-16.3824504	-0.1016848	0.0000000	0.0000000
neighborhoodMontclare	-38.2503236	-16.6583570	-3.8520331	-3.8520331
neighborhoodMorgan Park	-32.9152677	-1.6053685	0.0000000	0.0000000
neighborhoodNear West Side	26.7968999	19.6655090	18.6386725	18.6386725
neighborhoodNorth Center	-5.5272313	1.4398366	4.0488151	4.0488151
neighborhoodNorth Park	11.9856722	18.4931142	13.2363548	13.2363548
neighborhoodO'Hare	11.2016756	33.0654411	20.4629661	20.4629661
neighborhoodPortage Park	-21.8747216	-11.1138816	0.0000000	0.0000000
neighborhoodPullman	-12.5569569	25.6507941	8.6457862	8.6457862
neighborhoodRogers Park	0.9365428	-8.4023182	-8.9646521	-8.9646521
neighborhoodSouth Chicago	-18.5016086	-2.2626565	0.0000000	0.0000000
neighborhoodSouth Shore	-31.5848250	-3.8328060	-5.9843692	-5.9843692
neighborhoodThe Loop	58.6592702	31.8439967	33.6537014	33.6537014
neighborhoodUptown	28.3106488	9.3949138	7.6467561	7.6467561
neighborhoodWashington Park	-2.7018463	21.1629845	2.7401862	2.7401862
neighborhoodWest Elsdon	-22.1003751	1.4776742	0.0000000	0.0000000
neighborhoodWest Englewood	-131.2382045	-72.4798383	-82.6083220	-82.6083220
neighborhoodWest Lawn	-65.4192271	-33.2272699	-29.9506327	-29.9506327
neighborhoodWest Town	14.0481586	5.7421476	11.2033667	11.2033667
neighborhoodWoodlawn	-32.3252089	-3.5041323	-13.3645024	-13.3645024
districtFar North	-16.3103408	-5.6777545	0.0000000	0.0000000

	LSLR	Ridge	Lasso	Elastic
districtFar Southeast	1.1463278	6.8217270	2.4686950	2.4686950
districtFar Southwest	0.7882503	5.5820938	0.0000000	0.0000000
districtNorth	1.7855736	-3.0388828	0.0000000	0.0000000
districtNorthwest	2.0063410	7.4890742	0.0000000	0.0000000
districtSouth	0.2874515	-4.3576904	0.0000000	0.0000000
districtSouthwest	0.2581522	-3.9238686	0.0000000	0.0000000
districtWest	-0.8057127	-0.7506925	0.0000000	0.0000000
WalkScore	-0.1497608	0.5673138	0.0000000	0.0000000
TransitScore	0.3819037	0.9825056	1.3645074	1.3645074
BikeScore	0.2676216	0.4666456	0.4218521	0.4218521
PctRentals	9.6758770	-6.5860032	0.0000000	0.0000000

Evaluation

Coincidentally, this pair of the tuning parameter corresponds to the tuning parameter of the lasso regression. This indicates that the lasso regression is potentially the best choice to solve the prediction problem. Because at (1.0, 0.5), the elastic net performs as the lasso regression, we expect that the outputs are identical to the outputs from Section 4, so we should evaluate the LSLR, Ridge, and Lasso models to make the optimal choice.

From the trained elastic net model, the MSE of the elastic net at $\alpha = 1$, and $\lambda = 0.5$ is around 3459.3, and the sum of coefficients is 34.3, which is identical as the trained lasso regression model. This result support our statement above.

Section 6: Comparison and Conclusions

Summary Table

Table 8: Summary Table: Lasso vs Ridge vs LSLR

	Lambda	MSE	Sum Coefficients (abs)	Num Features
Lasso	0.5	3459.3	34.3	38
Ridge	14.5	3480.2	71.2	60
LSLR	0.0	3449.7	510.6	60

The summary table indicates that the lasso regression obtained comparable predictive accuracy compared with LSLR and ridge regression (all around 3450). Regardless of external factors, the most accurate model is the least-square linear regression model, which has the lowest MSE (3449.7) among all models. Therefore, for accuracy purposes only, ridge regression should be selected. In this task, we only dealing with the predictive task. However, the lasso regression obtained significantly less sum of coefficients and number of features, which means less complexity and variance in prediction. Less complexity improved the readability of the model and training accuracy since a more complex model requires more training data to obtain the pattern of the data. Smaller variance also ensures the applicability of the model in real life. Therefore, we recommend that the lasso regression should be the model for price prediction.