TECHNICAL UNIVERSITY OF DENMARK

Supervised learning: Classification and regression
Project 2

November 16, 2023

Author:

ZHENLIN XIE	s232268
ANDRO KRANJCEVIC	s204704
JOÃO LUÍS GONÇALVES MENA	s223186

Technical University of Denmark
Course: 02450 - Introduction to Machine Learning and Data Mining

Contents

Contributions Table			
	Zhenlin	Andro	Joao
Regression (a)	0%	60%	40%
Regression (b)	0%	20%	80%
Classification	0%	80%	20%
Discussion	0%	40%	60%
Exam Problems	33.33%	33.33%	33.33%

1 Regression

The following section aims to create a model that can predict prices for a wide range of listings based on the patterns it has learned from the data through regression analysis. By utilizing the most significant features as input variables, the price for each listing (target variable) will be estimated.

1.1 Regression (a)

Feature transformation

Since this project was built on thew work reported previously, most of the feature transformations remain the same as before. For organization purposes, a new script was created, but the same data manipulations were applied, with exception of an additional log transformation being applied to the *price*, *minimum_nights*, *number_of_reviews* and *bedroom* attributes to correct right skeweness, and a cubic root transformation in the *bath* attribute to correct left skeweness.

With that, all numerical attributes have been normalized, have mean 0 and standard deviation, while the remaining attributes are either binarized or one-out-of-k encoded.

Regularization and K-Fold Cross-Validation

The regularization parameter λ is introduced for the purpose of controlling the impact on bias and variance, by penalizing the flexibility of the model. Depending on the value of λ, the generalization error changes. For each value of λ, K = 10 fold cross-validation is implemented to estimate the generalization error.

The range of 10^{-2} up to $10^{0.2}$ linearly distributed by 50 points of λ values was selected, where the generalization error is estimated by:

$$E^{gen} = \sum_{k=1}^{K} \frac{N_k^{test}}{N} E_k^{test} \tag{1}$$

$$E_i^{test} = \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} (y_i - \hat{y_i})^2 \tag{2}$$

Where, $(y_i - \hat{y_i})^2$ term represents the squared error between the actual value y_i and the predicted value $\hat{y}_i$ for the i^{th} observation. In Figure 1, a generalization error is shown for different values of λ.

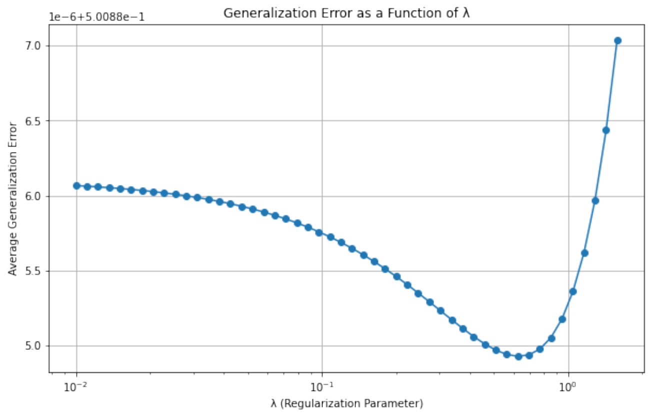

Figure 1: Ridge Regression Coefficient Plot

The generalization error as a function of the regularization parameter λ shows a clear U-shaped pattern, resembling the expected behavior. As the regularization parameter λ increases, the model initially overfits at smaller values, which is observed by a high error due to fitting noise in the data. As λ rises, the model reduces its complexity, decreasing the error and mitigating overfitting. However, at larger λ values, the model becomes too simple, which then leads to underfitting and a subsequent error rise.

In the following part, the Ridge Regression is implemented to compare the model coefficients to understand the effect of individual features on the predicted income. The output, y, of the linear model, is computed as a weighted sum of the input features in x, where the weights are the coefficients of the linear model. Additionally, if one wants to compute a linear model with the lowest generalization error, the model coefficients $\beta_{0,..,n}$ should be determined during the training phase to minimize the error.
The ridge regression "penalizes" the variable coefficients, such that those that are the least effective, "shrink" the fastest. In general, as alpha increases, the penalty for large coefficients also increases. If certain coefficients flatten out sooner than others, it represents reaching the regularisation state. If the coefficient values remain large, it means that it may be a significant feature when making predictions.

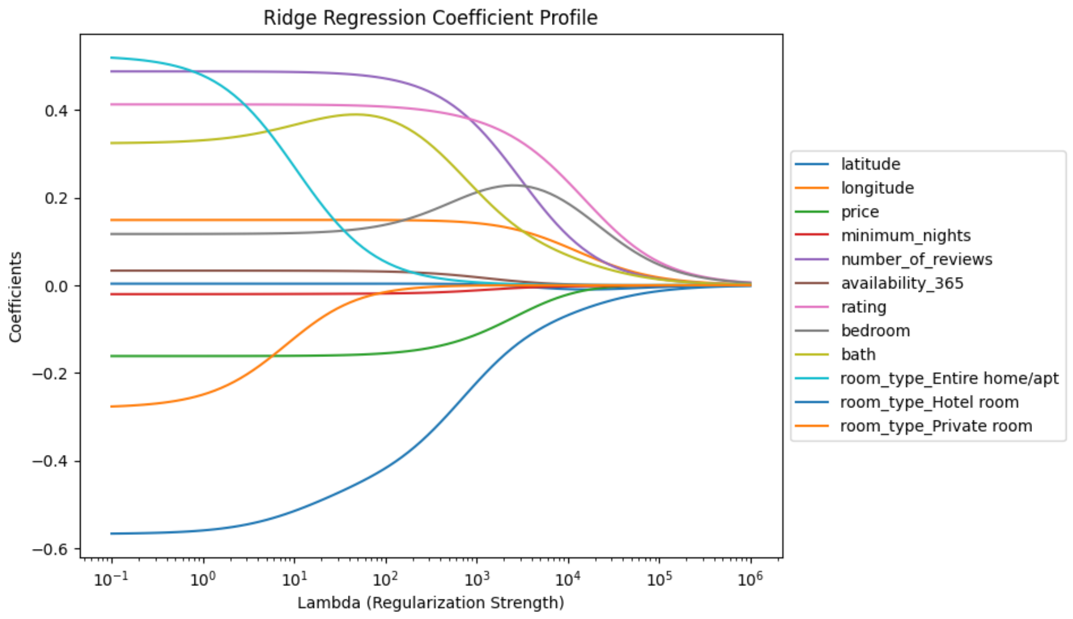

Figure 2: Ridge Regression Coefficient Plot

From the Figure 2, it's evident that certain features ('rating', 'number of reviews', and 'bedroom') retain their influence on the output over a broader range of λ values as compared to others. These features might be inherently more important in predicting the target variable ('price'). On the other hand, features with the label 'minimum nights' and 'availability 365' have coefficients that approach zero relatively quickly as λ increases, suggesting they might be less influential. Such results are understandable, as room type and required minimum stay, do not necessarily have a significant impact on the price. On the other hand, rating, number of bedrooms, and number of reviews often impact the listing's price.

1.2 Regression (b)

In this section, two more methods and test will be implemented, and their results compared with the results obtained in the previous section for the Linear Regression model. These models will also be implemented with Cross Validation, where $K = 10$, which allows for a better comparison of the models. The first method is a Baseline model. This was implemented by implementing a simple linear regression with no features, which simply uses the computed mean for the y training data to predict the y result in the test data. Then, an Artificial Neural Network model was implemented, where the number of hidden units h was utilized as a complexity-controlling parameter. The table 1 below contains the error results of the tests for each model, for each of the K folds used for the cross-validation. The baseline model presents one error value for each fold. The Linear Regression model columns contain the error for the $(\lambda)_i$ value with the lowest error in each fold. The Artificial Neural Network columns show the lowest error and corresponding hidden layer units h_i in each fold. For this, multiple h_i values were tested until finding what is believed to be the optimal hidden layer units size for this model. The range of values selected for the final model was 1, 10, 25, 40, 50.

Outer fold	ANN		Linear regression		Baseline
	h_i	E_{test}	λ_i	E_{test}	E_{test}
1	40	0.4416	0.0001	0.546290	1.016001
2	40	0.3905	1.5848	0.546975	0.956523
3	40	0.4795	1.5848	0.575295	1.000422
4	40	0.4795	0.5907	0.500884	0.958535
5	40	0.4709	0.0001	0.608098	1.032881
6	40	0.4575	0.0001	0.534399	1.023477
7	40	0.4304	1.5848	0.537243	1.030870
8	40	0.4601	0.0001	0.513137	0.967696
9	40	0.4143	0.0001	0.526894	1.005856
10	40	0.4377	1.5848	0.571590	1.013628

Table 1: Two-level cross-validation table used to compare the three models in the regression problem.

1.3 Comparison

All three models present different characteristics and results. The baseline model is clearly the simplest to implement, but as a consequence, it presents by far the worst results of all three. On the opposite end of the spectrum, the ANN model achieves the lowest error measured in our tests, but it is also the more complex and requires a lot more computational power, and from our experience, much more time to execute.

Taking this into account, the questions "Is either model better than a trivial baseline?" and "Is one model better than the other?" can be answered from two different perspectives: holistically and statistically. A statistical analysis of the values displayed in table 1 can be found bellow. However, from a holistic point of view, it can be said with confidence that both the Linear Regression and ANN models are better than the baseline model - despite it's simple implementation and execution, the different in errors simply makes it too much of an unreliable option. Regarding the remaining two models, it is difficult to compare them without proper metrics. Since both models have pros and cons, it comes out to a question of tradeoffs and further analysis should be performed.

1.4 Statistical Testing

The performance of three distinct models, a Baseline model, Linear Regression, and an Artificial Neural Network (MLP) was statistically evaluated using paired t-test. The primary goal of the test was to determine if there were statistically significant differences in the performance of these models.

Table 2: Comparison of model performance using paired t-test

Model Comparison	t-statistic	p-value	95% CI Lower	95% CI Upper
LinReg vs. ANN	-7.766	2.803e-05	-0.129	-0.071
LinReg vs. Baseline	-49.828	2.649e-12	-0.475	-0.434
ANN vs. Baseline	-44.959	6.661e-12	-0.582	-0.526

From table 2, it is visible that Linear Regression model outperforms the ANN model and is also superior to the baseline, indicating it is the most effective model for this analysis. As expected, the baseline model has the worst performance, since it is a linear regression model using only the mean of the target variable y, as input feature. That is justified through extremely large negative t-statistic and p-value in comparisons where baseline model is involved, signalizing significant differences in the performance of compared models. These findings underscore the importance of model selection based on statistical significance and practical relevance to the given context.

2 Classification

In the following section a multiclass classification will be conducted. In Multiclass, the 'medium' will be added for prices between the 33rd and 66th percentile. Prior data transformation is crucial, involving standardization, feature selection, and handling missing values. Additionally, logistic regression, MLP (Multilayer perceptron) and a baseline model are compared.

2.1 Parameter selection for classification

Evaluating complexity-controlling parameters helps determine their impact on model generalization from training to unseen data, aiming to minimize overfitting while capturing key data patterns.

The complexity-controlling parameter chosen for Logistic regression classifier is the regularization parameter λ. It adjusts the model's emphasis on minimizing the coefficients, balancing between fitting the training data and maintaining simplicity to prevent overfitting. A higher λ value weakens regularization for a more complex model, while a lower λ value strengthens it for simplicity. The values chosen for examination are defined through function 'np.logspace(-1, 3, 50)', which generates 50 values logarithmically spaced between 10^-1 and 10^3. This range is chosen, as it captures the lambda with lowest observed generalization error, visible in Figure 3 below:

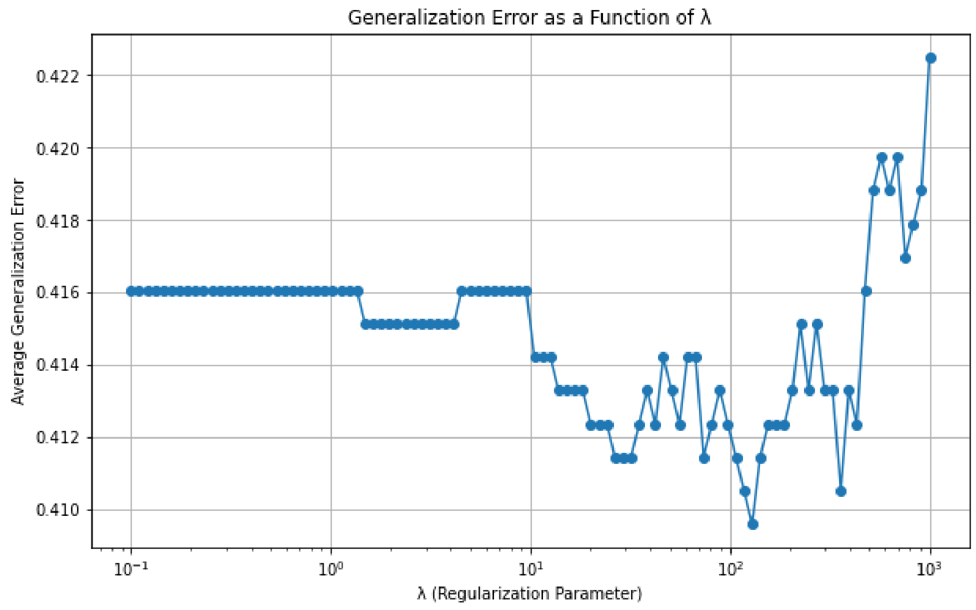

Figure 3: Generalization Error as a Function of λ for Logistic Regression Classifier

The complexity-controlling parameter for the ANN model, more precisely the MLP (Multilayer perceptron) classifier, is the number of neurons in hidden layers. It influences the model's complexity and its ability to learn patterns, too many may cause overfitting, while too few can result in underfitting. For practical reasons, the focus remains on testing different numbers of neurons within a single hidden layer. This approach allows to investigate the impact of the network's width—how many neurons it has—without the added complexity and computational cost that come with adding depth (more layers). The configurations selected for testing are the following: '[(1,), (15,), (30,),(50,)]'. This includes networks with single hidden layer containing 1, 15, 30, or 50 neurons (hidden units).

2.2 Classification models comparison

The set of 50 potential values for λ that span over a logarithmic scale between 10^{-2} and 10^{2} has been investigated. Thus, for each fold, training, validation, and comparison of 50 different models with varying regularization strengths has been performed. The λ values listed in the Table 1 are selected values from this generated set that are the best-performing for each of the 10 folds during the cross-validation procedure.

Outer fold	Logistic regression*	ANN	Baseline	λ (Logistic Regression*)	α (ANN)
1	0.433628	0.367994	0.652655	9.102982	(15,)
2	0.433628	0.373894	0.665929	0.100000	(50,)
3	0.432891	0.387168	0.676991	2.023590	(50,)
4	0.419188	0.369004	0.670849	10.000000	(15,)
5	0.434686	0.390406	0.655351	7.543120	(50,)
6	0.412546	0.354982	0.645018	4.714866	(50,)
7	0.411808	0.392620	0.682657	3.237458	(15,)
8	0.435424	0.383026	0.646494	0.145635	(50,)
9	0.433948	0.394096	0.648708	0.145635	(15,)
10	0.440590	0.391144	0.661993	10.000000	(50,)

Table 3: Two-level cross-validation results comparing three models.

It can be observed that ANN classifier outputs the best results (lowest generalization errors), for each of the ten K-folds. As expected, the baseline model displays the worst performance, since it simply predicts the most common class observed in the training data for all instances in the test set.

*'multinomial' has been passed as argument within the scikit-learn LogisticRegression() object.

Logistic Regression using optimal λ

In the following section, the Logistic regression model is trained using a most suitable value of regularization parameter λ. More precisely, the chosen value of λ, is the one which usage produces the smallest generalization error for Logistic regression model, as shown in table 1. The performance of Logistic Regression classifier for each of the three classes, was evaluated using precision, recall, and f1-score, as displayed in table 4 below.

Table 4: Classification Report for the Logistic Regression Classifier

Class	Precision	Recall	F1-Score	Support
High	0.62	0.62	0.62	458
Low	0.63	0.64	0.63	440
Medium	0.42	0.41	0.42	457

It can be observed that the Logistic Regression classifier performs the best for high and low price classes, while having a poor performance for medium price class. The more detailed performance for each class can be observed through confusion matrix (shown as heatmap) in Figure 4 below.

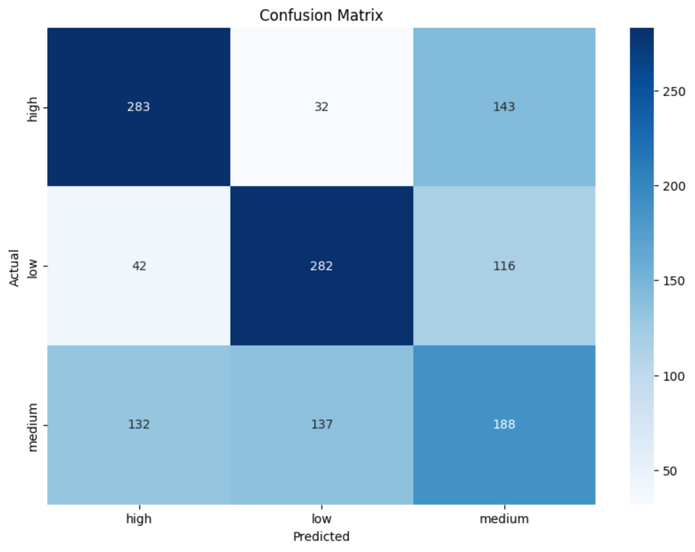

Figure 4: Confusion matrix for Logistic Regression Classifier as heatmap

For each instance to be classified, the Logistic Regression model calculates a linear combination of the feature values, each multiplied by its corresponding weight learned during the training process. This linear combination is then passed through a logistic (sigmoid) function to map the value into a probability. The function outputs the probability distribution over multiple classes (high, medium, low).

2.3 Statistical testing

The performance of three distinct models, a Baseline model, Logistic Regression, and an Artificial Neural Network (MLP) was statistically evaluated using McNemar's test. The evaluation focused on determining the presence of significant differences in predictive accuracy between each pair of models.
McNemar's test was applied to the results obtained from the cross-validated predictive models. Additionally, approximate 95 % confidence intervals for the differences in paired proportions were calculated to assess the precision of the differences.

Table 5: Comparison of model performance using McNemar's test

Model Comparison	p-value	95% CI Lower	95% CI Upper
LogReg vs. ANN	0.00024	0.343	0.449
LogReg vs. Baseline	3.876e-32	0.699	0.768
ANN vs. Baseline	8.366e-47	0.751	0.814

As expected, the model that delivered the lowest error predicted the test splits nearly the same, very high p-values for LogReg vs ANN are seen. The best performing models are making the mistakes for the same instances in dataset, increasing the p-values, as they deliver nearly the exact predictions. Since there is a statistically significant difference in the performance of the two models being compared to the baseline model, its p-value is much smaller compared to the mutual p-value.

3 Discussion

A lot was learned about classification and regression in this project. The first key insight is the importance of a good, clean and normalized dataset. Another important takeaway was that different models are ideal for different machine learning problems, both for classification and regression. This models should also be tested and compared, and to do this properly, the use of cross validation is fundamental. Moreover, getting hands on experience in applying neural networks for regression and classification problems was very positive, and understanding the impact of changes in complexity control variables and tuning hyperparameters was super interesting.

After closely investigating literature on Airbnb listings, only two papers have been identified that closely examine the data using both classification and regression techniques, indicating a promising area of research. In [1], Logistic regression and Random Forest classifiers have been implemented to predict the Airbnb listings price for 28 most popular US cities. However, the research was based on binary classification, as for this project the multiclass classification is implemented. The Random forest model from the paper outperforms all the models implemented in this project, while the Logistic regression classifier does not make predictions for one of the classes. Therefore, it can be concluded that Logistic regression classifier from this project is more reliable than the one displayed in the mentioned paper.
In [2], The Neural Network and XGBoost regressor have been implemented to predict the Beijing Airbnb listings prices. Both regressors implemented in the paper slightly outpeform the ones from this project. What probably has high impact on results is that the paper defined Neural network with multiple layers, while this project has been focused on optimizing hidden units within one layer only, due computational complexity (expensiveness).

Finally, the objectives for this project were accomplished successfully and a lot of learning was done.

4 Appendix A

4.1 Exam Problems

Question 1

Observing the given ROC curve, and taking into account the formula of True Positive Rate and False Positive Rate

$$FPR = \frac{FP}{FP+TN}; TPR = \frac{TP}{TP+FN}$$

We can conclude that the correct prediction is D.

Answer: **D**

Question 2

Using the formula of the impurity gain, which is $\Delta = I(r) - \sum_{k=1}^{K=2} \frac{N(v_k)}{r} I(v_k)$ we can conclude that $\Delta \approx 0.0074$.
Answer: **C**

Question 3

The number of parameters that has to be trained to fit the neural network can be obtained by the connections between the input and hidden layers, including their biases, and the connections between the hidden and output layers, along with their biases. In this problem there are 7 input features, 10 hidden units, and 4 output units, so by using the formula $P = n_i \times n_h + n_h + n_h \times n_o + n_o$, we get that the number of parameters will be 124.

Answer: **A**

Question 4

Answer: **A**

Branch containing congestion level 4 (white square) as input turns True when reaching point A. The related white square on a plot is in interval (0.16, 3) for b1 axis. Therefore, options B and D are ruled out. Branch containing congestion level 1 (darkest grey square) as input turns True when reaching point D. The related darkest grey square on a plot is in interval (3, -0.16) for b1 axis. Since values >= -0.16 shouldn't be considered, option C is ruled out. The only correct answer could then be A.

References

[1] Jasleen Dhillon, Nandana Priyanka Eluri, Damanpreet Kaur, Aafreen Chhipa, Ashwin Gadupudi, Rajeswari Cherupulli Eravi, and Matin Pirouz. Analysis of airbnb prices using machine learning techniques. In *2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0297–0303. IEEE, 2021.

[2] Siqi Yang. Learning-based airbnb price prediction model. In *2021 2nd International Conference on E-Commerce and Internet Technology (ECIT)*, pages 283–288. IEEE, 2021.