Danmarks Tekniske Universitet

DTU

Introduction to Machine Learning | Assignment 2

Copenhagen Apartment Price Prediction and Energy Class Classification

Authors

Dikaiopoulos Michail - s242816
Papantzikos Dimitrios - s242798
Tzimoulias Konstantinos - s242796

Contribution Table

Section	**s242816**	**s242798**	**s242796**
Regression a	30%	30%	40%
Regression b	30%	40%	30%
Classification	40%	30%	30%

Table 1: Contribution of group members per section.

November 14, 2024

Contents

1 Introduction

In this project, we aim to develop predictive models for two distinct but interconnected tasks focused on analyzing **Copenhagen apartment data**. The first task is a regression analysis to **predict apartment rent prices**, which helps understand the impact of various factors on rent, such as apartment size, location, and amenities. The initial step in our regression analysis uses a linear regression model, where we will systematically evaluate the performance through regularization and cross-validation techniques. Later, we will compare the linear model with an artificial neural network (**ANN**) and a baseline model to establish the most accurate approach for predicting rent prices.

The second task is a classification problem aimed at **predicting the energy mark of each apartment**, which is important for assessing energy efficiency. This is a multi-class classification task, but due to our small dataset we will focus on a binary classification between **energy mark A and Lower Classes**, where we will test three different models: logistic regression, a random forest classifier (selected based on initial tests), and a baseline model (most frequent class). Similar to the regression task, we will conduct a two-level cross-validation to optimize model parameters and determine each modelâs effectiveness.

Through both tasks, we will employ statistical techniques to **validate model performance**, comparing each method against baselines to **assess their generalizability**. The insights gained from these models are intended to guide future data-driven decisions related to rental pricing and energy efficiency improvements in Copenhagenâs housing market.

2 Regression on Total Monthly Rent

2.1 Simple Linear Model with Regularization

2.1.1 Overview

The primary focus of this **part a.** is to predict **'total_monthly_rent'**, which represents the monthly cost associated with renting a property. By developing a model to predict this target variable, we aim to identify the significant factors that drive rental prices and quantify their influence on the monthly rent.

2.1.2 Methodology

The primary focus of this **part a.** is to predict **'total_monthly_rent'**, which represents the monthly cost associated with renting a property. By developing a model to predict this target variable, we aim to identify the significant factors that drive rental prices and quantify their influence on the monthly rent.

To prepare the data for regression analysis, we applied two main transformations: **one-hot encoding** for categorical variables and **standardization** of numerical features. This approach creates a binary feature for each category, ensuring that each unique value of the categorical variable is treated as an independent feature. One-hot encoding is especially suitable here, as it prevents the model from assuming any ordinal relationship between categories, which aligns with our understanding that categories like "1 month" or "immediate availability" don't imply a natural order.

Furthermore, Each feature in the predictor matrix was standardized to have a **mean of 0 and a standard deviation of 1**. Standardization ensures that all features are on the same scale, which is crucial when applying **regularization techniques like Ridge that we used in part a.** Also a quite important use of regularization is that promotes faster convergence during model training and enhances interpretability by making the effect of each feature's coefficient comparable across the model.

To improve our model's predictive performance and **prevent overfitting**, we introduce a **regularization parameter** λ in our linear regression model. Regularization helps control the model complexity by penalizing large coefficients, encouraging the model to generalize better to unseen data. In this case, we'll apply **Ridge regression (L2 regularization)**, which adds a penalty proportional to the sum of squared coefficients. To determine the optimal regularization strength, we evaluate the generalization error across a range of values for λ.We select a range that spans values from very small (almost unregularized), in our case 10^{-5} to larger values i.e 10^5 that significantly constrain the model coefficients. Ideally, this range will allow us to observe a point where **the generalization error initially decreases as λ increases ,indicating a reduction in overfitting and subsequently increases, showing that excessive regularization is causing underfitting.**

For each value of λ , we estimate the generalization error using **10-fold cross-validation (K=10)**. This technique splits the data into ten equal parts, or "folds," using nine of them for training and the remaining one for testing in each iteration. This process repeats across

all folds, ensuring that each subset of the data serves as a validation set exactly once. The cross-validation method provides a reliable estimate of the generalization error by capturing how the model performs across different portions of the data.

The output y in the linear regression model with the lowest generalization error is computed by applying the modelâs optimized parameters to a given input x. For a linear model, this relationship is defined by the following equation:

$$y = \beta_0 + \sum_{i=1}^{n} \beta_i x_i$$

where:

- y: The predicted value of the target variable, in this case, *total monthly rent*.
- β_0: The intercept term, representing the baseline prediction when all features x_i are zero. This is the value of y when no predictors influence the result.
- β_i: The coefficient of the i-th feature, representing the weight or importance of that feature in the prediction.
- x_i: The value of the i-th feature for a given input x, corresponding to property characteristics like *months on website*, *availability in*, and other relevant variables.

The coefficients β_i were obtained by minimizing the generalization error using cross-validation over different regularization strengths (values of λ). The regularization step helped us avoid overfitting, resulting in coefficients that generalize well to new data.

Each attribute x_i affects the output y in proportion to its corresponding coefficient β_i. Specifically:

- A positive coefficient β_i means that increasing x_i leads to an increase in the predicted monthly rent y.
- Conversely, a negative coefficient β_i implies that increasing x_i will reduce the predicted monthly rent.

For example:

- *rooms* has a positive $\beta_i = 0.09$, so it suggests that listings with a more rooms tend to have higher monthly rent.
- *student_affordable* has a negative $\beta_i = -0.06$, it suggests that suggests properties labeled as "student affordable" are associated with lower monthly rent and thus, when a property is designated as "student affordable," it typically reduces the predicted rent amount compared to other properties.

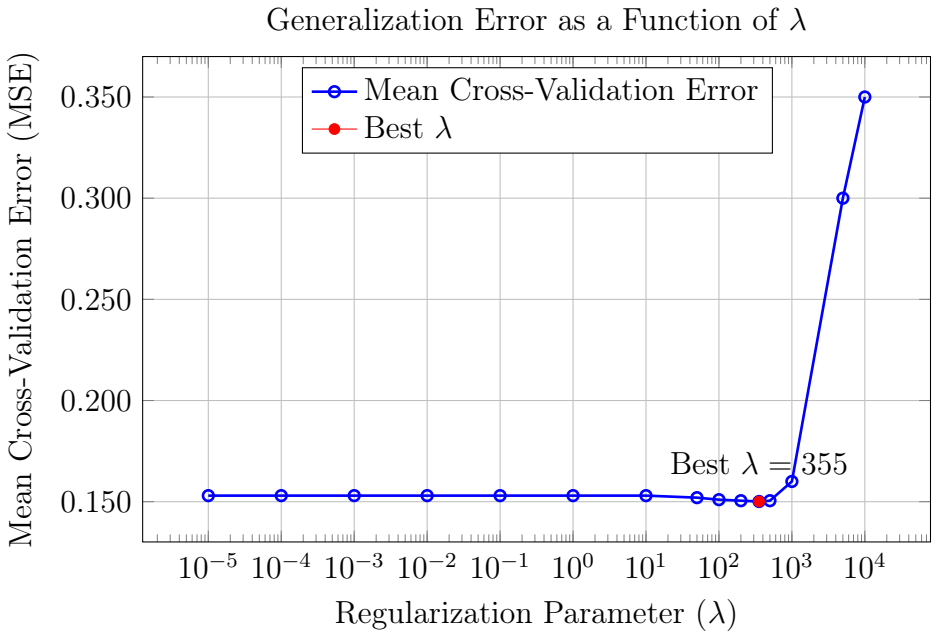


Figure 1: Generalization Error as a Function of λ, with Best $\lambda = 355$

2.2 Linear Model with Regularization and Artificial Neural Network Comparison

In this section, we compare three models: the regularized linear regression model from the previous section, an artificial neural network (ANN), and a baseline model. We aim to address two key questions:

- **Is one model significantly better than the others?**
- **Are any models significantly better than a trivial baseline?**

These questions are explored using two-level cross-validation with $K_1 = K_2 = 10$ folds. It is worth to mention that all the analysis below has been done with a fixed random state (42).

2.2.1 Hyperparameter tuning and generalization error estimation with 2-level 10-fold cross validation

To fairly assess each model, we implement two-level cross-validation. For each outer fold, we find the optimal values for the complexity-controlling parameters in the inner fold. Specifically, we tune:

- **The regularization parameter** (λ) for linear regression

– **The number of hidden units** (h) for the ANN

This procedure is repeated across all outer folds, and the resulting models are evaluated on the corresponding test sets to estimate their generalization errors.

Outer fold	**ANN**		**Ridge regression**		**Baseline**
i	h_i^*	E_i^{test}	λ_i^*	E_i^{test}	E_i^{test}
1	64	147.54	50000	198.3	198.65
2	64	172.61	10000	167.54	168.06
3	64	193.56	25000	241.9	242.69
4	64	343.19	50000	413.61	413.91
5	64	1.64	50000	0.44	0.51
6	64	232.1	50000	242.23	242.66
7	64	0.28	10000	0.17	0.31
8	64	3.6	50000	0.64	0.72
9	64	43.02	10000	139.98	140.3
10	64	126.1	25000	154.77	155.31

Table 2: Parameter values and test error of each model per outer fold in scale of 10^8

– In general, ANN with 64 hidden units exhibits lower test errors in most folds compared to Ridge Regression and the Baseline model, suggesting it may have better predictive performance overall.

– In specific folds (e.g., Fold 1, 3, and 4), Ridge Regression and the Baseline model have notably higher errors, indicating potential variability in the model's performance depending on the data in each fold.

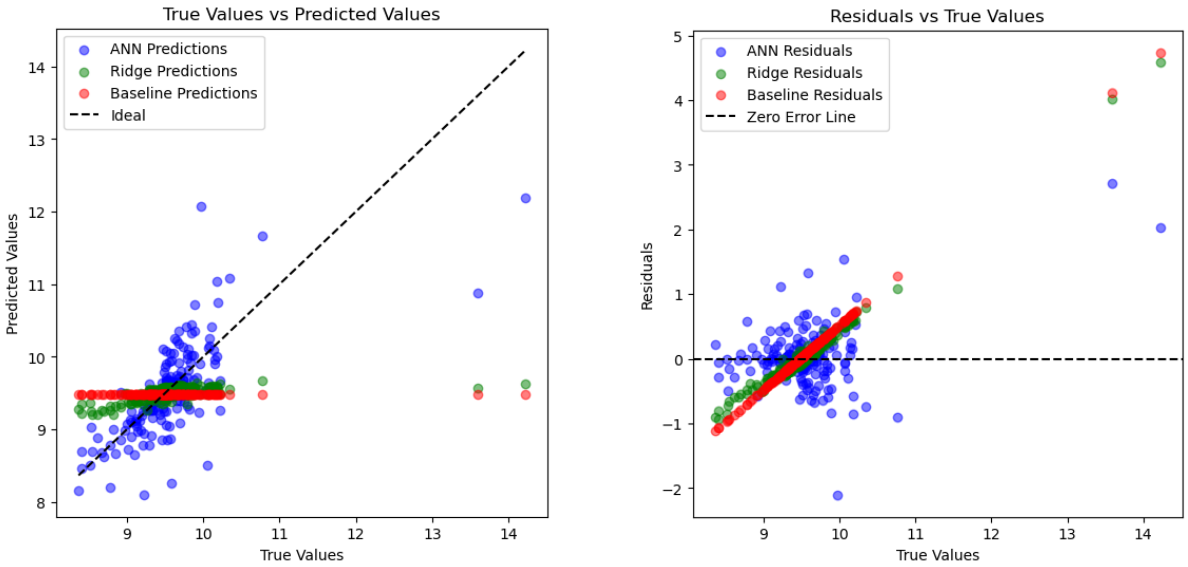
– The Baseline test errors are relatively consistent across folds, serving as a benchmark for comparing the performance of the trained models.

2.2.2 Results and Analysis

To visualize the results, we present 2 plots for the overall best models, namely, the ANN with 2 hidden layers and 64 hidden units in the first layer and the Ridge Regression with $\lambda = 50000$.

– **True vs. Predicted Plot (Figure 2a)**:

∗ The Artificial Neural Network (ANN) predictions (blue points) are more closely aligned with the ideal line (dashed line) compared to Ridge Regression (green points) and the Baseline model (red points). This indicates that ANN generally provides predictions that are closer to the true values, suggesting a higher accuracy.



(a) True vs Predicted

(b) True vs Residuals

Figure 2: True vs Predicted and Residuals

  * The Ridge and Baseline predictions tend to cluster around a narrow range, showing limited variation in predictions, which suggests they are less responsive to the true values compared to ANN.

- **Residuals vs. True Values Plot (Figure 2b)**:
  * The residuals of the ANN model (blue points) are distributed more closely around the zero error line (dashed line), indicating lower errors and a better fit.
  * The Ridge and Baseline models exhibit higher residuals for certain values, with more deviation from the zero error line, especially for larger true values. This suggests that these models are less accurate and may have greater prediction errors.
- **Overall Comparison**:
  * The ANN model appears to outperform both Ridge Regression and the Baseline model, as evidenced by its closer alignment with the ideal line in the True vs. Predicted plot and its lower residuals in the Residuals vs. True Values plot. This suggests that ANN provides a better fit and more accurate predictions across the dataset. On top of that, the ANNs residuals seems like random, showcasing a good capture of variability, whereas the residuals for the Ridge seem to follow the residuals of the baseline predictions, which show a positive linear trend.

2.2.3 Model comparison with statistical tests on Setup II

To determine if the models perform significantly differently, we conducted pairwise comparisons of test errors using the paired t-test. The results, including p-values and confidence intervals, are shown below.

	ANN vs Ridge Regression	**ANN vs Baseline**	**Ridge Regression vs Baseline**
Mean Difference	-0.0548	-0.0987	-0.0439
Std. Dev	0.0764	0.0797	0.0080
P-Value	0.0495	0.0035	0.0000
Confidence Interval	(-0.1095, -0.0002)	(-0.1557, -0.0416)	(-0.0496, -0.0382)

Table 3: Comparison of ANN, Ridge Regression, and Baseline Models

The results of the comparisons among the models are as follows:

- **ANN vs Ridge Regression:** With a mean difference of -0.0548, a significant p-value of 0.0495, and a 95% confidence interval of $(-0.1095, -0.0002)$, ANN with 64 hidden units seems to perform better than Ridge Regression with $\lambda = 50000$.
- **ANN vs Baseline:** ANN outperforms the Baseline model, indicated by a mean difference of -0.0987, a strong p-value of 0.0035, and a confidence interval of $(-0.1557, -0.0416)$ showing improvement.
- **Ridge Regression vs Baseline:** Ridge Regression also outperforms the Baseline with a mean difference of -0.0439, a highly significant p-value of less than 0.00001 (rounded), and a narrow confidence interval of $(-0.0496, -0.0382)$, indicating reliable improvement.

3 Classification on Energy Mark

3.1 Overview

In our classification problem, we chose to classify our apartments based on their energy consumption. As we were examining our dataset, we found out that 45% percent of the apartments had missing energy mark data. With a little bit of research, we found out that in many cases, apartment owners choose not to publish their apartment's energy mark in a rental website, either because they do not want to lower their chance of renting the apartment due to poor energy ratings, or because they want to avoid the cost and the complexity of getting an official energy certification for their apartment.

However, it becomes increasingly important for future tenants to know the consumption and the energy footprint that their apartment has, so a rental mediator should be able to at least provide a prediction for this information.

3.2 Methodology

We modeled our problem as a binary classification problem, where we classify our apartments as "Energy class A" or "Lower classes". Our data has more detailed energy marks like "A", "B", "C", etc., but we chose this simplification due to the limited amount of data that we currently have and the importance that we want to give to class A apartments, compared to the rest of the energy classes. To evaluate the efficiency of the logistic regression classifier, we compare it with a baseline and a random forest classifier using a 2-level 10-fold cross validation (for optimising hyperparameters and estimating generalization error) and statistically evaluate the performance of each model.

3.2.1 Models and parameters

We will evaluate and train the following three classifiers:

- **Logistic Regression Classifier** Our focus will be on evaluating the performance of this classifier, compared to the other two. The parameter we will be tuning in the inner cross validation layer is the L2 regularization parameter (or simply λ). We will use a range of 10 different λ values from 10^{-4} to 10^{5}
- **Random Forest Classifier** This model uses the output of multiple decision trees to reach a single result. We chose it as a comparison model, because it tends to perform well in high dimensional data with an observable imbalance in the data classes like our dataset. The hyperparameter that we will tune is the number of estimators that the model uses, i.e. the number of the individual decision trees in the forest. We will use a range of 10 different number of estimators, from 10 to 1500.
- **Baseline Classifier** This model will always predict the majority class in the training data and is used as the simplest benchmark for logistic regression's performance. It has no parameters to tune.

3.2.2 Challenges

During the implementation of the three models, we faced the following challenges:

- **Skewed variable distributions and variables in different scales**. To overcome this, we transformed logarithmically that had extreme outliers and then standardized our data.

– **Insufficient number of data for multi-class classification**. As we mentioned before, due to the lack of data for several energy classes, we reformed our problem into a binary classification one.

3.3 Results

3.3.1 Cross validation output

Our nested 10-fold cross validation for the three aforementioned models produces the following results:

Outer fold	**Logistic Regression**		**Random Forest**		**Baseline**
i	λ_i^*	E_i^{test}	$n\_estimators_i^*$	E_i^{test}	E_i^{test}
1	1.0000	0.121212	100	0.131313	0.424242
2	0.1000	0.111111	500	0.141414	0.434343
3	0.0100	0.191919	300	0.141414	0.404040
4	0.1000	0.121212	750	0.131313	0.424242
5	0.0001	0.080808	500	0.101010	0.454545
6	0.1000	0.071429	200	0.102041	0.418367
7	1.0000	0.132653	1500	0.112245	0.448980
8	0.1000	0.142857	50	0.102041	0.459184
9	0.1000	0.255102	1000	0.224490	0.438776
10	1.0000	0.163265	50	0.183673	0.448980

Table 4: Parameter values and test error of each model per outer fold

3.3.2 Statistical evaluation of the results

If we now take the generalization errors produced for each model in each outer loop of the cross validation technique we previously used and perform a statistical evaluation of the three models with a 95% confidence interval we have:

– **Logistic Regression vs Baseline**

Confidence interval is **{-0.3378, -0.2551}** and P_{value} is **0.0000001**. The confidence interval for the mean difference is negative and quite big and P_{value} is very small, below the 0.05 significance level, so we are confident that logistic regression outperforms significantly the baseline model.

– **Random Forest vs Baseline**

Confidence interval is **{-0.3303, -0.2666}** and P_{value} is **0.0000000**. The interpretation of the results is the same as in the logistic regression vs baseline case.

– **Logistic Regression vs Random Forest**

Confidence interval is **{-0.0198, 0.0239}** and P_{value} is **0.0005576**. In this case, the confidence interval includes 0 and is smaller than the previous two cases. P_{value} is much bigger than 0.05, so there is insufficient evidence to suggest a statistically significant difference and we can't tell for sure which model performs better than the other.

3.3.3 Logistic regression classifier training with optimal λ

From cross-validation, we saw that setting $\lambda = 0.1$ we achieve the best generalization error, so we will use this value for our regularization value while training. Using 90% of our data for training we achieve an accuracy of 0.837% in test data. Below is the confusion matrix of our test results.

		Predicted Class	
		Energy class A	**Lower classes**
Actual Class	**Energy class A**	53	3
	Lower classes	8	35

.

4 Exam Problems for the Project

Question 2. Spring 2019 question 15

For this problem the first thing we need to do is calculate the parent node (overall) classification error E_{parent} which is calculated based on the maximum class proportion:

$$\text{Total Observations} = 135$$
$$p(y=1) = \frac{33+4+0}{135} = 0.274$$
$$p(y=2) = \frac{28+2+1}{135} = 0.230$$
$$p(y=3) = \frac{30+3+0}{135} = 0.244$$
$$p(y=4) = \frac{29+5+0}{135} = 0.252$$

The maximum class proportion is $\max(p(y=1), p(y=2), p(y=3), p(y=4)) = 0.274$. So that gives us,

$$E_{\text{parent}} = 1 - 0.274 = 0.726$$

Now we need to calculate the impurity of both left and right branches

For the left branch (where $x_7 = 2$), we only have one observation in class $y = 2$, so the classification error E_{left} is:

$$E_{\text{left}} = 0$$

For the right branch (where $x_7 \neq 2$), we combine the counts from $x_7 = 0$ and $x_7 = 1$:

$$\text{Total Observations (Right Branch)} = 120 + 14 = 134$$

Class proportions in the right branch are:

$$p_{\text{right}}(y = 1) = \frac{37}{134} \approx 0.276$$
$$p_{\text{right}}(y = 2) = \frac{30}{134} \approx 0.224$$
$$p_{\text{right}}(y = 3) = \frac{33}{134} \approx 0.246$$
$$p_{\text{right}}(y = 4) = \frac{34}{134} \approx 0.254$$

The maximum class proportion in the right branch is $\max(0.276, 0.224, 0.246, 0.254) = 0.276$. Therefore,

$$E_{\text{right}} = 1 - 0.276 = 0.724$$

Now for the impurity gain Δ for the split at $x_7 = 2$ we use the formula:

$$\Delta = E_{\text{parent}} - \left(\frac{n_{\text{left}}}{n_{\text{total}}} E_{\text{left}} + \frac{n_{\text{right}}}{n_{\text{total}}} E_{\text{right}} \right)$$

Substituting the values:

$$\Delta = 0.726 - \left(\frac{1}{135} \cdot 0 + \frac{134}{135} \cdot 0.724 \right)$$

$$\Delta \approx 0.0074$$

The impurity gain $\Delta \approx 0.0074$, corresponds to option **C**.