Danmarks Tekniske Universitet

DTU

02450 Project 2

Authors

Group 192:
Amine Lbath - s221881
Eirik Runde Barlaug - s221409
Sara Bakken Heiberg - s221630

Student	Section 1	Section 2	Section 3	Section 4	Section 5	Exam questions
s221881	33.33%	30%	40%	30%	33.33%	33.33%
s221409	33.33%	30%	30%	40%	33.33%	33.33%
s221630	33.33%	40%	30%	30%	33.33%	33.33%

Table 1: Contribution of each member

November 15, 2022

Contents

1 Introduction **1**

2 Regression, part a **1**

3 Regression, part b **3**
3.1 Model comparison: two-level cross-validation 3
3.2 Statistical evaluation of the models: setup II 4

4 Classification **5**
4.1 Model comparison 6
4.2 Statistical evaluation of the classifiers: The McNemar's test 7
4.2.1 Contingency tables 7
4.2.2 Result 8
4.3 Implementing the multinomial regression classifier 9

5 Discussion **10**

References **11**

6 Appendix A: The coefficients for the multinomail regression classifier **12**

7 Appendix B: Theory behind statistical testing **12**
7.1 Hypothesis and p-value 12
7.2 Confidence intervals 13

8 Appendix C: Exam problems **14**

1 Introduction

This project continues of project 1 about feature extraction and visualization in the course 02450 Machine Learning and Data Mining. We are using the same Energy Efficiency data set [1], composed of energy analysis of 768 unique buildings. The goal of this project is to use supervised learning in order to solve both a relevant classification and regression problem for our data, i.e. predicting cooling and heating loads based on buildings features.

2 Regression, part a

The overall idea of regression is to examine whether a set of explanatory variables can be used to predict a target variable, and to see which variables in particular are significant predictors of the response variable. We start off by examining the most elementary model for regression, namely linear regression. Here, our goal is to predict the cooling load based on the independent variables, i.e. the eight building parameters. In particular, this means finding a model where the cooling load is given as a linear combination of the eight building features.

Before we start feeding the data into our machine learning model, it needs to be preprocessed. The linear regression model can get improved if our data is standardized, because no input feature overtake the others. Therefore, we applied a feature transformation on the building feature matrix $\mathbf{X}$ such that each column (building feature) has mean 0 and standard deviation 1.

In order to avoid overfitting of our machine learning model, we introduce a regularization parameter, λ. Instead of using a simple residual sum of squares to measure the discrepancy between the output data and the output of our linear model, we use ridge regression to prevent learning an over-complex model. Ridge regression minimizes the residual sum of squares plus a penalty (λ) multiplied by the sum of squares of the regression coefficients. As λ increases, the coefficients approach zero, while when λ is zero, the model is unregularized.

By adding different degrees of penalties to the weights in the linear regression, regularization allows us to build different regression models. The most appropriate choice of a regularization (λ) is then made using K fold cross-validation.

Regularization does not improve the performance on the training data that the algorithm used to learn the feature weights. Instead, it can improve the overall performance, i.e., the performance on new, unseen data - which is exactly what we aim to do. This is why we are interested in studying the generalization error. For each regularization parameter λ, we compute the generalization error when training the model with that specific regularization parameter. The optimal model is the one that minimizes the K fold cross-validation error.

Figure 1 shows the generalization error as a function of λ when we are using K=10 fold cross-validation for model selection.

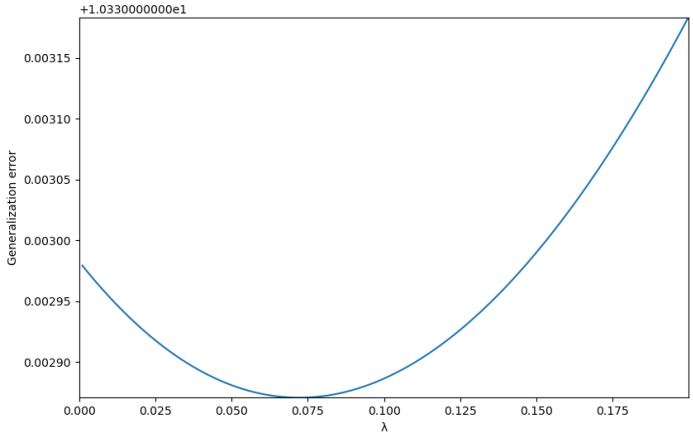


Figure 1: Generalization error as a function of λ

The plot shows that the optimal λ-value is around 0.07. Increasing or decreasing λ will lead to an increase in the generalization error. However, increasing λ will increase the slope of the generalization error even more than when decreasing it. This indicates that for our data, the model is doing best when λ is small, i.e. the model is nearly unregularized.

In order to see what impact the different building parameters have on the cooling, we need to look at the weighting of the attributes.

Attribute	Weighting
Relative compactness	-7.355
Surface area	-3.841
Wall area	0.063
Roof area	-3.776
Overall height	7.610
Orientation	0.098
Glazing area	2.034
Glazing area distribution	0.040

Table 2: Ridge regression coefficients

Table 2 shows the coefficients from the ridge regression, i.e. the weighting of the attributes. Clearly, the wall area, orientation and glazing area distribution have almost no impact. The surface area, roof area, glazing area, and especially the relative compactness and the overall height have a great impact on the cooling load. Positive weights indicate that if a building has a large value of this feature, it will more likely be predicted a higher cooling load. On the other hand, a negative weight indicates that a large value of the corresponding feature will more likely predict a lower cooling load. Intuitively, this makes sense as a compact building is more energy efficient and hence a large value will contribute with a

large reduction of the cooling load. On the contrary, a tall building will increase the cooling load.

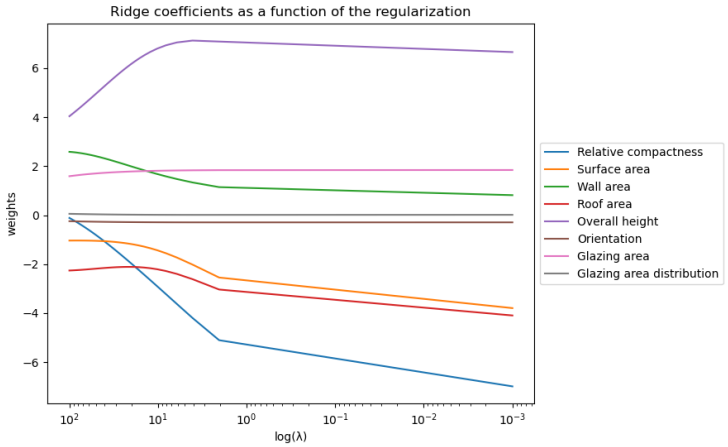


Figure 2: Ridge coefficients as a function of the regularization

In the figure above 2, the ridge coefficients are plotted as a function of λ. Each color represents a different feature of the coefficient vector. The figure shows the importance of using the right value of λ. When λ is large, the regularization effect dominates and the coefficients tend to zero. As λ tends toward zero and the solution tends towards the ordinary residual sum of squares, the coefficients for overall height increases and the coefficient for relative compactness decreases. The other coefficients seems to be more unaffected by the choice of λ. This reflects the results in Table 2.

3 Regression, part b

The goal of this section consists of finding a good model to predict the cooling load of buildings based on their characteristics. We will train on our dataset using a linear regression model and an artificial neural network (ANN). First, we are going to find the best set of hyperparameters to use to train our models. Then we will compare our regularized linear regression model to our ANN. Finally, we will try to understand if our models actually perform better than a trivial baseline model.

3.1 Model comparison: two-level cross-validation

To compare the three models, i.e. linear regression, ANN and baseline, and find the best set of hyperparameters, we will use a 2-level cross-validation with 10 folds on each level. As done in the previous section, we are going to test different regularization strength λ for the linear regression. As for the ANN, we will use the number of hidden units h as a complexity-controlling parameter. The baseline model simply corresponds to a linear regression model with no features, i.e. it always returns the mean of the cooling load on the training data.

The chosen hyperparameters should cover a range of values as big as possible to ensure it contains the optimal value, i.e. the one that yields the minimum test error E_i^{test} for each outer fold i. We chose the following set of hyperparameters:

$$\lambda\text{-grid} = \{\lambda_1, \lambda_2, \lambda_3, ...\} = \{10^{-7}, 10^{-6}, 10^{-5}, \ldots, 10^{0}, 10^{1}, 10^{2}\}$$
$$h\text{-grid} = \{h_1, h_2, h_3, ...\} = \{32, 64, 128, 256, 512, 1024\}$$

After running the running the cross-validation for almost five consecutive hours, we obtained the following values:

Outer fold	Linear regression		ANN		Baseline
i	λ_i^*	E_i^{test}	h_i^*	E_i^{test}	E_i^{test}
1	0.10	8.484	256	0.676	76.638
2	0.10	9.597	128	0.485	90.248
3	0.01	12.432	128	0.798	93.629
4	0.10	15.093	128	0.572	102.308
5	0.10	9.958	128	0.727	107.844
6	0.10	8.523	128	0.592	70.230
7	0.10	12.025	512	0.292	81.191
8	0.10	9.894	128	0.466	102.685
9	0.10	7.893	256	0.421	85.439
10	0.10	10.830	256	0.493	96.222

Table 3: Two-level cross-validation table for comparison of a regularized linear regression model, an ANN and a baseline model in the regression problem.

The above table 3 displays the best hyperparameters for each fold, as found after running the corresponding inner folds. It also shows the corresponding estimated generalization errors. It includes as well the baseline test error for each fold.

As we can see the optimal λ for the linear regression corresponds to 0.1. This confirms the value we obtained in the previous section. For the artificial neural network, the optimal number of hidden units h is equal to 128. Besides, it seems that the overall the ANN outperforms the linear regression which outperforms the baseline model. However, this intuition does not account for a proof in any way. This leads us to the next section.

3.2 Statistical evaluation of the models: setup II

In this section we are going to statistically evaluate if there is a significant performance difference between the linear regression, the fitted ANN and the baseline models. We will compare these models pairwise using the setup II, a statistical test of performance considering a dataset of size N. The setup II is more relevant than the setup I (only considering the specific training set D) to compare the models as it leads to more general conclusions. In this section, we consider a statistical evaluation which based on the correlation heuristic compares estimates of the true generalization error $E^{gen} = \int[\int L(f_{_{D^{train}}}(x), y)dxdy]dD^{train}$.

By running a 10-fold cross validation on a randomly shuffled dataset, we obtain the results summarised in the table 4 bellow:

Pair	θ_L	θ_U	p-value	Reject H_0?
(ANN, Baseline)	80.59	99.15	4.07e-09	Yes
(Regression, Baseline)	72.64	87.87	1.91e-09	Yes
(ANN, Regression)	7.64	11.58	1.54e-06	Yes

Table 4: The results of the pair wise tests for the three regression models

As we can see the estimated difference in performance between ANN / Baseline, and Regression / Baseline is quite high, around 70-100. The very low p-value of about 10^{-9} suggests that the chance is very high that both ANN and Regression is better than baseline.

The estimated difference in performance between the ANN and the Regression is not as high as the previous but still quite high, around 10. The p-value for this test is still very much lower than the threshold, $\alpha = 0.05$. We can therefore reject H_0 and conclude that the ANN is better than the linear regression.

To sum up, the artificial neural network is more efficient than both the baseline model and the linear regression. This not much of a surprise as the ANN allows to solve more complex and non-linear regression tasks.

4 Classification

In this part of the report, we will solve a relevant classification problem for the data and statistically evaluate the results. The tasks will closely mirror what we did in the last section. The three methods we will compare is a baseline, multinomial regression and k-nearest-neighbors (KNN) classification.

For the abovementioned models, the complexity controlling parameters are $k = 1, 2, 3, ... =$ number of neigbours for the KNN, and $\lambda \geq 0$ for the multinomial regression (as described in the regression part).

For this task, we will classify the discretized heating load. The data has been discretized as follows: Given some data point $\mathbf{x}$ and its heating load $HL(\mathbf{x})$ we have that:

$$\begin{aligned}
\mathbf{x} \in \text{Class } 0 &\iff 0 \leq HL(\mathbf{x}) \leq Q_1(HL(\mathbf{x})) = 12.9925 \\
\mathbf{x} \in \text{Class } 1 &\iff Q_1(HL(\mathbf{x})) < HL(\mathbf{x}) \leq Q_2(HL(\mathbf{x})) = 18.95 \\
\mathbf{x} \in \text{Class } 2 &\iff Q_2(HL(\mathbf{x})) < HL(\mathbf{x}) \leq Q_3(HL(\mathbf{x})) = 31.6675 \\
\mathbf{x} \in \text{Class } 3 &\iff HL(\mathbf{x}) \geq Q_3(HL(\mathbf{x})),
\end{aligned}$$

where Q_i is quantile i. For that reason, our classification is that of a *multiclass classification problem*, with 4 classes, and the *heating load* is our target variable. The number of instances in each class is the same for all four classes, due to the splitting based on quantiles.

4.1 Model comparison

When comparing the three models we will use two-level cross-validation. For the baseline classifier, we classify all instances from the test dataset as belonging to the largest class in our training dataset. Furthermore, the error from each fold will be calculated as $E_i^{test} = \frac{\#\text{ misclassified observations}}{\#\text{ observations in the test dataset}}$.

The two-level cross-validations splits the data into $K_1 = 10$ outer folds and $K_2 = 10$ inner folds. We find the optimal hyperparameters λ and k (for a given range) for our two models for each inner fold, and computes the test error E_i^{test} for each outer fold i, given the hyperparameter chosen. The best hyperparameter for each run is the one that yields the smallest test error. By performing 10 test runs on 10 different (shuffeled) train-test splits of the data, we obtained an average error for each k and λ for a large range of the two hyperparameters. The average error across the 10 test runs for each k and λ is given in figure 3. **Note: By convention, sklearn use the inverse of the regularization, $\mathbf{c = \frac{1}{\lambda}}$, to control complexity so from here on c is sometimes used.** To obtain λ we simply apply the inverse of c.

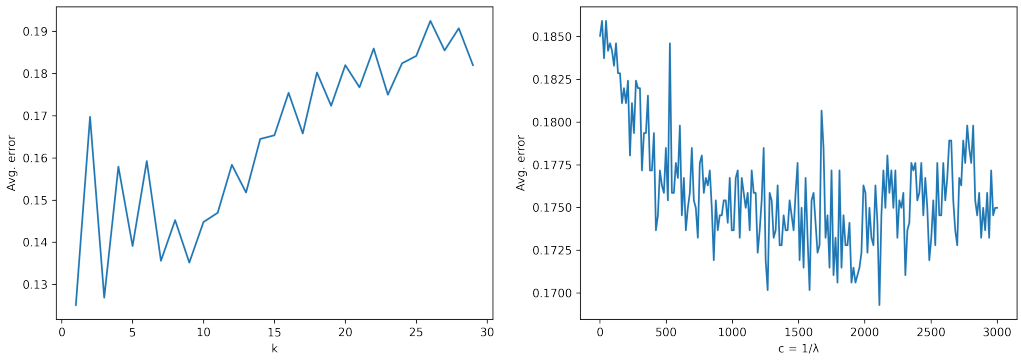


Figure 3: Mean prediction error as a function of k and $c = \frac{1}{\lambda}$, across 10 runs with shuffeled train-test splits.

From the plots, as well as some additional trial and error runs, we created the smaller ranges for the hyperparameters. Namely,

$$c\text{-grid} = \{\frac{1}{\lambda_1}, \frac{1}{\lambda_2}, \frac{1}{\lambda_3}, ...\} = \{1500, 1550, 1600, 1650, \ldots, 1950, 2000\}$$
$$k\text{-grid} = \{k_1, k_2, k_3, ...\} = \{\texttt{np.linspace(0.1, 10, num=20)}\}.$$

Decreasing the number of neighbors k and *decreasing* the regularization term λ (i.e. increasing c in the code) both leads to *increased* model complexity (i.e. higher variance and lower bias). The results of the two-level cross-validation with the given hyperparameter grids are given in table 5. For this part we used one third of the data for testing, which means the test set consisted of 253 observations, and the training set consisted of 515 observations.

Outer fold	KNN		Multinomial regression		Baseline
i	k_i^*	E_i^{test}	$c_i^*(\lambda_i^*)$	E_i^{test}	E_i^{test}
1	3	0.1039	1650 (6.06e-04)	0.1688	0.8442
2	1	0.0390	1600 (6.25e-04)	0.2208	0.8182
3	1	0.0909	1900 (5.26e-04)	0.1688	0.8312
4	1	0.0779	1650 (6.06e-04)	0.1299	0.7792
5	7	0.1688	1650 (6.06e-04)	0.0519	0.7792
6	1	0.1298	1700 (5.88e-04)	0.1818	0.8182
7	1	0.0779	1700 (5.88e-04)	0.2078	0.8961
8	1	0.0909	1750 (5.71e-04)	0.2208	0.7662
9	1	0.0789	1600 (6.25e-04)	0.1447	0.8158
10	1	0.1316	1600 (6.25e-04)	0.2763	0.7632

Table 5: Two-level cross-validation table for comparison of the three models.

From table 5, the choice of k obviously fell on $k^* = 1$. For λ, we chose $\lambda^* = \frac{1}{1600}$. From the table, it seems as if the KNN-classifier performs better than the multinomial regression classifier. However, on some splits the multinomial regression performs better than the KNN. Furthermore, this difference in performance is not yet proven statistically - which brings us to the next chapter.

4.2 Statistical evaluation of the classifiers: The McNemar's test

In this section we will perform a statistical evaluation of the three models similar to that of the previous section. We will compare the models pair-wise with a *McNemar's test*. The McNemar's test is a paired non-parametric or distribution-free statistical hypothesis test where we compare the hit-or-miss vectors for the predictions of two classifiers. Our goal is to find out if two models differ in performance or not. For all the three pair-wise tests we obtain a p-value and a confidence interval. The McNemar's test algorithm applied in this project is taken from *Tue Herlau, Mikkel N. Schmidt and Morten Morup: "Introduction to Machine Learning and Data Mining", Lecture notes, Spring 2022, version 1.0, p.204 - 207.*

4.2.1 Contingency tables

The McNemars test is based on the 2×2 contingency table of the classifiers. If we define the elements of our hit-or-miss vectors for a given classifier C_k as

$$c_i^{[k]} = \begin{cases} 1, & \text{if } \hat{y}_i^{[k]} = y_i \\ 0, & \text{else,} \end{cases} \tag{1}$$

then the contingency table for two classifiers C_A, C_B with n test samples reads as in table 6.

	C_B correct	C_B incorrect
C_A correct	$n_{11} = \sum_{i=1}^{n} c_i^{[A]} c_i^{[B]}$	$n_{12} = \sum_{i=1}^{n} c_i^{[A]} (1 - c_i^{[B]})$
C_A incorrect	$n_{21} = \sum_{i=1}^{n} (1 - c_i^{[A]}) c_i^{[B]}$	$n_{22} = \sum_{i=1}^{n} (1 - c_i^{[A]})(1 - c_i^{[B]})$

Table 6: The 2x2 contingency table.

Given the chosen hyperparameters from chapter 4.1 we obtained the contingency tables in figure 4.

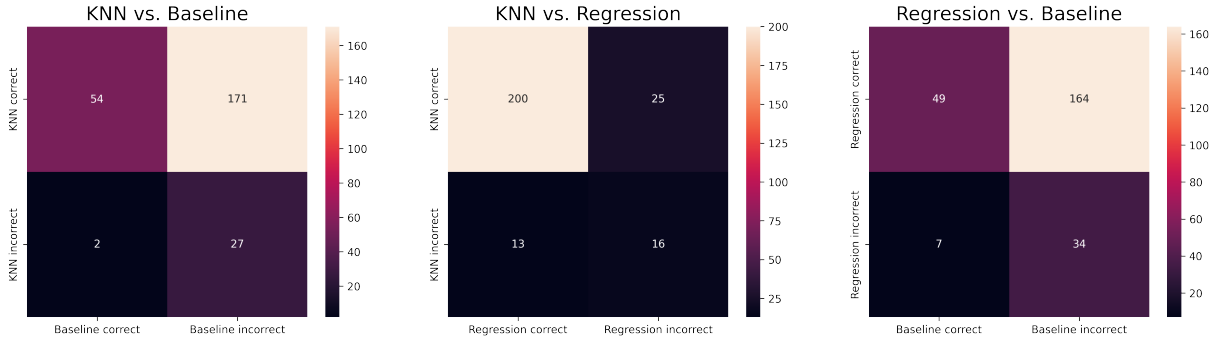


Figure 4: Contingency tables for the pair-wise model comparisons.

In the appendix 7, we detailed a bit more the theory behind statistical testing.

4.2.2 Result

Based on the contingency tables displayed in figure 4, we then computed the estimated difference in performance, $\hat{\theta}$, the confidence interval , $[\theta_L, \theta_U]$, and the p-values for the three pair-wise tests. The results from the tests are given in table 7.

Pair	$\hat{\theta}$	θ_L	θ_U	p-value	Reject H_0?
(KNN, Baseline)	0.6654	0.6034	0.7212	0.000000	Yes
(KNN, Regression	0.0472	0.0001	0.0943	0.0730	No
(Regression, Baseline)	0.6181	0.5498	0.6821	0.000000	Yes

Table 7: The results of the pair wise tests for the three classifiers

From table 7 it is clear that both the KNN classifier and the multinomial regression classifier is better than the baseline. The estimated difference in their performance is also high, around 0.6-0.7 in both cases, with their respective confidence intervals. The very low p-value suggests that the chance is very high that both KNN and Regression is better than baseline. For the comparison between the KNN and Regression, the estimated difference in performance is low. It may of course be lower or higher than the estimate, but most likely (with 95% certainty) it is inside the confidence interval. The p-value for this test not as small as for the two other tests, in fact it is larger than our threshold, $\alpha = 0.05$. We can

therefore *not* reject H_0 and conclude that the KNN classifier is better than the multinomial regression classifier. We can only be certain that both models beat the baseline, but the models may be equally accurate.

4.3 Implementing the multinomial regression classifier

Finally, the multinomial regression classifier (MRC) was implemented with $\lambda^* = \frac{1}{1600}$. The accuracy and confusion matrix for the classifier is given in equation 2 and figure 5, respectively.

$$\text{Acc}_{\text{MRC}} = 0.8189 \tag{2}$$

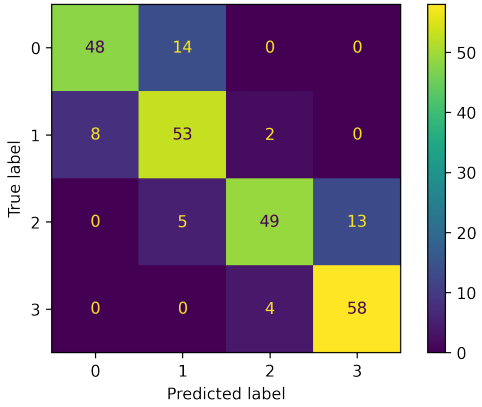


Figure 5: The confusion matrix for the multinomial regression classifier implemented with the optimal λ.

These are quite reasonable results, and the performance of the multinomial regression classifier is satisfactory. The coefficients of the multinomial regression did seem to be similar to the ones in the part on linear regression for some of the variables. The interpretation of the coefficients is the same for multinomial regression, however they are now associated with the probability assigned to each class, and is this a (n_classes, n_attributes) shaped matrix. The similarities are for the variable X_5, which seems to be important in both models, as well as X_6, which seems to be unimportant in both models. This is true for all classes. The other coefficients differ across classes - seems as if the model has quite high variance.

5 Discussion

In this project, we have used two different approaches to create models to predict energy performances of buildings: classification and regression. In the regression section we predicted a continuous quantity of buildings cooling load based on building characteristics. We made predictions using a linear regression model using L_2 regularization and an artificial neural network. We also used a two-level cross validation to determine the best set of hyperparameters for these models, i.e., regularization strength and number of hidden units. Ridge regression showed that regularization did not improve that much our linear regression model. After performing statistical testing, we found out that both models performed better than the oversimplistic baseline model. Our results showed that the ANN also outperforms the regression model to predict the cooling load.

The original 2012 paper *Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools* [2] written by Athanasios Tsanas and Angeliki Xifara also focused on predicting energy loads of buildings. More specifically, they used a random forest model to predict cooling loads. As a best result they obtained a mean squared error of 6.59 ± 1.56. By comparison, our ANN never performed worse than 0.8 in the cross-validation. Hence, it seems that our model outperforms by far the 2012 state of the art random forest model from the paper. Though, some statistical testing should still be performed to compare more formally the two models.

In classification we discretized the heating load and predicted the discrete class that the input data is attributed to from the identified class labels in the training dataset. The classification shows that both the KNN-model and the multinomial regression model outperforms the baseline when it comes to predicting the class (that is, the quantile) of the heating load. However, the difference in the twoâs performance is not high enough to statistically guarantee that one is better than the other. Furthermore, the lowest error rate we could obtain with multinomial regression was around 0.15 when we implemented it with the optimal hyperparameter λ^*.

The original paper did not use classification to predict heating loads. Hence, we cannot compare our classifiers with the reference paper. However, using the same method as in the regression part, we trained an ANN to predict the heating load. The resulting mean squared error was always lesser than 0.05, whereas in the paper their error was around 1.03 ± 0.54. Again, our model seems to perform better than the one from the paper.

In this project we managed to find and compare regression and classification models using cross-validation and statistical testing. In the end, we obtained really good models that seem to outperform the ones from our 2012 source paper.

References

[1] A. Tsanas and A. Xifara, "Energy efficiency data set," 2012. data retrieved from UCI machine learning repository, `https://archive.ics.uci.edu/ml/datasets/Energy+efficiency`.

[2] A. Tsanas and A. Xifara, "Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools," *Energy and Buildings*, vol. 49, pp. 560–567, 2012.

6 Appendix A: The coefficients for the multinomail regression classifier

Class	$\mathbf{X_1}$	$\mathbf{X_2}$	$\mathbf{X_3}$	$\mathbf{X_4}$	$\mathbf{X_5}$	$\mathbf{X_6}$	$\mathbf{X_7}$	$\mathbf{X_8}$
C_1	3.7184	10.9417	-9.6886	15.3488	-20.1526	0.2450	-17.1545	0.7088
C_2	0.7934	-2.3179	-1.7037	- 1.4375	-23.2953	0.4038	- 13.8508	0.3820
C_3	-23.3175	-20.0724	-0.6878	-19.2412	8.5750	-0.2342	14.5083	-14.5083
C_4	18.8057	11.4487	12.0801	5.3300	34.8730	-0.4146	16.4970	-0.4367

Table 8: Multinomial regression coefficients.

7 Appendix B: Theory behind statistical testing

7.1 Hypothesis and p-value

We want to obtain a measure for how certain we are that our two models are different. We therefore take into considerations the predictions where the two models predict differently, i.e. n_{12} and n_{21}. We may say, for a given test data set, that C_A is better than C_B if $n_{12} > n_{21}$, and vice versa. If we have $n_{12} \lesseqgtr n_{21}$ then we can define

$$\hat{r} = \frac{n_{12}}{n_{12} + n_{21}} \lesseqgtr \frac{1}{2}. \tag{3}$$

If we let p_{ij} be the probability that a random pair of observations x_q^A (given to C_A), x_q^B (given to C_B) will increase n_{ij} by one, then in our case we want to check if $p_{12} = p_{21}$. In other words, our null-hypothesis, that the classifiers have the same performance, becomes

$$r = \frac{p_{12}}{p_{12} + p_{21}} = \frac{1}{2}. \tag{4}$$

This is turned into a p-value via the binomial distribution: $p(n_{12}, n_{21}|(n_{12}+n_{21})) = p_{\text{binom}}(n_{12}|\theta = r, N = (n_{12} + n_{21}))$. We thus obtain the following hypotheses for the two classifiers, C_A and C_B:

$$H_0 : C_A \text{ and } C_B \text{ have the same performance.}$$
$$H_1 : C_A \text{ and } C_B \text{ have different performance.}$$

The p-value is the probability of observing a value of n_{12} and n_{21} more extreme than the one observed, assuming H_0 is true. When H_0 is true we have that $r = \frac{1}{2}$ (i.e. $p_{12} = p_{21}$). We can consider the more extreme value as the minimum of n_{12} and n_{21}, such that our p-value eventually will be calculated as

$$p = 2\text{cdf}_{\text{binom}}(min\{n_{12}, n_{21}\}|\theta = \frac{1}{2}, N = n_{12} + n_{21}) \tag{5}$$

We then let our threshold be $\alpha = 0.05$, and we conclude that C_A is better than C_B if $p < \alpha = 0.05$ and the difference between the performance of the two is positive.

7.2 Confidence intervals

Further, we create a $(1-\alpha)$ confidence interval for the difference in performance between the two classifiers. Again, $\alpha = 0.05$, so when a CI is found, we may claim with 95% certainty that the difference in the performance between the two classifiers will lie in the interval we obtain. We let the difference between the performance of the classifiers be denoted $\theta = \theta_A - \theta_B$, where $\theta_{A,B}$ is the true chance that classifier A or B is correct, respectively. In this formulation, if $\theta > 0$ then C_A is better than C_B, and vice versa.

First, we obtain an estimator for the difference in accuracy between C_A and C_B as

$$\hat{\theta} = \frac{n_{12} - n_{21}}{n}. \tag{6}$$

An *approximation* of the confidence interval around θ, that is $[\theta_L, \theta_U]$, is found via the Beta-distribution and is given by

$$\begin{aligned}
\theta_L &= 2\text{cdf}^{-1}_{\text{Beta}}(\frac{\alpha}{2}|\alpha = f, \beta = g) - 1 \\
\theta_U &= 2\text{cdf}^{-1}_{\text{Beta}}(1 - \frac{\alpha}{2}|\alpha = f, \beta = g) - 1,
\end{aligned}$$

where $\text{cdf}^{-1}_{\text{Beta}}(\cdot)$ is the inverse cumulative Beta-distribution function, and f and g is given by

$$f = \frac{E_\theta + 1}{2}(Q - 1), \quad g = \frac{1 - E_\theta}{2}(Q - 1), \tag{7}$$

where

$$E_\theta = \frac{n_{12} - n_{21}}{n}, \quad Q = \frac{n^2(n+1)(E_\theta + 1)(1 - E_\theta)}{n(n_{12} + n_{21}) - (n_{12} - n_{21})^2}. \tag{8}$$

8 Appendix C: Exam problems

1. **C.** Prediction C. Calculate FPR and TPR for particular choices of threshold value $\hat{y}$, and . See calculations below.

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

$\hat{y} = 0.4$: $TPR = \frac{4}{4+0} = 1$ $FPR = \frac{4}{4} = 1$

$\hat{y} = 0.5$: $TPR = \frac{3}{3+1} = \frac{3}{4}$ $FPR = \frac{4}{4+0} = 1$

$\hat{y} = 0.55$: $TPR = \frac{3}{3+1} = \frac{3}{4}$ $FPR = \frac{3}{3+1} = \frac{3}{4}$

$\hat{y} = 0.6$: $TPR = \frac{3}{3+1} = \frac{3}{4}$ $FPR = \frac{2}{2+2} = \frac{1}{2}$

$\hat{y} = 0.7$: $TPR = \frac{2}{2+2} = \frac{1}{2}$ $FPR = \frac{2}{2+2} = \frac{1}{2}$

$\hat{y} = 0.8$: $TPR = \frac{1}{1+3} = \frac{1}{4}$ $FPR = \frac{1}{1+3} = \frac{1}{4}$

2. **C.** The impurity gain of the split $x_7 = 2$ is $\Delta \approx 0.0074$. See calculations below.

$p(1|l) = \frac{33+4}{134}$

$p(2|l) = \frac{28+2}{134}$

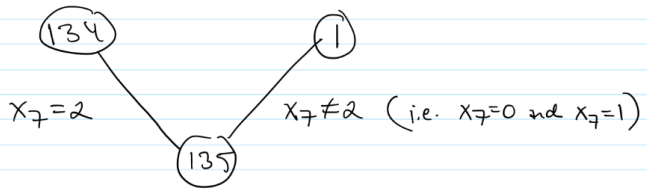
$p(3|l) = \frac{30+3}{134}$

$p(4|l) = \frac{29+5}{134}$

$p(1|r) = 0$
$p(2|r) = 1$
$p(3|r) = 0$
$p(4|r) = 0$

$I(r) = \text{ClassError}(r) = 1 - 1 = 0$

$I(l) = \text{ClassError}(l) = 1 - \frac{37}{134}$



$p(1|root) = 37/135$
$p(2|root) = 31/135$
$p(3|root) = 33/135$
$p(4|root) = 34/135$

$I(root) = \text{ClassError}(root) = 1 - \max_c p(c|root) = 1 - \frac{37}{135}$

$$\Delta = I(root) - \sum \frac{N(v_k)}{N(root)} I(v_k)$$

$$= I(root) - \left[\frac{N(l)}{N(root)} \cdot I(l) + \frac{N(r)}{N(root)} \cdot I(r)\right]$$

$$= \left(1 - \frac{37}{135}\right) - \left[\left(1 - \frac{37}{134}\right) \cdot \frac{134}{135} + \frac{1}{135} \cdot 0\right]$$

$$\approx 0.0074$$

3. **A.** The network contains 124 parameters.

   There are 7 nodes in the input layer, 10 nodes in the hidden layer and 4 nodes in the output layer. Hence, the numer of weights needed to be trained is:

$$7 \times 10 + 10 \times 4 = 110 \tag{9}$$

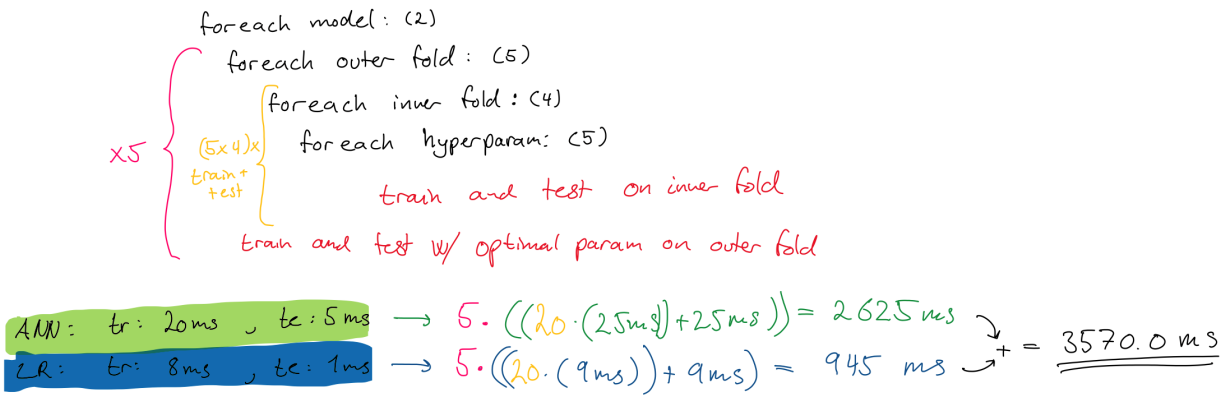
   The number of bias terms needed to be trained is:

$$10 + 4 = 14 \tag{10}$$

   So in total, the number of parameters in the neural network is 124.

4. **D.** If we follow the decision rules from alternative D, we see that they match with the classification boundaries.

5. **C.** ANN train and test time = 25ms, LR train and test tmie = 9ms. See picture below for calculations.



6. **B.** See picture below.

$$P(y=k=4 \mid \hat{\vec{y}}) = \frac{1}{1+\sum_{k'=1}^{3} e^{\hat{y}_{k'}}}, \quad P(y \leq k=3 \mid \hat{\vec{y}}) = \frac{e^{\hat{y}_k}}{1+\sum_{k'=1}^{3} e^{\hat{y}_{k'}}} \quad \text{where} \quad \hat{\vec{y}}_k = \begin{bmatrix} 1 \\ b_1 \\ b_2 \end{bmatrix}^T w_k$$

Find $\hat{y}_k$, $k=1..3$, as function of b_1, b_2:

$$\hat{\vec{y}}_1 = [1 \; b_1 \; b_2] \begin{bmatrix} 1,2 \\ -2,1 \\ 3,2 \end{bmatrix} = 1,2 - 2,1 b_1 + 3,2 b_2$$

$$\hat{y}_2 = \ldots = 1,2 - 1,7 b_1 + 2,9 b_2$$

$$\hat{y}_3 = 1,3 - 1,1 b_1 + 2,2 b_2$$

Quite similar shape, prediction differences probably lie in different b, b_2 and their signs.

We find for each observation $P(y=4 \mid \hat{\vec{y}})$, $P(y<4 \mid \hat{\vec{y}})$

$P(y=4) > P(y \neq 4)$ when $e^{\hat{y}_k} < 1$ bc the denominator is the same in $P(y=4)$ and $P(y \neq 4)$

→ Thus we want an observation that yields $\hat{y}_k < 0$ for $k=1,2,3$.

↳ This seems to be answer B, but we check:

B: $\hat{y}_1 = -2,66$, $\hat{y}_2 = -2,42$, $\hat{y}_3 = -1,56$

↳ Since $e^{\hat{y}_k} < 1$ for $k=1,2,3$ we know that

$P(y=4 \mid \underline{b} = \begin{bmatrix} -0,6 \\ -1,6 \end{bmatrix}) > P(y<4 \mid -\!\!-\!\!-)$ so B is correct.