University of St Andrews

Practical 1

Predicting energy use of appliances

Student ID: 120022067

University of St Andrews

CS5014 Machine Learning

Contents

Contents **2**

1 Introduction **3**

1.1 Libraries 3

1.1.1 Pandas 3

1.1.2 Matplotlib 3

1.1.3 Jupyer 3

1.1.4 Scikit-learn 3

2 Dataset **3**

2.1 Analysing and Visualising the Data 3

2.2 Cleaning the data 4

2.3 Preparing Inputs 4

2.3.1 Features Selection 5

2.3.2 Create a Test Set 5

2.3.3 Feature Scaling 6

2.3.4 Categorical Features 6

3 Training models **6**

3.1 Regression Models 6

3.2 The Performance of Regression Models 6

3.3 Final Model 7

4 Conclusion **8**

Bibliography **9**

1 Introduction

The aim of this assignment is to gain experience in working with real data. In order to achieve this, a regression model needs to be designed, implemented and trained on the provided dataset.

1.1 Libraries

This section covers the third party libraries used in this assignment.

1.1.1 Pandas

Pandas is used to load the data and perform analysis.

1.1.2 Matplotlib

The nature of this practical requires a tool to visualise the data and results. This python package serves that purpose.

1.1.3 Jupyer

This is a useful package for doing analysis on the dataset.

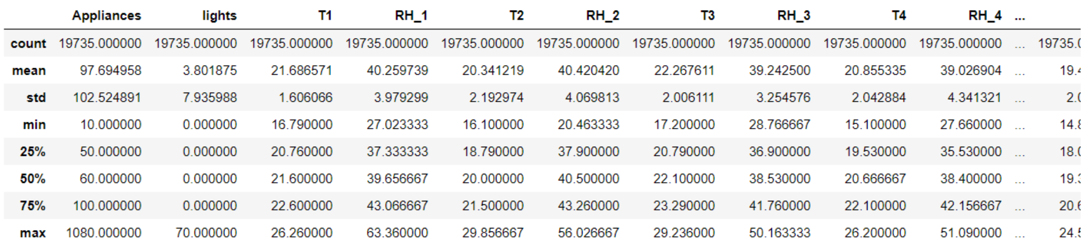
1.1.4 Scikit-learn

This package provides several regression algorithms. This allows training of regression models to be performed with ease.

2 Dataset

2.1 Analysing and Visualising the Data

This step was done in jupyter notebook. The codes can be found in ***/src/analysis.ipynb***.



	Appliances	lights	T1	RH_1	T2	RH_2	T3	RH_3	T4	RH_4	...	
count	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	...	19735.
mean	97.694958	3.801875	21.686571	40.259739	20.341219	40.420420	22.267611	39.242500	20.855335	39.026904	...	19.
std	102.524891	7.935988	1.606066	3.979299	2.192974	4.069813	2.006111	3.254576	2.042884	4.341321	...	2.
min	10.000000	0.000000	16.790000	27.023333	16.100000	20.463333	17.200000	28.766667	15.100000	27.660000	...	14.
25%	50.000000	0.000000	20.760000	37.333333	18.790000	37.900000	20.790000	36.900000	19.530000	35.530000	...	18.
50%	60.000000	0.000000	21.600000	39.656667	20.000000	40.500000	22.100000	38.530000	20.666667	38.400000	...	19.
75%	100.000000	0.000000	22.600000	43.066667	21.500000	43.260000	23.290000	41.760000	22.100000	42.156667	...	20.
max	1080.000000	70.000000	26.260000	63.360000	29.856667	56.026667	29.236000	50.163333	26.200000	51.090000	...	24.

Figure 1: Summary of each numerical attribute.

The summary in Figure 1 shows that there is no missing value in the dataset (value of count matches the number of rows). The histogram in Figure 3 also shows that the attributes have very different scales, which indicates that feature scaling needs to be performed before training the regression models.

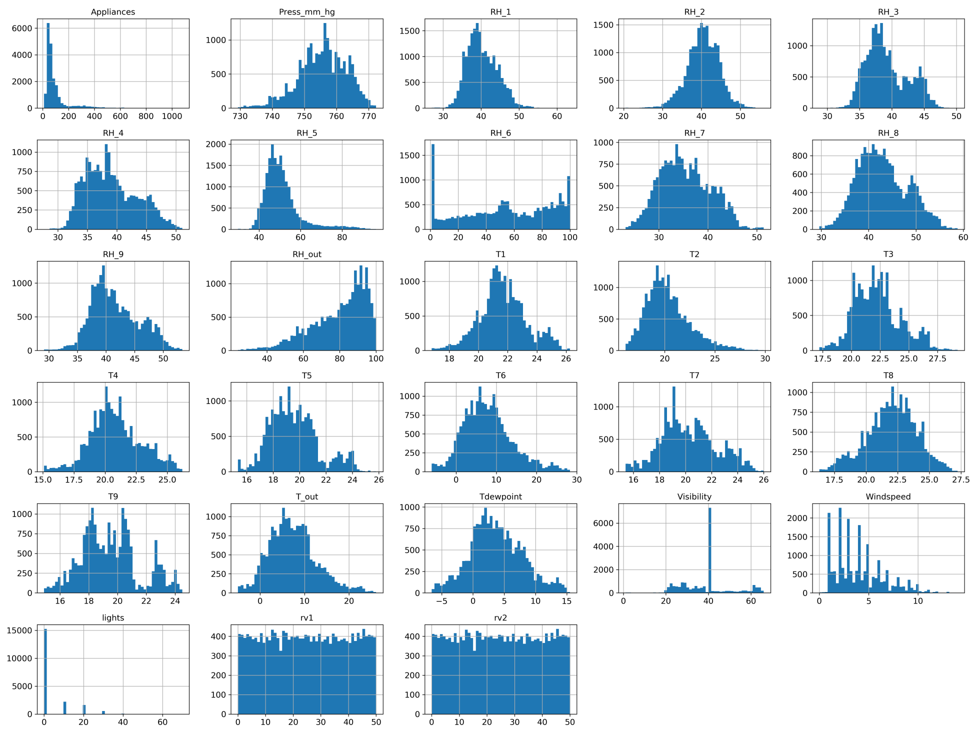


Figure 2: A histogram for each numerical attribute.

2.2 Cleaning the data

There is not a lot of data cleaning task involve because the provided dataset is already cleaned. There is no missing value or mismatched value found within the dataset.

2.3 Preparing Inputs

This section explains the steps of preparing features and label for the regression model. Inputs preparation codes can be found in ***/src/preprocess.py***

2.3.1 Features Selection

```
for _,row in df.iterrows():
  dt_str = row["date"]
  dt_obj = datetime.strptime(dt_str, '%Y-%m-%d %H:%M:%S')
  nsm.append(calculate_nsm(dt_obj))
  day_of_week.append(get_day_of_week(dt_obj))
  week_status.append(get_week_status(dt_obj))
  df["nsm"] = nsm
  df["day_of_week"] = day_of_week
  df["week_status"] = week_status
return df
```

Listing 1: The code used to generate attributes from sate/time variable.

According to Candanedo et al. (2017), there are 3 important predictors generated from date/time variable. These include the number of seconds from midnight for each day (NSM), the week status (weekend or workday) and the day of the week. The extra attributes are added to the dataset with the code in Listing 1. The full code can be found in ***/src/util/attributes_util.py***.

As shown in Figure 3, Candanedo et al. (2017) used Boruta's algorithm to identify important predictors. They identified that rv1 and rv2 are not important predictors. Date is also another feature that cannot be used in the model. Therefore, these features are removed from the dataset.

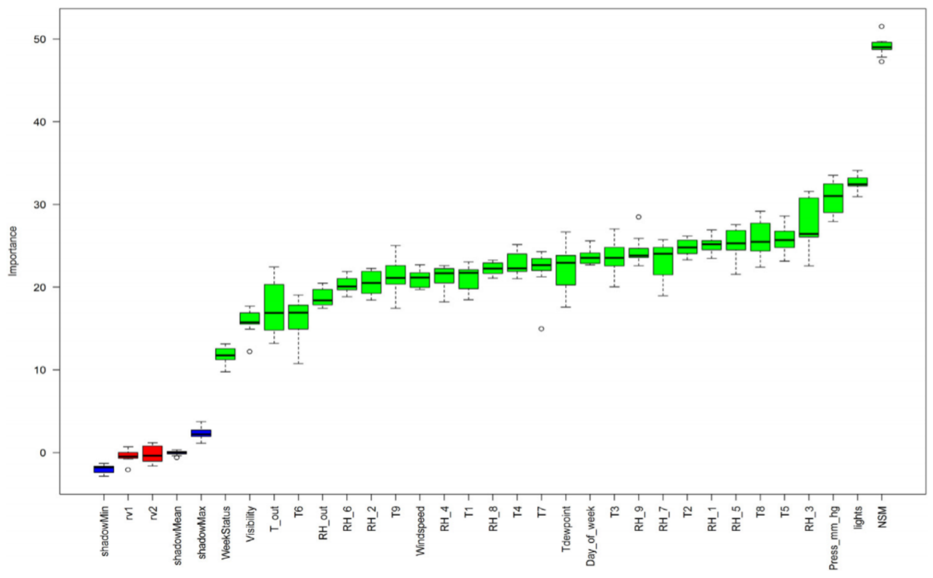


Figure 3: Feature importance and selection from Boruta's algorithm (Candanedo et al. 2017).

2.3.2 Create a Test Set

For this practical, the size of the test set is 20% of the dataset. The rest of the data will be used to train the model.

2.3.3 Feature Scaling

As mentioned before, the numerical attributes have very different scales. This transformation needs to be applied to the numerical features because machine learning algorithm don't perform well when the input numerical features have very different scales (Géron 2017). Therefore, standardization was applied to the numerical attributes. Standardization was chosen over min-max scaling because it is much less affected by outliers (Géron 2017).

2.3.4 Categorical Features

There are two categorical features in the dataset: the week status (weekend or workday) and the day of the week. **LabelBinarizer** class from sklearn was used to handle these attributes. This class allows one-hot encoding on the categorical attributes. Once this is completed, prepared data was saved as text files, so that it can be loaded into regression models with ease.

3 Training models

3.1 Regression Models

For this practical, four regression models are compared against each other. The models are linear regression, regularized linear regression (ridge), polynomial regression, and regularized polynomial regression (ridge). For polynomial models, the polynomial degree 2 is assigned. The training algorithms do not work properly with higher polynomial degrees. This may be because there are too many predictors. The regularized models are also trained with 10 fold cross validation to prevent overfitting and dataset bias. All model objects are saved in ***models*** directory after training. The code used to train the models can be found in ***/src/training_model.py***.

3.2 The Performance of Regression Models

In order to compare the performance of each of the regression models, different performance measures are used here: the root mean squared error (RMSE) and the coefficient of determination or R^2. The performance measures can be calculated with the equations shown below.

$$RMSE = \sqrt{\frac{\Sigma_{i=1}^{n}(Y_i - \hat{Y_i})^2}{n}} \tag{1}$$

$$R^2 = 1 - \frac{\Sigma_{i=1}^{n}(Y_i - \hat{Y_i})^2}{\Sigma_{i=1}^{n}(Y_i - \bar{Y})^2} \tag{2}$$

Model	Train RMSE	Train R^2	Test RMSE	Test R^2
Linear regression	93.83	0.17	90.84	0.17
Linear ridge regression	93.81	0.17	90.81	0.17
Polynomial regression	81.35	0.37	83.02	0.31
Polynomial ridge regression	81.29	0.37	82.81	0.31

Table 1: The performance of regression models

The performance of each model can be found in Table 1. It is clear that polynomial regression models perform better than the linear regression models. It seems that adding regularization does not affect the performance that much. The model with the best performance is polynomial ridge regression with RMSE equals to 81.29 for the training set and 82.81 for the testing set. Therefore, this model is chosen as final model.

3.3 Final Model

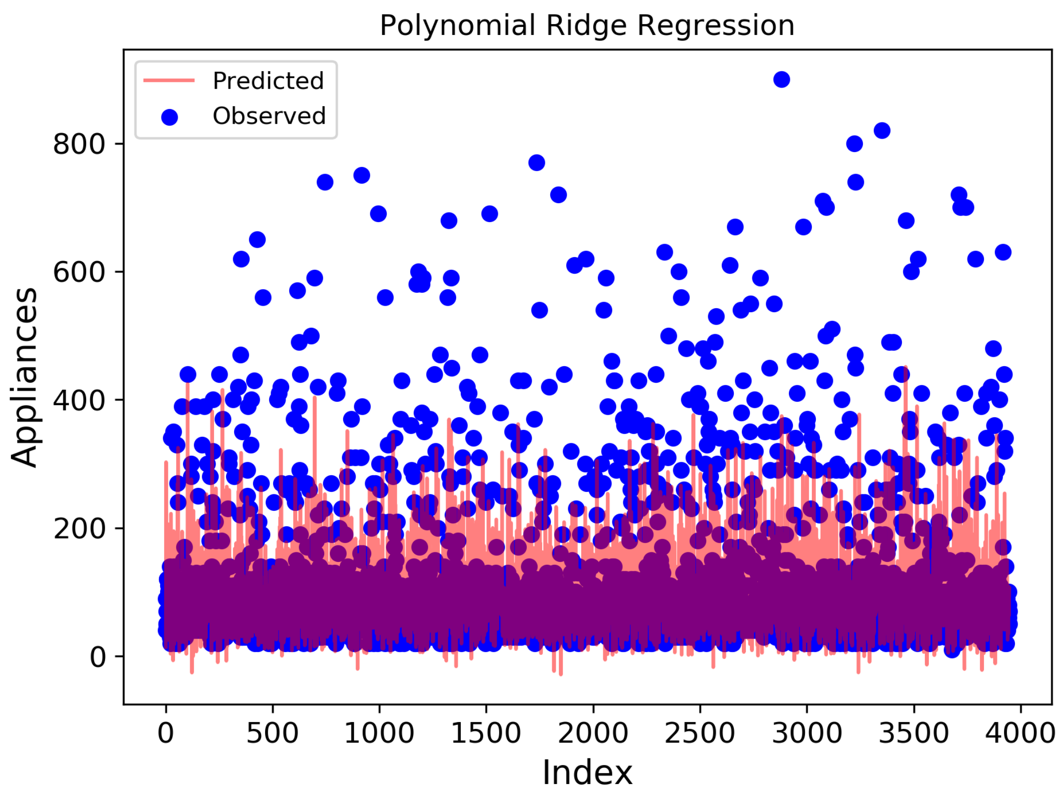


Figure 4: Predictions from polynomial ridge regression model.

Figure 4 shows the predictions of the test set. It is clear that the model performance was greatly affected when the value of Appliances goes beyond 300. Since increasing polynomial degree is not an option as discussed in Section 3.1, one way of improving the performance is to try different algorithms. According to Candanedo et al. (2017), GBM, RF, and SVM radial performs better on this dataset. Due to time constraints, it is not possible to implement and train with those algorithms. The code that generates the plot of the final model can be found in ***/src/final_model.py***.

4 Conclusion

Overall, this practical provides understanding of the procedures of machine learning methodology. The objective of this practical is to gain experience in working with real data. Third party libraries used in this assignment are pandas, matplotlib, jupyter, and sklearn.

The majority of the tasks were done in preprocessing stage. The dataset provided is already cleaned, but there are extra attributes that need to be added in order to improve the prediction. These include the number of seconds from midnight for each day (NSM), the week status (weekend or workday) and the day of the week. Standardization was applied to the numerical attributes because machine learning algorithm do not perform well when the input numerical features have very different scales. The handling of categorical attributes was done by LabelBinarizer class from sklearn.

The training process involves training four regression models on the dataset. The models are linear regression, regularized linear regression (ridge), polynomial regression, and regularized polynomial regression (ridge). Regularized polynomial regression was chosen as final model because it has the lowest RMSE. Due to time constraints, it is not possible to apply other algorithms.

Bibliography

Candanedo, L. M., Feldheim, V. & Deramaix, D. (2017), ‘Data driven prediction models of energy use of appliances in a low-energy house’, *Energy and Buildings* **140**, 81–97.

Géron, A. (2017), *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems*, ” O’Reilly Media, Inc.”.