Introduction to Machine Learning (SS 2023)
Programming Project

Author 1
Last name: Cikalleshi
First name: Erik
Matrikel Nr.:

Author 2
Last name:
First name:
Matrikel Nr.:

I. Introduction

The nature of the task in the powerprediction problem is mainly regression. Regression tasks involve predicting a continous numerical values such as power consumption based on weather information. The goal is to predict the amount of power consumed using wheater and cities variables as input features.

The dataset consists of 39,997 instances (rows) and 67 features (columns). The features include a mix of numerical and categorical variables. The first column is an unnamed index column of integer type, and the remaining columns contain information such as temperature, weather conditions, humidity, wind speed, precipitation, and cloud coverage for various locations. There are no missing values in the dataset, as all columns have zero missing values.

II. Implementation / ML Process

Pre-processing Steps:

Encoding Categorical Features: The dataset contains categorical features, and these features are encoded using one-hot encoding. One-hot encoding ensures that the categorical features are represented numerically and can be used as inputs for machine learning models. After excluding the features starting with "_main" and "_description," we observed a reduction in the execution time of the models. On Erik's PC, the execution time for the random forest model was 2.42 seconds without these features, compared to 3.45 seconds when including them.

Erik's PC specifications are as follows: 13th Gen Intel Core i7-13700KF processor, 32 GB RAM, and an AMD Radeon RX 7900XTX graphics card (without graphic card optimization).

Handling Missing Values: Eventho we know that there are no missing values in the dataset, we still used these two steps to make sure that there are no missing values in the dataset:

- Dropping Rows with Missing Values: Rows with missing values are dropped from the dataset using the `dropna()` function. This ensures that the dataset used for training and evaluation does not contain any missing values.
- Filling Missing Values: Missing values are filled with the mean value of the respective column using the `fillna()` function. This step ensures that no NaN values remain in the dataset.

Splitting the Dataset: The dataset is split into training and validation sets using the `train_test_split()` function from scikit-learn. This step allows for model training on the training set and evaluation on the validation set to assess performance and tune hyperparameters if necessary.

The function `get_encodedV2()` is a modified version of encoding the categorical features in the dataset. It applies one-hot encoding to the categorical columns, fills missing values with column means, selects the top 40 correlated features (excluding the target feature), and adds back the target feature. However, this modification did not significantly affect the model's performance or feature selection.

To resolve this kind of problem we decided to implement linear Regression and neural network because we have labelled data. Logistic regression is used for binary problems but this kind of problem isn't. Linear Regression was easy to implement when using scikit. Neural network is generally good for most problems. As additional method we tried random forest, because we also wanted to try a non parametric method. Random forest improve accuray and avoid overfitting compared to decision tree.

The hyperparameters for the linear regression model are the scikit-learn default values. The hyperparameters for the neural network are as follows:

- 2 hidden layers with 64 neurons for the first layer and 32 neurons for the second layer
- ReLU activation function for the hidden layers and linear activation function for the output layer
- Adam optimizer with a learning rate of 0.001
- Mean squared error loss function
- 8 epochs was the best number of epochs for the neural network model
- Batch size of 32

The hyperparameters for the random forest model are just the number of estimators, which is 5.

III. Results

- Describe the performance of your model (in terms of the metrics for your dataset) on the training and validation sets with the help of plots or/and tables.
- You must provide at least two separate visualizations (plot or tables) of different things, i.e. don't use a table and a bar plot of the same metrics. At least three visualizations are required for the 3 person team.

IV. Results

- Describe the performance of your model (in terms of the metrics for your dataset) on the training and validation sets with the help of plots or/and tables.
- You must provide at least two separate visualizations (plot or tables) of different things, i.e. don't use a table and a bar plot of the same metrics. At least three visualizations are required for the 3 person team.

V. Discussion

- Analyze the results presented in the report (comment on what contributed to the good or bad results). If your method does not work well, try to analyze why this is the case.
- Describe very briefly what you tried but did not keep for your final implementation (e.g. things you tried but that did not work, discarded ideas, etc.).
- How could you try to improve your results? What else would you want to try?

VI. Conclusion

- Finally, describe the test-set performance you achieved. Do not optimize your method based on the test set performance!
- Write a 5-10 line paragraph describing the main take-away of your project.