Modelling of the Sound Pressure Level of Airfoil Sections

Vojtěch Matulík

December 14, 2024

Contents

1 Introduction

Airfoil noise is an important concern in applications such as aircraft design, wind turbine blades, and other aerodynamic surfaces. Minimizing noise pollution while maintaining aerodynamic efficiency is critical for modern engineering designs. Developing a predictive model for the sound pressure level (SPL) can help engineers test and optimize designs, reducing the reliance on costly experimental setups.

The task involves predicting the noise levels generated by airfoil sections under various aerodynamic conditions. Specifically, the goal is to model the *sound pressure level* as a function of key aerodynamic parameters such as frequency, angle of attack, chord length, free stream velocity, and suction side displacement thickness. This model can assist in reducing noise pollution in applications such as aviation and wind energy.

This problem falls into the domain of supervised learning, as it involves a regression task, in which the objective is to predict continuous values 2 of SPL rather than discrete classes.

The implementation part of the project is done in Python.

2 Exploratory Data Analysis

For this task, we use the *NASA Airfoil Self-Noise Dataset*, available from Kaggle (`https://www.kaggle.com/datasets/fedesoriano/airfoil-selfnoise-dataset`). This dataset contains measurements from aerodynamic and acoustic tests conducted in an anechoic wind tunnel.

First step in EDA is displaying the first several rows of the dataset.

```
index          f  alpha       c  U_infinity     delta     SSPL
    0        800    0.0  0.3048        71.3  0.002663  126.201
    1       1000    0.0  0.3048        71.3  0.002663  125.201
    2       1250    0.0  0.3048        71.3  0.002663  125.951
    3       1600    0.0  0.3048        71.3  0.002663  127.591
    4       2000    0.0  0.3048        71.3  0.002663  127.461
```

The dataset includes the following variables.

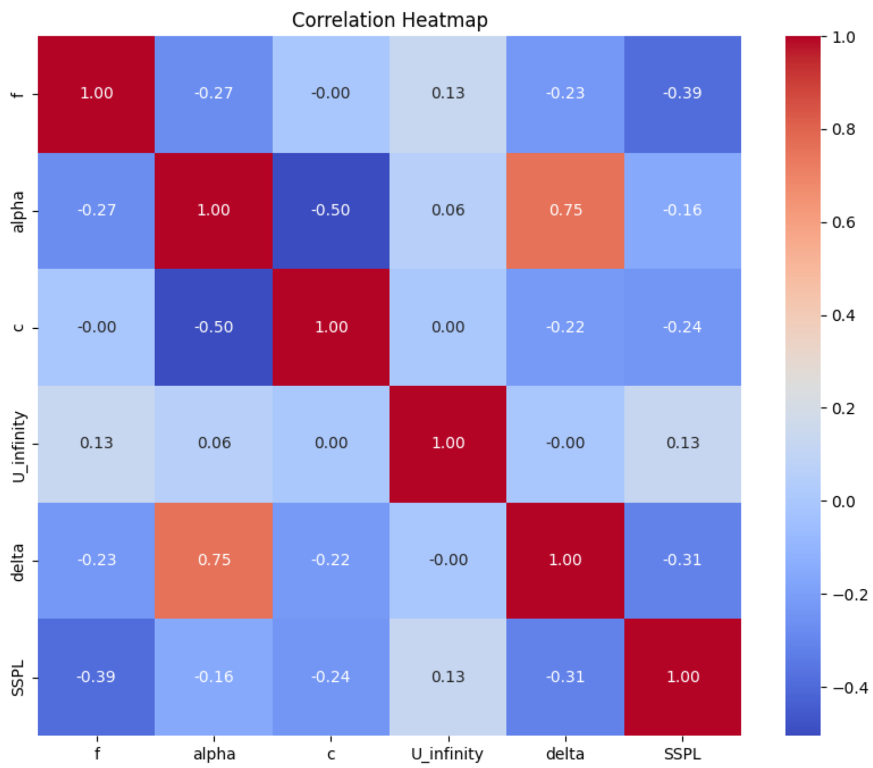
- **Frequency (unit Hz, variable f)**: Frequency of the sound wave.
- **Angle of Attack (unit degrees, variable alpha)**: Angle between the airfoil chord line and the free-flow velocity.
- **Chord Length (unit meters, variable c)**: Distance from the leading edge to the trailing edge of the airfoil.

- **Free-Stream Velocity (unit m/s, variable U_infinity)**: The velocity of air approaching the airfoil.
- **Suction Side Displacement Thickness (unit meters, variable delta)**: Measurement of the boundary layer on the suction side of the airfoil.
- **Sound Pressure Level (unit dB, variable SSPL)**: Noise level measured in decibels.

Now, the summary of the statistical properties of the dataset is displayed and analyzed.

	f	alpha	c	U_infinity	delta	SSPL
count	1503.000	1503.000	1503.000	1503.000	1503.000	1503.000
mean	2886.380	6.782	0.136	50.860	0.011	124.835
std	3152.573	5.918	0.093	15.572	0.013	6.898
min	200.000	0.000	0.025	31.700	0.000	103.380
25%	800.000	2.000	0.050	39.600	0.002	120.191
50%	1600.000	5.400	0.101	39.600	0.004	125.721
75%	4000.000	9.900	0.228	71.300	0.015	129.995
max	20000.000	22.200	0.304	71.300	0.058	140.987

In the table, a total of 1503 records can be seen in the dataset. SPL values are concentrated around the mean (124.835 dB), with a slight positive skew. Next, the correlation between the features is visualized using a heatmap.



Strong positive correlation (0.75) between α and δ suggests potential multicollinearity, which does not necessarily have to be handled when using deep learning. Neural networks, unlike linear models, usually do not suffer from multicollinearity.

As the next exploration tool, the pairplot is used, which visualizes the relationships between SPL and the input features.

The near-normal distribution of the SPL is ideal for regression modeling, as it ensures consistent prediction performance across the target range even though it is slightly positively skewed. No pattern is found between the target (SPL) and the features. A logarithmic relation can be seen between α and δ.

3 Data Transformation

Based on the results of the previous section, data transformations are performed. The data is split into a training and testing set.

```
# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

And feature scaling is performed.

```
# Feature scaling
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
```

4 Neural Network Model

Preliminary Model

As a first step, a simple neural network is implemented to evaluate its performance and identify potential areas for improvement. The preliminary model consists of the following configuration:

- **Input Layer:** The model accepts five input features corresponding to the aerodynamic parameters $(f, \alpha, c, U_\infty, \delta)$.
- **Hidden Layers:** Three dense layers, each containing 8 neurons with the ReLU activation function.
- **Output Layer:** A single neuron with a linear activation function to predict the Sound Pressure Level (SPL).
- **Loss Function:** Mean Squared Error (MSE).
- **Optimizer:** Adam with a learning rate of 0.001.

The preliminary model demonstrates moderate performance, achieving a R^2 value of approximately 0.2 and a mean squared error (MSE) of approximately 2420. These results indicate the need for further improvements to achieve better predictive accuracy.

Hyperparameter Tuning

To enhance the performance of the model, hyperparameter tuning is performed:

- **Number of Neurons:** The number of neurons in each layer is increased from 8 to 16 to improve the capacity of the model.
- **Number of Layers:** The model is expanded to include three hidden layers for a better representation of the data's characteristics.
- **Batch Size:** A batch size of 1 produces the best results for this dataset, as it allows fine-grained updates during training.
- **Regularization:** L2 regularization is tested but is ultimately not applied, as overfitting is not observed.

Final Model

The final neural network model is implemented with the following architecture:

- **Input Layer:** Accepts five features.
- **Hidden Layers:** Three dense layers, each containing 16 neurons with the ReLU activation function.
- **Output Layer:** A single neuron with a linear activation function to predict SPL.
- **Loss Function:** Mean Squared Error (MSE).
- **Optimizer:** Adam optimizer with a learning rate of 0.001.
- **Training Configuration:** The model is trained for 100 epochs using a batch size of 1.

The code implementation for the final model is as follows:

```
# Build the ANN
ann = tf.keras.models.Sequential()

ann.add(tf.keras.layers.Input(shape=(5,)))  # Input layer
ann.add(tf.keras.layers.Dense(units=16, activation='relu'))
ann.add(tf.keras.layers.Dense(units=16, activation='relu'))
ann.add(tf.keras.layers.Dense(units=16, activation='relu'))
ann.add(tf.keras.layers.Dense(units=1))  # Output layer

# Compile the model
optimizer = tf.keras.optimizers.Adam(learning_rate=0.001)
ann.compile(optimizer=optimizer, loss='mean_squared_error')

# Train the model
history = ann.fit(X_train, y_train, batch_size=1, epochs=100, verbose=0)
```

The final model achieves the following performance metrics:

- **Mean Squared Error (MSE):** 6.81
- **R-squared (R^2):** 0.855

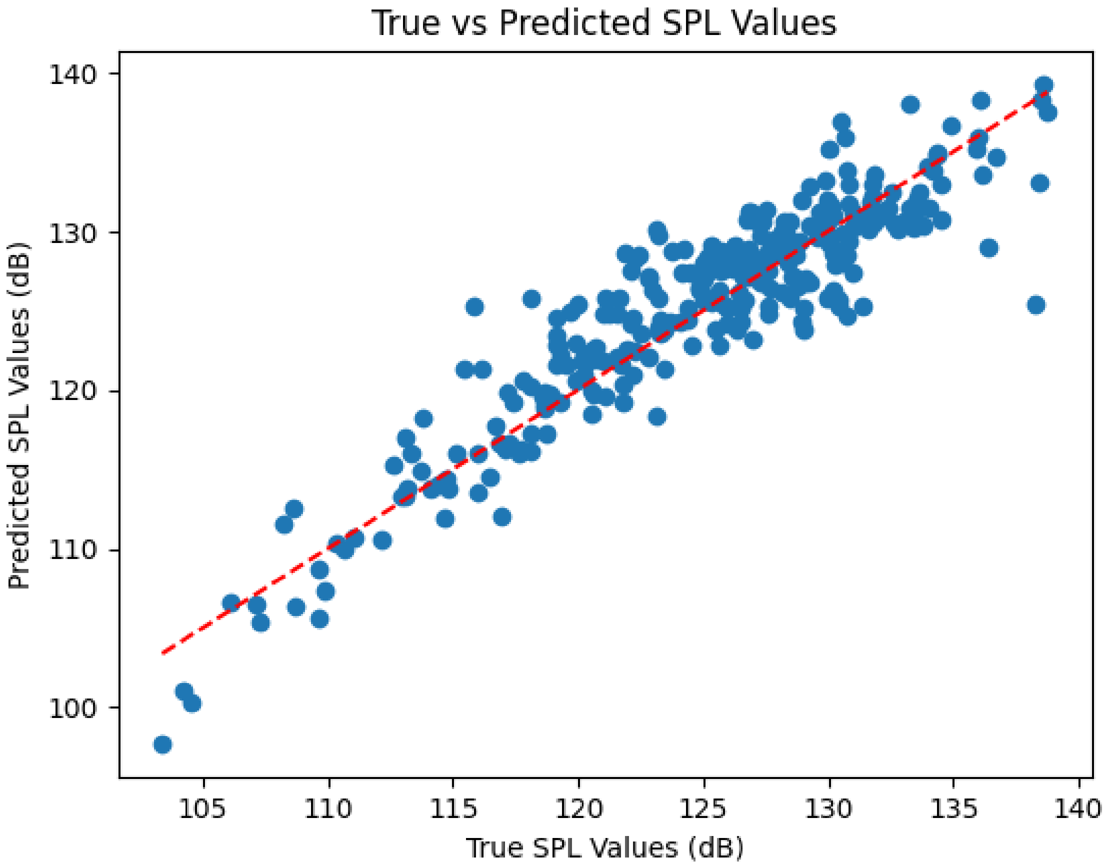
These results indicate that the final model provides a significant improvement over the baseline model, with better generalization and predictive precision.

Visualisation

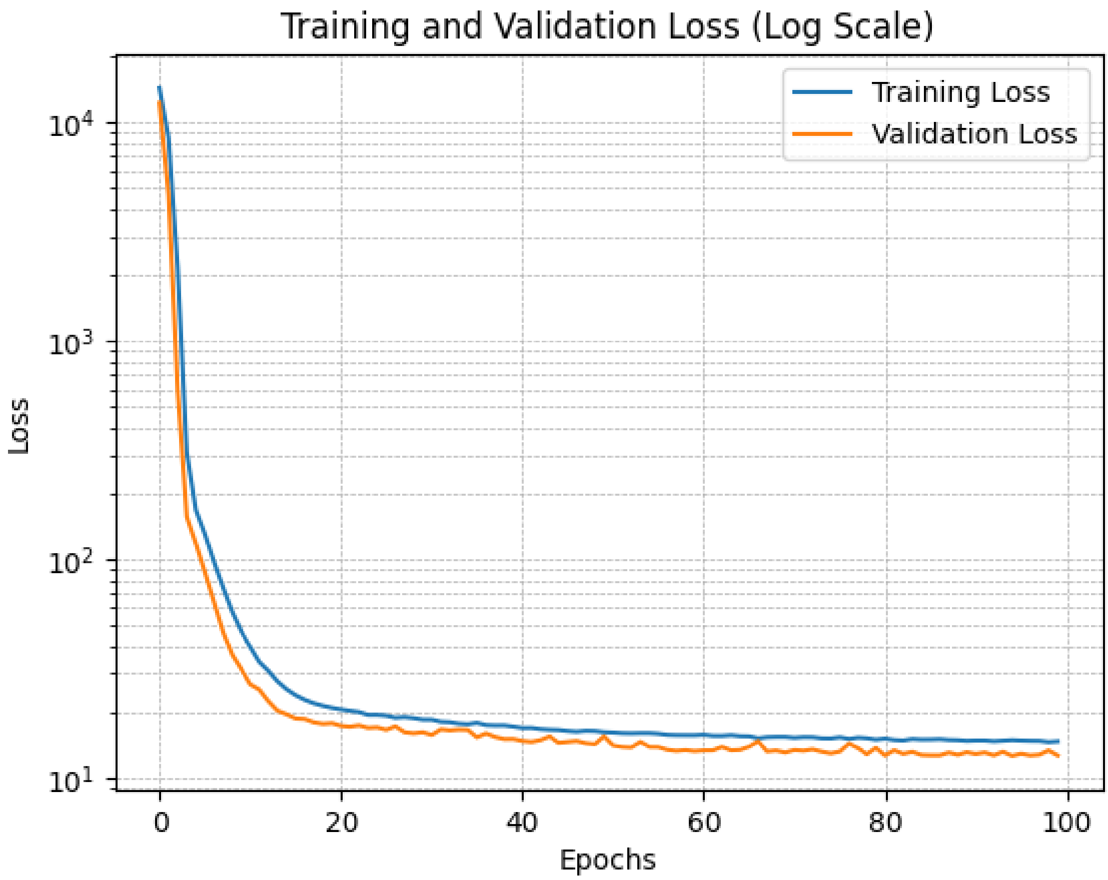
The performance of the model during hyperparameter tuning is evaluated based on the metrics R^2 and MSE, together with the following visualizations.

- **True vs. Predicted Values:** A scatter plot of true SPL values against predicted values.
- **Training and Validation Loss:** A line plot of the loss values over epochs.

The following graphs illustrate the results of the final model.



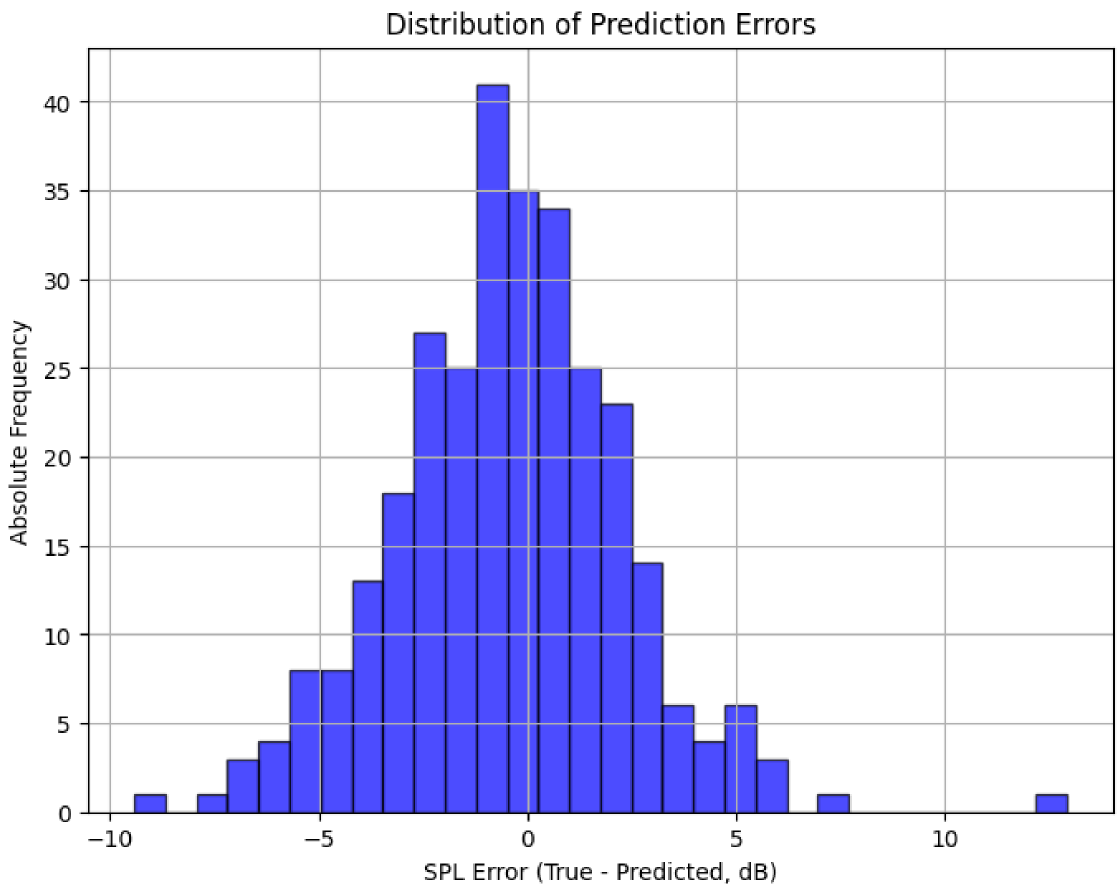
The true vs. predicted values show a strong linear relationship, indicating accurate predictions.



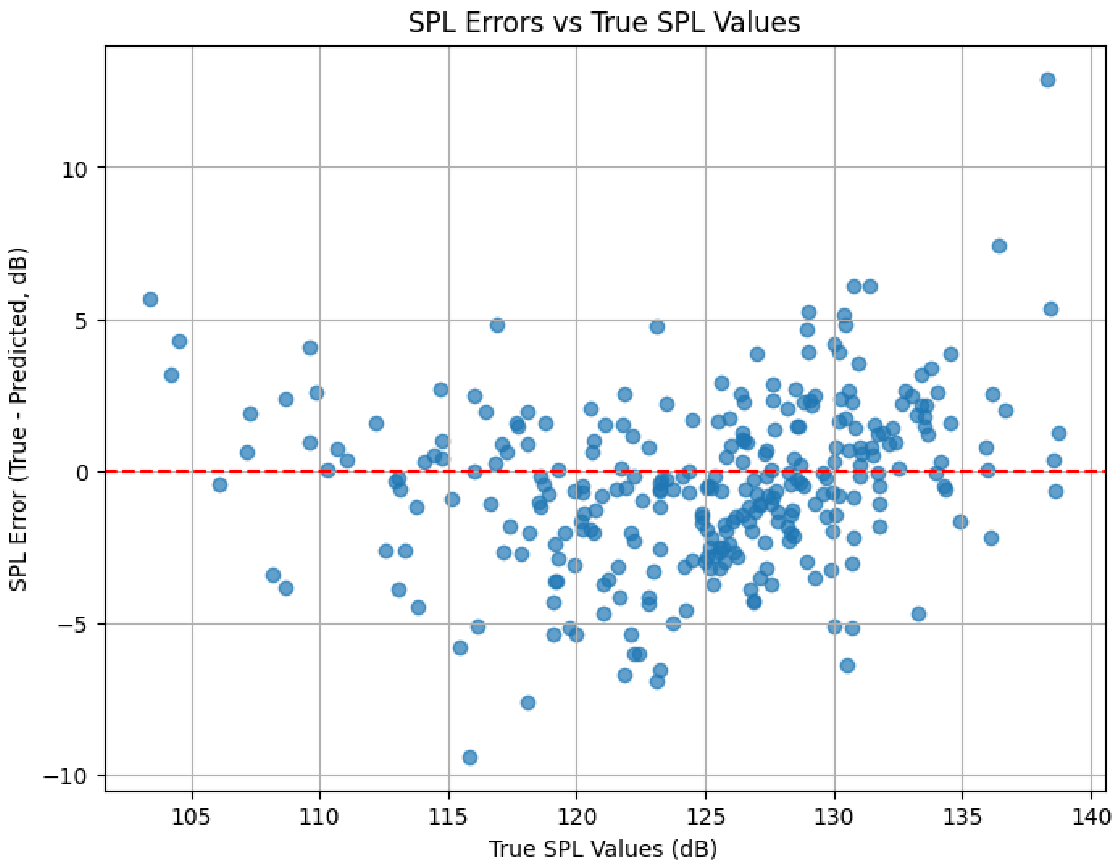
The loss curves demonstrate a consistent decrease in training and validation losses without divergence, suggesting that overfitting does not occur. There is a slight fluctuation in the validation loss, but not significant to consider the model as overfitting.

Error Analysis

After training the final neural network model, it is important to analyze the errors (residuals) to identify potential patterns and evaluate the performance of the model. (The residuals are calculated as the difference between the true values and the predicted values). The figure shown below displays the distribution of errors visualized using a histogram.



The histogram shows that the errors are centered around zero with a roughly symmetric distribution, indicating that there is no significant prediction bias. A scatter plot of errors against the true SPL values is created to examine whether the errors vary systematically with the true values.



The plot does not reveal significant trends, suggesting that the model performs consistently at different SPL values.

5 Conclusion

In this project, a predictive model for the sound pressure level of the airfoil sections is successfully developed using a neural network. The project demonstrates the power of neural networks for regression tasks in aerodynamic modeling. The final model achieves a high level of predictive accuracy and generalization, making it a valuable tool to reduce noise pollution in airfoil applications such as wind energy.