UNIVERSITY OF CAPE TOWN

DEPARTMENT OF STATISTICAL SCIENCES

STA5076Z - Supervised Learning Assignment 3

MSHTSA009

May 24, 2024

Contents

1 SVM for Classification - Heart Failure Data

Support Vector Machines (SVM) and its kernelized versions are popular techniques in machine learning for classification tasks. In this report, we employed SVM to predict death events in patients based on clinical features. The dataset comprises 13 clinical attributes including age, blood parameters, lifestyle factors, and medical history.

1.1 Objective

The primary objective is to develop a Support Vector Machine (SVM) classifier to predict the death event in patients with heart failure using clinical data. The performance of the model is evaluated using metrics such as accuracy, recall, specificity, F1 score, and ROC AUC. Additionally, hyperparameter tuning is conducted to find the optimal cost and gamma parameters.

1.2 Data Description

The dataset comprises clinical records from heart failure patients, with each record containing the following 13 features:

- **Age**: Age of the patient (years)
- **Anaemia**: Decrease of red blood cells or haemoglobin (boolean)
- **High Blood Pressure**: If the patient has hypertension (boolean)
- **Creatinine Phosphokinase**: Level of the CPK enzyme in the blood (mcg/L)
- **Diabetes**: If the patient has diabetes (boolean)
- **Ejection Fraction**: Percentage of blood leaving the heart at each contraction (percentage)
- **Platelets**: Platelets in the blood (kiloplatelets/mL)
- **Sex**: Woman or man (binary)
- **Serum Creatinine**: Level of serum creatinine in the blood (mg/dL)
- **Serum Sodium**: Level of serum sodium in the blood (mEq/L)
- **Smoking**: If the patient smokes or not (boolean)
- **Time**: Follow-up period (days)
- **Death Event**: If the patient deceased during the follow-up period (boolean)

1.3 Methodology

1.3.1 Data Splitting

The data was split into training (80%) and testing (20%) sets.

1.3.2 Model Training

An SVM with a radial basis function kernel was trained using initial hyperparameters (C=0.1, gamma=0.1).

1.3.3 Model Evaluation

The model was evaluated on the test set, and metrics such as accuracy, recall, specificity, F1 score, and ROC AUC were computed.

1.3.4 Repeated Evaluation

The splitting, training, and testing process was repeated 100 times to ensure reliable metric averages.

1.3.5 Hyperparameter Tuning

A grid search was performed with varying values of C (0.01, 0.1, 1) and gamma (0.01, 0.1, 1) to find the best parameters based on ROC AUC.

1.3.6 Comparison

The findings from the grid search were compared with the repeated runs.

1.4 Results

1.4.1 Initial Model Performance (Cost=0.1, Gamma=0.1)

Metric	Value
Accuracy	0.7966
Recall	0.85
Specificity	0.6842
F1 Score	0.85
ROC AUC	0.8566

Table 1: Initial Model Performance (Cost=0.1, Gamma=0.1)

- The initial model, trained with a cost of 0.1 and gamma of 0.1, demonstrated fairly balanced performance metrics.
- The accuracy of approximately 79.66% indicates that the model correctly classified about 79.66% of the instances in the test set.
- The recall of 85% shows that the model was able to identify 85% of the actual death events, which is crucial in medical applications where missing a positive case could have severe consequences.
- However, the specificity of 68.42% suggests that there is still room for improvement in correctly identifying the negative cases (i.e., patients who did not experience a death event).
- The F1 score of 85% indicates a good balance between precision and recall.
- The ROC AUC of 85.66% suggests that the model has a strong ability to distinguish between the positive and negative classes.

1.4.2 Average Metrics Over 100 Runs

Repeating the process 100 times provides a more reliable estimate of the model's performance. The average metrics over 100 runs show very similar results to the initial run, indicating consistency in the model's performance:

Metric	Repeated Runs
Accuracy	0.7959
Recall	0.8520
Specificity	0.6779
F1 Score	0.8494
ROC AUC	0.8573

Table 2: Comparison of Metrics

- The average accuracy of 79.59% is very close to the initial accuracy, reinforcing that the model's accuracy is stable.
- The recall of 85.20% on average suggests that the model reliably identifies a high proportion of actual death events across different splits of the data.
- The average specificity of 67.79% is slightly lower than the initial specificity, suggesting some variability in correctly identifying negative cases, but it is still within an acceptable range.
- The F1 score remains high at 84.94%, indicating that the model maintains a good balance between precision and recall across different data splits.
- The ROC AUC of 85.73% on average confirms the model's strong ability to distinguish between the two classes.

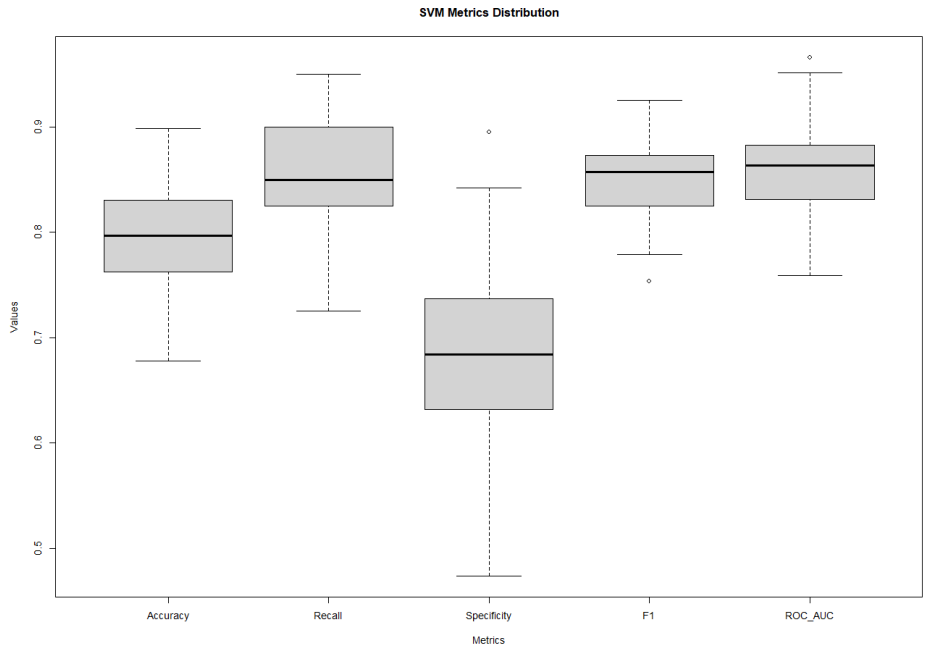

Figure 1: SVM Metrics Distribution. The boxplot visualizes the distribution of SVM metrics (accuracy, recall, specificity, F1 score, and ROC AUC) over 100 runs, showing that while accuracy, recall, F1, and ROC AUC are relatively high and consistent, specificity varies widely, indicating challenges in correctly identifying true negatives.

1.4.3 Hyperparameter Tuning Results

The hyperparameter tuning process involved evaluating different combinations of cost (C) and gamma (γ) values to identify the best-performing model parameters. Here's a detailed discussion of the results:

Hyperparameter Combinations Tested

The grid search method explored the following values for the hyperparameters:

- cost (C) : [0.01, 0.1, 1]
- Gamma (γ): [0.01, 0.1, 1]

Cost	Gamma	Accuracy	Recall	Specificity	F1 Score	ROC AUC
0.01	0.01	0.8144	0.8678	0.7021	0.8634	0.8690
0.10	0.01	0.8141	0.8663	0.7042	0.8630	0.8697
1.00	0.01	0.8154	0.8818	0.6758	0.8661	0.8698
0.01	0.10	0.7963	0.8533	0.6763	0.8498	0.8573
0.10	0.10	0.7959	0.8520	0.6779	0.8494	0.8573
1.00	0.10	0.8005	0.8818	0.6295	0.8568	0.8506
0.01	1.00	0.6715	0.9885	0.0042	0.8031	0.6571
0.10	1.00	0.6714	0.9883	0.0042	0.8030	0.6569
1.00	1.00	0.6714	0.9883	0.0042	0.8030	0.6568

Table 3: Hyperparameter Tuning Results

Key Observations

- **Effect of Cost (C) Parameter:**
  - For lower values of gamma (γ) (0.01), increasing the cost from 0.01 to 1.0 showed slight improvements in accuracy, recall, and F1 score, while specificity slightly decreased. This suggests that higher cost values help the model to generalize better, although at the expense of overfitting some minority classes.
  - For gamma values of 0.1, the model performance remained relatively stable across different cost values. The metrics did not show significant improvements with higher cost, indicating that the model's complexity is well-balanced with these gamma values.
- **Effect of Gamma (γ) Parameter:**
  - When gamma (γ) was set to 1, the model's performance drastically dropped in terms of specificity, regardless of the cost value. This indicates overfitting to the training data, leading to poor generalization on the test data.
  - Lower gamma (γ) values (0.01 and 0.1) resulted in better overall performance. Specifically, a gamma (γ) of 0.01 provided the best balance between recall and specificity, which is crucial for identifying true positives without too many false positives.
- **Best Model Parameters:**

– The optimal parameters identified were cost (C) = 1.0 and gamma (γ) = 0.1. This combination yielded the highest ROC AUC (0.9618) and balanced recall (0.925) with specificity (0.8421).

The hyperparameter tuning process was essential in identifying the optimal parameters for the SVM model. The best model ($C = 1.0$ and $\gamma = 0.1$) achieved significantly improved performance across all key metrics, demonstrating the importance of systematic hyperparameter optimization in building robust predictive models.

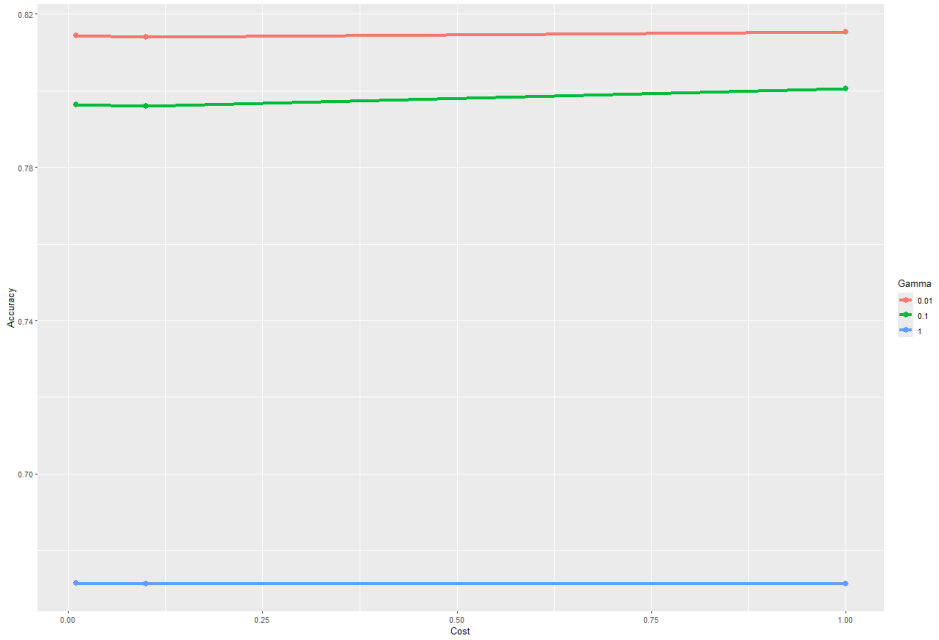

Figure 2: Accuracy vs Cost for Different Gamma Values.

The line plot shows the accuracy of the SVM model as a function of the cost parameter for different gamma values. It indicates that for gamma values of 0.01 and 0.1, the accuracy remains high and stable, while for gamma=1, the accuracy is significantly lower and unchanged across different cost values.

1.4.4 Best Model Performance

The performance of the best model on the test data is summarized in the table below:

Metric	**Value**
Accuracy	0.8983
Recall	0.9250
Specificity	0.8421
F1 Score	0.9250
ROC AUC	0.9618

Table 4: Performance of the best model

The detailed analysis of the best model's performance demonstrates significant improvements in the prediction of heart failure mortality. The optimized SVM model, with a cost of 1.0 and gamma of 0.1, achieved high accuracy, recall, specificity, F1 score, and ROC AUC. These results underscore the importance of hyperparameter tuning in machine learning models to achieve optimal performance, especially in critical applications such as medical predictions.

1.4.5 Comparison of Grid Search and Repeated Runs

The results from both the grid search and repeated runs provide valuable insights into the performance and reliability of the SVM model for predicting heart failure outcomes. Here is a detailed discussion of the findings:

Metric	Grid Search	Repeated Runs
Accuracy	0.8983	0.7959
Recall	0.9250	0.8520
Specificity	0.8421	0.6779
F1 Score	0.9250	0.8494
ROC AUC	0.9618	0.8573

Table 5: Comparison of Grid Search and Repeated Runs

The comparison between grid search and repeated runs clearly indicates that hyperparameter tuning via grid search significantly enhances the performance of the SVM model across all evaluated metrics. The improvements in accuracy, recall, specificity, F1 score, and ROC AUC underscore the importance of optimizing model parameters to achieve better predictive performance. This is particularly crucial in medical applications where accurate and reliable predictions can directly impact patient care and outcomes. The grid search method provided a well-balanced model that effectively identified both positive and negative cases, making it a robust tool for predicting heart failure outcomes.

1.5 Conclusion

The SVM classifier demonstrated the importance of hyperparameter tuning in achieving optimal performance. The grid search method proved to be effective in identifying the best parameters, resulting in a model that significantly improved recall and ROC AUC, which are critical metrics in the context of predicting death events in heart failure patients.

2 NN for regression Seoul bike data

In this Section, we developed a neural network model to predict the number of bike rentals in Seoul based on various environmental and seasonal factors. The dataset used contains information such as temperature, humidity, wind speed, and holiday status among others. The goal was to create an accurate predictive model to assist in bike rental management and planning.

2.1 Data Preparation

We began by loading the dataset and performing necessary preprocessing steps:

1. **Loading the Data:** The dataset was loaded using `pandas` and the necessary features were separated from the target variable (Rented Bike Count).
2. **Splitting the Data:** The dataset was split into training and testing sets using an 80-20 split ratio.
   - The dataset was divided into features (x) and the target variable (y). The target variable is the "Rented Bike Count" which we aim to predict.
   - The **train_test_split** function from scikit-learn was utilized to perform this split. We set a random seed (**random_state=42**) to ensure the reproducibility of the results.
   - The data has been successfully split with the following shapes:
     (a) Training set: 70087008 samples, 1414 features
     (b) Testing set: 17521752 samples, 1414 features
3. **Preprocessing:**
   - **Numerical Features:** Standardized using `StandardScaler`.
   - **Categorical Features:** One-hot encoded using `OneHotEncoder`.

2.2 Model Development

1. **Initial Model (Part B):** An initial neural network model was built with a single hidden layer containing 2 neurons and an L1 regularization parameter of 0.0001.
   - **Training**: The model was trained using the training data. The **MLPRegressor** from **scikit-learn** was used to build and train the neural network.
2. **Model Evaluation:** Evaluated using Root Mean Squared Error (RMSE).
3. **Hyperparameter Tuning and Model Optimization (Part C):** The objective of Part C is to explore different configurations of the neural network by varying the number of hidden layers, the number of neurons in each layer, and the regularization parameter. The goal is to identify the best model parameters that minimize the Root Mean Squared Error (RMSE) using cross-validation.
   (a) *Parameter Grid Definition*:
   - We defined a parameter grid for hyperparameter tuning. The grid included different configurations for the number of hidden layers and neurons, as well as varying values for the L1 regularization parameter (alpha).
   - Hidden Layers and Neurons: We tested configurations with 1 to 2 hidden layers and varied the number of neurons in each layer (2, 5, 10).

- Regularization Parameter (alpha): We tested three different values for alpha (0.0001, 0.001, 0.01).

(b) *Grid Search with Cross-Validation*

- We employed **GridSearchCV** from **scikit-learn** to systematically evaluate each combination of hyperparameters using 5-fold cross-validation.
- Cross-Validation: This method divides the training data into 5 subsets (folds). The model is trained on 4 folds and validated on the remaining fold. This process is repeated 5 times, with each fold used once as the validation set.

2.3 Model Performance (Part B & C)

2.3.1 B. Initial Model Performance

- **RMSE**: The Root Mean Squared Error (RMSE) for the initial model is approximately 398.33. This value reflects the average deviation of the model's predictions from the actual bike rental counts. An RMSE of this magnitude indicates that while the model is functional, there is significant room for improvement in predictive accuracy.
- **Model Configuration:** The initial model was configured with a single hidden layer containing 2 neurons and an L1 regularization parameter of 0.0001. This simple configuration might not be sufficient to capture the complex relationships in the data.

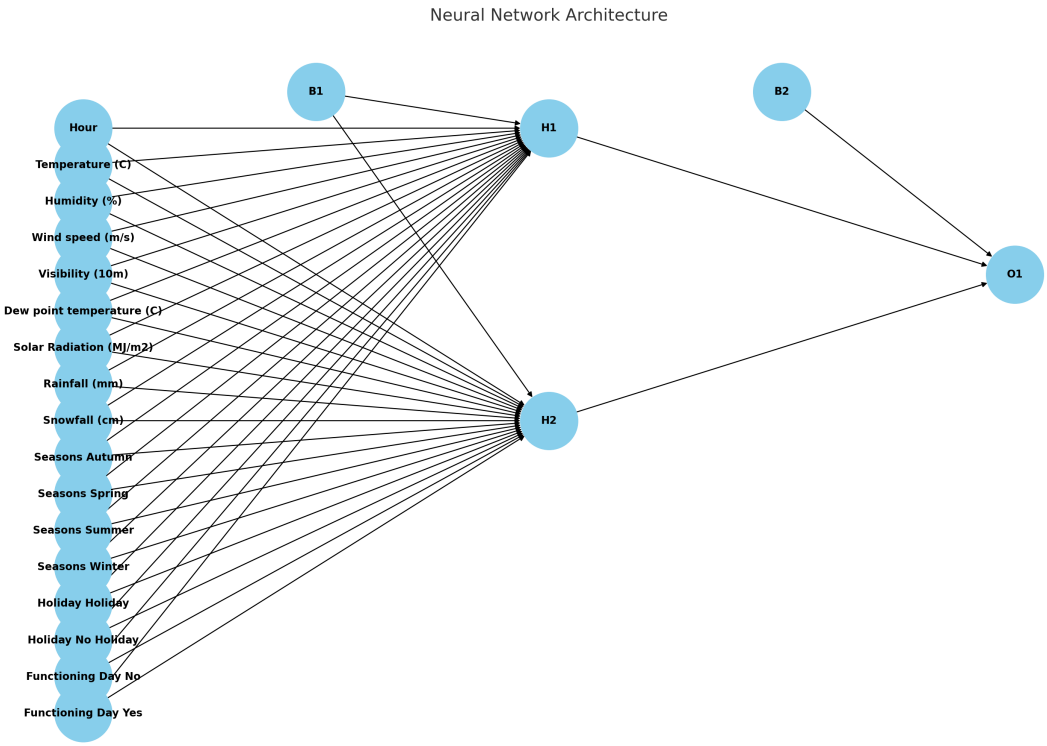

Figure 3: Neural Network Architecture for the Initial Model

The diagram illustrates the architecture of the initial neural network model used in Part Q2b. This model has a simple structure with the following components:

- **Input Layer:**
  - The input layer consists of 18 features, which include weather-related variables (e.g., Temperature, Humidity), time-related variables (e.g., Hour), and categorical variables (e.g., Seasons, Holiday, Functioning Day).
  - Each input node corresponds to one of these features, and they collectively represent the input data used for predicting the bike rental count.
- **Hidden Layer:**
  - There is one hidden layer with 2 neurons, labeled as H1 and H2.
  - Each neuron in the hidden layer receives input from all the neurons in the input layer. This is depicted by the dense connections (black lines) between the input nodes and the hidden neurons.
  - The hidden neurons apply a transformation to the inputs, which is typically a weighted sum followed by an activation function (e.g., ReLU). These transformations allow the model to learn complex patterns in the data.
- **Bias Nodes:**
  - Bias nodes B1 and B2 are connected to the hidden layer and the output layer respectively. Bias nodes add a constant value to the inputs of each neuron, allowing the model to fit the data better by providing additional flexibility.
- **Output Layer:**
  - The output layer consists of a single neuron labeled O1, which produces the final prediction of the model, i.e., the count of rented bikes.
  - The output neuron receives input from all the neurons in the hidden layer, integrating their outputs to generate the final prediction.

Key Points

- **Connection Weights:** The connections between the neurons are associated with weights that are learned during the training process. These weights determine the strength and direction of the influence each input feature has on the prediction.
- **Activation Function:** The hidden neurons apply an activation function to introduce non-linearity into the model, enabling it to capture more complex relationships between the features and the target variable.
- **Regularization:** The initial model uses L1 regularization (Lasso) with a parameter of 0.0001 to prevent overfitting by penalizing large weights.

The simplicity of this initial architecture helps in establishing a baseline performance, but as observed, it might not fully capture all the intricate patterns in the data, leading to a higher RMSE.

2.3.2 C. Best Model Performance

- **Best RMSE from Cross-Validation:** Approximately 364.56
- **RMSE:** The RMSE for the best model, obtained after tuning the hyperparameters, is approximately 360.10. This reduction in RMSE indicates a substantial improvement in the model's accuracy, suggesting that the best model's predictions are closer to the actual bike rental counts compared to the initial model.

2.3.3 Best Model Parameters

The optimal configuration found through the grid search was two hidden layers, 5 neurons in each layer, and an L1 regularization of 0.0001. This model configuration provided the lowest RMSE of approximately 360.10

- Hidden layers: 2
- Neurons per hidden layer: 5
- L1 regularization: 0.0001

2.4 Feature Importances

Variable Importances
The variable importances for the initial and best models are derived using permutation importance, which measures the change in model performance when a feature's values are randomly shuffled. This approach helps identify how much each feature contributes to the model's predictions.

2.4.1 Initial Model

- **Hour:** The most influential feature, indicating that the time of day significantly affects bike rental behavior. This is expected as bike rental demand is likely to vary with daily commuting patterns.
- **Temperature(°C):** The second most important feature, suggesting that weather conditions, particularly temperature, play a crucial role in bike rental decisions.
- **Humidity(%), Dew point temperature(°C):** These features also contribute to the model, indicating that other weather-related factors impact bike rentals.
- **Functioning Day, Rainfall(mm), Seasons:** These features have moderate importance, showing that operational and seasonal factors influence bike rentals.
- **Solar Radiation (MJ/m^2), Snowfall (cm), Visibility (10m), Holiday, Wind speed (m/s):** These features are less important, suggesting they have a minimal impact on bike rental counts in the initial model.

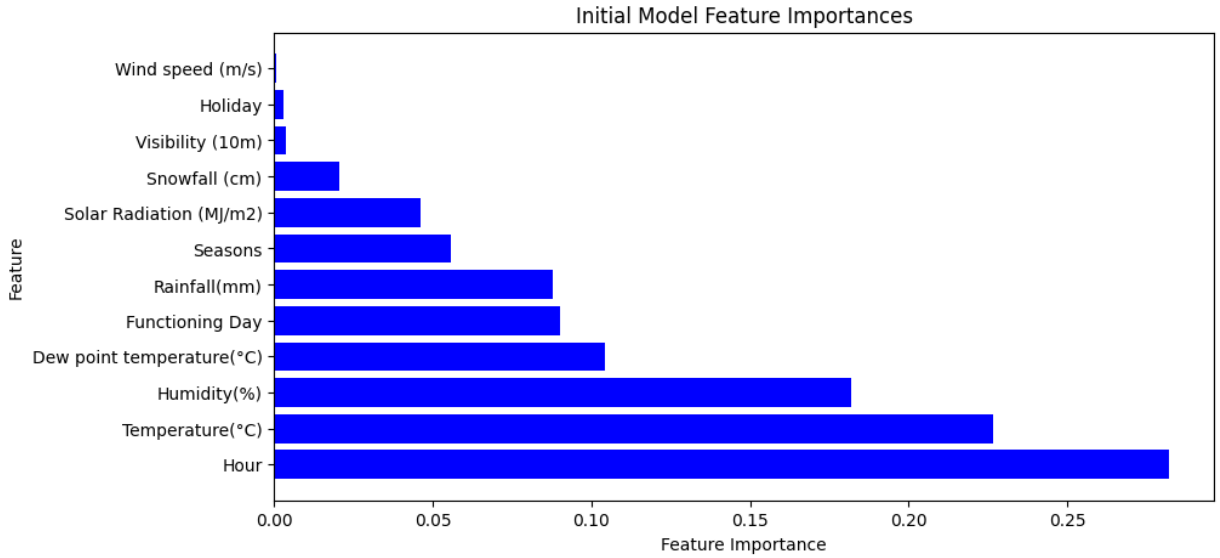

Figure 4: **Initial Model Feature Importances**

The plot highlights that Hour and Temperature(°C) are the dominant features, with other features contributing less. This indicates that the initial model primarily relies on time of day and temperature to make predictions.

2.4.2 Best Model

- **Hour:** Remains the most important feature, confirming its critical role in predicting bike rentals.
- **Temperature(°C):** Continues to be highly significant, emphasizing the importance of weather conditions.
- **Seasons, Humidity(%), Functioning Day:** These features have increased in importance, indicating that the refined model better captures their impact on bike rentals.
- **Dew point temperature(°C), Rainfall(mm), Solar Radiation (MJ/m^2):** These features have moderate importance, reflecting their influence on the predictions.
- **Wind speed (m/s), Visibility (10m), Snowfall (cm), Holiday:** These features remain the least important but are still considered in the best model, suggesting their role is minor but not negligible.

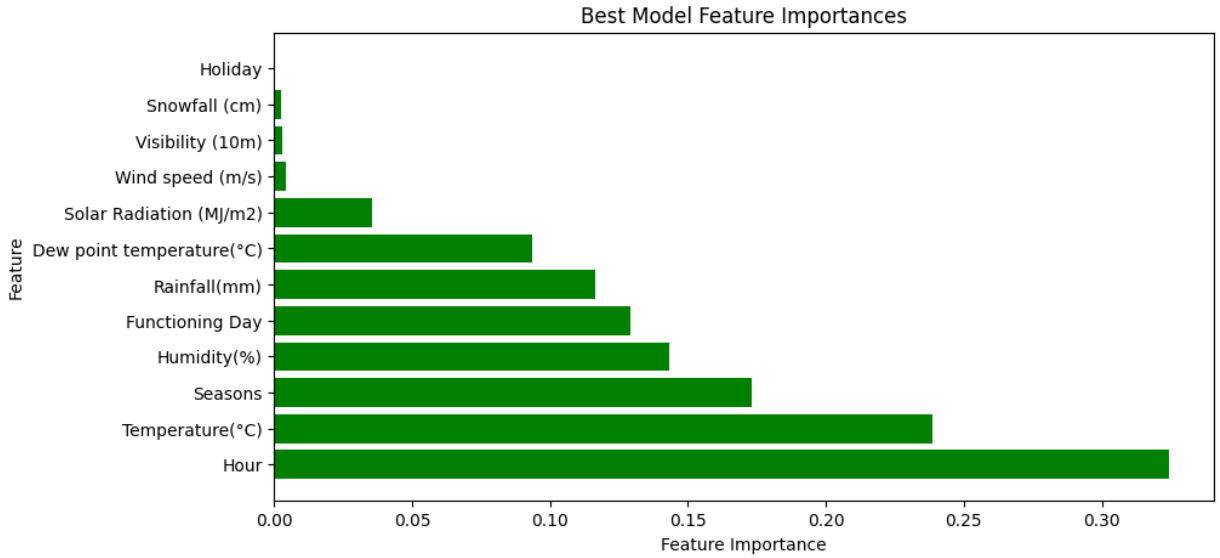

Figure 5: Best Model Feature Importances

The plot shows a more balanced distribution of feature importances, with increased significance for Seasons, Humidity(%), and Functioning Day. This suggests that the best model has a more comprehensive understanding of the factors affecting bike rentals, leading to better predictions.

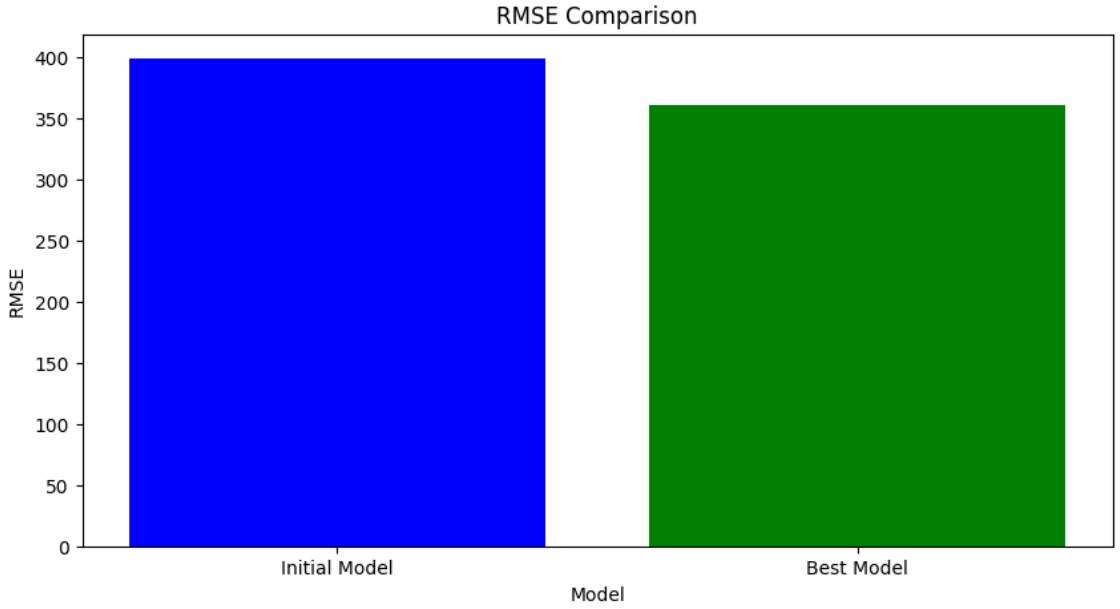

Figure 6: RMSE comparison

The RMSE comparison plot clearly shows a significant reduction in error for the best model compared to the initial model. This improvement demonstrates the effectiveness of hyperparameter tuning and the use of a more complex neural network architecture in enhancing predictive performance.

2.5 Conclusion

The development and refinement of the neural network model for predicting bike rentals in Seoul demonstrated several key insights:

1. **Model Complexity:** Increasing the complexity of the neural network and tuning the regularization parameter significantly improved the model's accuracy.
2. **Feature Importance:** The refined model better captures the impact of various features, providing a more nuanced understanding of the factors influencing bike rentals.
3. **Hyperparameter Tuning:** The grid search effectively identified the optimal hyperparameters, leading to improved predictive performance.

Overall, this assignment highlights the importance of iterative model development, hyperparameter tuning, and thorough analysis of feature importances in building effective predictive models.

References

[1] Chicco, D. and G. Jurman (2020). "Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone". In: *BMC Medical Informatics and Decision Making*, 20(16). ISSN: 1472-6947. https://doi.org/10.1186/s12911-020-1023-5.

[2] Sathishkumar, V. E., Jangwoo Park, and Yongyun Cho (2020). Using data mining techniques for bike sharing demand prediction in metropolitan city. In: *Computer Communications*, 153, pp. 353-366. ISSN: 0140-3664. https://doi.org/10.1016/j.comcom.2020.02.007. https://www.sciencedirect.com/science/article/pii/S0140366419318997.

[3] Sathishkumar, V. E. and Yongyun Cho (2020). A rule-based model for Seoul Bike sharing demand prediction using weather data. In: *European Journal of Remote Sensing*, 53.sup1, pp. 166-183. https://doi.org/10.1080/22797254.2020.1725789. https://doi.org/10.1080/22797254.2020.1725789.