University of Cape Town

Course Code: STA5076Z

Course Name: UNSUPERVISED LEARNING

SVMs AND ANNs REPORT

Author:
Dibanisa Fakude

Student Number:
FKDDIB001

May 23, 2024

Contents

Authorship Declaration

I, Dibanisa Fakude, declare that:

1. This research report and the work presented in it, is my own.
2. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
3. These calculations/report/plans are my own work.
4. I have not allowed and will not allow anyone to copy my work with the intention of passing it off as his or her own work.

Signature: D. Fakude

1 Project Overview

This project involves two tasks, the first task, the goal is to develop a Support Vector Machine (SVM) model to predict the occurrence of death events based on clinical features in heart failure patients. The dataset contains 13 clinical features, including age, anaemia, high blood pressure, and others. The approach involves splitting the data into training and testing sets, building SVM models with different hyperparameters using grid search, evaluating model performance metrics such as accuracy, recall, specificity, F1 score, and ROC AUC, and comparing the results from multiple runs and hyperparameter combinations. The best model parameters are determined using grid search and tested on the test data.

For the second task, the objective is to predict the number of bicycle share rentals per hour in Seoul, South Korea, using neural networks. The dataset includes various features such as temperature, humidity, windspeed, and others. The process entails splitting the data into training and testing sets, building neural network models with different configurations (e.g., number of hidden layers, neurons, regularization parameters) using cross-validation or grid search, selecting the best model parameters, rebuilding the model with the best parameters, and evaluating its performance on the test data. Additionally, the importance of variables in the best model is explained and compared with the model's performance in the initial configuration.

2 Support Vector Machines

2.1 Data Wrangling and Exploration

In the data wrangling phase, various numeric variables such as Anemia, Diabetes, High Blood Pressure, Sex, and Smoking were converted into factor variables using the mutate function. This transformation enhances the interpretability and categorization of these variables. For instance, Anemia was converted into a factor with two levels: "No" and "Yes", representing the absence or presence of anemia, respectively. Similarly, Diabetes, High Blood Pressure, Sex, and Smoking were also converted to factor variables with corresponding labels to facilitate analysis and interpretation. Additionally, the correlation matrix is computed to explore relationships between numerical features. Following this exploration, visualizations are employed to further understand the data. A stacked bar plot is generated to illustrate the distribution of death events by sex, providing insight into potential associations between these variables. Through these sequential steps, a comprehensive understanding of the dataset is obtained, laying the groundwork for subsequent analysis and modeling endeavors.

The data was split into training and testing sets using a random 80-20 split. To achieve this, the dataset was first randomly partitioned into two subsets: 80% of the data was allocated to the training set, while the remaining 20% was allocated to the testing set. This random splitting process helped ensure that the model was trained on a diverse range of data and evaluated on unseen instances. Additionally, to address class imbalance in the training set, upsampling was performed on the minority class (DEATH_EVENT) using the 'upSample' function. This technique involved artificially increasing the number of instances in the minority class which Incorporated 32% of the sample to match the major class which Incorporated 68% of the class distribution. The upsampling process was conducted solely on the training set to prevent data leakage. After upsampling, the class distribution in the upsampled training set was verified to ensure a balanced representation of both classes. Table 1&2 Below is the class distribution of the response variable "DEATH_EVENT" before and after resampling.

No	**Yes**
162	77

Table 1: Class Distribution (before upsampling)

No	**Yes**
162	162

Table 2: Class Distribution (after upsampling)

b)The results depict the performance metrics of a support vector machine (SVM) model trained using a radial kernel function with cost=0.1 and gamma=0.1. The

classification accuracy of approximately 78.33% indicates that the model correctly predicted the class labels for about 78.33% of the instances. Furthermore, the recall, representing the proportion of actual positive instances correctly predicted as positive, stands at approximately 60.71%, suggesting that the model captured around 60.71% of the true positive instances. On the other hand, the specificity, measuring the proportion of actual negative instances correctly predicted as negative, is approximately 93.75%, indicating a high rate of correctly identified true negative instances. The F1 score, a harmonic mean of precision and recall, is around 72.34%, indicating balanced performance in terms of precision and recall. Additionally, the ROC AUC, which evaluates the model's ability to discriminate between classes, is approximately 81.32%, reflecting good discriminatory power. Overall, these results illustrate the SVM model's effectiveness in accurately classifying instances and its ability to balance precision and recall while discriminating between classes.

Metric	**Value**
Classification Accuracy	0.7833333
Recall	0.6071429
Specificity	0.9375
F1 Score	0.7234043
ROC AUC	0.8132221

Table 3: Model Performance Metrics

c)The reported results present the average performance metrics of a support vector machine (SVM) model over 100 repetitions of the training and testing process. The average accuracy, computed as approximately 81.58%, indicates the proportion of correctly classified instances across the 100 runs. Moreover, the average recall, standing at approximately 66.53%, signifies the average proportion of actual positive instances correctly predicted as positive by the model. The average specificity, approximately 92.41%, reflects the average proportion of actual negative instances correctly predicted as negative. Additionally, the average F1 score, calculated as around 74.7%, represents the harmonic mean of precision and recall, providing a balanced measure of the model's accuracy across the repetitions. Finally, the average AUC value, approximately 82.68%, assesses the model's ability to distinguish between the two classes across the 100 runs. These results collectively demonstrate the consistent performance of the SVM model over multiple repetitions, showcasing its effectiveness in accurately classifying instances and discriminating between classes.Figure 1 below shows the box-plots of the SVM metrics over the 100th repetition.

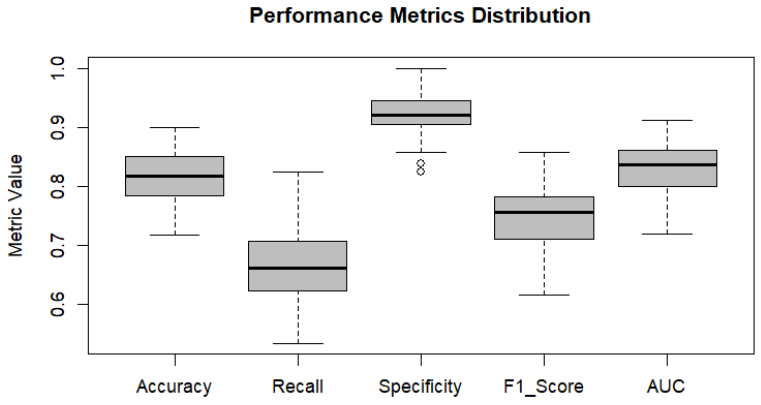

Figure 1: SVM Metrics Boxplot over 100 Runs

d)The reported results depict the performance of support vector machine (SVM) models trained with different combinations of cost and gamma parameters. The table includes the values of cost, gamma, accuracy, recall, specificity, F1 score, and AUC for each combination. Across the various parameter settings, the highest accuracy achieved was 96.67%, observed when the cost was set to 10.0 and gamma to 0.1. The second-highest accuracy of 90% was attained with several combinations, indicating the robustness of the model across different parameter settings. Conversely, the lowest accuracy was 85%, observed with a cost of 0.1 and gamma of 10.0, suggesting suboptimal performance under these parameters. Similarly, the highest recall of 100% was attained with several combinations, indicating the model's ability to correctly identify all positive instances. The second-highest recall of 100% was observed with various parameter combinations, highlighting the consistency of the model's performance in capturing positive instances. Conversely, the lowest recall was 75%, noted with a cost of 0.1 and gamma of 0.1, indicating potential shortcomings in capturing positive instances under these parameters. Specificity values were generally high, indicating the model's capability to correctly identify negative instances. However, the F1 scores and AUC values varied across parameter settings, suggesting that the model's overall performance was influenced by the choice of hyperparameters. Table 4 below shows the comparison of different metrics across different hyperparameters.

Cost	**Gamma**	**Accuracy**	**Recall**	**Specificity**	**F1 Score**	**AUC**
0.1	0.1	0.883	0.750	0.972	0.837	0.901
0.1	1.0	0.900	1.000	0.872	0.813	0.842
0.1	10.0	0.850	1.000	0.820	0.690	0.763
1.0	0.1	0.883	0.773	0.947	0.829	0.886
1.0	1.0	0.900	1.000	0.872	0.813	0.842
1.0	10.0	0.900	1.000	0.872	0.813	0.842
10.0	0.1	0.967	1.000	0.953	0.944	0.947
10.0	1.0	0.900	1.000	0.872	0.813	0.842
10.0	10.0	0.900	1.000	0.872	0.813	0.842

Table 4: SVM Performance Metrics with Different Cost and Gamma Parameters

Figure 2 below is visualizing the performance metrics across different parameter combinations, which provided additional insights.The heatmap displayed the performance metrics of a Support Vector Machine (SVM) model with varying cost and gamma parameters, with each subplot representing a different metric: Accuracy, Recall, Specificity, F1 Score, and AUC. The x-axis represents the cost parameter, and the y-axis represents the gamma parameter, with the color scale indicating the value of the performance metric from red (lowest) to green (highest). For accuracy, the highest values are seen at cost values of 10 and gamma values of 0.1, 1, and 10, indicating better overall performance. Recall shows high values consistently at gamma values of 1 and 10, across different cost values, highlighting effective identification of positive cases. Specificity is highest at cost values of 0.1 and 10, with gamma values of 0.1 and 1, suggesting the model excels at identifying negative cases in these regions. The F1 Score, combining both precision and recall, is highest at cost values of 10 and gamma values of 1 and 10, indicating a good balance between precision and recall. Lastly, the AUC, measuring the model's ability to distinguish between classes, is highest at cost values of 10 and gamma values of 1 and 10, signifying strong discriminatory power. Overall, the heatmap reveals that a cost value of 10 combined with gamma values of 1 or 10 yields the best performance across all metrics, while lower cost and gamma values result in poorer performance.

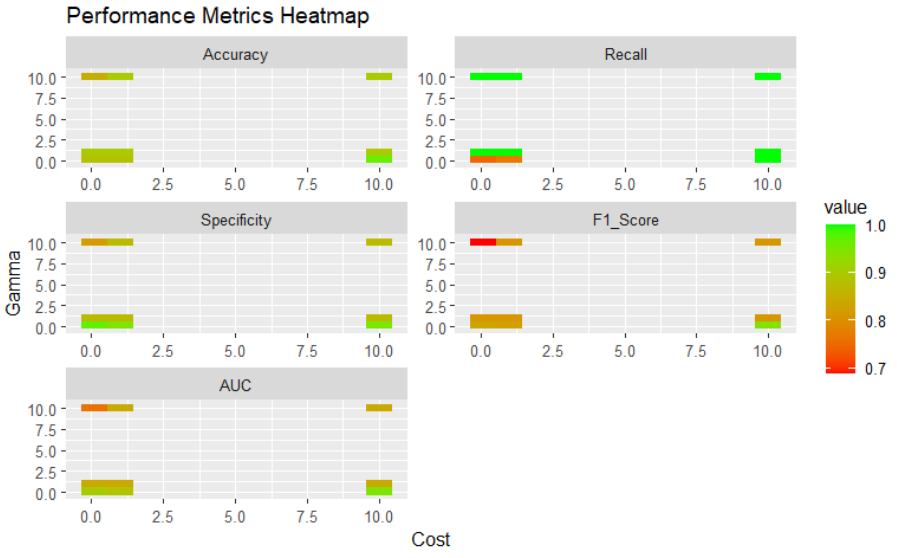

Figure 2: Perfomance metrics heatmap

e)After performing a grid search to tune the cost and gamma parameters for the SVM model, the best parameters identified were a cost of 1 and a gamma of 1. Using these parameters, the model was tested on the test data, resulting in a test accuracy of 0.9. This indicates that the model, with the optimized parameters, is

able to correctly classify 90% of the instances in the test set. The grid search was conducted over a range of values for both parameters: cost values of 0.01, 0.1, 1, 10, and 100, and gamma values of 0.01, 0.1, 1, 10, and 100. This comprehensive search ensured that the best combination of parameters was identified to maximize the model's performance.

3 Artificial Neural Networks

3.1 Data Wrangling and Exploration

Firstly, the dataset was explored to identify underlying patterns and anomalies. The distribution of the response variable "Rented Bike Count" was plotted using a distribution plot, which revealed that the data follows a normal distribution with some outliers(see figure 3).

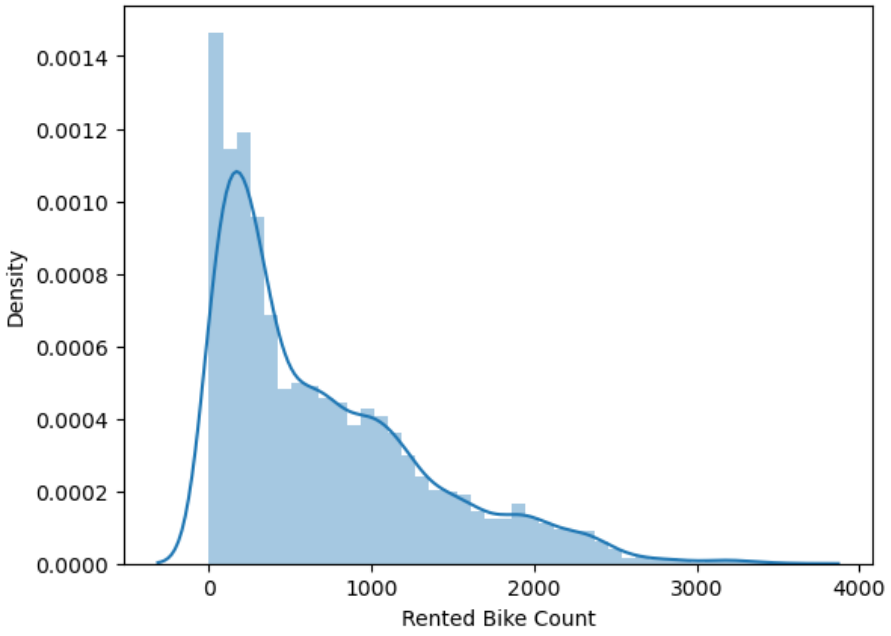

Figure 3: Distribution of the response variable

Secondly, categorical variables were visualized using boxplots, as shown in Figure 4. The first boxplot indicated that people tend to rent more bikes on non-holidays compared to holidays. The second boxplot illustrated bike rentals based on seasons, showing explicitly that more bikes are rented in summer, followed by autumn and spring, with winter having the lowest rental counts. The third boxplot displayed bike rentals on functioning days, demonstrating that more bikes are rented on functioning days compared to non-functioning days..Using a correlation heatmap, the relationships between numerical variables were analyzed. This analysis revealed how each variable correlates with the others. Notably, temperature and dew point temperature exhibited a strong positive correlation of 0.91. Additionally, rented bike count and temperature showed a positive correlation of 0.54 as shown if figure 5 below.

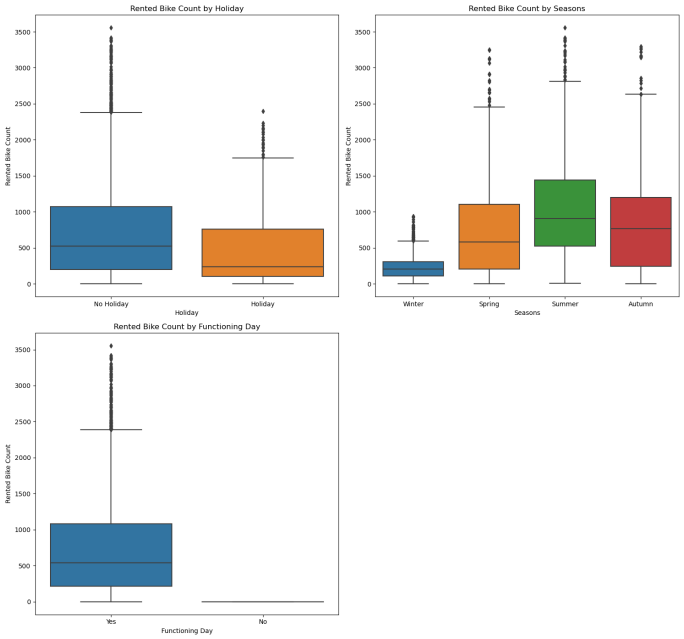

Figure 4: Distribution of catergorical variables

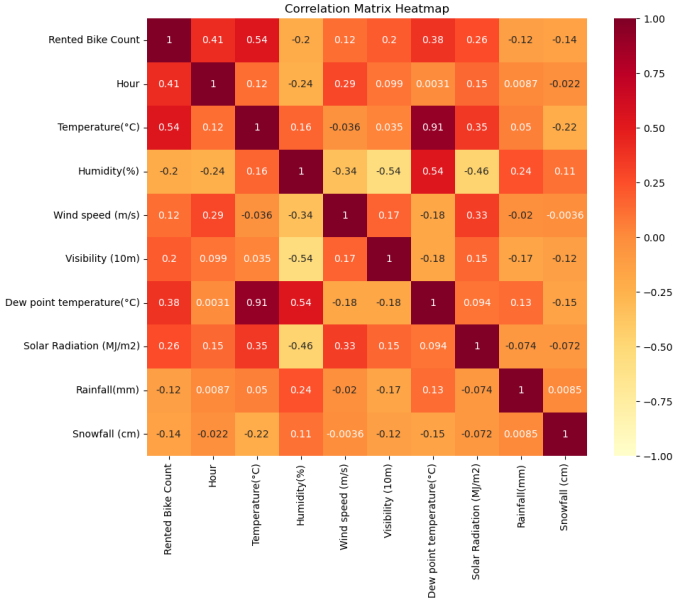

Figure 5: Heatmap of Numeric Varibles

This was followed by label encoding of categorical variables to convert them into numerical format suitable for machine learning models. Specifically, the last two columns in ‘X‘ were label encoded using ‘LabelEncoder‘ from ‘sklearn.preprocessing‘. This step transformed categorical values into numerical labels. Next, the ”Season” variable, which is also categorical, was one-hot encoded using the ‘ColumnTransformer‘ and ‘OneHotEncoder‘ from ‘sklearn.compose‘. This encoding created separate binary columns for each season, allowing the neural network to interpret this categorical information more effectively. The resulting feature matrix, ‘X‘, now included the one-hot encoded season variables along with the original features, all in numerical format, ready for further model development.After preprocessing, the dataset underwent a pivotal step in machine learning: splitting into training and testing sets. This was accomplished by allocating 20% of the data for testing and reserving the remaining 80% for training. This division ensures that the model's performance can be accurately assessed on unseen data.Figure 3 below shows the distribution of the response variable ”Rented Bike Count”.

b)In this analysis, a neural network was constructed using TensorFlow in Python, with a specific architecture comprising one hidden layer housing two neurons (see Figure 6). To mitigate overfitting and enhance generalization, L1 (Lasso) regularization with a coefficient of 0.0001 was applied. After training the neural network on the dataset, the model's performance was evaluated using the Root Mean Squared Error (RMSE) metric. The computed RMSE value was found to be 390.799. RMSE serves as a crucial indicator of the model's predictive accuracy, quantifying the average magnitude of the errors between the actual and predicted values. This value provides insight into the model's ability to accurately predict the target variable, with lower RMSE values indicating better performance. Thus, in this context, the RMSE of 390.799 signifies the average deviation of the model's predictions from the actual values, serving as a benchmark for assessing its effectiveness in capturing the underlying patterns in the data.

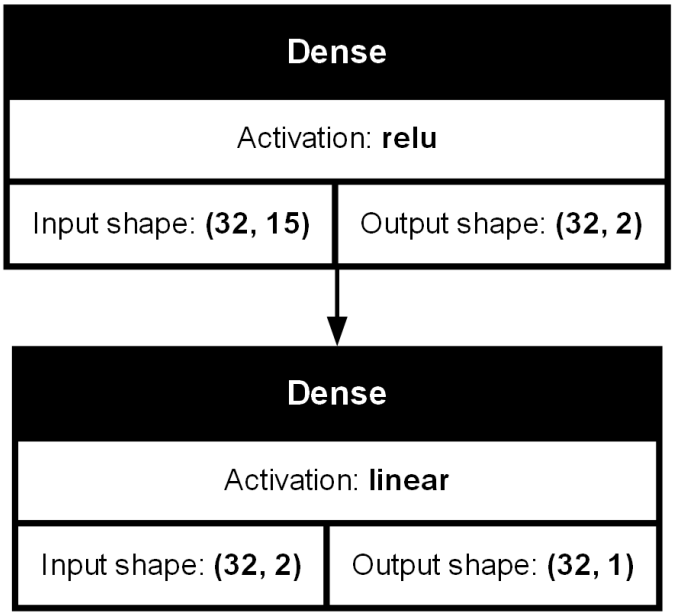

Figure 6: Artificial Neural Network architecture

c)Using a grid search approach with cross-validation, the aim was to determine the optimal model parameters for predicting continuous data by finding the best combination of hidden layers, neurons per layer, and regularization rate to minimize prediction error. After conducting an extensive grid search over multiple configurations, the best model parameters identified were three hidden layers, 32 neurons per layer, and L1 regularization with a coefficient of 0.001.

d)Before implementing parameter optimization through grid search, the initial model had a simpler architecture, consisting of only one hidden layer with two neurons and L1 regularization with a coefficient of 0.0001. This initial configuration yielded a higher RMSE on the test data, totaling 391.141. In contrast, after refining the model with the best parameters identified from the grid search—three hidden layers, 32 neurons per layer, and L1 regularization with a coefficient of 0.001—the model's performance improved significantly, resulting in a substantially lower RMSE of 247.962. Figure 7 below shows the loss curve helped in monitoring how well the model is learning access overfitting during the training process. A decreasing loss indicates that the model is improving and learning the patterns in the training data.

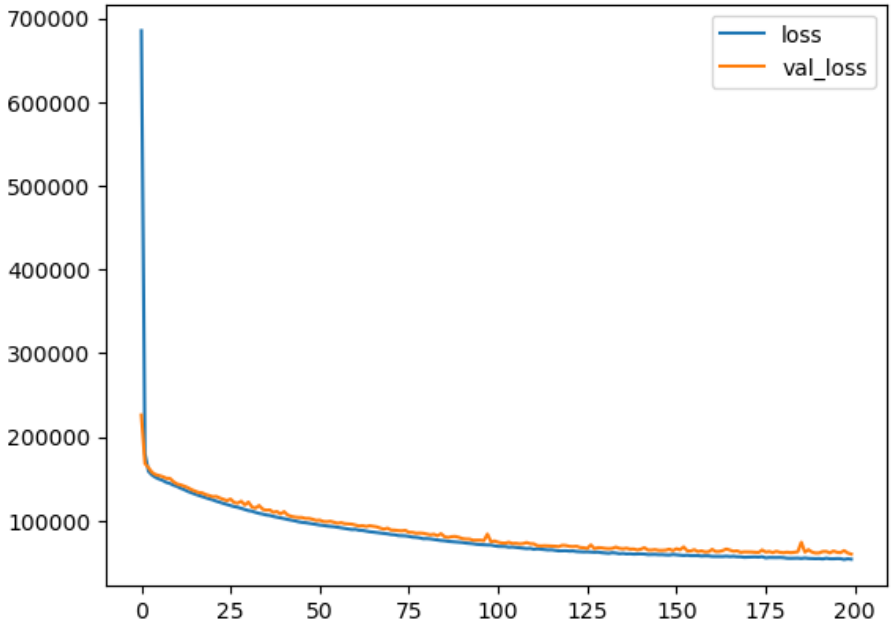

Figure 7: Loss curve

Notably, the deeper architecture (see Figure 8) and increased regularization in the optimized model allowed for better capture of complex data patterns and more effective mitigation of less significant features. Additionally, the optimized model

achieved an explained variance score of 0.852, further indicating its superior ability to account for the variability in the target data compared to the initial model.

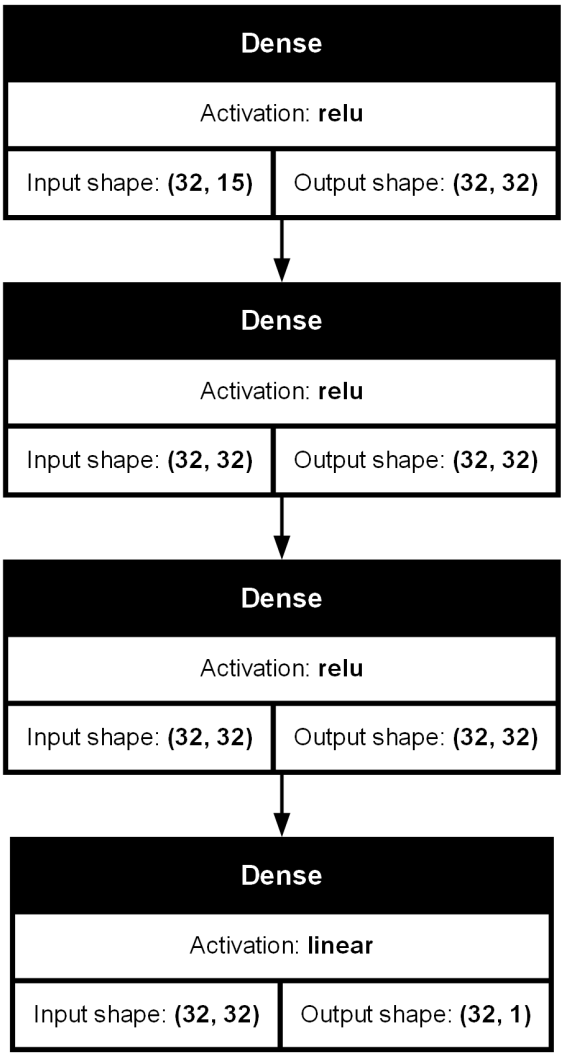

Figure 8: ANN architecture of best model parameters

Figure 9 illustrates that the variable importances in the optimized model show more pronounced and informative impacts of input features on the target variable compared to the flatter importances observed in the initial model (see Figure 10). This comparison highlights the crucial role of parameter optimization in enhancing model performance and clarifying the underlying relationships between input features and the target variable.In the best model's variable importance plot, 'Dew Point Temperature' and 'Hour' have high feature values on the positive side (right), indicating that they increase the predicted output (i.e., increase bike rentals). Conversely, in

the initial model (model a), 'Hour' is positioned in between, neither significantly increasing nor decreasing the outcome. 'Temperature' in the initial model shows an increase in the predicted outcome, while 'Rainfall' and 'Humidity' are high on the negative side (left), indicating a decrease in the predicted output due to their high negative values.

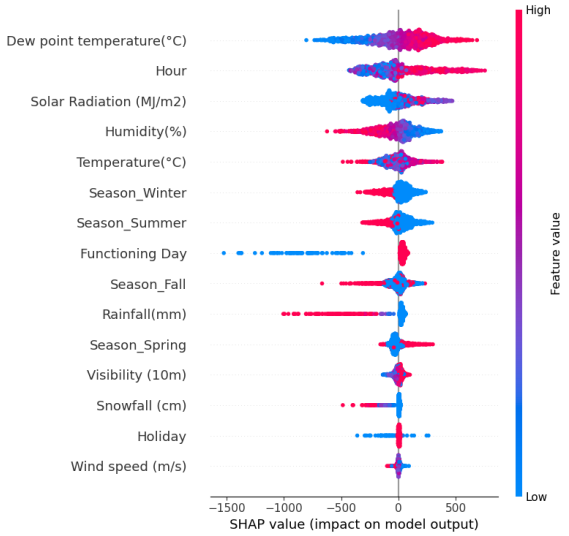

Figure 9: Variable importance for best model

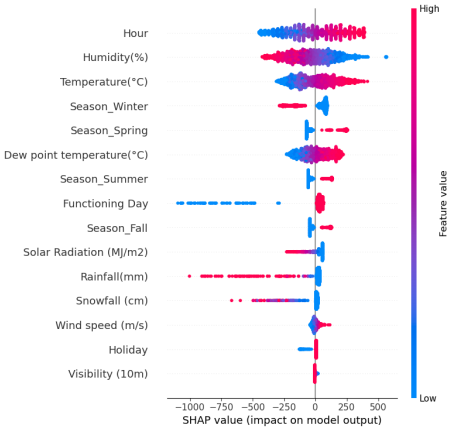

Figure 10: Variable importance for the (a) model

4 Conclusion

By plotting the actual versus predicted values, I could visually inspect how closely my model's predictions align with the true values.The points fell close to the diagonal line (y_test = predictions1), indicating that the predicted values match the true values and validating my best model.Refer to Figure 11 below.

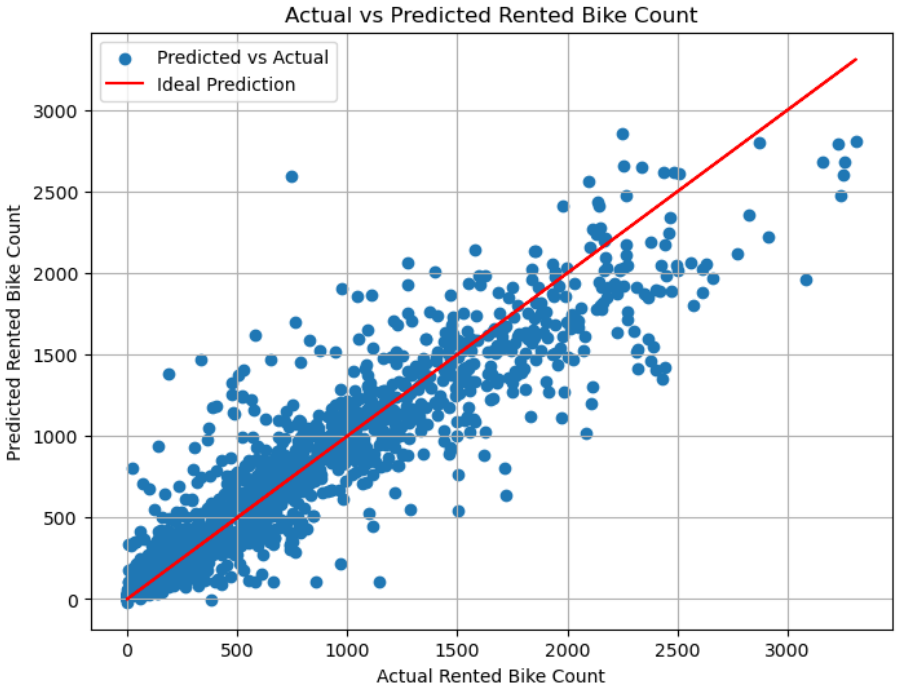

Figure 11: Actual Versus Predicted Rented Bike Count

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. doi: 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. doi: https://doi.org/10.1080/22797254.2020.1725789. eprint: https://doi.org/10.1080/22797254.2020.1725789.