I. ITNPBD6 Assignment 1, Student Number: 3071806

II. Methodology:

1. Data Import: The dataset was read and imported using pandas library.
2. Data Exploration: To visualize the data distribution and identify potential issues
   - Checked for missing values and duplicate rows. No missing values or duplicates were found, ensuring the data was complete and unique for analysis.
   - Histograms and box plots were employed to visualize the distribution of the data and identify potential outliers or anomalies. This helped to assess whether further data cleaning was necessary.
3. Data Cleaning: To ensure data quality and consistency data cleaning steps were performed
   - Corrupted strings 'Y' and 'N' in the 'Car park' column were replaced with 'Yes' and 'No' for consistency.
   - Rows with 'Staff' count outside the range of (0, 100) were removed, as they were considered outliers likely due to errors or measurement problems.
   - The only 'Village' location in the 'Location' column was removed to reduce dimensionality and avoid biased representation of data on a single element.
4. Data Preprocessing: To prepare it for machine learning algorithms
   - Encoded the target variable 'Performance' as 1 for 'Good' and 0 for 'Bad' for binary classification.
   - Dropped unwanted columns such as 'Town', 'Country', 'Store ID', and 'Manager name' to reduce dimensionality and focus on relevant features.
   - Encoded categorical variables using LabelEncoder and one-hot encoding to convert them into a suitable format for machine learning algorithms.
5. Feature Selection: To find the optimal features for each model to improve performance
   - Applied the SelectKBest method with f_classif as the scoring function to find the optimal k for each model (Logistic Regression, Decision Tree, and Neural Network) using cross-validation.
   - Selected the top k features for each model, combining the features from Logistic Regression and Decision Tree to create a final set of features for improved performance.
6. Train/Validate/Test Split: To evaluate models and tune hyperparameters on separate data
   - • Initially split the data into training and test sets with an 80/20 ratio. Then, further split the training set into training and validation sets using a 75/25 ratio, resulting in a 60/20/20 ratio overall for train/validation/test sets.
   - • Allowed for model evaluation and hyperparameter tuning on separate data to prevent overfitting and provide a more accurate assessment of model performance.
   - • Determined the best scaler for each model using the validation set.
7. Scaling and Hyperparameter Tuning: find the best scaler and to optimize model performance
   - Applied the best scalers to the training and validation sets for each model to normalize the data, making it suitable for algorithms sensitive to feature scales.
   - Trained models with scaled data and conducted hyperparameter tuning using GridSearchCV with F1-score as the scoring metric due to misclassification of stores as good or bad.
   - Trained the models again using the entire training set with the tuned hyperparameters for optimal performance.
8. Model Evaluation: To compare and select the best model
   - Evaluated the performance of each model (Logistic Regression, Decision Tree, and Neural Network) using the test set, comparing their accuracy, precision, recall, and F1-score.
   - Selected the best model based on F1-score to ensure a balance between precision and recall.

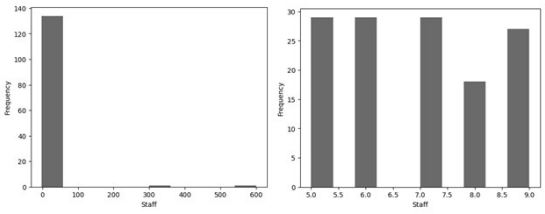
III. Model Features:

Feature	Data Type
Car park_0	Categorical - Binary
Clearance space	Numeric - Continuous
Competition number	Numeric - Discrete
Competition score	Numeric - Continuous
Floor Space	Numeric - Continuous
Location_0	Categorical - Binary
Location_2	Categorical - Binary
Staff	Numeric - Discrete
Window	Numeric - Discrete

One consequence of the choices made in treating the variables is that the categorical features like 'Car park' and 'Location' have been encoded as separate binary features using one-hot encoding. This approach allows the model to treat these categorical features correctly, without imposing any ordinal relationship between the categories. However, it also increases the dimensionality of the dataset, which might lead to overfitting if there aren't enough samples for the model to learn from effectively.

IV. Data preparation:

Number of staff employed



The initial histogram revealed negative values and extreme outliers, which suggested that there were errors in the data. To address these issues, I carried out the following data cleaning operations:
Removed rows with negative 'Staff' values, as they're illogical, ensuring data consistency.
Eliminated extreme outliers by defining an acceptable range (0-100) for 'Staff', enhancing data quality by removing potential errors.

V. Model Training and Hyper Parameters:

Model	Hyperparameters	F1 Score
Logistic Regression	{'C': 10, 'solver': 'liblinear'}	0.761905
Decision Tree	{'criterion': 'gini', 'max_depth': 2, 'min_samples_split': 2}	0.600000
Neural Network	{'activation': 'relu', 'alpha': 0.0001, 'hidden_layer_sizes': (100,)}	0.666667

I used GridSearchCV to explore possible hyperparameter combinations, selecting the best ones based on the F1 Score validation metric. To ensure a comprehensive evaluation of model effectiveness, I considered additional metrics such as accuracy, precision, and recall. This approach provided a holistic understanding of the model's performance for classifying stores as well-performing or poorly-performing, while placing emphasis on the balance between precision and recall through the F1 Score.

I identified key hyperparameters by considering the algorithm and the store classification problem at hand. For Logistic Regression, I chose 'C' (regularization strength) and 'solver' (algorithm to use in optimization). For the Decision Tree, I focused on 'criterion' (quality of a split), 'max_depth' (maximum depth of the tree), and 'min_samples_split' (minimum number of samples required to split a node). Lastly, for the Neural Network, I considered 'activation' (activation function for the hidden layers), 'alpha' (L2 penalty or regularization term), and 'hidden_layer_sizes' (number of neurons in the hidden layers).

VI. Final Model Results:

Based on the highest F1 Score, I selected the Logistic Regression model as the final model to classify store performance, ensuring a balance between precision and recall. In conclusion, the model proved to be the most effective in accurately predicting whether a store is well-performing or poorly-performing. Although, this model can be utilized by the organization to identify and analyse the factors contributing to store performance.

	Predicted Good	Predicted Bad
Actual Good	14	2
Actual Bad	3	8

The confusion matrix showed 14 well-performing stores (True Positives) and 8 poorly-performing stores (True Negatives) correctly identified, with 2 False Positives and 3 False Negatives. This indicates that Logistic Regression is suitable for the store performance classification task.

Test Set Logistic Regression:
Accuracy = 0.814, Precision = 0.8, recall = 0.727

VII. References:

scikit-learn: https://scikit-learn.org/stable/
pandas: https://pandas.pydata.org/
matplotlib: https://matplotlib.org/
Stack Overflow: https://stackoverflow.com/
Python documentation: https://docs.python.org/
GitHub: https://github.com/
OpenAI: https://openai.com/
Towards Data Science: https://towardsdatascience.com/
DataCamp: https://www.datacamp.com/