BIKE SHARING DEMAND PREDICTION

TECHNICAL DOCUMENTATION

Project Type – Regression

Name - Abhay Kumar

PROBLEM STATEMENT

Bike-sharing programs have revolutionized urban mobility by offering convenient access to rental bikes. However, the challenge of maintaining a consistent supply of bikes is crucial. These automated systems enable users to rent and return bikes at multiple locations, reducing waiting times and enhancing transportation accessibility. This project focuses on leveraging historical data, including temperature and time, to forecast demand for Seoul's bike-sharing program. Predicting the hourly bicycle count is vital for ensuring timely access to rental bikes. The interconnected network of automated stores streamlines the entire process. This dataset-driven predictive model aims to optimize bike supply, meeting the dynamic transportation needs of the city's residents.

BUSINESS OBJECTIVE

Predicting the optimal number of bikes required at any given moment and day is a complex yet crucial challenge for bike-sharing enterprises. Striking the right balance is paramount because too few bikes can lead to resource inefficiencies, encompassing increased maintenance costs and the need for additional parking and security infrastructure. Conversely, an excess of bikes can result in financial losses, including an immediate reduction in revenue due to limited customer engagement and potential long-term consequences such as eroding customer trust and loyalty.

Therefore, possessing a robust demand estimation mechanism is imperative for bike rental businesses. Such a mechanism enables efficient operations, maximizes resource utilization, and ensures a seamless customer experience. Accurate demand forecasts are the linchpin of informed decision-making and the overall success of bike-sharing ventures.

DATA OVERVIEW

Attribute Information:

- **Date:** Given in the format year-month-day.
- **Rented Bike count:** Reflects the count of bikes rented for each hour.
- **Hour:** Indicates the hour of the day.
- **Temperature:** Measured in Celsius.
- **Humidity:** Represented as a percentage.
- **Windspeed:** Measured in meters per second.
- **Visibility:** Measured in meters.
- **Dew point temperature:** In Celsius.
- **Solar radiation:** Measured in MJ/m2 (MegaJoules per square meter).
- **Rainfall:** Measured in millimeters.

- **Snowfall:** Measured in centimeters.
- **Seasons:** Categorized as Winter, Spring, Summer, or Autumn.
- **Holiday:** Classifies whether it's a holiday or a regular day.
- **Functional Day:** Distinguishes between Functional Hours and Non-Functional Hours.

DATASET INFORMATION

Our initial steps involved loading the dataset and importing necessary libraries. Subsequently, we conducted a comprehensive data inspection, which entailed examining the first and last rows, evaluating the dataset's dimensions, scrutinizing its column structure, and gaining a fundamental understanding of the data. This dataset encompassed a mix of continuous and categorical variables, presenting a multifaceted landscape for analysis and modelling.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8760 entries, 0 to 8759
Data columns (total 14 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Date                       8760 non-null   object
 1   Rented Bike Count          8760 non-null   int64
 2   Hour                       8760 non-null   int64
 3   Temperature(°C)            8760 non-null   float64
 4   Humidity(%)                8760 non-null   int64
 5   Wind speed (m/s)           8760 non-null   float64
 6   Visibility (10m)           8760 non-null   int64
 7   Dew point temperature(°C)  8760 non-null   float64
 8   Solar Radiation (MJ/m2)    8760 non-null   float64
 9   Rainfall(mm)               8760 non-null   float64
 10  Snowfall (cm)              8760 non-null   float64
 11  Seasons                    8760 non-null   object
 12  Holiday                    8760 non-null   object
 13  Functioning Day            8760 non-null   object
dtypes: float64(6), int64(4), object(4)
memory usage: 958.2+ KB
```

This project utilizes a dataset with 14 key columns, including date, temperature, humidity, and more, to build a predictive model for bike rental counts. The dataset contains 8760 observations, and 'casual' and 'registered' columns contribute to the 'count' target variable. The 'datetime' column serves as an index for efficient data access.

EXPLORATORY DATA ANALYSIS (EDA)

Exploratory Data Analysis (EDA) stands as a critical early stage in data science, dedicated to deepening our understanding of the dataset. Its fundamental objective is to unearth hidden insights, identify patterns, and discern relationships and trends inherent in the data. EDA represents an iterative journey, instrumental in flagging outliers, managing missing data, and addressing potential data anomalies that could impede accurate analysis and modeling. This methodical exploration not only helps in data preparation but also forms the bedrock for informed decision-making and the development of robust predictive models, enhancing the overall quality and reliability of data-driven insights.

DATA CLEANING

Data cleaning is a foundational step in data preparation, dedicated to identifying and rectifying inaccuracies, disparities, and missing data points within a dataset. Positioned as a critical phase in the data processing pipeline, it carries the vital responsibility of ensuring data accuracy, completeness, and its presentation in a format suitable for analysis. This procedure encompasses various actions such as removing duplicate entries, filling in missing information, correcting errors, and ensuring uniform data structures. The overarching goal of data cleaning is to elevate data quality, making it suitable for further analytical exploration and modeling, thereby enhancing the reliability and utility of the dataset for decision-making and insights.

FEATURE ENGINEERING & DATA PRE-PROCESSING

Feature engineering is a pivotal and strategic phase in the development of machine learning models. It involves the careful manipulation of data to create new features or to refine existing ones to enhance the model's ability to uncover patterns, relationships, and meaningful information within the data. The quality and relevance of features profoundly impact a model's success. Effective feature engineering not only optimizes a model's performance but also expedites the learning process. It can reveal hidden patterns, reduce noise, and improve the model's accuracy in making predictions or classifications. In essence, feature engineering equips the model with the right set of tools to extract meaningful insights from the data, serving as a cornerstone for successful machine learning endeavors. It transforms raw data into a more informative and interpretable format, aligning it with the model's requirements and ultimately contributing to the model's ability to make informed decisions and produce valuable results.

1 Regression plot

The majority of numerical features exhibit a positive correlation with the target variable, indicating that as these features increase or decrease, the target variable tends to follow a corresponding trend, highlighting their potential significance in influencing the target's behaviour.

2 Correlation Coefficient and Heatmap

The correlation coefficient measures the strength and direction of a linear relationship between variables. A high correlation (0.91) between dew_point_temperature and temperature led to dropping dew_point_temperature. This decision simplifies the model, reduces multicollinearity, and enhances its interpretability and performance.

3 Variance Inflation Factor(VIF) Analysis Explanation

VIF (Variance Inflation Factor) analysis is essential for detecting multicollinearity, where predictors are highly correlated in a regression model, introducing redundancy. It ensures reliable and interpretable models by identifying and addressing multicollinearity, ensuring that each predictor contributes unique information, enhancing model quality and accuracy.

4 Encoding in Feature Engineering

Encoding is vital in feature engineering, converting categorical variables into numeric forms for machine learning. Methods like one-hot, label, ordinal, and count encoding offer various approaches. The choice depends on data nature and model needs, ensuring effective use of categorical data in machine learning algorithms.

5 Normalization of Target Variable

Data normalization is crucial in data mining to transform data for efficient processing. It reduces duplicate data, enhancing quality and consistency. In this context, Rented Bike Count's moderately right-skewed distribution needs normalization for linear regression. Outliers in the column can impact the model's performance and should be addressed.

ML MODEL IMPLEMENTATION

The "train-test split" represents a fundamental practice in the realm of machine learning. This procedure involves partitioning a dataset into two distinct subsets: the "training set" utilized to teach a machine learning model, and the "test set" employed to assess the model's performance. This division is essential to gauge how effectively the model generalizes its learning to unseen data. Typically, a significant portion of the data, such as 70-80%, is allocated to training, while the remainder is preserved for testing. By subjecting the model to data it has not encountered during training, we can gauge its capacity to provide accurate predictions for novel, real-world data, thereby evaluating its overall efficacy.

MODEL IMPLEMENTATION

We Employed 9 ML models named

1. Linear Regression:

Linear regression connects a target variable to predictor variables, finding the best linear equation for predictions. It's fundamental for outcome forecasting in diverse fields, serving as a foundation for advanced modeling.

2 Lasso

Lasso (Least Absolute Shrinkage and Selection Operator) is a regularization technique for linear regression. It simplifies the model and improves generalization by penalizing feature coefficients. Lasso's unique feature is driving some coefficients to zero, aiding feature selection and reducing model complexity.

3 Ridge

Ridge Regression, a form of regularized linear regression, counters multicollinearity and overfitting by adding an L2 regularization penalty based on coefficient magnitudes' squares. It promotes smaller, non-zero coefficients for all features, different from Lasso, which drives some coefficients to zero. Ridge Regression balances model fit and complexity for stable predictions.

4 Elastic Net

ElasticNet combines L1 (Lasso) and L2 (Ridge) regularization in linear regression to prevent overfitting. It balances feature selection and multicollinearity reduction, inheriting Lasso's feature selection and Ridge's coefficient reduction properties. It's versatile for complex datasets requiring both regularization and feature selection.

5 K-Nearest Neighbors

K-nearest neighbor (KNN) is a versatile supervised learning algorithm used for classification and regression without strict data assumptions. It relies on similarity among data points, utilizing distance

metrics like Euclidean and Manhattan distances. In classification, it predicts by majority vote of the K-nearest neighbors, while in regression, it averages their values. Careful selection of K and distance metric is crucial for optimal performance.

6 Support Vector Machine

The Support Vector Machine (SVM) is a powerful machine learning algorithm for classification and regression. It finds optimal hyperplanes to separate classes or predict values. SVM excels in high-dimensional data and offers versatile kernel functions. The regularization parameter (C) balances margin and classification errors, making SVM suitable for various applications.

7 Decision Tree

A decision tree is a core machine learning model for prediction and decision-making. It breaks down data by applying rules to input features, with nodes representing feature tests and branches indicating outcomes. Leaves contain final class predictions or decisions. Customizable parameters in models like DecisionTreeRegressor allow tailored model adjustments.

8 Random Forest

Random Forest, an ensemble learning algorithm, stands out by constructing numerous decision trees and aggregating their outputs to arrive at a final prediction. It's a departure from single decision trees, as it mitigates the risk of overfitting by blending the outcomes of multiple trees, each trained on a distinct subset of the data. The ensemble approach leverages the wisdom of many trees, and it's widely appreciated for its robustness, ability to handle high-dimensional data, and suitability for both classification and regression tasks.

Hyperparameter Tunning using GridSearchCV

Hyperparameter tuning using scikit-learn's `GridSearchCV` with a Random Forest Regressor. It defines a parameter grid, initializes the search, and performs 5-fold cross-validation to find the best hyperparameter combination for maximizing the model's performance in predicting the target variable.

9 Light GBM

LightGBM is an efficient, open-source gradient boosting framework ideal for large datasets and high-performance machine learning tasks. It distinguishes itself with a histogram-based node splitting approach, making it faster and memory-efficient. It's a valuable tool for tackling large-scale machine learning challenges. Additionally, the code demonstrates hyperparameter tuning for an XGBoost Regressor model using GridSearchCV. The parameters considered include the number of trees, tree depth, minimum samples for splitting, and minimum samples in a leaf node, optimizing the model's performance through cross-validation and R-squared scoring.

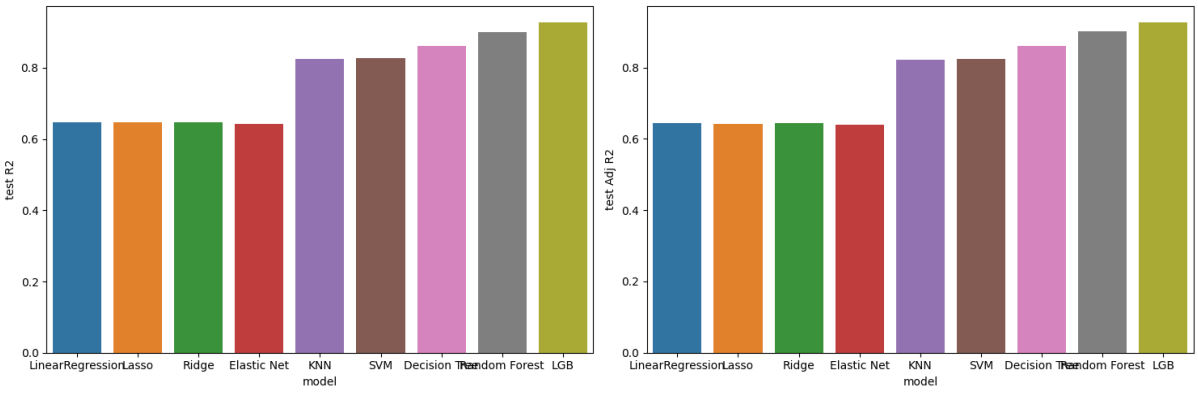
HyperParameter Tunning using GridSearchCV

Code snippet illustrates the process of hyperparameter tuning for an XGBoost Regressor using scikit-learn's GridSearchCV. It explores parameters like the number of trees, tree depth, node splitting criteria, and minimum samples in a leaf. The model is trained and assessed through 5-fold cross-validation, aiming to find the best hyperparameter combination for optimal XGBoost Regressor performance.

MODEL RESULT

The R-squared measure is useful for assessing the relationship between dependent and independent variables, but it doesn't address overfitting. In complex regression models, Adjusted R-squared is valuable as it considers variable relationships and penalizes unnecessary ones, reducing overfitting risks. R-squared is still reliable for understanding predictability in rented_bike_count.

	model	train MSE	test MSE	train RMSE	test RMSE	train MAE	test MAE	train R2	test R2	train Adj R2	test Adj R2
8	LGB	0.000	12.645	0.000	3.556	0.002	2.340	1.000	0.916	1.000	0.915
7	Random Forest	8.112	15.330	2.848	3.915	1.921	2.691	0.948	0.898	0.948	0.897
6	Decision Tree	13.689	20.914	3.700	4.573	2.542	3.108	0.912	0.861	0.912	0.860
5	SVM	20.542	26.094	4.532	5.108	2.685	3.326	0.868	0.827	0.868	0.825
4	KNN	12.718	26.575	3.566	5.155	2.361	3.577	0.918	0.824	0.918	0.822
0	LinearRegression	54.120	53.132	7.357	7.289	5.639	5.622	0.653	0.648	0.652	0.645
2	Ridge	54.119	53.108	7.357	7.288	5.639	5.621	0.653	0.648	0.652	0.645
1	Lasso	54.273	53.316	7.367	7.302	5.648	5.636	0.652	0.647	0.651	0.643
3	Elastic Net	54.593	53.911	7.389	7.342	5.677	5.680	0.650	0.643	0.649	0.639



The analysis points to XGBoost (XGB) or LightGBM (LGB) Regressor as suitable final models, given their low RMSE and high R2 scores on test data. Choosing LGB is preferred for its strength with large datasets and its ability to offer more informative feature interpretations.