MLops Assignment-2 Report

By- Tanishq Kankerwal (B21BB032)

Bike Sharing Dataset: Predicting Bike Rentals

Introduction

The objective of this report is to extend and enhance the MLOps pipeline for predicting target variables (in this case number of bike rentals). This includes feature engineering, preprocessing techniques, model selection, and pipeline automation.

Data Preprocessing

1. Handling Missing Values

To ensure the dataset is clean and ready for modeling, we first checked for missing (NaN) values in both the features and target variable.

2. Imputation Strategy

For missing values in numerical features, we used the SimpleImputer with the mean strategy, which replaces missing values with the mean of the respective feature. For categorical features, we used the most_frequent strategy, which fills in missing values with the most common value in each category.

Feature Engineering

1. Interaction Features

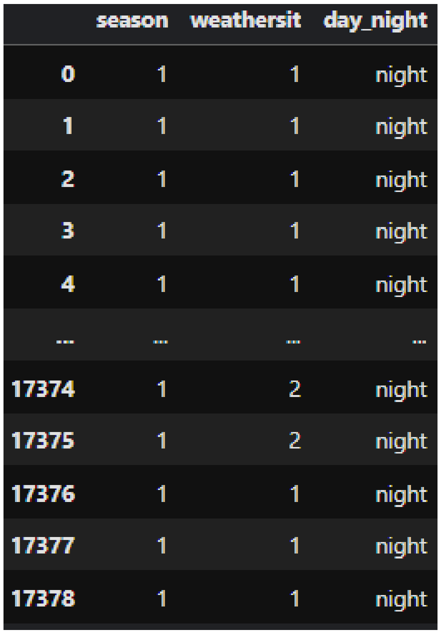
To capture potential interactions between numerical variables, we created two new features:

- **`temp_hum_interaction`**: Interaction between temperature (`temp`) and humidity (`hum`).
- **`windspeed_temp_interaction`**: Interaction between windspeed (`windspeed`) and temperature (`temp`).
- **`hum_windspeed_interaction:`** Interaction between windspeed (`windspeed`) and humidity(hum).

These features were chosen because temperature and humidity together could influence the comfort level, which may affect bike rentals. Similarly, windspeed combined with temperature could influence outdoor conditions for biking.

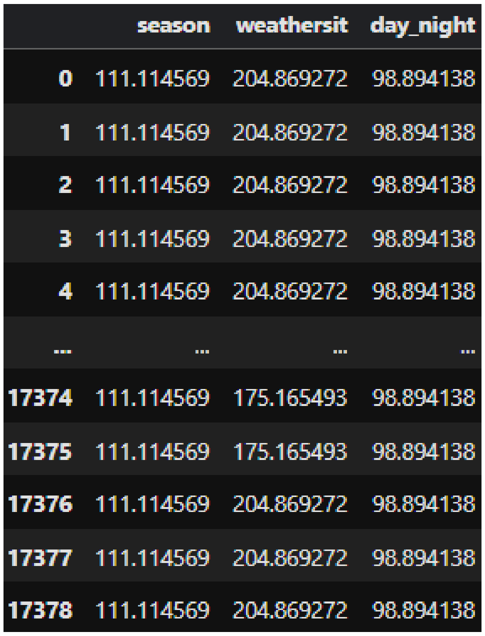
2. Use of Target Encoder for encoding

TargetEncoder() works by taking the mean of all the target values after encoding them.



	season	weathersit	day_night
0	1	1	night
1	1	1	night
2	1	1	night
3	1	1	night
4	1	1	night
...	...	...	...
17374	1	2	night
17375	1	2	night
17376	1	1	night
17377	1	1	night
17378	1	1	night

----->

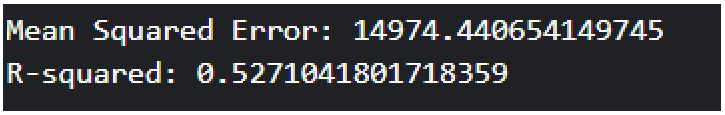


	season	weathersit	day_night
0	111.114569	204.869272	98.894138
1	111.114569	204.869272	98.894138
2	111.114569	204.869272	98.894138
3	111.114569	204.869272	98.894138
4	111.114569	204.869272	98.894138
...	...	...	...
17374	111.114569	175.165493	98.894138
17375	111.114569	175.165493	98.894138
17376	111.114569	204.869272	98.894138
17377	111.114569	204.869272	98.894138
17378	111.114569	204.869272	98.894138

Comparing with one-hot encoding ->

Not much difference found. Almost identical results.

In both cases

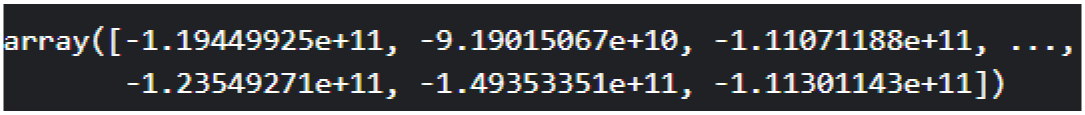


```
Mean Squared Error: 14974.440654149745
R-squared: 0.5271041801718359
```

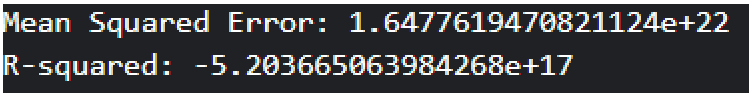
3. Training with Linear Regression

With scratch using for loop:

Predictions:



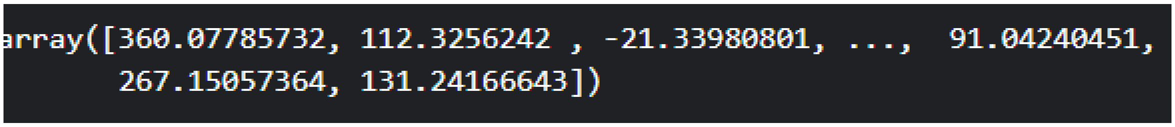
```
array([-1.19449925e+11, -9.19015067e+10, -1.11071188e+11, ...,
       -1.23549271e+11, -1.49353351e+11, -1.11301143e+11])
```



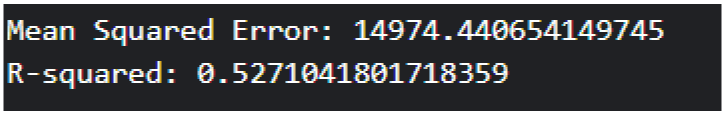
```
Mean Squared Error: 1.6477619470821124e+22
R-squared: -5.203665063984268e+17
```

Using in-built:

Predictions:

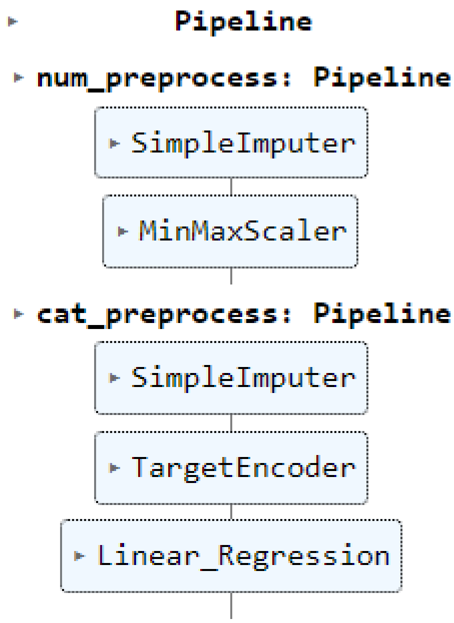


```
array([360.07785732, 112.3256242 , -21.33980801, ...,  91.04240451,
       267.15057364, 131.24166643])
```



```
Mean Squared Error: 14974.440654149745
R-squared: 0.5271041801718359
```

MLops Pipeline:


▸ Pipeline
▸ num_preprocess: Pipeline
▸ SimpleImputer
▸ MinMaxScaler
▸ cat_preprocess: Pipeline
▸ SimpleImputer
▸ TargetEncoder
▸ Linear_Regression