Report

Enhancing and Optimizing an MLOps Pipeline

Report on MLOps Week 2 Assignment:

Abstract:

This Report has been prepared to have a record of experiments and activities done using Machine learning Algorithms like “Random forest ”,”Linear Regression”(Built from scratch and Inbuilt) over bike sharing data set.This Experiment gives us an understanding of how the Quality of the data, representation, pre-processing ,feature engineering affects the performance of Machine learning Algorithms.Hands on experience on creating ML activities lifecycle pipeline .Also gives an exposure towards OpenSource Library “MLFlow”.How this tool helps in keeping track of different experiments, performance and results of ML algorithms with variation in dataset,Hyper - Parameters.

Table of Contents:

About Data:

Data initially had -17 columns

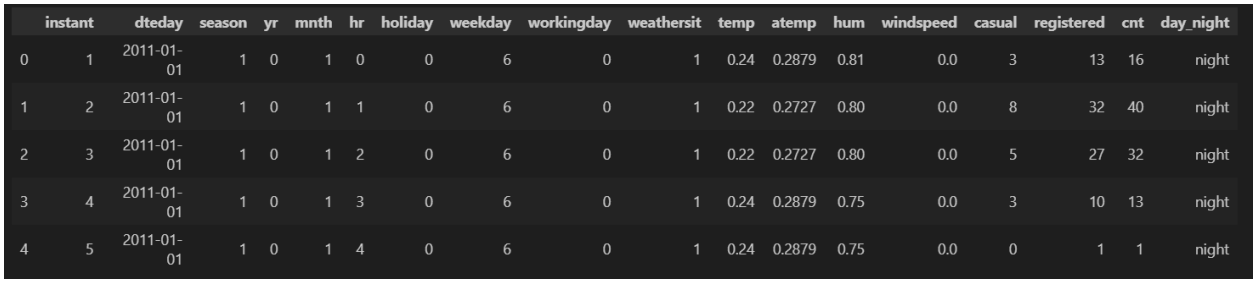
```
'instant', 'dteday', 'season', 'yr', 'mnth', 'hr', 'holiday', 'weekday',
       'workingday', 'weathersit', 'temp', 'atemp', 'hum', 'windspeed',
       'casual', 'registered', 'cnt', 'day_night'
```

Of which our endeavor was to predict ‘cnt’ using other feature attributes.
These data did not consist of any Missing values.

```
instant       0
dteday        0
season        0
```

```
yr            0
mnth          0
hr            0
holiday       0
weekday       0
workingday    0
weathersit    0
temp          0
atemp         0
hum           0
windspeed     0
casual        0
registered    0
cnt           0
```

The following image represents the first few instances of the data set.

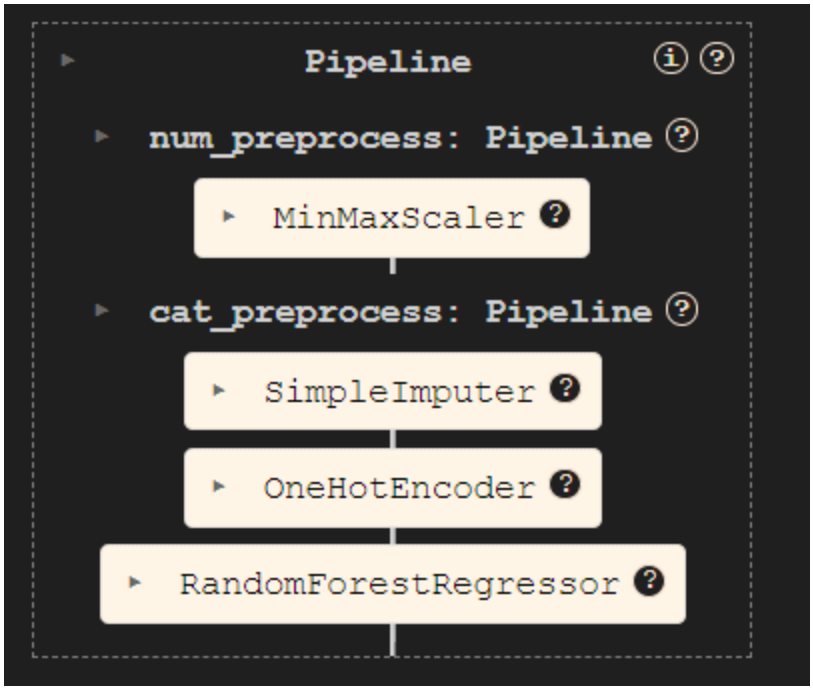


	instant	dteday	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt	day_night
0	1	2011-01-01	1	0	1	0	0	6	0	1	0.24	0.2879	0.81	0.0	3	13	16	night
1	2	2011-01-01	1	0	1	1	0	6	0	1	0.22	0.2727	0.80	0.0	8	32	40	night
2	3	2011-01-01	1	0	1	2	0	6	0	1	0.22	0.2727	0.80	0.0	5	27	32	night
3	4	2011-01-01	1	0	1	3	0	6	0	1	0.24	0.2879	0.75	0.0	3	10	13	night
4	5	2011-01-01	1	0	1	4	0	6	0	1	0.24	0.2879	0.75	0.0	0	1	1	night

On observation the feature ‘cnt’ was unaffected by ‘dteday’.

Initial Lab Work:

On the above given data-set we followed the pipeline:



We did Data Engineering like dropped the unnecessary, handling missing values.Then created **Pipelines** for performing preprocessing on- Features like Numerical preprocessing i.e. Normalization on few features using “Min-Max Scaler” which brings the data in range(0 to 1). And then also performed Categorical preprocessing using OneHotEncoder.

One Hot encoded features:(With one hot encoder):
These are features generated by One Hot Encoder
['season_2', 'season_3', 'season_4', 'weathersit_2', 'weathersit_3',
'weathersit_4', 'day_night_night']

Then data was passed to Random Forest Regressor, a Machine learning algorithm using utilities present in Scikit Learn Library.

Random Forest Regressor:
Using this Algorithm on the processed data, The performance was:

Metrics (2)

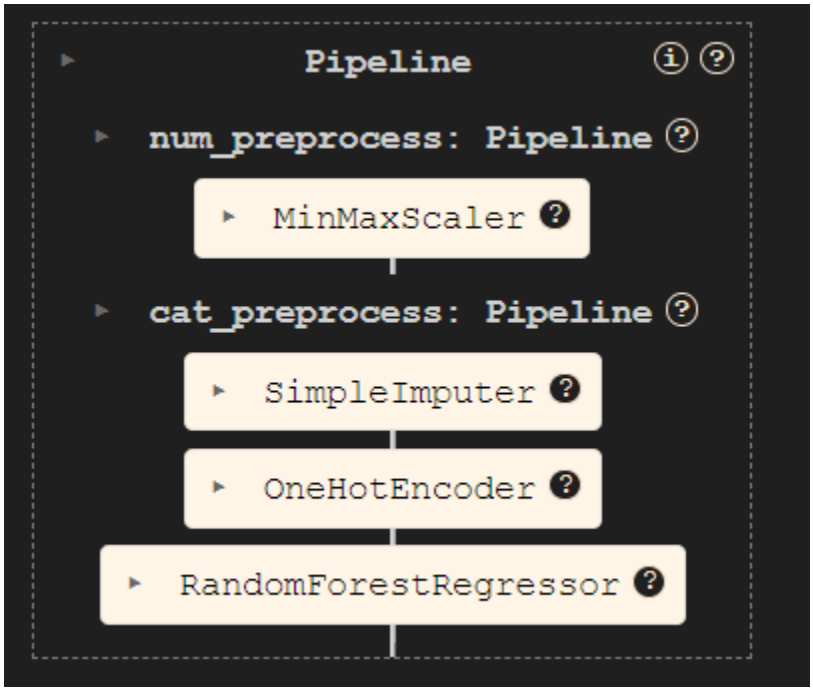
Search metrics

Metric	Value
mse	1808.4074990292243
r2	0.9428901308176855

RandomForestRegressor

RandomForestRegressor(random_state=42)

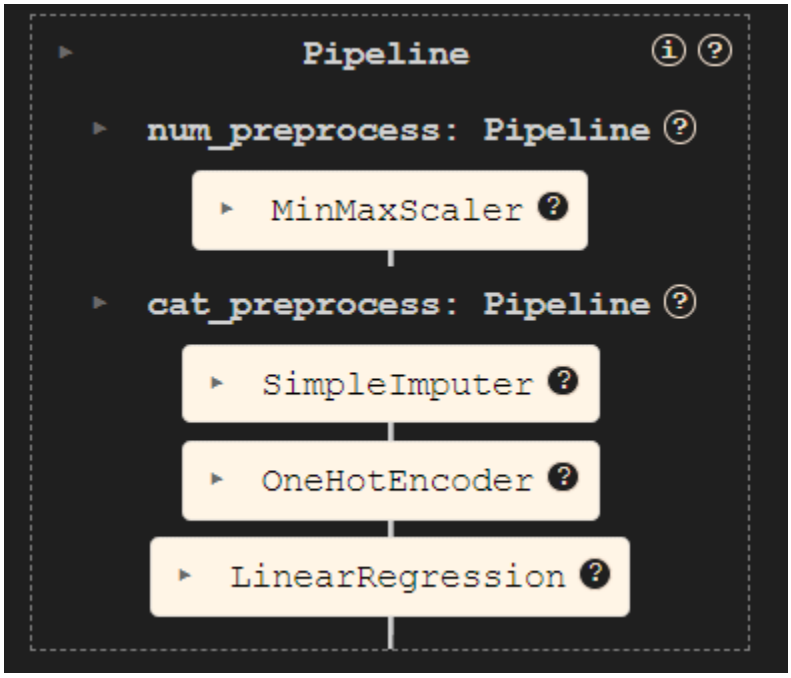
Mse(Mean Squared Error) :1808
R2: 0.943



Post Lab Work:(Assignment)-

- On the Same processed data the Multivariate Linear Regression was used to predict the feature ”cnt”.

Linear Regression implemented inbuilt(with One Hot encoder)



Metrics (2)

Search metrics

Metric	Value
mse	14896.15062084329
r2	0.5295765950245783

This was the performance metrics of the Multivariate Linear Regression using Scikit Learn .

MSE(Mean Squared Error):14896
R2=0.529

Linear Regression implemented from Scratch (With One Hot Encoder)

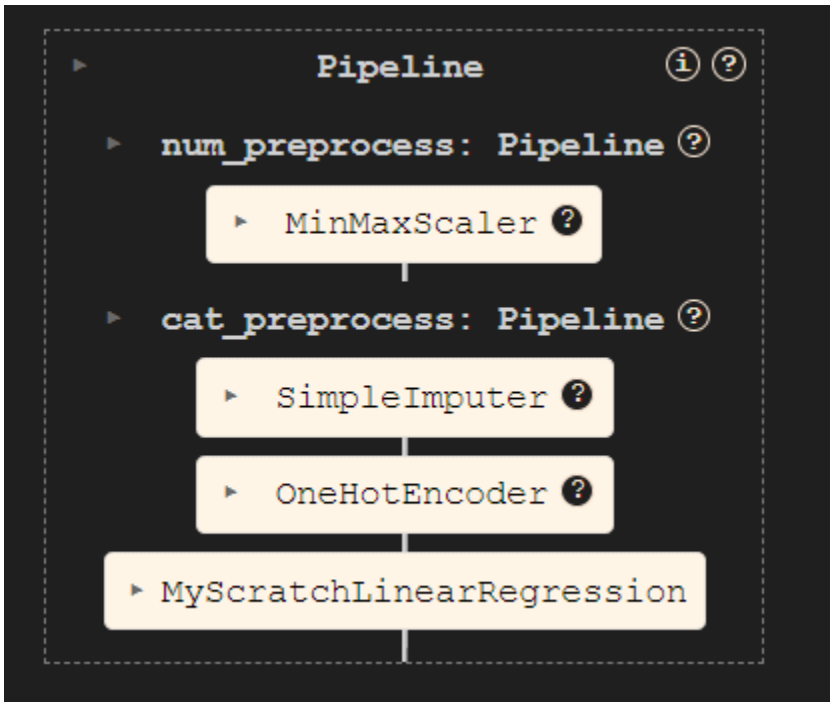
- On the same processed data the Linear Regression Algorithm was applied.Which was built by me from scratch. I did the training process using the following parameters to minimize the Objective function,

Epochs:10000
Learning rate: 0.0001

Metrics (2)

Search metrics

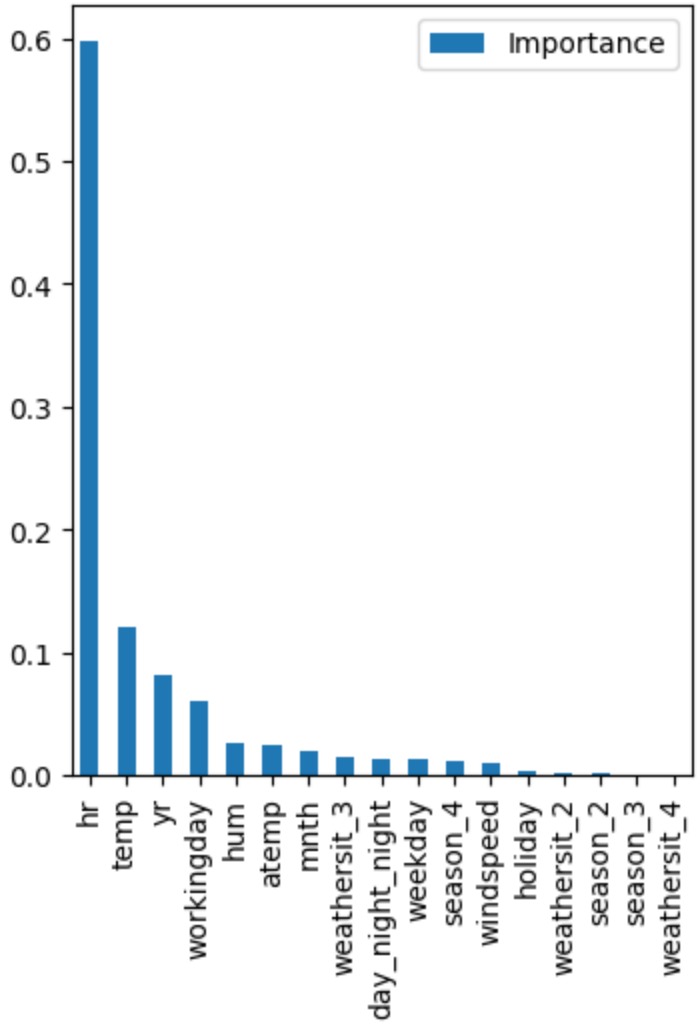
Metric	Value
mse	15207.771554108473
r2	0.5197355438550831



Epoch wise Mean Square Error while Training:

```
Epoch 0, MSE = 69521.54434294757
Epoch 1000, MSE = 26546.762281553543
Epoch 2000, MSE = 25637.97261805185
Epoch 3000, MSE = 24814.209531497196
Epoch 4000, MSE = 24067.07256089985
Epoch 5000, MSE = 23389.165835472046
Epoch 6000, MSE = 22773.836544350746
Epoch 7000, MSE = 22215.083518022228
Epoch 8000, MSE = 21707.490368071423
Epoch 9000, MSE = 21246.167489341522
Epoch 10000, MSE = 20826.70016833644
Epoch 11000, MSE = 20445.101914313305
Epoch 12000, MSE = 20097.772444607264
Epoch 13000, MSE = 19781.459846558086
Epoch 14000, MSE = 19493.226492667814
Epoch 15000, MSE = 19230.41833023578
Epoch 16000, MSE = 18990.637205985717
Epoch 17000, MSE = 18771.715921180046
Epoch 18000, MSE = 18571.695743953478
Epoch 19000, MSE = 18388.806133523354
Epoch 20000, MSE = 18221.446455914684
Epoch 21000, MSE = 18068.16949319964
Epoch 22000, MSE = 17927.66656828091
```

Performance metrics of **Linear Regression Algorithm Developed from Scratch**:
MSE (Mean Squared Error):15207
R2= 0.52



Using Random Forest Algorithm we could figure out that the most of the dependency of output lies with **hr, temp,yr,workingdat,hum, atemp**

To improve the data ,it would be better to use features like **'hum**' and **'atem'** (relatively less important). In combination with the **hr, temp,yr,workingdat,hum, atemp** (High Importance).

Feature Engineering Task:

(*Create at least two new interaction features between numerical variables (e.g., temp * hum). Justify your choice of features and explain how they might improve the model's predictive performance.*)

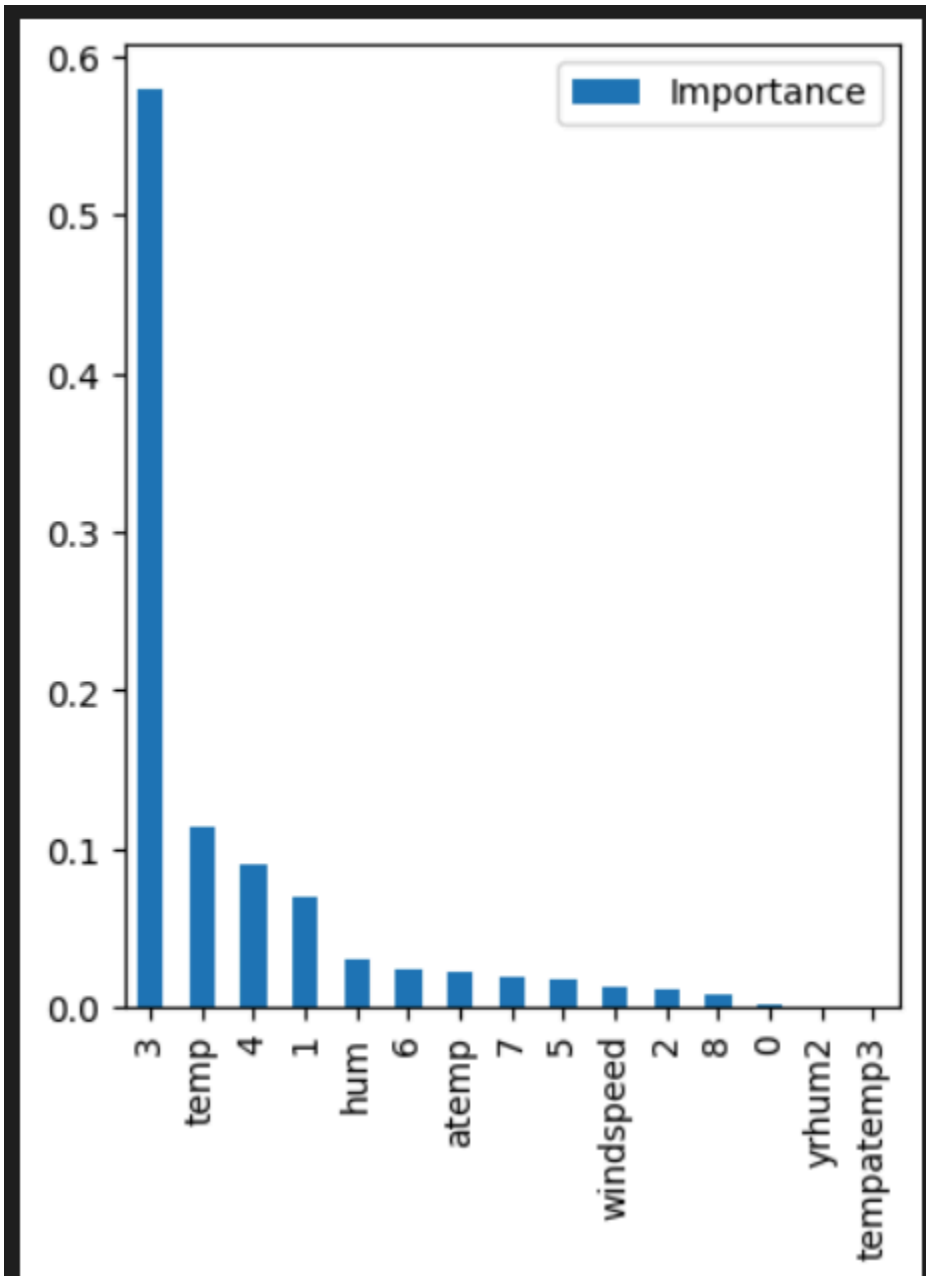
1) '**yr**'*'(**hum**')^2
2) '**temp**'*('**atemp**')^3

I choose this because as mentioned 'yr' , 'temp' were high importance feature as per the inference brought from "**Random Forest Regressor**". We could also observe that 'hum','atemp' were relatively less importance , what i infer is the output is not linear dependent on these

features , so to introduce non - linearity i raised the exponent of the feature anticipating a better importance .

Target Encoder:

(Replace the OneHotEncoder with TargetEncoder for categorical variables.
Evaluate how this change impacts the model's performance compared to one-hot encoding.)
Now we brought a change in categorical preprocessing pipeline, Earlier we used ‘One Hot Encoder’.Now on incorporating Target Encoder into categorical preprocessing pipeline.We get anonymous features -0 1 2 3 4 5 6 7 8- these were the features generated by Target Encoder.



Among the features generated by Target Encoder, The Random Forest Regressor on using later showed highest importance.

- Difference we made now as compared to initial pipeline was we did categorical preprocessing first and later the numerical preprocessing, in order to process i.e. normalize the features generated by Target Encoder in exchange with Categorical variables.

This method improved the model's performance :

Linear Regression(Inbuilt)	Mean Squared Error(MSE):	R2:
One Hot Encoder +Without Addition Features	10412.43	0.6711
Target Encoder+With Addition Features	10412.43	0.6711

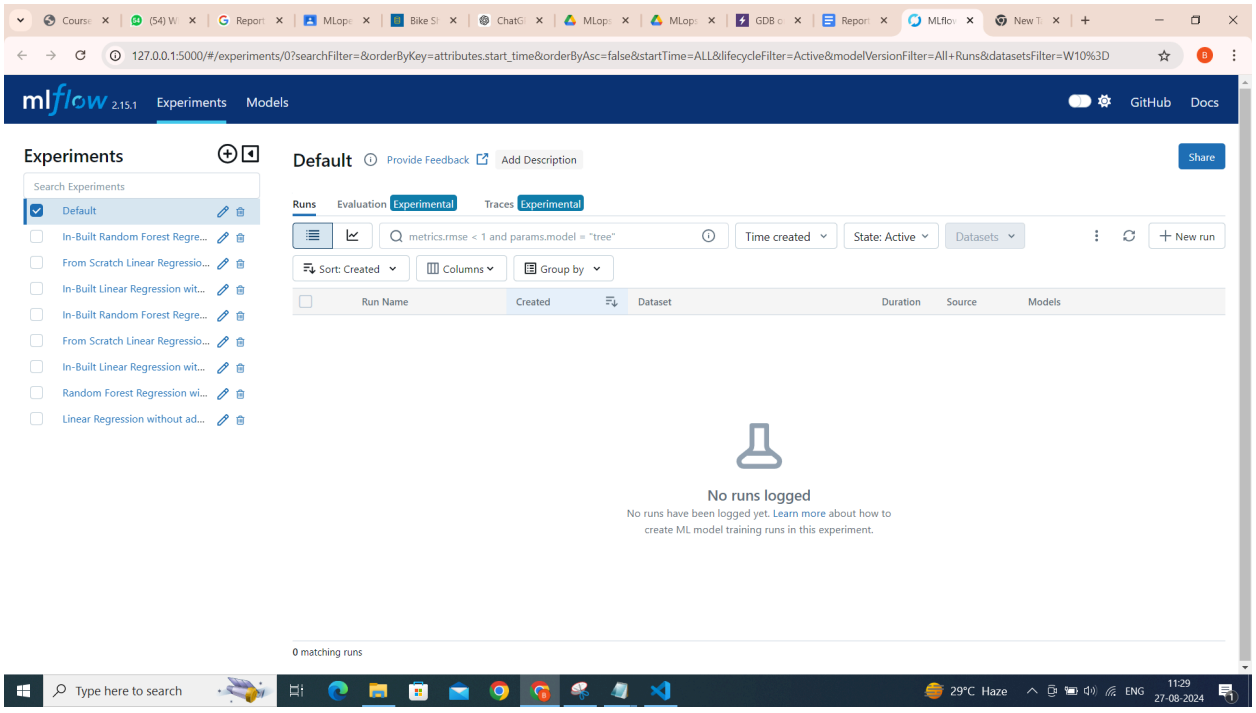
Linear Regression(Built by me from Scratch)	Mean Squared Error(MSE):	R2:
One Hot Encoder+Without Addition Features	15207.77	0.51
Target Encoder+With Addition Features	11820.58	0.62

Random Forest Regressor(Inbuilt)	Mean Squared Error(MSE):	R2:
One Hot Encoder+Without Addition Features	1808.40	0.9428
Target Encoder+With Addition Features	1874.73	0.9407

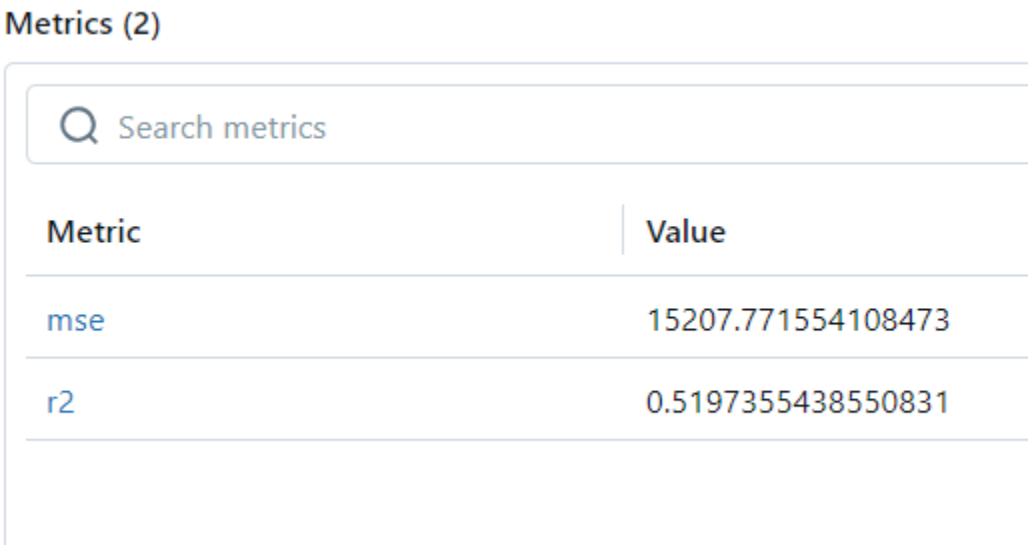
We can observe that both the approach did not bring significant difference in performance of Linear Regression(In Built) and Random Forest Regressor(in Built). Something which cannot be ignored is it improved the R2 performance metric of Linear Regression (Built from scratch) from 0.51 to 0.62.

Results and Pipeline:

From Scratch Linear Regression without additional features:



Host : Screenshot of experiment instances



From Scratch Linear Regression with additional features

Metrics (2)

Search metrics

Metric	Value
mse	11820.585428149978
r2	0.6267035566804461

In-Built Random Forest Regressor with additional features

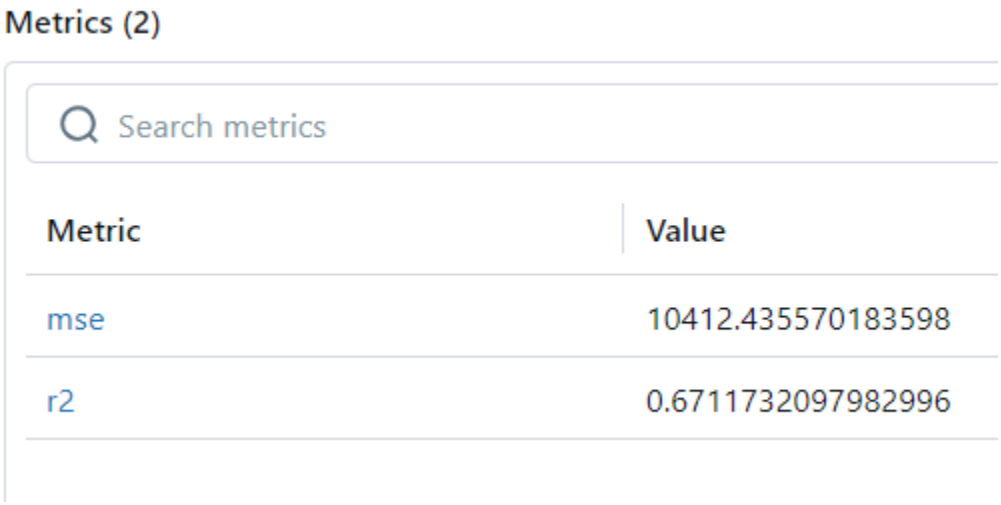
Metrics (2)

Search metrics

Metric	Value
mse	1874.7331110005055
r2	0.9407955547748686

In-Built Random Forest Regressor without additional features

In-Built Linear Regression with additional features



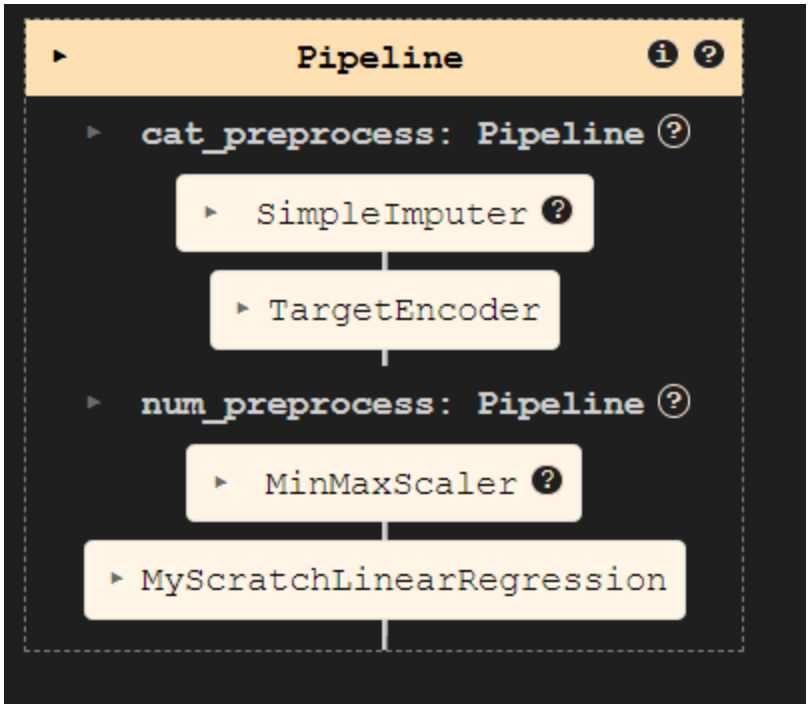
In-Built Random Forest Regressor without additional features

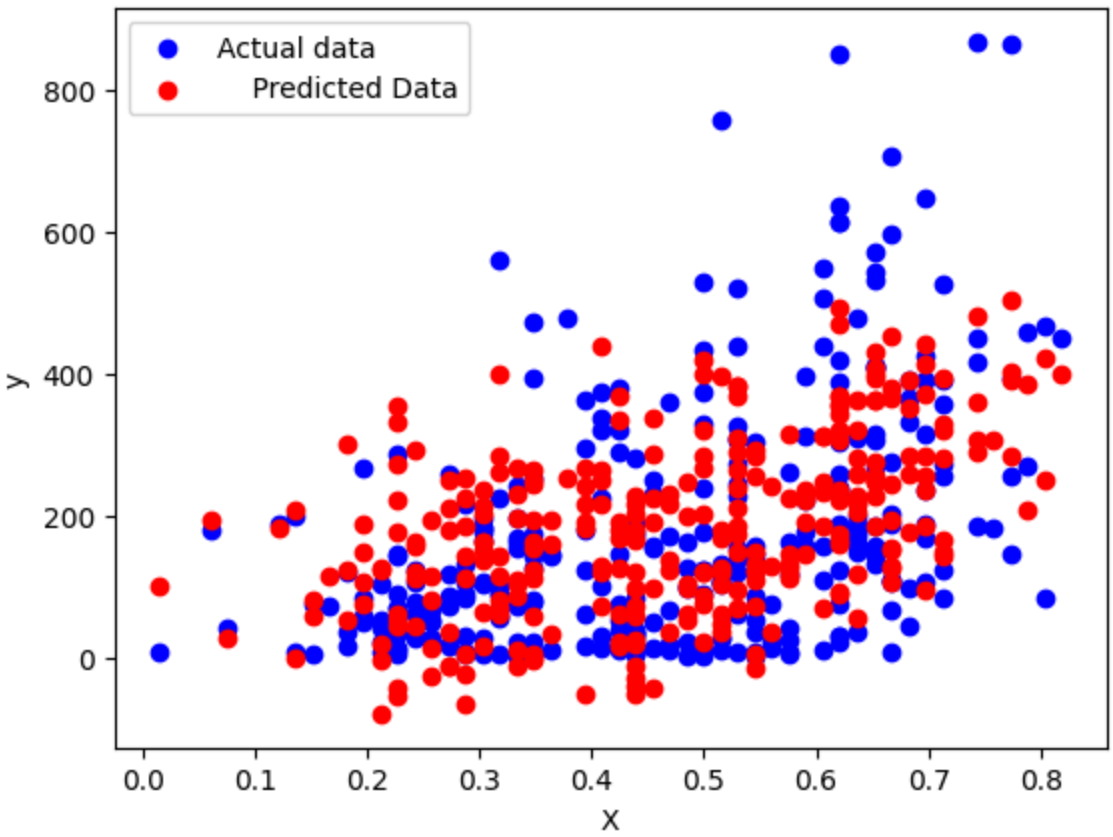
Metrics (2)

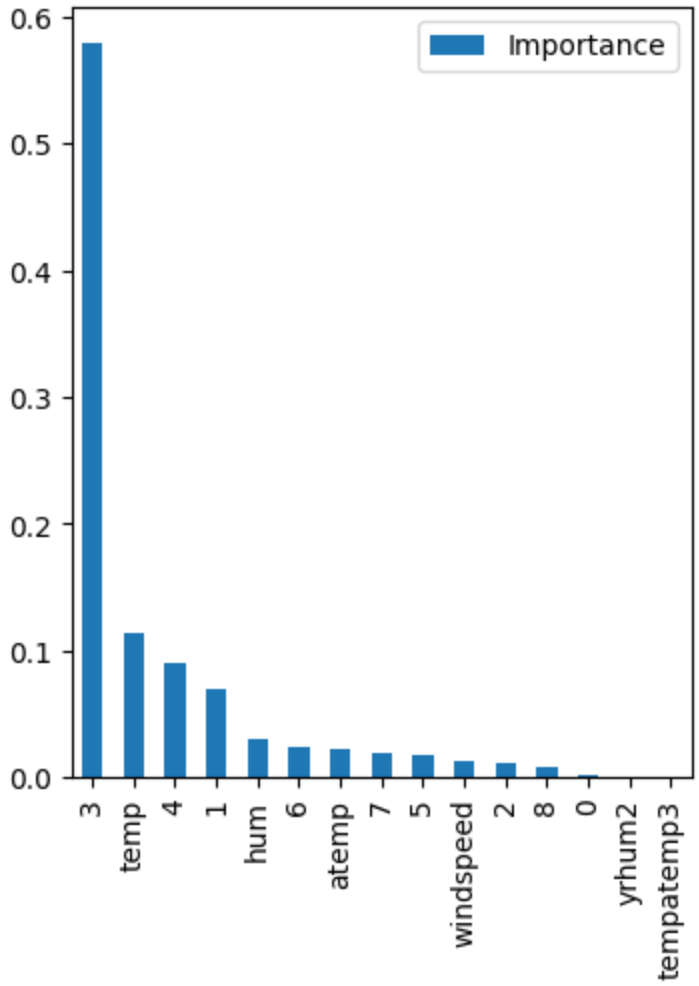
Search metrics

Metric	Value
mse	1808.4074990292243
r2	0.9428901308176855

PipeLine followed for MyScratchLinearRegression Model:


Pipeline
cat_preprocess: Pipeline
SimpleImputer
TargetEncoder
num_preprocess: Pipeline
MinMaxScaler
MyScratchLinearRegression


Actual data
Predicted Data
800
600
400
200
0
y
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
X



The newly added features “yr*hum^2”,”temp*atemp^3” has less importance showed by Random Forest Regressor.

Track of MLFlow work:

could be found at the following link

http://127.0.0.1:5000 - This link opens up to MLflow workspace