Enhancing and Optimizing an MLOps Pipeline

Colab Link: mlops_assignment2.ipynb

We had already implemented a MLOps pipeline during the lab session, this assignment is an extension on top of that.

We had been tasked with predicting the number of bike rentals (target variable: cnt) based on various features such as weather conditions, season, and time of day. The problem is a regression task where the goal is to predict a continuous variable.

Dataset Description:

Bike-sharing rental process is highly correlated to the environmental and seasonal settings. For instance, weather conditions,precipitation, day of week, season, hour of the day, etc. can affect the rental behaviors.

- instant: record index
    - dteday : date
    - season : s   eason (1:springer, 2:summer, 3:fall, 4:winter)
    - yr : year (0: 2011, 1:2012)
    - mnth : month (1 to 12)
    - hr : hour (0 to 23)
    - holiday : weather day is holiday or not (extracted from http://dchr.dc.gov/page/holiday-schedule)
    - weekday : day of the week
    - workingday : if day is neither weekend nor holiday is 1, otherwise is 0.
    + weathersit :
        - 1: Clear, Few clouds, Partly cloudy, Partly cloudy
        - 2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist
        - 3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds
        - 4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog
    - temp : Normalized temperature in Celsius. The values are divided to 41 (max)
    - atemp: Normalized feeling temperature in Celsius. The values are divided to 50 (max)
    - hum: Normalized humidity. The values are divided to 100 (max)
    - windspeed: Normalized wind speed. The values are divided to 67 (max)
    - casual: count of casual users
    - registered: count of registered users
    - cnt: count of total rental bikes including both casual and registered

Feature Engineering:

We have chosen two new interaction features between numerical variables ['temp', 'hum', 'windspeed'] which are:

1. temp_windspeed(temp * windspeed)
2. temp_humidity(temp * humidity)

We are now going to explore the reasoning behind choosing these two interaction variables.

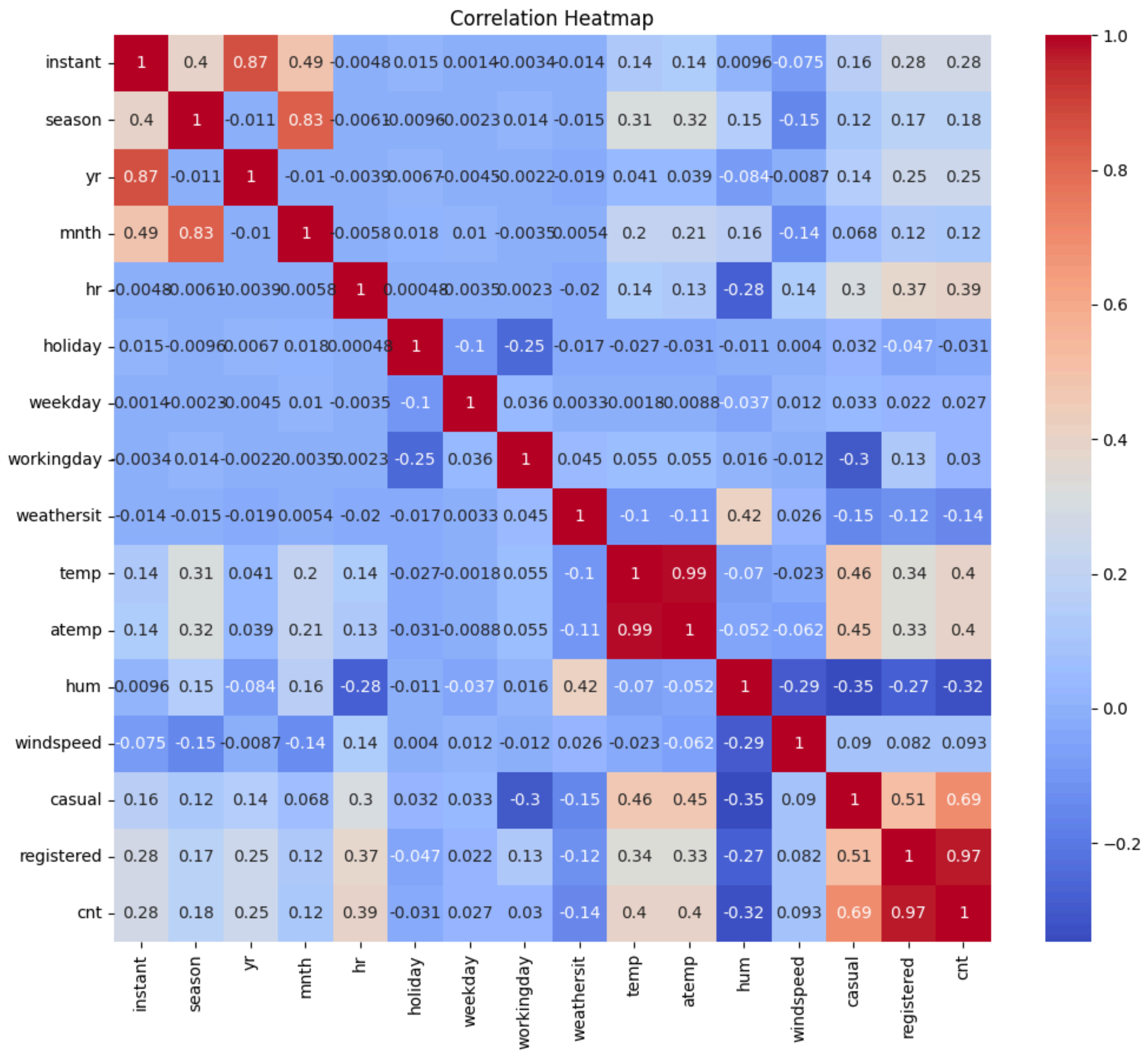


Fig 1: Correlation heatmap

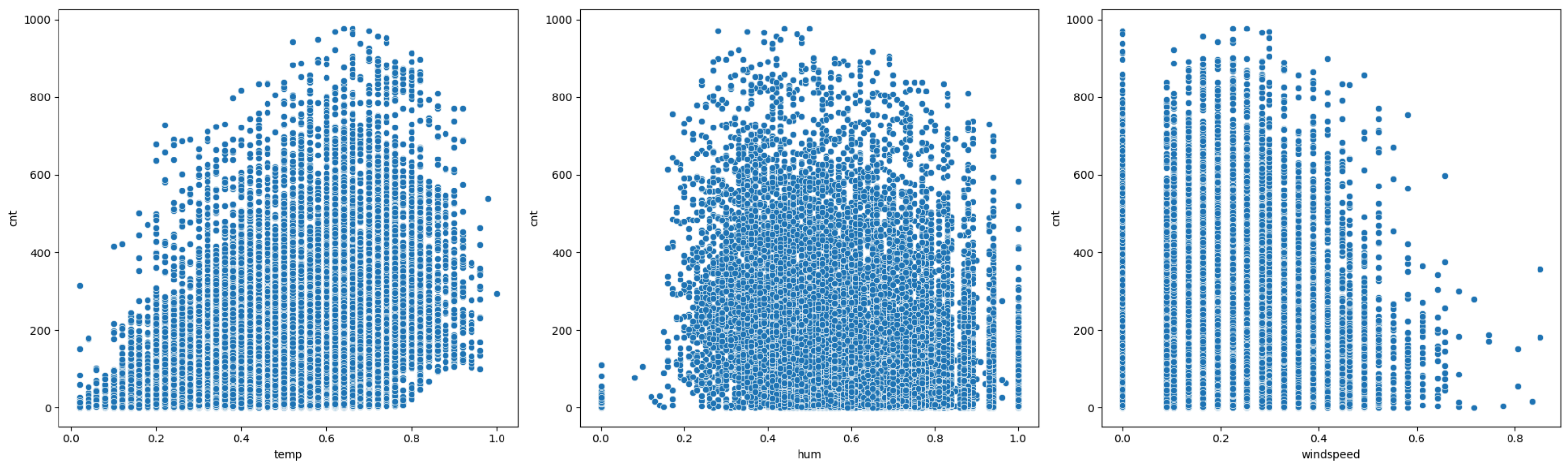


Fig 2: Scatter plots of numerical features with ‘cnt’

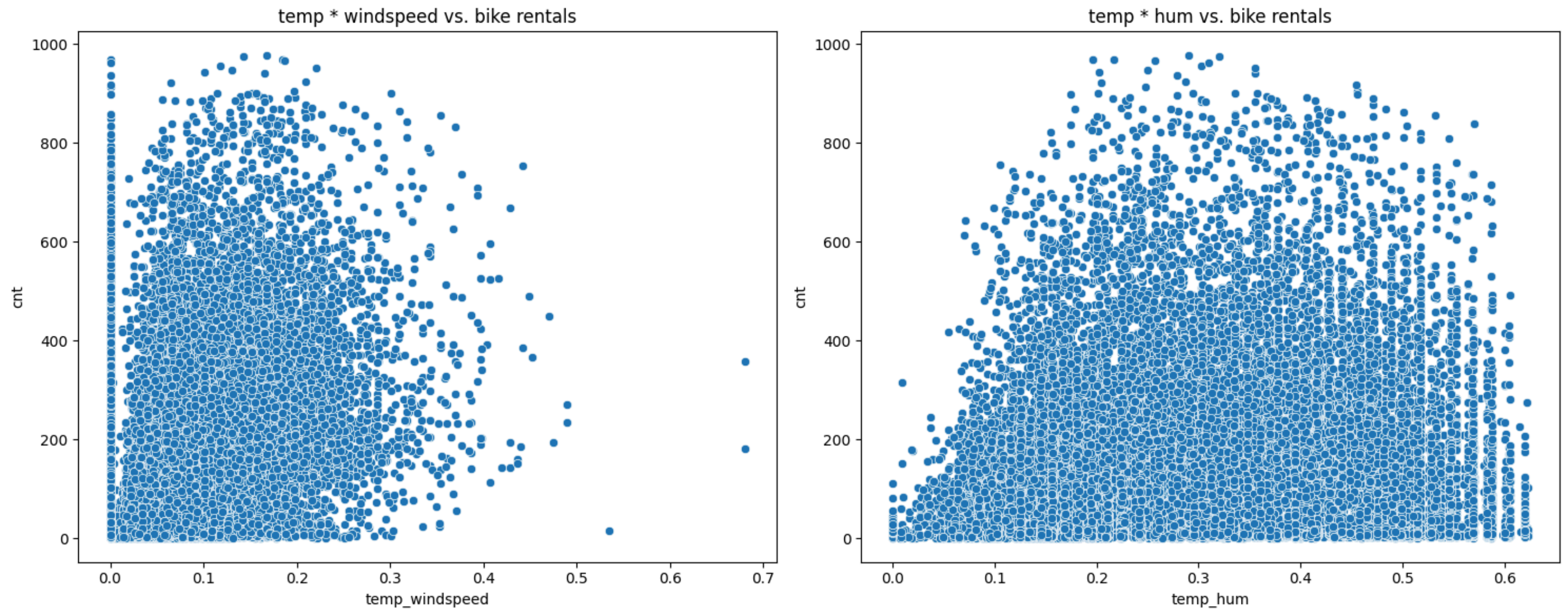


Fig 3: interaction variables vs ‘cnt’

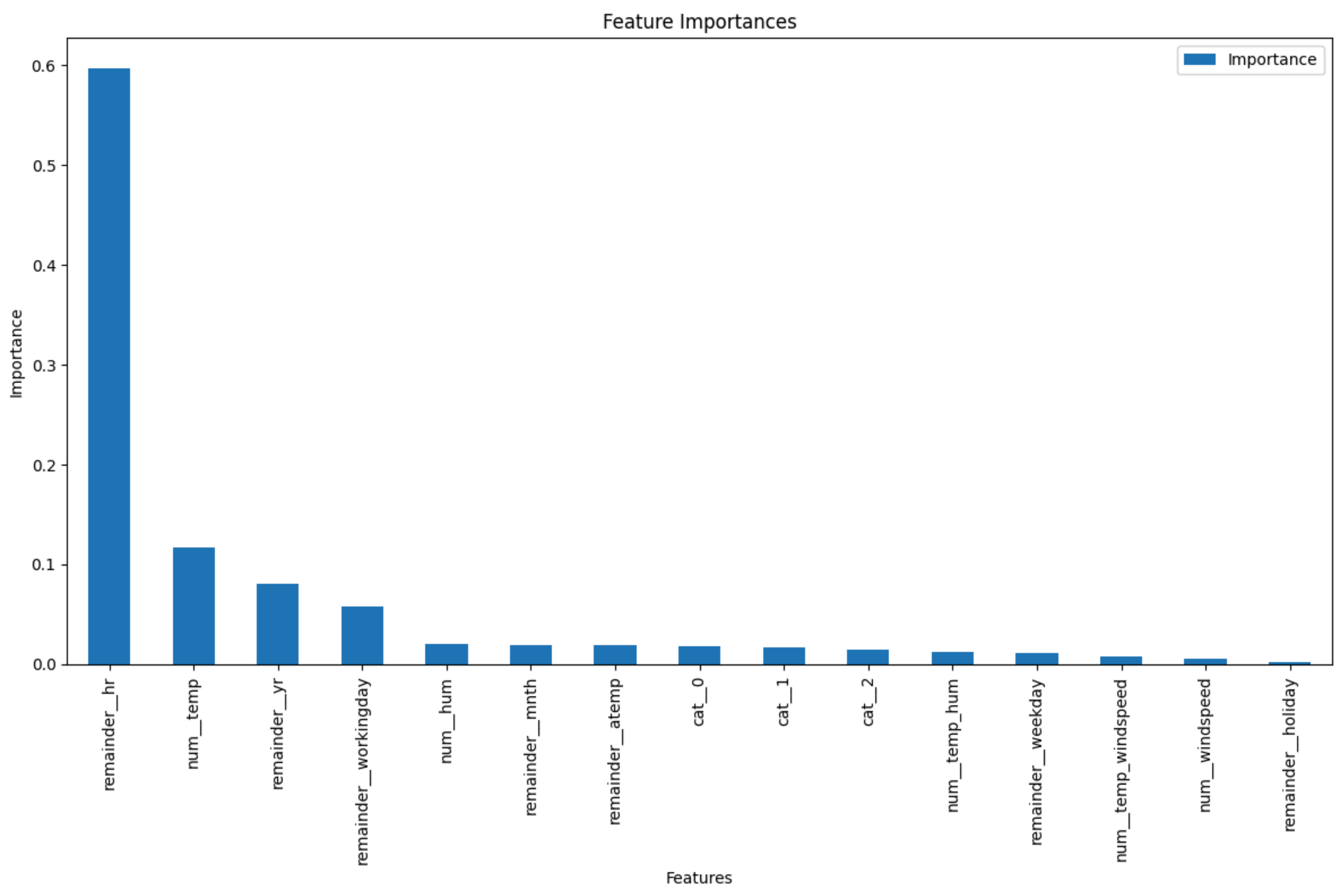


Fig 4: Feature Importance Map

1. Correlation Heatmap (Fig 1):

- Temperature ('temp') and apparent temperature ('atemp') have the highest positive correlation with bike rentals ('cnt') at 0.4.
- Humidity ('hum') has a moderate negative correlation (-0.32) with bike rentals.
- Windspeed has a weak positive correlation (0.093) with bike rentals.
- Hour ('hr') has a moderate positive correlation (0.39) with bike rentals.

2. Scatter plots (Fig 2):

- Temperature shows a clear positive relationship with bike rentals, with higher temperatures generally associated with more rentals.
- Humidity shows a slight negative relationship, with lower humidity associated with more rentals.
- Windspeed doesn't show a clear linear relationship with bike rentals.

3. Interaction features (Fig 3):

- temp * windspeed: The scatter plot shows a non-linear relationship with bike rentals. There's a concentration of higher rental counts in the middle range of this interaction term, suggesting it might capture some useful information.
- temp * hum: This plot also shows a non-linear relationship. There's a clear pattern where rental counts are highest for moderate values of this interaction term, decreasing for both low and high values.

4. Feature Importance Plot(Fig 4):

- We can see though the new interaction features are not the most important but still important then some. We can further see that the temp_hum is more important than the other.

Comparison on MSE loss and R2 before and after inclusion:

	MSE	R2
Before	1813.5	0.943
After	1834.5	0.942

Certainly the performance has not improved, but it doesn’t mean that the features are causing degradation. The features are useful and are adding value to our dataset it’s just that now the task of learning itself has become harder for model because of inclusion of more features.

Replacing OneHotEncoder with TargetEncoder:

Using TargetEncoder we can have the following advantages:

- Dimensionality Reduction: TargetEncoder can significantly reduce the number of features by encoding categorical variables into a single continuous feature. This is particularly advantageous for Linear Regression as it can prevent overfitting due to high dimensionality.
- Capturing Target Relationship: Since TargetEncoder uses the target variable for encoding, it can capture the relationship between categories and the target, potentially improving the model's performance. However, there's a risk of overfitting, which you can mitigate by using techniques like cross-validation or smoothing in the target encoding.

Comparison on MSE loss and R2 before and after inclusion:

	MSE	R2
OneHotEncoder	1834.5	0.943
TargetEncoder	1785.3	0.944

Target encoding can help Random Forest models capture subtle relationships between categories and the target variable that might be diluted or missed with one-hot encoding.

Incorporating Linear Regression:

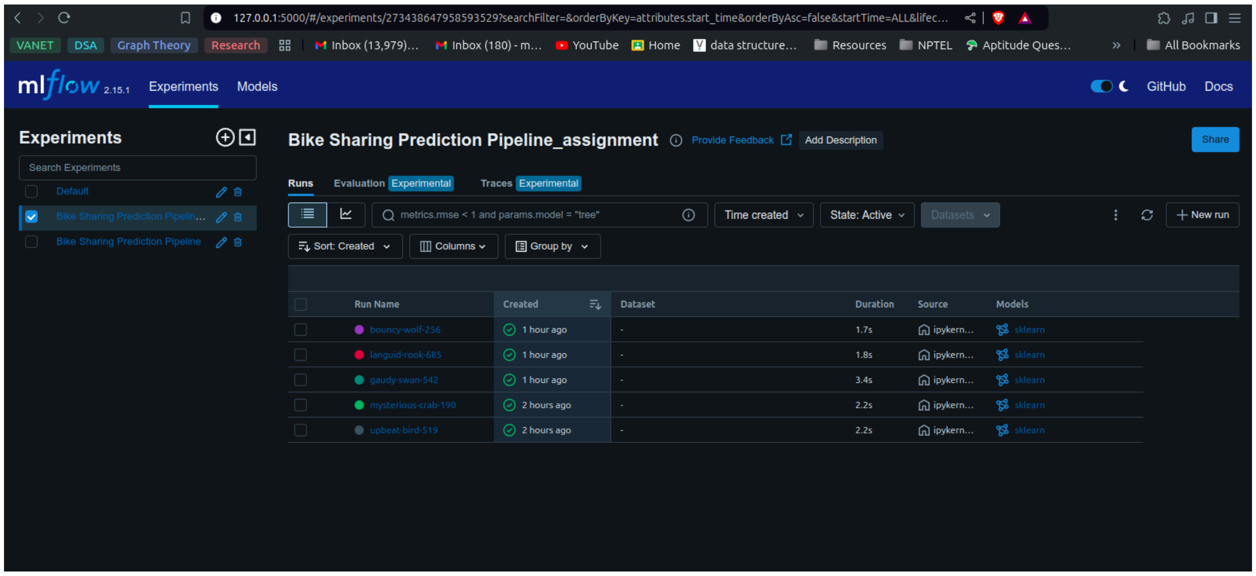
We have added linear regression to our model both library and scratch versions and the performance of them are as follows:

	MSE	R2
Library	14979.1	0.527
Scratch	14979.1	0.527

We observe that performance of both library and scratch models have been identical, as both the models do have closed form equations for linear regression.

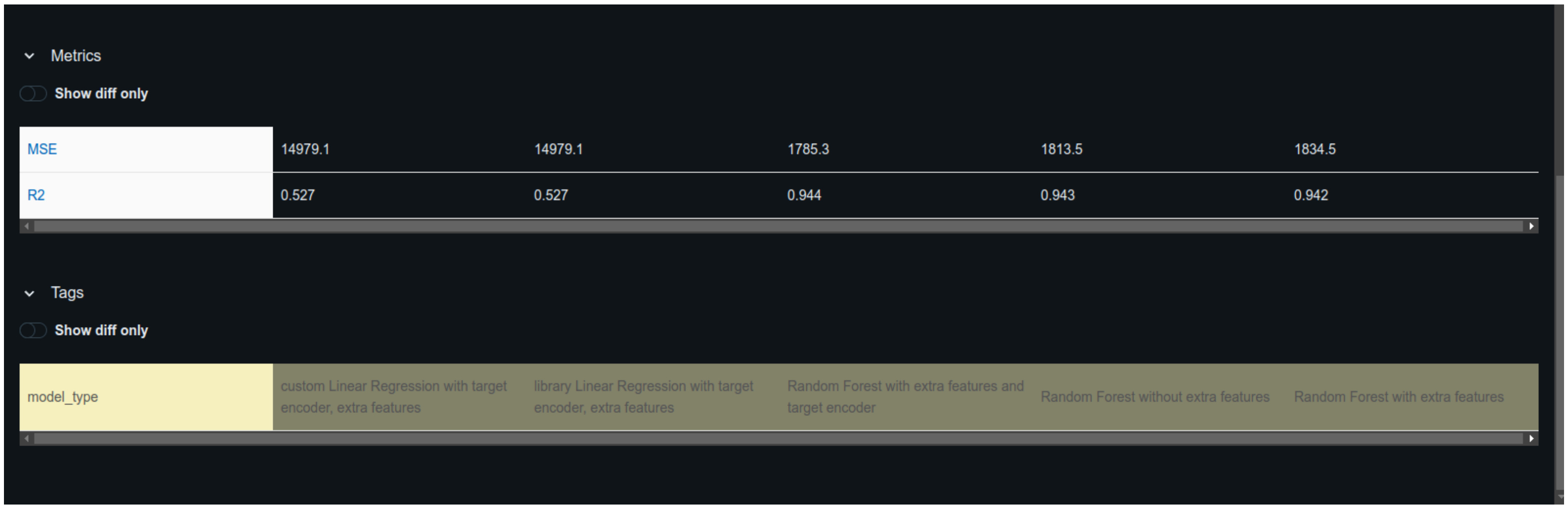
Integration with MLFlow

We have integrated MLFlow to our pipeline as well which helps us in model management, experiment tracking, version control and reproducibility


127.0.0.1:5000/#/experiments/273438647958593529?searchFilter=&orderByKey=attributes.start_time&orderByAsc=false&startTime=ALL&lifec...
mlflow 2.15.1
Experiments
Models
GitHub
Docs
Experiments
Search Experiments
Default
Bike Sharing Prediction Pipelin...
Bike Sharing Prediction Pipeline
Provide Feedback
Add Description
Share
Runs
Evaluation Experimental
Traces Experimental
metrics.rmse < 1 and params.model = "tree"
Time created
State: Active
Datasets
New run
Sort: Created
Columns
Group by

Bike Sharing Prediction Pipeline_assignment

Run Name	Created	Dataset	Duration	Source	Models
bouncy-wolf-256	1 hour ago	-	1.7s	ipykern...	sklearn
languid-rook-685	1 hour ago	-	1.8s	ipykern...	sklearn
gaudy-swan-542	1 hour ago	-	3.4s	ipykern...	sklearn
mysterious-crab-190	2 hours ago	-	2.2s	ipykern...	sklearn
upbeat-bird-519	2 hours ago	-	2.2s	ipykern...	sklearn



Metrics

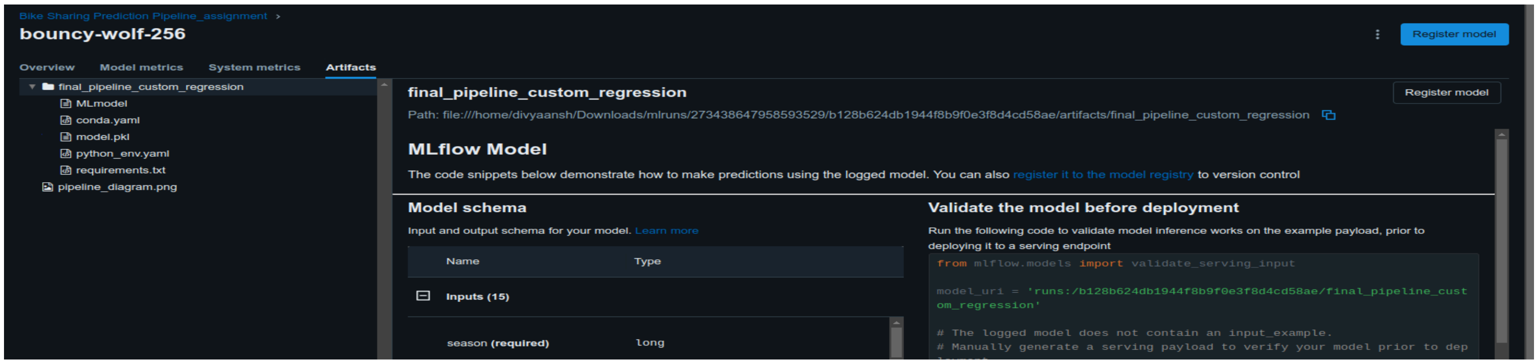
Show diff only

MSE	14979.1	14979.1	1785.3	1813.5	1834.5
R2	0.527	0.527	0.944	0.943	0.942

Tags

Show diff only

model_type	custom Linear Regression with target encoder, extra features	library Linear Regression with target encoder, extra features	Random Forest with extra features and target encoder	Random Forest without extra features	Random Forest with extra features



bouncy-wolf-256

final_pipeline_custom_regression

Path: file:///home/divyaansh/Downloads/mlruns/273438647958593529/b128b624db1944f8b9f0e3f8d4cd58ae/artifacts/final_pipeline_custom_regression

MLflow Model

The code snippets below demonstrate how to make predictions using the logged model. You can also register it to the model registry to version control

Model schema

Input and output schema for your model. Learn more

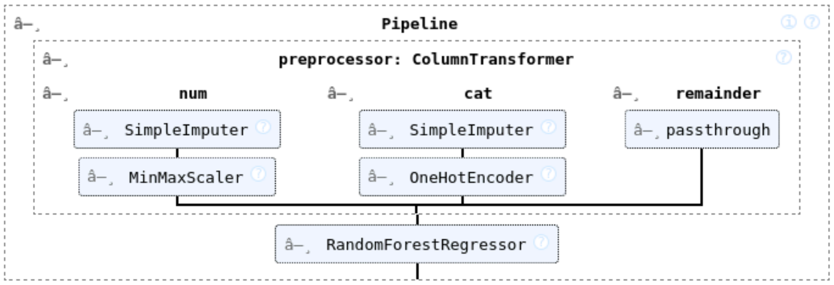
Name	Type
Inputs (15)	
season (required)	long

Validate the model before deployment

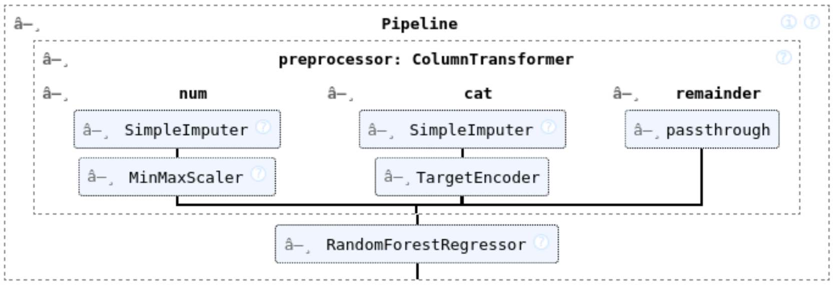
Run the following code to validate model inference works on the example payload, prior to deploying it to a serving endpoint

```
from mlflow.models import validate_serving_input

model_uri = 'runs:/b128b624db1944f8b9f0e3f8d4cd58ae/final_pipeline_cust
om_regression'

# The logged model does not contain an input_example.
# Manually generate a serving payload to verify your model prior to dep
```

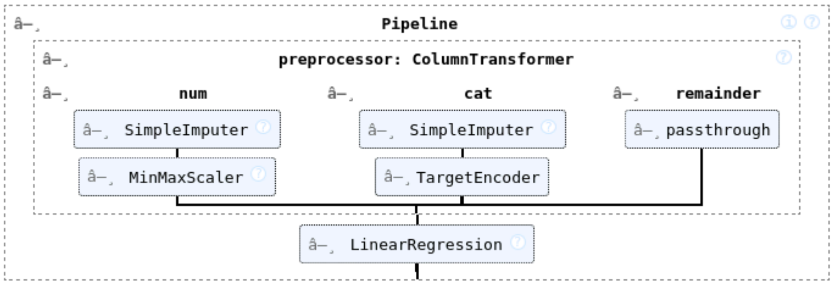
Screenshots of Pipelines:



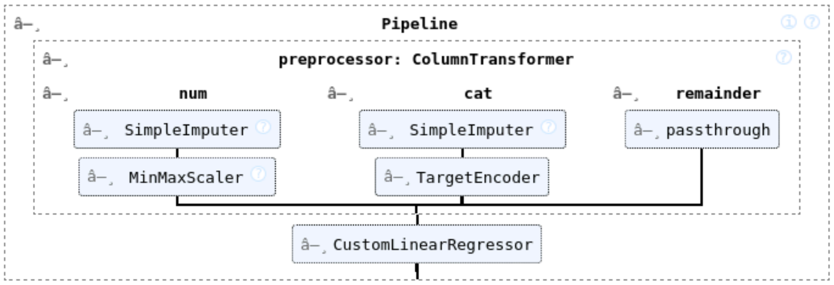
Randomforest with OneHotEncoder



Randomforest with TragetEncoder



Library Linear Regression



Custom Linear Regression