Report: Predict Bike Sharing Demand with AutoGluon Solution

Fernando Daniel Tapia Lozano

Initial Training

What did you realize when you tried to submit your predictions? What changes were needed to the output of the predictor to submit your results?

First, there were extra features inside the train and the test datasets, these have to be ignored during the prediction.

Second, the feature "date" needed to be parsed in both datasets, later, this feature was splitted in four other features.

Later, some features needed to change their data types.

What was the top ranked model that performed?

The top ranked model in the three runs was:

WeightedEnsemble_L3

Exploratory data analysis and feature creation

What did the exploratory analysis find and how did you add additional features?

During the EDA, two features needed to be changed to category types since they were considered as integers from the beginning.

Then, the date features were splitted in years, months, days and hours using the function date.dt, with this, in the next steps the prediction would improve its performance.

How much better did your model perform after adding additional features and why do you think that is?

```
Using new additional features, the metric increased from -52.885514 to
-30.044220 because there are new criteria which help the  accuracy of the
predictions.
```

Hyper parameter tuning

How much better did your model perform after trying different hyper parameters?

```
The metric of the top ranked model decreased from -30.044220 to -36.118848 but
the Kaggle score went from 0.70397 to 0.46940 showing a better performance.
```

If you were given more time with this dataset, where do you think you would spend more time?

I made few runnings of the model using the examples given in the Autogluon documentation, it helped me to understand that enhancements can be made varing some numbers of hyperparameters and the combination can be extense until finding the optimal score. This process could be benefitial, I would like to do it, however, it is time consuming as well as high in monetary cost due to the capacity and services needed for processing.

Create a table with the models you ran, the hyperparameters modified, and the kaggle score.

model	hpo1	hpo2	hpo3	score
initial	?	?	?	?
add_features	?	?	?	?
hpo	?	?	?	?

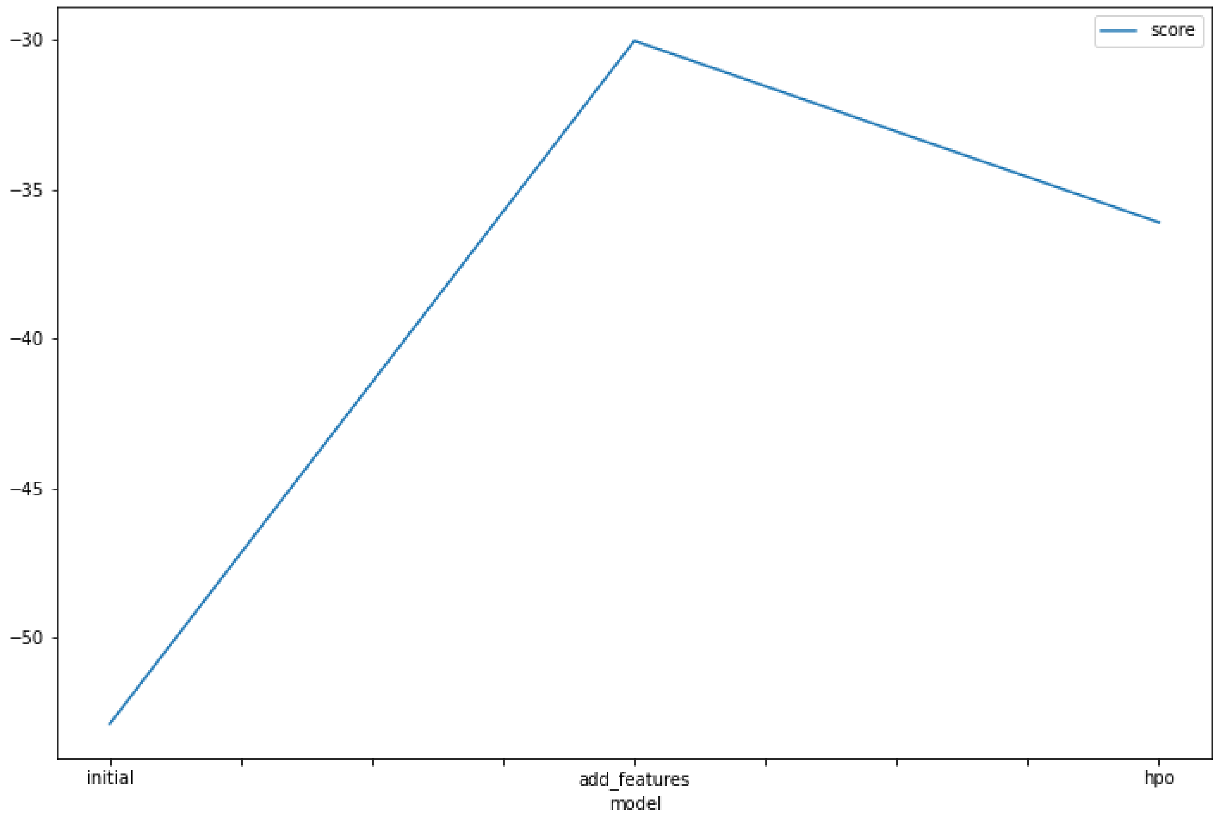
In [2]:
```
import pandas as pd
pd.DataFrame({
    "model": ["initial", "add_features", "hpo"],
    "metric_predict": ["root_mean_squared_error", "root_mean_squared_error", "roo
    "new_features": ["no", "yes", "yes"],
    "hyperparam_optimization": ["no", "no", "yes"],
    "score": [1.80129, 0.70397, 0.46940]
})
```

Out[2]:

	model	metric_predict	new_features	hyperparam_optimization	score
0	initial	root_mean_squared_error	no	no	1.80129
1	add_features	root_mean_squared_error	yes	no	0.70397
2	hpo	root_mean_squared_error	yes	yes	0.46940

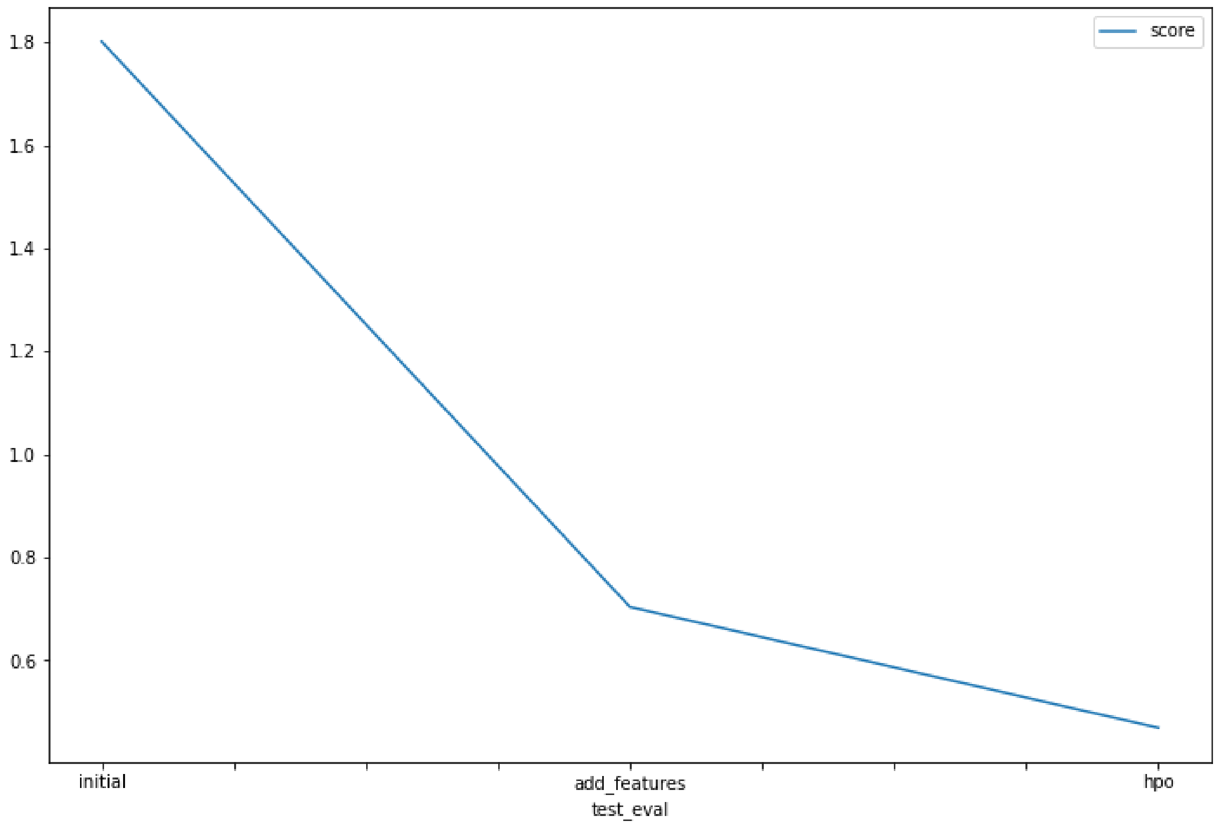
Create a line plot showing the top model score for the three (or more) training runs during the project.

In [3]:
```
fig = pd.DataFrame(
    {
        "model": ["initial", "add_features", "hpo"],
        "score": [-52.885514, -30.044220, -36.118848]
    }
).plot(x="model", y="score", figsize=(12, 8))
```



Create a line plot showing the top kaggle score for the three (or more) prediction submissions during the project.

In [4]:
```
fig = pd.DataFrame(
    {
        "test_eval": ["initial", "add_features", "hpo"],
        "score": [1.80129, 0.70397, 0.46940]
    }
).plot(x="test_eval", y="score", figsize=(12, 8))
```


score
initial
add_features
hpo
test_eval