Report: Predict Bike Sharing Demand with AutoGluon Solution

Aditi Choudhari

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?

TODO: I realized that I had to drop values that were below 0. This had to be done, since the negative values of future bikes could not be predicted. Therefore, they had to be set to 0, since in essence, predicting a negative value would translate to predicting 0 in reality

What was the top ranked model that performed?

TODO: Team Bolaka Mukherjee, with a score of 0.33756 in the overall leaderboard. In terms of the training run for my individual data, it was the Weighted_Ensemble_L3, as shown with the predictor leaderboard.

Exploratory data analysis and feature creation

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

TODO: There was a continuous distribution for the datetime, while others didn't seem to follow such a continuous trend, taking weather and season for example. Breaking up the datetime into smaller components would allow for trends that were not discovered before to be seen. So, I chose to add the day feature to the train and test data set from their individual datetime variables

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

TODO: There was improvement. The rmse has seen a decrease, going from 1.85267 to 1.77833. Including the full datetime is intuitive, but by adding the hour feature, a different trend is uncovered that allows for underlying patterns to be recognized, therefore making the model perform better as it can form stronger predictions

Hyper parameter tuning

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

TODO: There was a significant improvement – the error rate decreased by a significant amount. From the starting value of 1.77833 to 1.33006, the decrease in the error rate is apparent. One example of a hyperparameter that was changed was the n_estimators. Increasing this led to a decrease in the rmse, as more 'trees' means greater learning capacity and improved averaging.

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

TODO: Hyperparameter tuning. There was a clear improvement in the model performance with it, and so I would most definitely spend more time on it and on adjusting the hyperparameters of other such models in the Auto Gluon, for example, considering Random Forest or Linear Regression models.

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

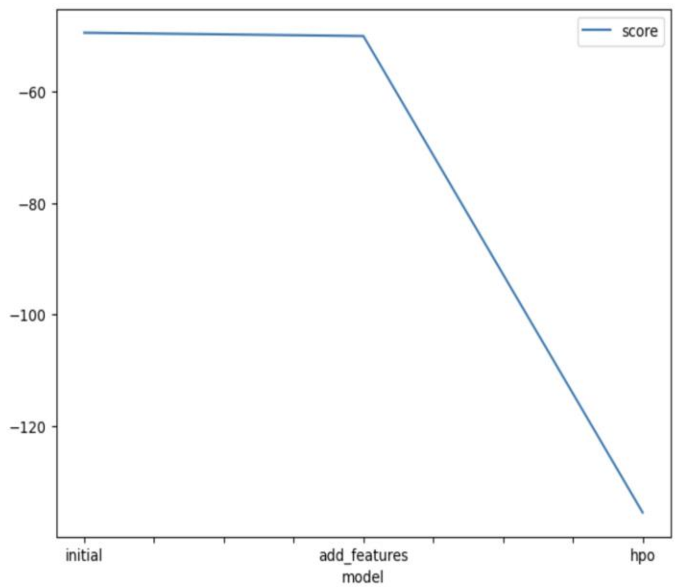
XGBoost

model	hpo1 (learning_rate)	hpo2 (max_depth)	hpo3 (n_estimators)	score

initial	0.3	6	100	1.85267
add_features	0.3	6	100	1.77833
hpo	0.1	4	200	1.33006

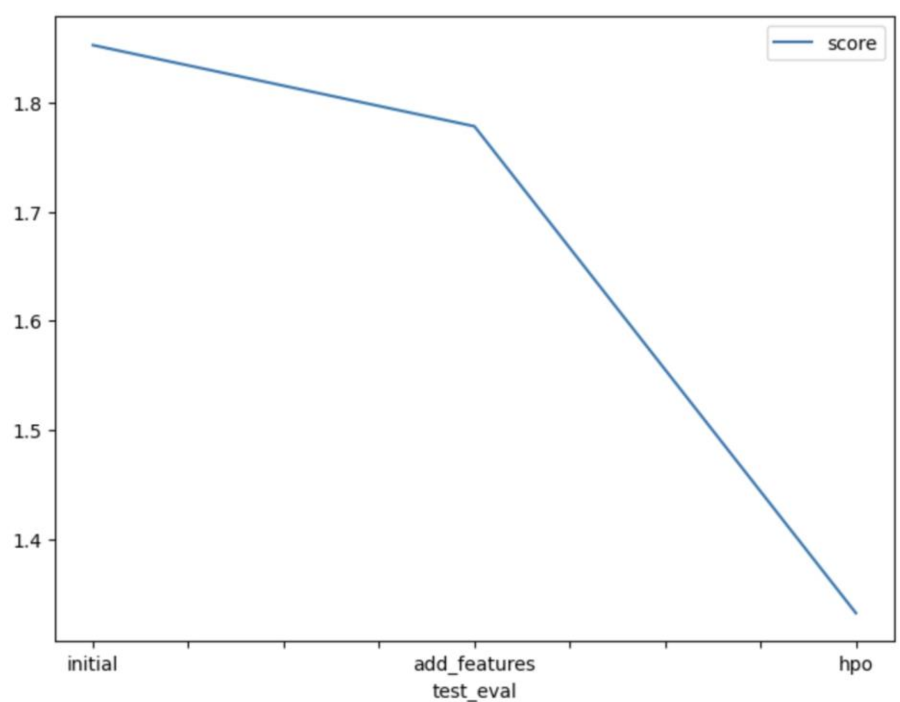
Create a line plot showing the top model score for the three (or more) training runs during the project.

TODO: Replace the image below with your own.



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

TODO: Replace the image below with your own.



Summary

TODO: To summarize the findings of the experiment, it is evident that the rmse in this demonstration went from 1.85267 in the initial state, 1.77833 to when the added feature (in this case ‘hours’) was integrated into the model, and 1.33006 when hyperparameters were tuned. In essence, both the training instances and test instances demonstrate that both feature engineering and hyperparameter tuning are vital in improving the performance of models.