# Report: Predict Bike Sharing Demand with AutoGluon Solution

#### MARK ABRENIO

## 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: When I first tried to submit my predictions I forgot to remove the negative values. And Therefore, I kept getting
errors during submission. I realized I had to use the df.clip(lower=0) metho in order to clip negative values to ZERO.

### What was the top ranked model that performed?

TODO: The top ranked model I trained was with the following hyperparameters. It scored 0.50321 on the KAGGLE RMSE score calculated from the test data.

```
gbm_options = {  # specifies non-default hyperparameter values for lightGBM gradient boosted
trees
    'num_boost_round': 100,  # number of boosting rounds (controls training time of GBM
models)
    #'num_leaves': space.Int(lower=26, upper=66, default=36),  # number of leaves in trees
(integer hyperparameter)
}

WeightedEnsemble_L3_options = {
    'ensemble_size': 70

}

nn_options = {
    'hidden_size' : 256,
    'num_layers' : 10
}

hyperparameters = {  # hyperparameters of each model type
                   'GBM': gbm_options,
                     'ENS_WEIGHTED': WeightedEnsemble_L3_options,
                   #'NN_TORCH': nn_options,  # NOTE: comment this line out if you get errors on Mac
OSX
                  }  # When these keys are missing from hyperparameters dict, no models of that type
are trained
```

## Exploratory data analysis and feature creation

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

TODO: From the histogram it is apparent that the weather and season are actually categorical features. It was also pointed out in the notebook. I did turn those into additional features per the notebook and once I did that the KAGGLE RMSE score improved from 1.83644 to 0.62941. Now looking at the histogram as I am filing out this document it also seems that the workingday, hiliday, and humidity features should also probably be turned into categorical data before training. I am running out of time for this project so I am not going to convert those and retrain but I think the model performance would likely improved if those features were turned into categorical data.

### How much better did your model preform after adding additional features and why do you think that is?
TODO: The RMSE error from kaggle imroved from 1.83644 to 0.62941 by encoding the weather and feason features from integer to categorical data. I think this is because it is much easier for the stastical model to interpret the meaning of their values as categorical than it is with something like an integer that is infinital consinuous. For example its easier for your brain to differentiate the colors red and blue than it is to differentiate the meaning between the integers 50 and 100, because its not clear there is a pattern in those integer values as they are continuous in nature.

## Hyper parameter tuning
### How much better did your model preform after trying different hyper parameters?
TODO: The best my model did from tuning different hyperparameters was 0.50321. For this I tuned the gmb_options to num_boost_round:100 and the weightedensemble_l3_options to ensemble_size:70. I tried to train the model with a very heavy nueral network and it still did not do better than the WeightedEnsemble_L3. I guess nueral networks dont always outperform other model types.

### If you were given more time with this dataset, where do you think you would spend more time?
TODO: I would convert the workingday, holiday, and humidity features to categorical. I would also experiment more with the ensemble size and the number of boosting rounds. I would probably try to reduce the GMB learning_rate too.

model	hpo1	hpo2	hpo3	score

ini	`gbm_options = {  # specifies non-default hyperparameter values for`	`WeightedEnsemble_L3_options = {`
		`'ensemble_size': 70`
			`hyperparameter_tune_kwargs = {  # HPO is not performed unless hyperparamete`	**RMSE: 23.85277**
				KAGGLE_SCORE: 0,50321

	lightGBM gradient boosted trees
	'num_boost_round': 100, #
	}
		}	r_tune_kwargs is specified
			'num_trials': num_trials,
			'scheduler' : 'local',
			'searcher': search_strategy,
			}
			num_trials = 10
			search_strategy = 'auto'
			Time_limit=600	

Model 2	gbm_options = { #
	'num_boost_round': 500, #
	}
		WeightedEnsemble_L3_options = {
		'ensemble_size': 70
		}
			'GBM': gbm_options,
			'ENS_WEIGHTED': WeightedEnsemble_L3_options,
			num_trials = 10
			search_strategy = 'auto'
			Time_limit
			600
				WeightedEnsemble_L2 -33.342861 root_mean_squared_error
				KAGGLE_SCORE: 0.55004

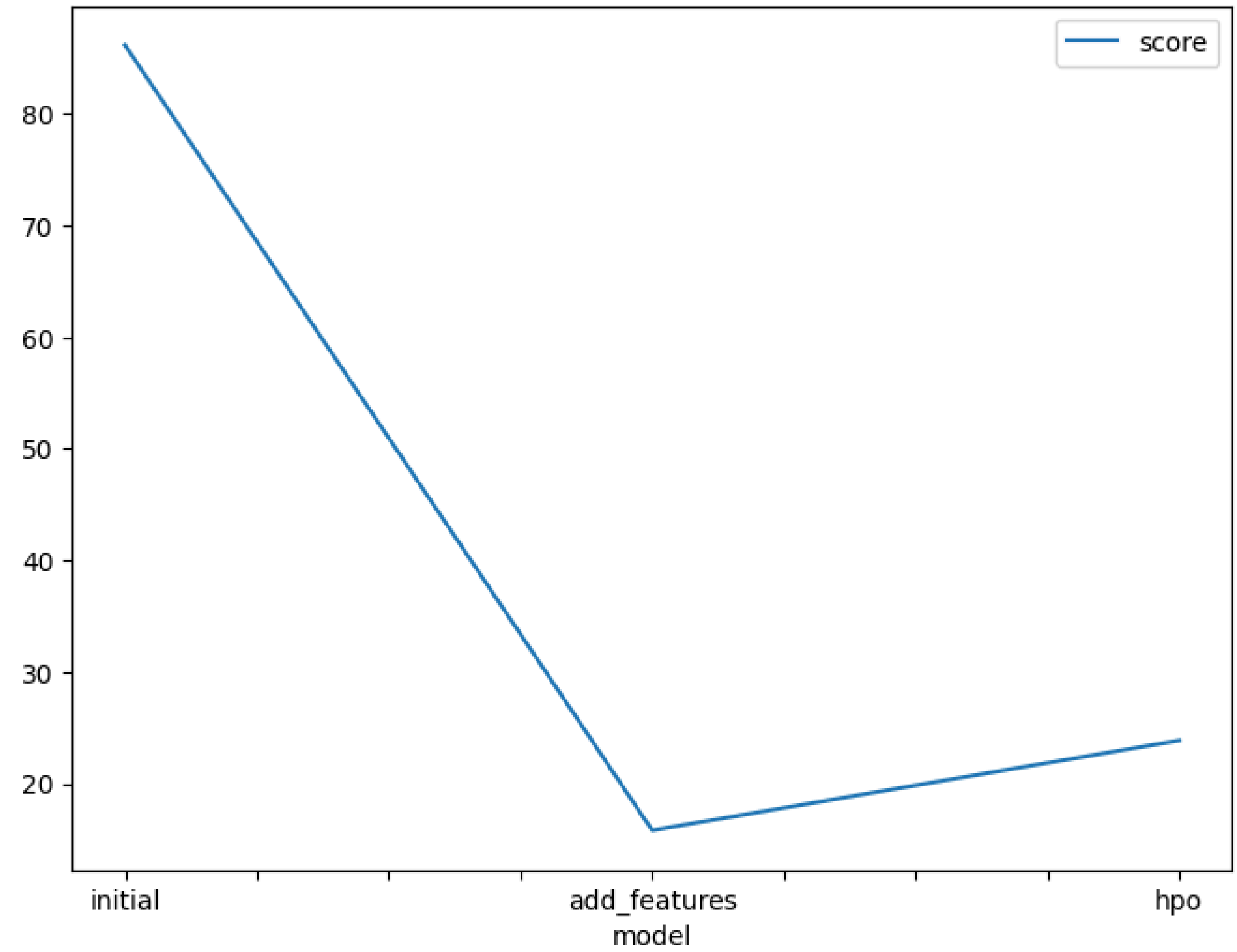
Model 3	gbm_options = { # 'num_boost_ro	WeightedEnsemble_L3_options = {	num_trials = 10 search_strate	**WeightedEnsemble_BAG_L2 -33.052860 -33.611532**

	und': 1000,
	}
		'ensemble_size': 70
		}
			gy = 'auto'
			hyperparameter_tune_kwargs = {
			'num_trials': num_trials,
			'scheduler' : 'local',
			'searcher': search_strategy,
			}
			Time_limit = 600
				root_mean_squared_error
				KAGGLE_SCORE: 0.54060

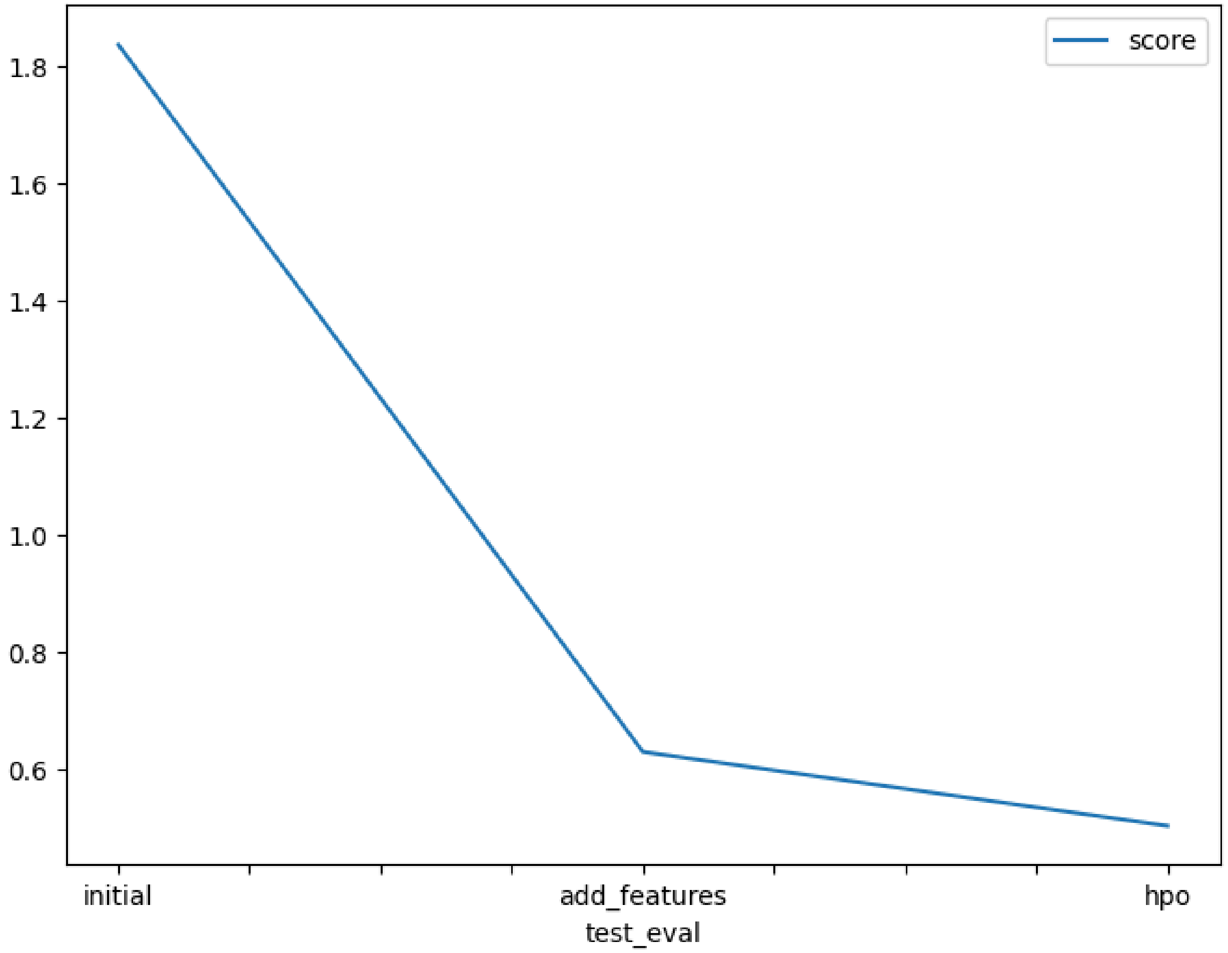
Model 4	hyperparameters = { 'NN_TORCH': [{}, {'activation': 'elu', 'dropout_prob': 0.10077639529843717, 'hidden_size': 256, 'learning_rate': 0.002735937344002146, 'num_layers': 12, 'use_batchnorm': True, 'weight_decay	Time_limit = 600	Time_limit = Default (No Time Limit)	**WeightedEnsemble_L3 -40.406118 root_mean_squared_erro** **KAGGLE_SCORE: 0.55657**

	': 1.356433327634438e-12, 'ag_args': {'name_suffix': '_r79', 'priority': -2}}, {'activation': 'elu', 'dropout_prob': 0.11897478034205347, 'hidden_size': 256, 'learning_rate': 0.0010474382260641949, 'num_layers': 12, 'use_batchnorm': False, 'weight_decay': 5.594471067786272e-10, 'ag_args': {'name_suffix': '_r22', 'priority': -7}}], }			

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



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



## Summary

I learned a lot from this project. I wish I had more time to dig through all of this. I was very surprised that the heavier neural network trained by autogluon even with a hidden size of 256 an 12 layers did not even match the performance of WeightenEnsemble_BAG_L2 Ensemble created by autogluon with the ensemble_size: 70 and num_boost_round: 1000 hyperparameters. I really thought the neural network would outperform that Ensemble model. I think the ensemble models perform better on tabular data than the neural networks do. Additionally, they are much quicker to train. In order to trian that neural network model on a CPU I had to remove time_limit hyperparameter and just let it train for over an hour, and it still did not perform as well on the Kaggle test data. This was a very interesting project with Autogluon and I absolutely need to study this material more in depth.