Practical 1:Predicting the Efficiency of Organic Photovoltaics

Haoqing Wang, Willie Jin, Youbin Kim

February 10, 2017

quantity in your submission.

1 Technical Approach

Overall, we tried several approaches to attempting to solve the problem. This can be divided into feature engineering, random forest tuning, and gradient boosting tuning.

Before trying various regression techniques, we realized the importance of feature engineering. In order to accurately predict the HOMO-LUMO gap for various molecules, we need to learn using relavent features. We believe that molecules that are structurally similar are likely to have similar HOMO-LUMO gaps. In order to judge similarity between molecules, we use RDkit to generate structural fingerprints of each molecule. The type of fingerprint we used are Morgan (circular) fingerprints, which hashes sections of the molecule with increasing radii up to a preset radius. Given a wide diversity of molecules, a diameter of 4 or 6 gives the best results in measuring similarity (O'Boyle & Sayle, 2016). We thus fingerprinted each molecule with a diameter of 4 into a 256 bit vector. Finally, we appended these 256 new "features" to the existing data file given.

Just listing things we should talk about: First we analyzed the features we had already–¿ looked for repetitions,not avaliable values, first writing for loops to try to tune RF parameters–¿ gridsearch–¿ random search(because took too much time) attempting to use oob for rf but got error (UserWarning: Some inputs do not have OOB scores. This probably means too few trees were used to compute any reliable oob estimates. warn("Some inputs do not have OOB scores. " done loading)

Thus did 80/20 split on test data. Attempted to do 5-fold cross-validation for parameter tuning using grid search, took too much time.

Based gradient boosting tuning on this website: https://www.analyticsvidhya.com/blog/2016/02/complete-guide-parameter-tuning-gradient-boosting-gbm-python/

Tuned random forest with help from this website: https://www.analyticsvidhya.com/blog/2015/06/tuning-random-forest-model/

Chose original values based off of these suggestions.

Random Forest Tuning	
Hyperparameter	Utilized Value
Max Depth	8
Min Samples Leaf	1
N Estimators	26

Table 1: The results of our attempted Randomized Search CV

Model	MSE
BASELINE LINEAR	0.089
BASELINE RF	0.074
TUNED RANDOM FOREST*	0.073
GRADIENT BOOST LEARNING RATE 0.1	0.077
GRADIENT BOOST LEARNING RATE 0.05	0.077
TUNED RANDOM FOREST ON ALL FEATURES**	0.013

Table 2: This table displays the predicted mean square error for every model on a set randomized 20% of the test data. *n estimators: 64, max depth:9, min samples leaf:50

2 Methods

2.1 Feature Engineering

2.2 Linear Regression

2.3 Random Forest

2.4 Gradient Boosting

3 Results

This section should report on the following questions:

- Did you create and submit a set of predictions?
- Did your methods give reasonable performance?

You must have *at least one plot or table* that details the performances of different methods tried. Credit will be given for quantitatively reporting (with clearly labeled and captioned figures and/or tables) on the performance of the methods you tried compared to your baselines.

4 Discussion

Ultimately, analyzing such a large amount of data proved to be a unique challenge for us because of the high computing power and long run times required to process the data. It was difficult to tune the models as best as we would of liked, but overall our results were promising.