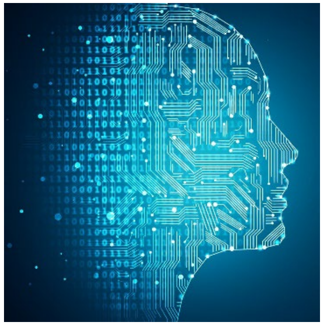
Machine Learning

Feature Engineering

Kevin Moon (kevin.moon@usu.edu)

STAT/CS 5810/6655



Outline

1. Feature engineering motivation
2. An empirical study
3. Feature engineering examples

What is feature engineering?

- The process of creating new features from raw data (i.e. existing features)
  - Often uses domain knowledge
  - Examples include ratios, differences, delays (in time series), or other mathematical transformations
  - Similar to transformations that human analysts do to help understand interactions of existing features
    - E.g. BMI, wind chill
- Feature engineering has historically been important in many Kaggle competitions
- Goal is to develop new features that improve the performance of machine learning algorithms

Why feature engineering?

- It can be shown that neural networks can compute any continuous function
- Shouldn't neural networks be able to *automatically* learn the relevant features?
- Furthermore, what about the ***data processing theorem***?

Theorem: Consider the random variables from this Markov chain: $X \rightarrow Y \rightarrow Z$. Then

$$I(X;Y) \geq I(X;Z)$$

- I.e., no processing of Y can increase the information that Y contains about X
- Similar to the concept of *garbage in, garbage out*

Why feature engineering?

- Yet, we'll see some examples where feature engineering clearly helps
- How can this be?
- **Answer**: just because the information is present in the data, this doesn't mean that a given machine learning algorithm is capable of extracting it from the raw data
  - While neural networks can do this in theory (they're universal), in practice it's hard
  - May require more training data or training time than you have access to or a larger network than is practical for a neural network to be successful
  - Feature engineering can speed up the learning process since less time is needed to learn the relevant features

Feature learning and neural networks

- Certain neural network architectures seem to do very well at feature learning
- **Example**: CNNs are very good at learning relevant and even universal features from images
  - Transfer learning has been quite successful in image problems
- **Example**: Attention in RNNs and Transformers has led to large improvements in NLP
- However, in both of these examples, the <u>architectures</u> were engineered to force the networks to pay attention to certain aspects of the data
- So in some sense, neural networks can exchange ***feature engineering*** for ***architecture engineering***

When to use feature engineering?

- You have domain knowledge that could lead to relevant features
  - This includes data preprocessing. You’re usually better off doing known best practices in preprocessing than expecting the machine learning method to figure it out
- Your performance without it isn’t what you need it to be
- You have the time and money for it

An Empirical Study

An Empirical Study

- Heaton (2016) did an empirical study of which features are harder to learn for various machine learning methods.
- Created transformed features from inputs
- Trained different ML methods to do regression and predict the transformed features using the inputs
  - SVM regression
  - Random forests (RF)
  - Neural network
  - Gradient boosted machines (GBM)
- Significant time was NOT spent on tuning
  - Thus the results show how well the methods learn with general tuning parameters

The Transformations

- Logarithms and power functions

$$y = \log x, \qquad y = x^2, \qquad y = \sqrt{x}$$

- Differences and ratios

$$y = x_1 - x_2, \qquad y = \frac{x_1}{x_2}$$

- Counts (how many elements of a vector satisfy a certain condition)
- Polynomials

$$y = 1 + 5x + 8x^2$$

The Transformations

- Rational differences

$$y = \frac{x_1 - x_2}{x_3 - x_4}$$

- Rational polynomials

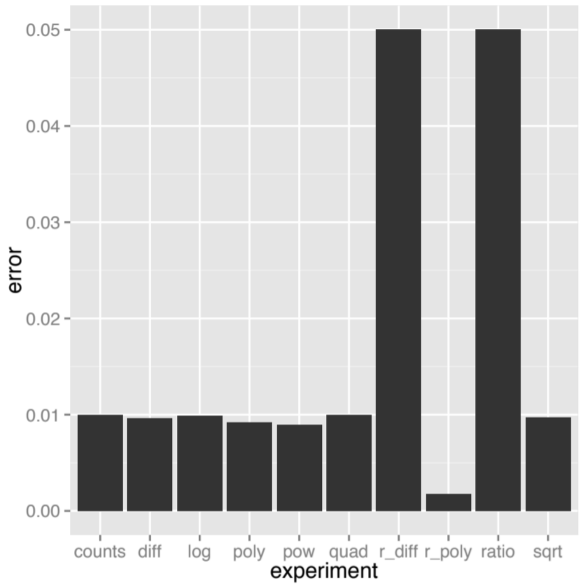
$$y = \frac{1}{5x + 8x^2}$$

- Distance between quadratic equation roots

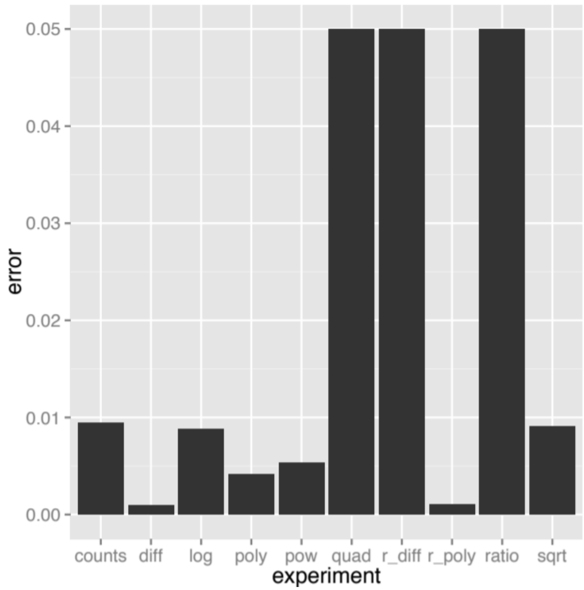
- For the results, MSE was capped at 0.05. An MSE higher than this indicated failure to learn this feature.

Results

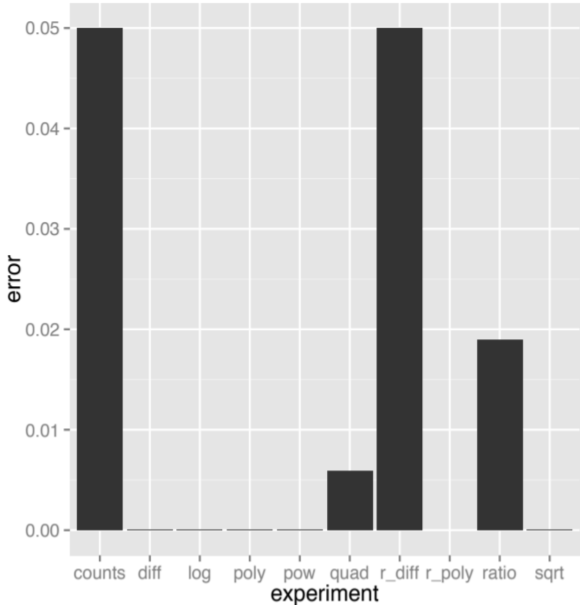
Neural Network



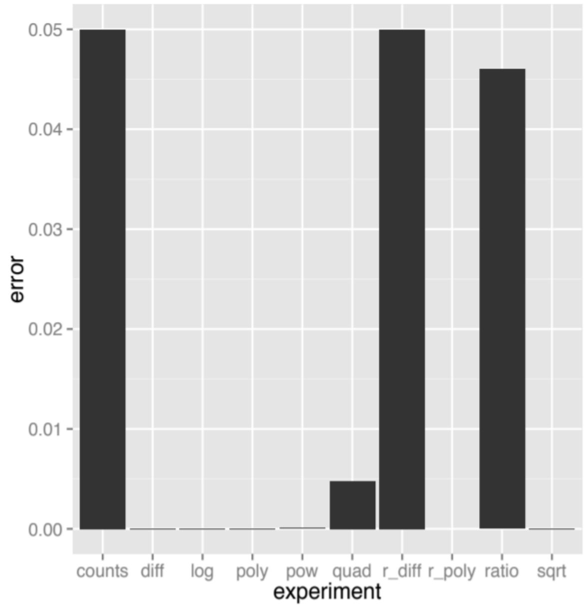
SVM



Random Forests



GBM



Feature Engineering Examples

NYC Taxi Fare Prediction

- Data from a Kaggle competition
- **Goal**: predict the taxi fare amount (including tolls) using features such as lat-lon coordinates, time of day, date, and # of passengers
- Feature engineering can help a lot with this data
- **Example**: create a binary variable whether the passenger is being picked up at an airport
  - Usually there is an extra charge for this
  - While this information is technically present in the lat-lon coordinates, creating a feature that extracts this directly skips the need to learn it

Trade Sign Classification

- **Goal**: classify high frequency trades as buyer or seller initiated
- **Motivation**: a lot of finance research relies on this trade sign
  - Obtaining real trade signs is expensive
  - Accurate trade sign classification would help reduce costs and improve accuracy of financial research
- We (Jared Hansen) did feature engineering to improve classification performance

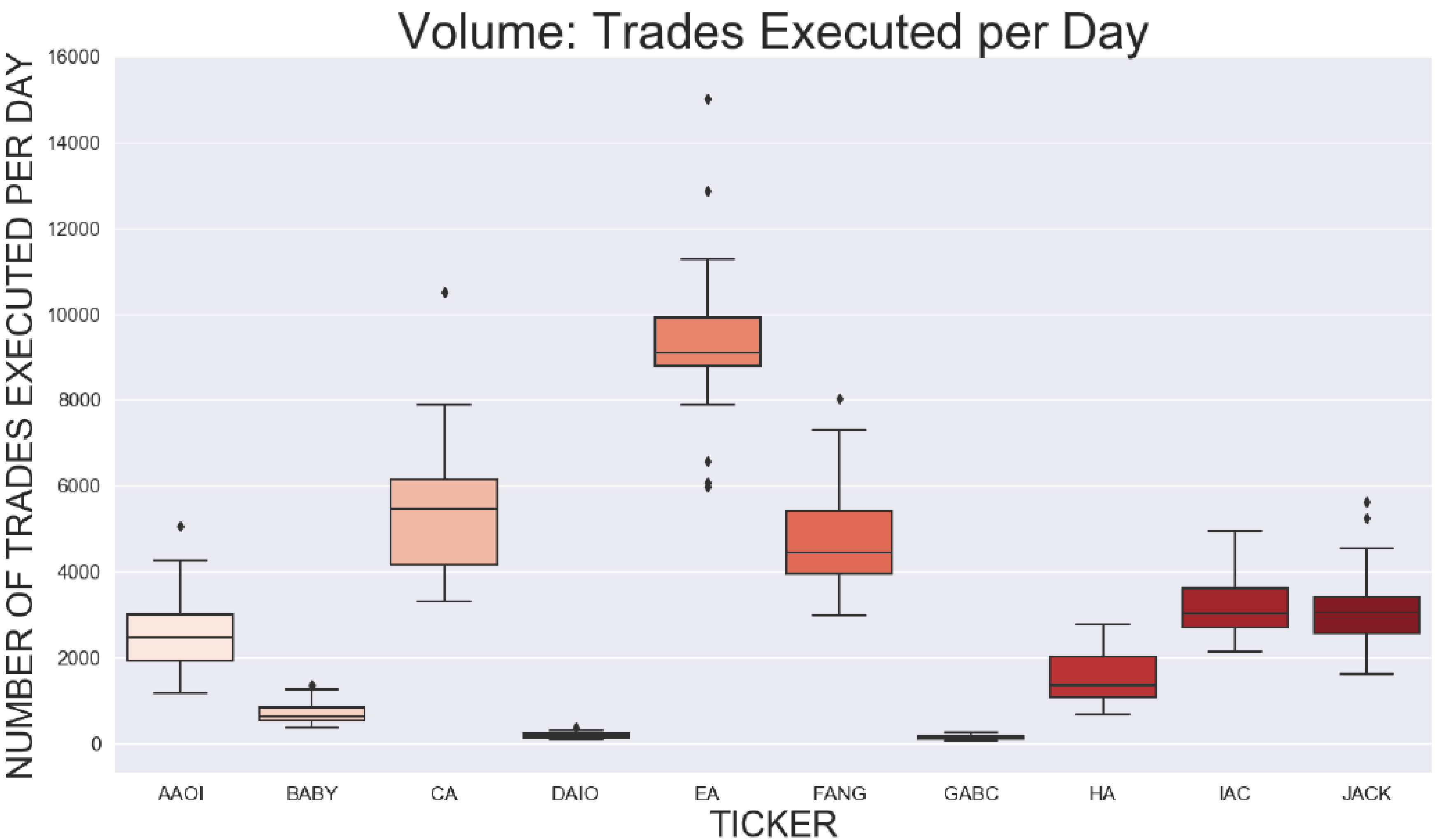
Trade Sign Data

- Stocks from different financial sectors and sizes

STUDY DATA: FINANCIAL DESCRIPTIONS				
Ticker	Market Cap ($ MM)	Cap Category	Average Spread	Average Price
AAOI	525	Small	0.0110	26.96
BABY	1,099	Small	0.0548	33.15
CA	14,684	Large	0.0100	35.24
DAIO	68	Small	0.0200	8.27
EA	38,444	Large	0.0219	125.35
FANG	12,490	Large	0.0257	127.21
GABC	788	Small	0.0243	34.35
HA	1,870	Mid	0.0500	36.52
IAC	12,193	Large	0.0705	158.86
JACK	2,564	Mid	0.0167	86.84

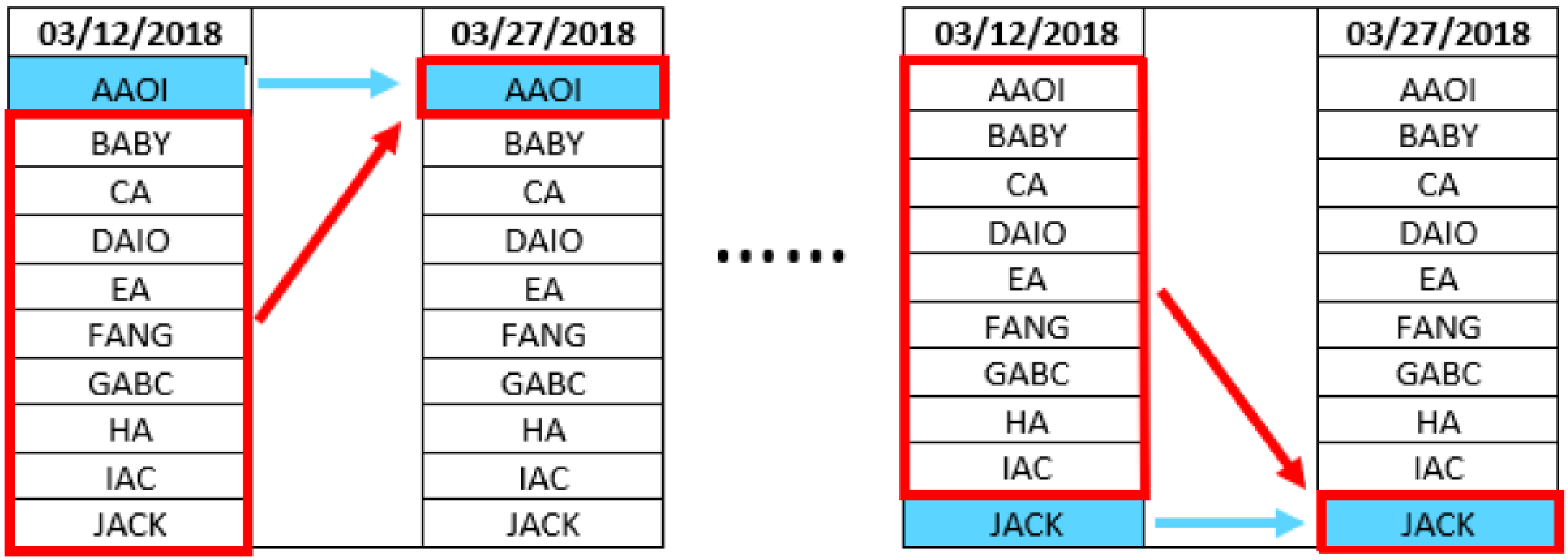
Trade Sign Data

- Variety of trading volume



Validation Approach

- Same-ticker and all-but cross-validation schemes.

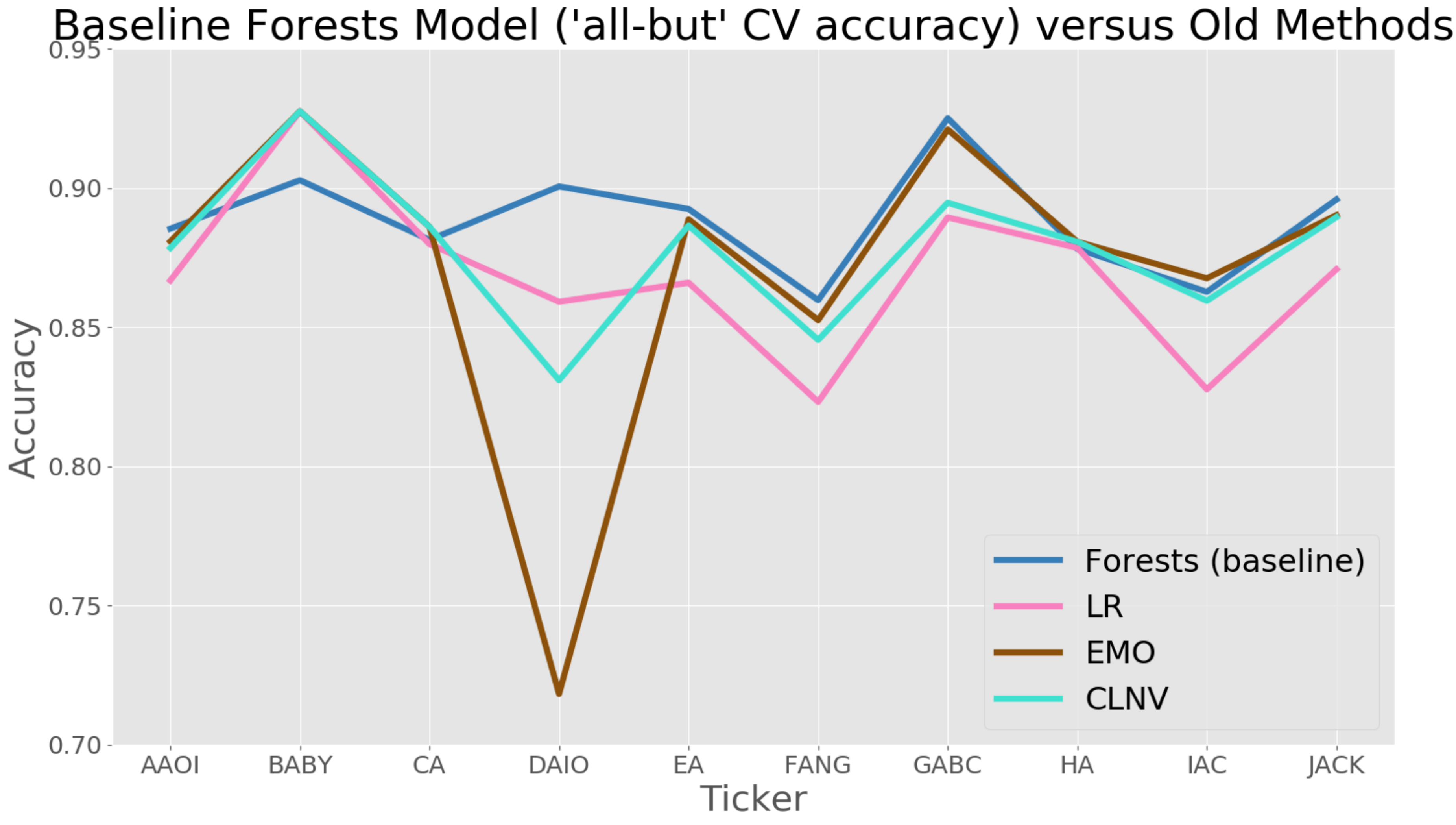


- **Key metric**: $\text{wab-PCC} := \left[\dfrac{\sum_{i=AAOI}^{JACK} (length_i)(\text{all-but PCC}_i)}{\sum_{i=AAOI}^{JACK} (length_i)} \right]$

Example base features

- Price (normalized)
- Time of day
- # of shares exchanged
- NBO (best offer/ask)
- NBB (best bid)
- Midpoint (between NBO and NBB)
- Some standard finance features created from features in earlier time points
- Predictions from classical finance methods (LR, EMO, CLNV, tick test, quote test, etc.)

Without feature engineering

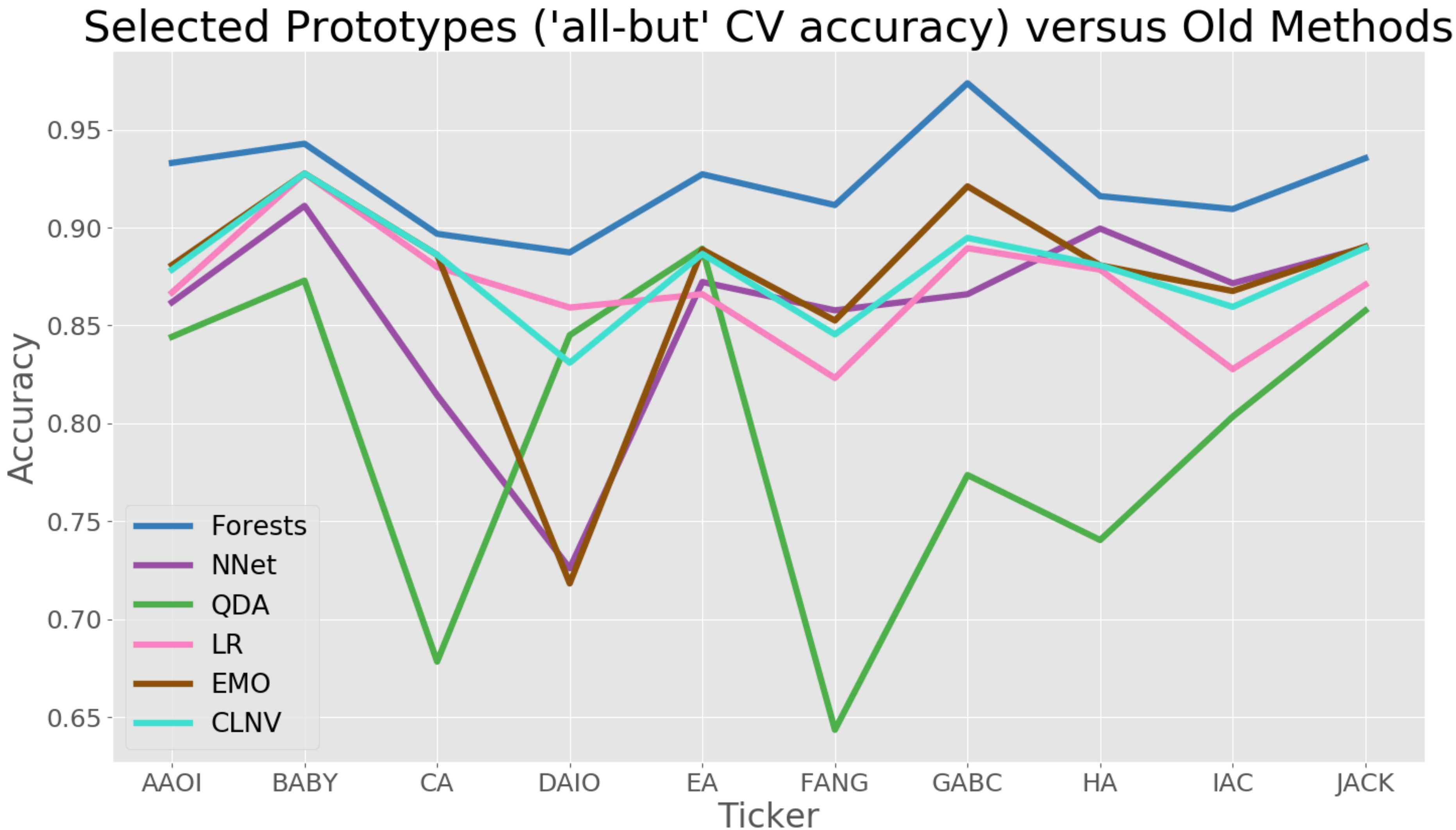


Initial feature engineering

Applied these transformations to some existing features

- **scaleOneVar(x)**: the new column is $\left[\frac{x_i - median_x}{IQR_x}\right]$
- **nominalDiff**(x, y, n): the new column is $\left[x_i - y_{i-n}\right]$
- **scaledDiff**(x, y, n): the new column is $\left[\frac{x_i - y_{i-n}}{y_{i-n}}\right]$
- **colsAgree**(x, y): the new column is $\left[x_i == y_i\right]$
- **lagVar**(x, n): the new column is $\left[x_{i-n}\right]$

With some feature engineering

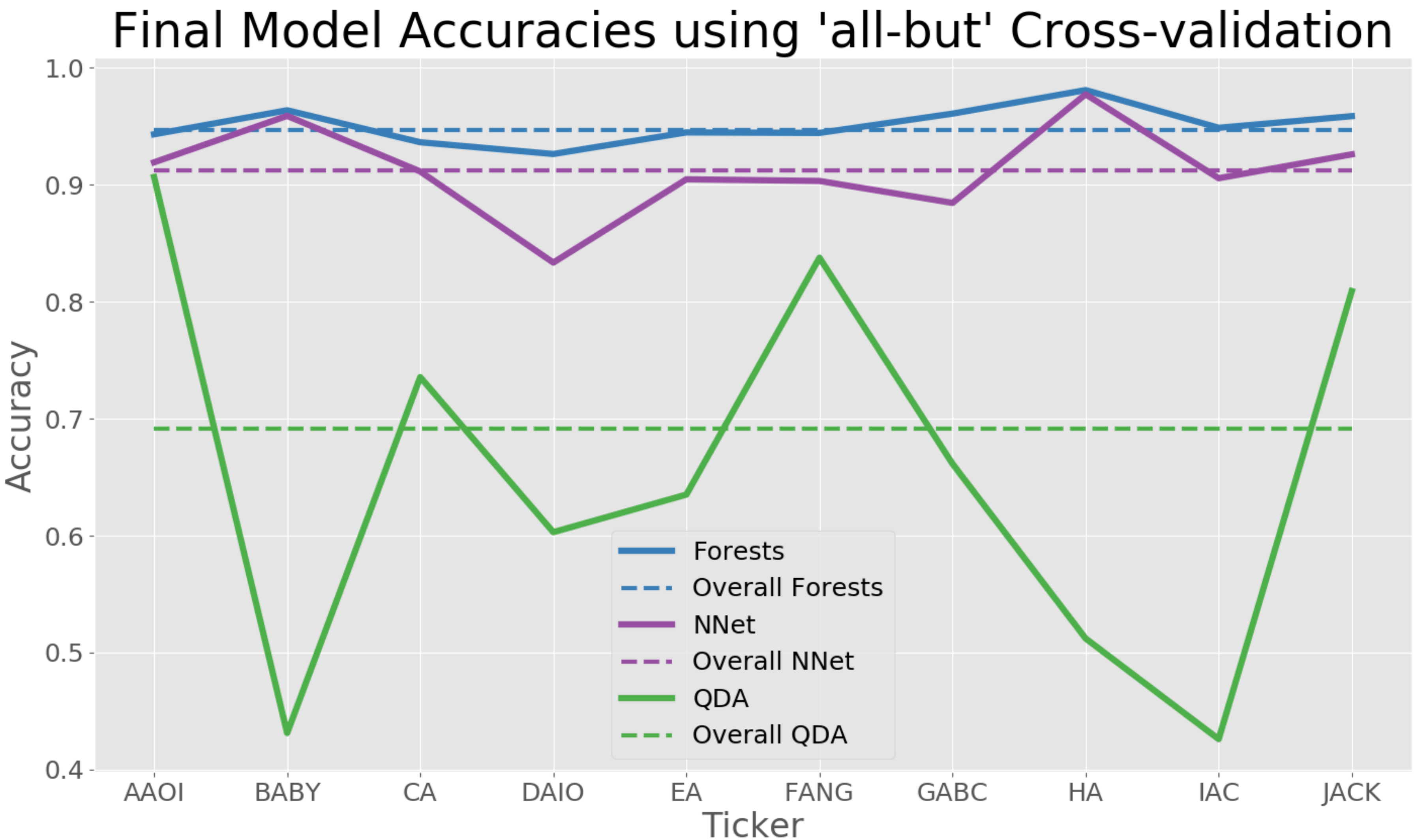


More feature engineering

Added features that Heaton (2016) found to be difficult to learn

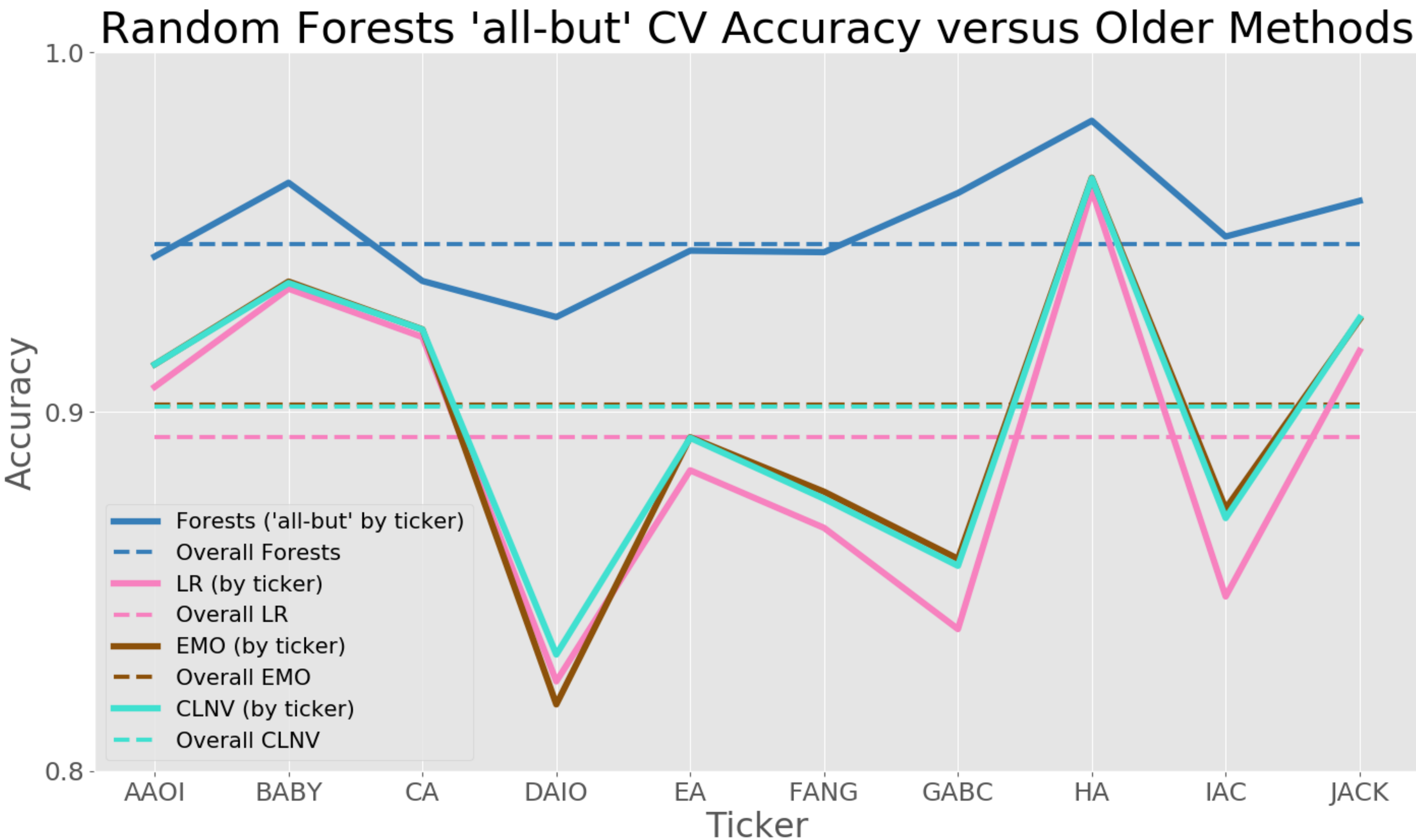
- Summed the old predictions (sumOfBuysellCols)
  - This is a “counts” type of feature
- Included some rational differences
  - E.g. rational difference between price and quotes
    - A variation on this feature was used in older models
  - Other rational differences

Final Results



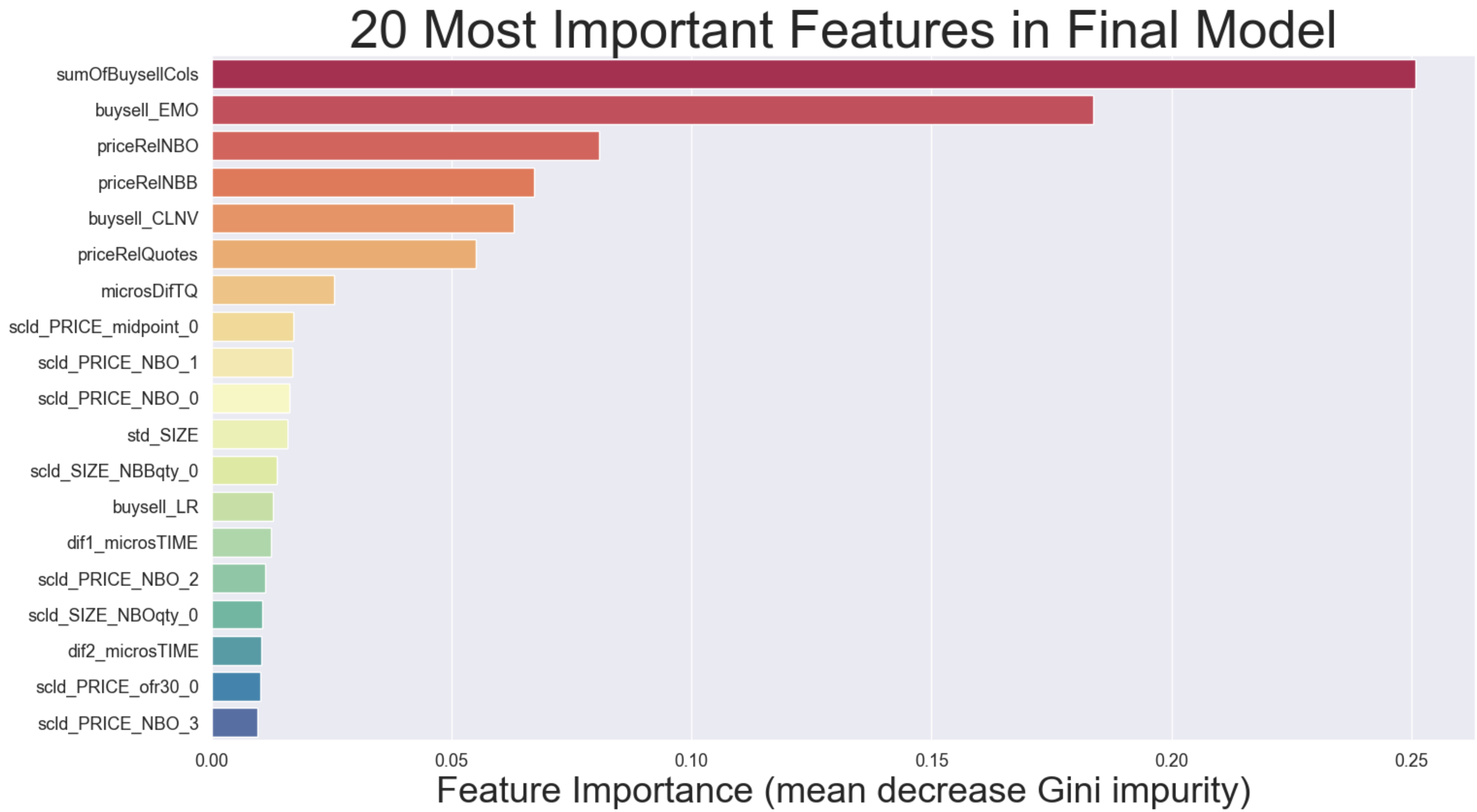
Final Results

- wab-PCC: {RF: 94.7}, {LR: 89.3}, {EMO: 90.2}, {CLNV: 90.1}



Feature Importance

- Highlights the importance of feature engineering in improving accuracy



Further reading

- Heaton, "An empirical analysis of feature engineering for predictive modeling," in *Southeast Con 2016*, IEEE, 2016.
- https://www.kaggle.com/c/new-york-city-taxi-fare-prediction
- Jared Hansen's MS thesis, "Applications of Machine Learning in High-Frequency Trade Direction Classification"
- Feature Engineering from A-Z: https://feaz-book.com/