Electricity Price Predictions

Eric Yehl, John Stuart, Priya Subramanian, Adam Yankelevits, Aaron Drew, Arbaaz Shakir

UC Berkeley Data-X | Spring 2018

Problem:

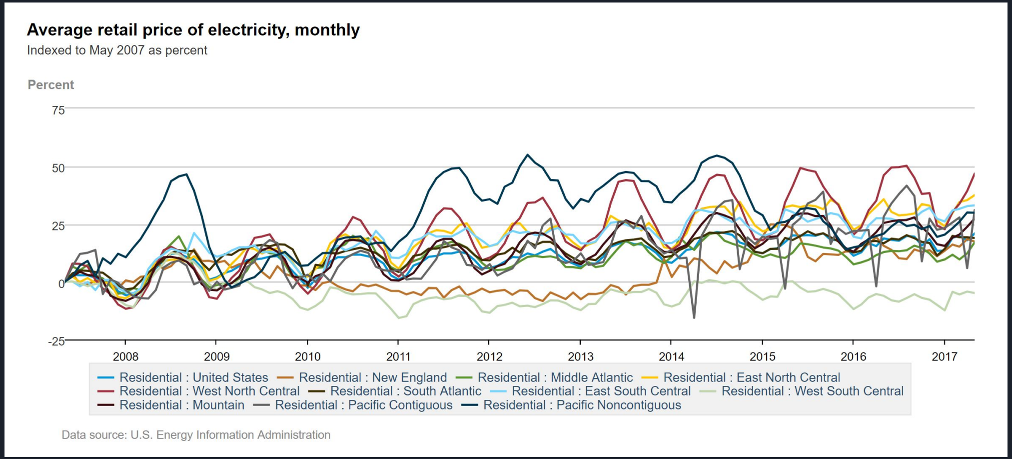
Electricity prices are unpredictable.

Energy infrastructure investments are gambles on future earnings.

Financing is slow for progressive power projects.

Electricity prices are volatile and dependent on many factors

- Location
- Sector
- Time of day
- Time of year
- Supply and Demand
- New Technology



Source: https://news.energysage.com/residential-electricity-prices-going-up-or-down/

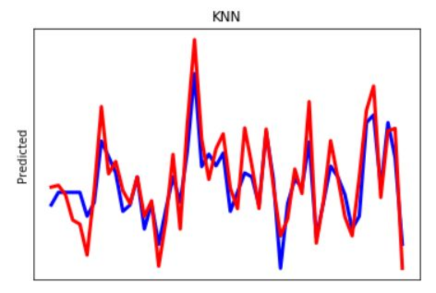
Learning Path

Last Semester

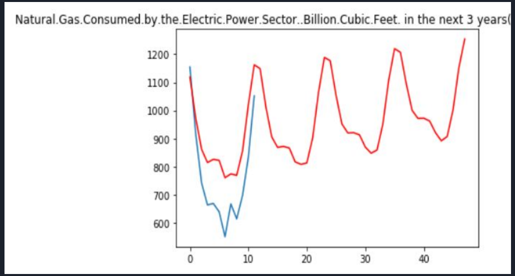
Our Ah-ha Moment

Learning Path: Last Semester

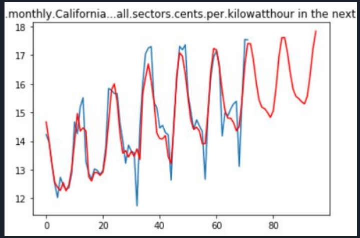
Established monthly electricity generation from various sources as good features for standard regression

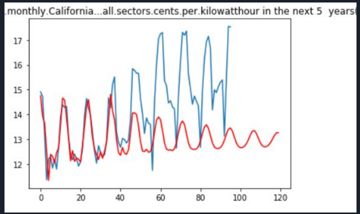


Used S-ARIMA to project each feature, but not all were accurate in 5 year projections



Ultimately only built a single feature projection model, not using any potentially better predictors besides price





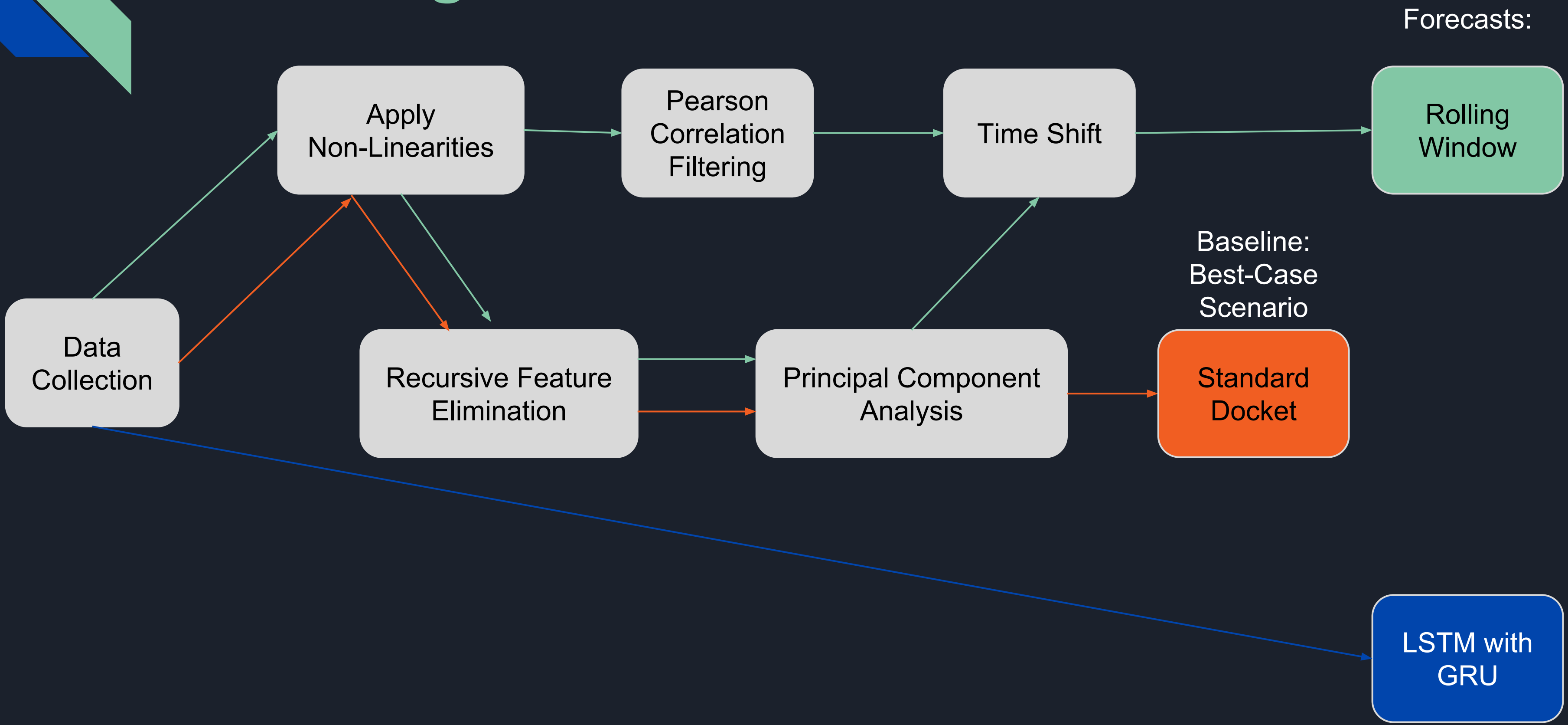
Learning Path: Our Ah-ha Moment

- Build an automated feature construction, selection and price prediction *pipeline*
- Make accurate price forecasts for any commodity
- All that is required of the user is to input relevant “hunch” features

Our Solution: *The Pipeline*

- Data Collection
- Exploratory Data Analysis
- Feature Construction & Filtering
  - Dimensionality Transformation
- Model Implementation

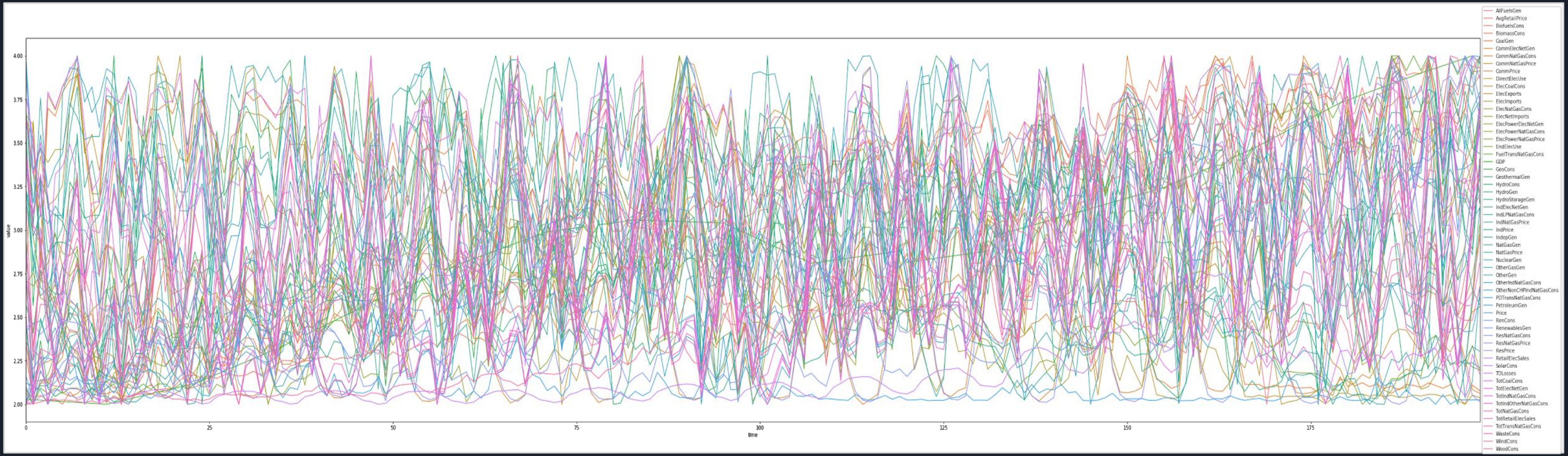
Learning Process: An Overview



Our Solution: Data Collection

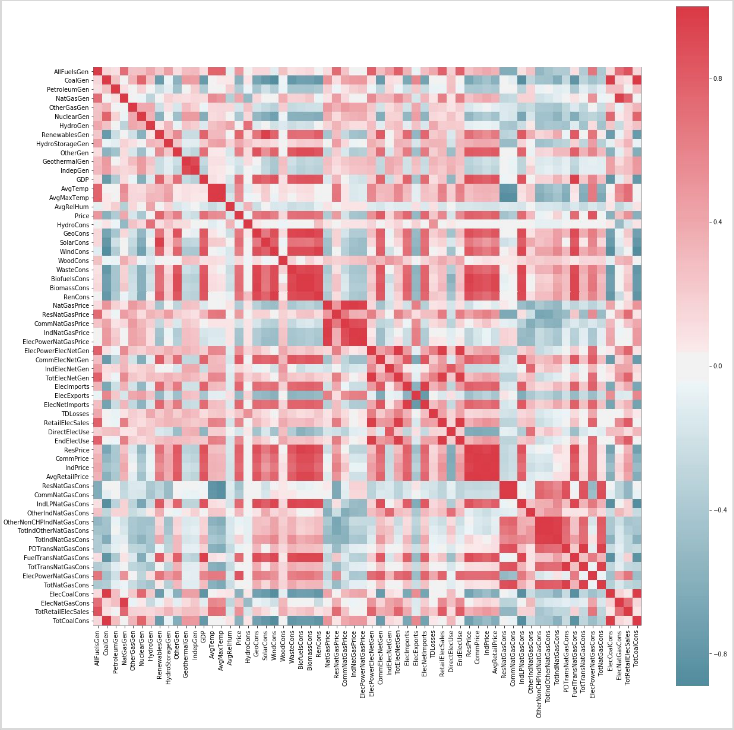
- Clean and update data from last semester
- Add new features
    - Electric Vehicles
    - Generation Capacity
    - Gasoline/Diesel Spot Prices
    - Utility Stock Prices
    - Economic Indicators
- Google Sheets API for flexible data collection and input to pipeline

Our Solution: EDA: Mess of Signals

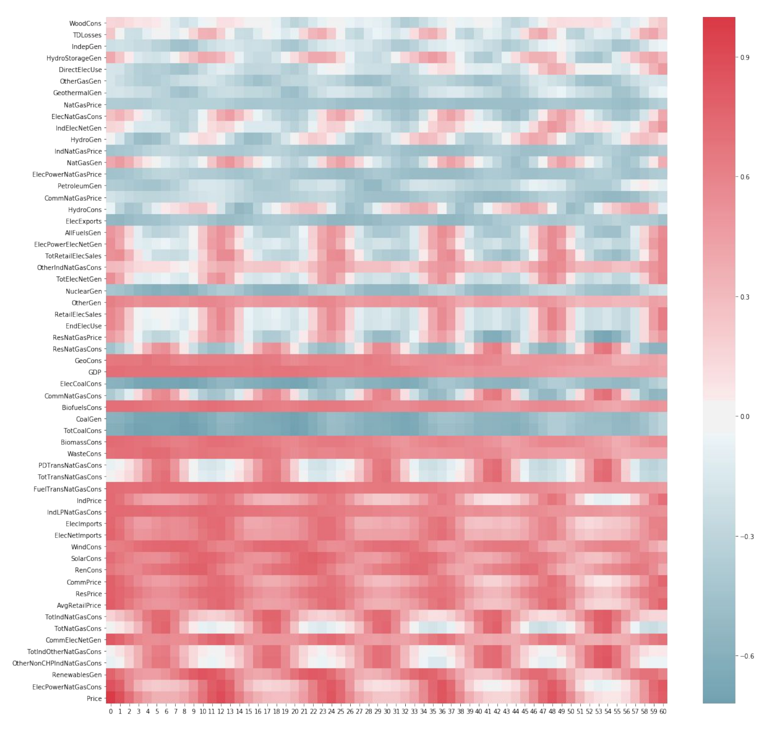


Our Solution: EDA: Heat Maps

Standard Pearson correlation map



Time offset correlations with electricity price



Our Solution: Feature Construction

Functions on signals

- Integral
- Difference
- Log
- Square Root
- Square
- Exponential
- Moving Average

→

Combinations of signals

- Addition
- Subtraction
- Multiplication
- Division

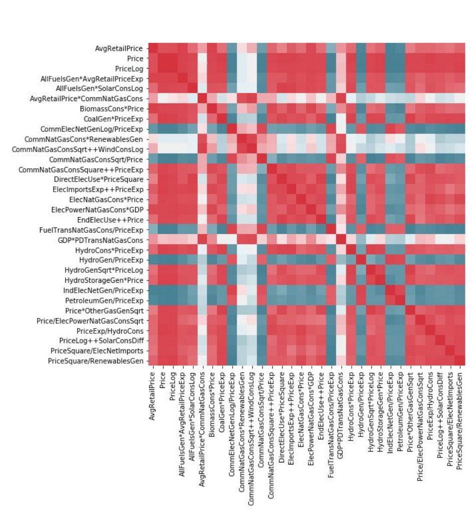
130+ features → 1,000,000,000+ possible new features

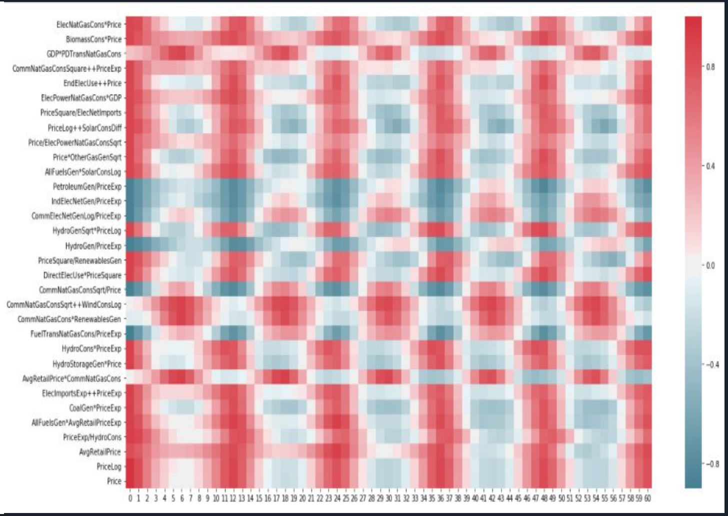
Our Solution: Feature Filtering

Methods of feature selection

- Throw out features with low price correlation or high correlation with other features

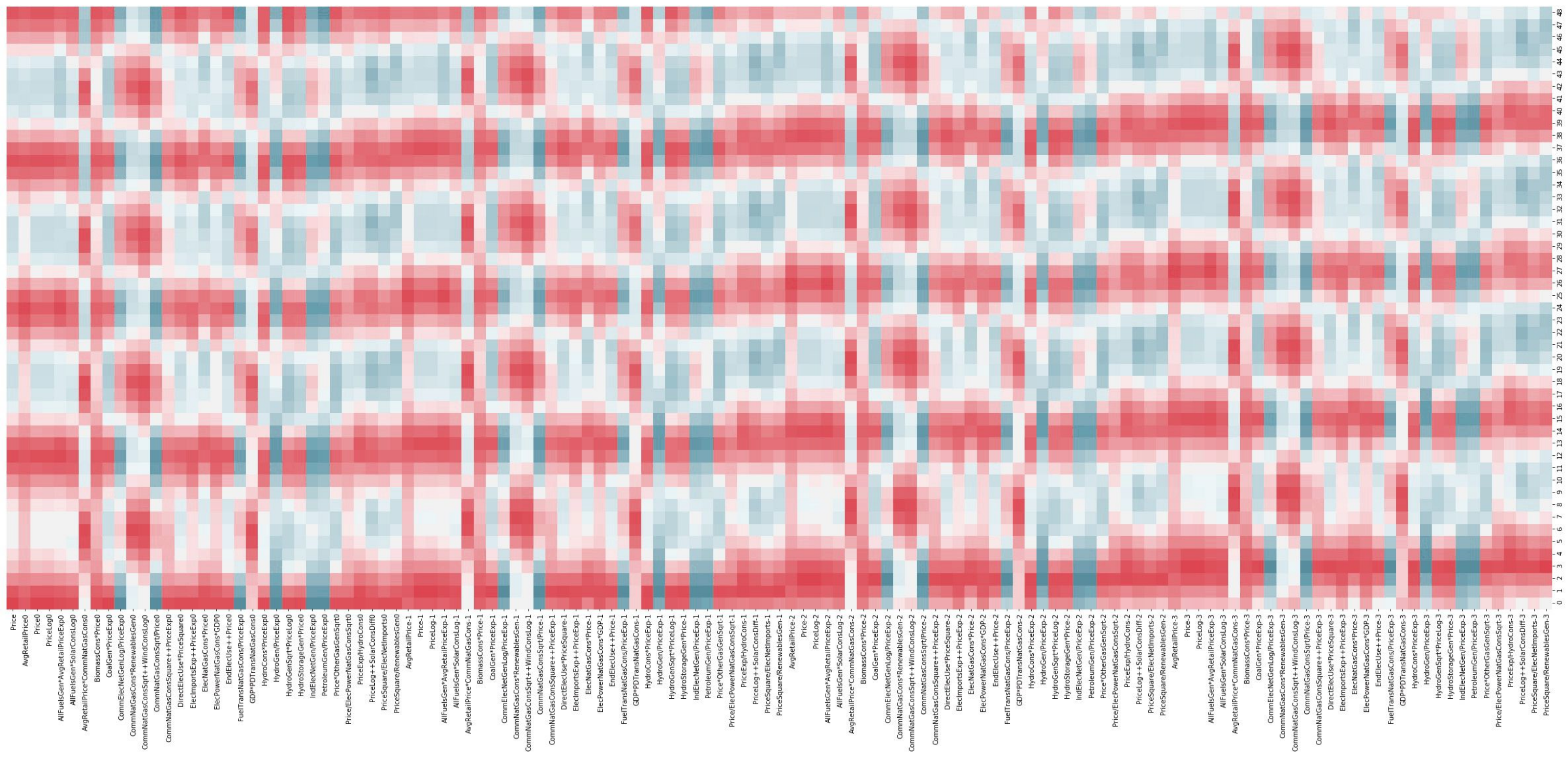
- **Recursive feature elimination**
  - drop features with the lowest regression coefficient until only n features remain.





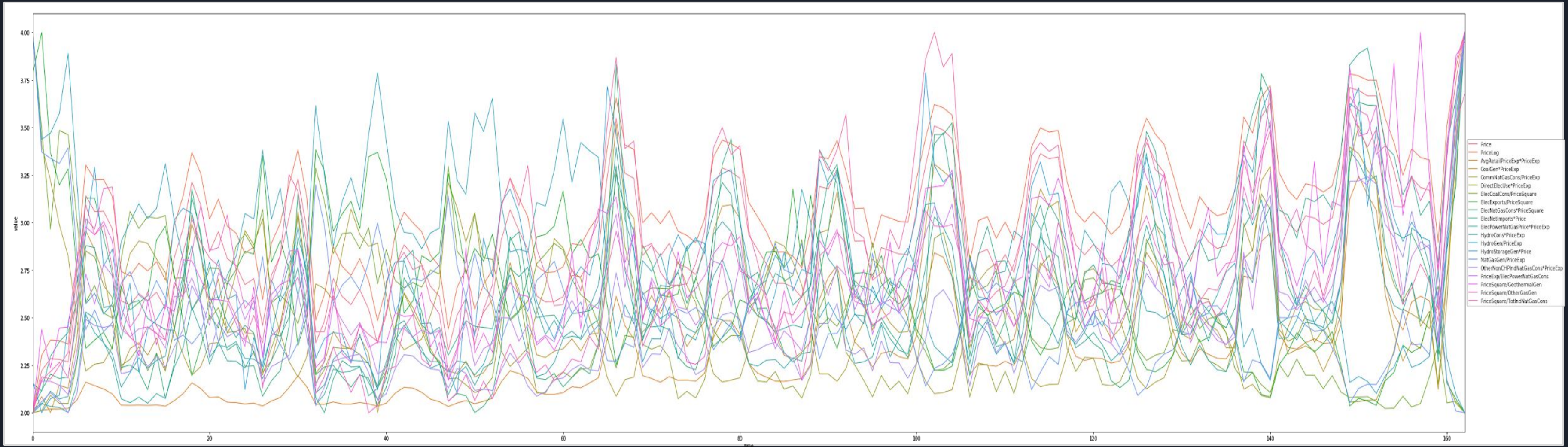
```
['PetroleumGen',
 'CommNatGasPrice',
 'Federal Interest Rates',
 'Inflation', 'SolarConsDiff',
 'RenConsLog',
 'ResNatGasPriceSquare']
```

- Seasonal correlations with price
- Different features and delays are best for different forecast offsets



Our Solution: Feature Filtering

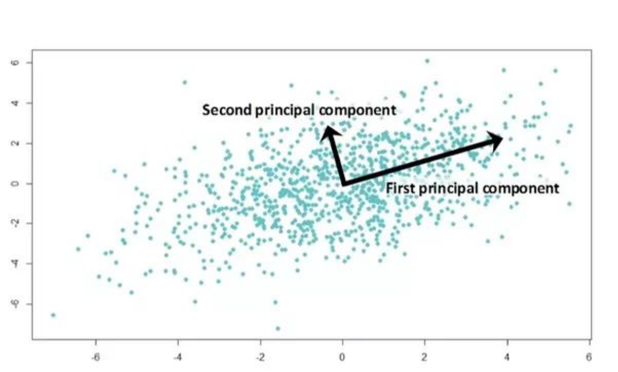
Newly constructed and selected signals show structure



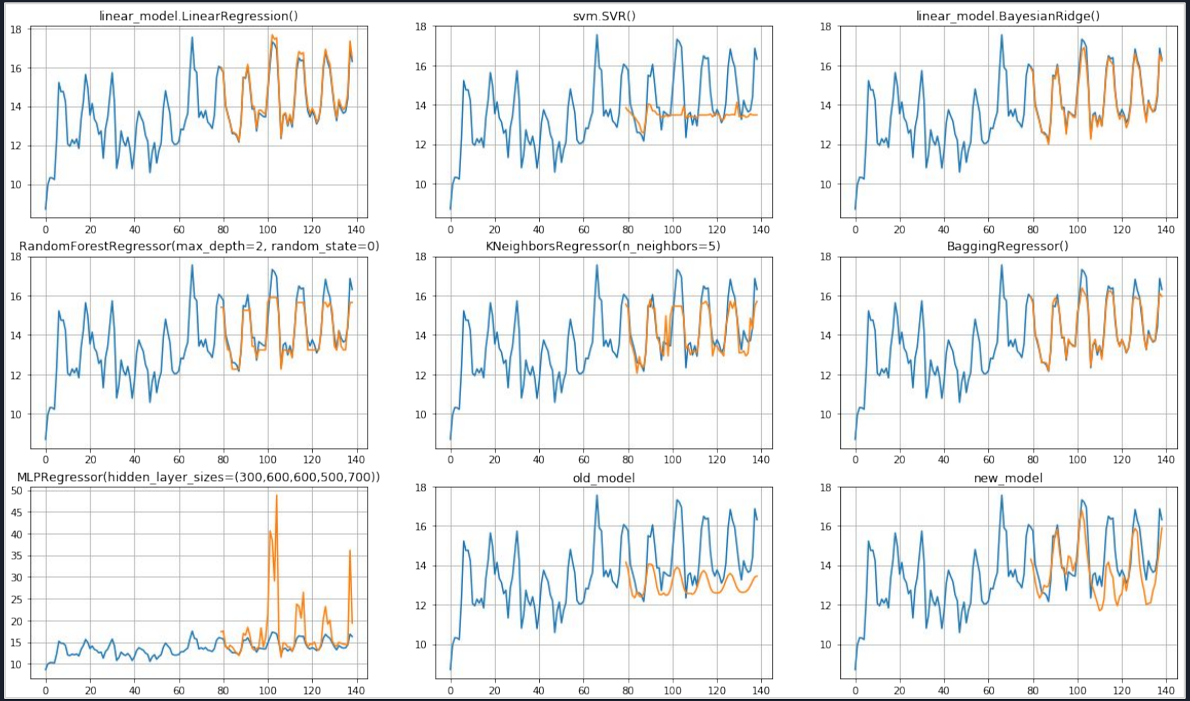
Our Solution: Dimensionality Transformation

Principal Component Analysis Transformation

- Reduce dimensionality of the thousands of features created through feature construction
- Reduction in dimensionality directly resulted in a reduction in computation time
- Keeps as much relevant information as possible from all features without completely eliminating them



Learning Path: Standard Docket



These are not forecasts, only predictions from standard regression models

Our Solution: Model Implementation

Rolling Window Regression

- Algorithm conceived and built from scratch
- Simple, flexible, fast
- Scalable, improves with more data streams
- Classic train-test splitting and hyperparameter optimization

offset	look_back	train_prop	neg_alpha	len_data	n_iter	num_feats	test_error	test_num	train_error	train_num
		0.80	**-0.00150**	127	47	5	0.037212	26	0.036994	101
			-0.00115	127	47	6	0.037122	26	0.037308	101
			-0.00080	127	55	6	0.048016	26	0.032996	101
			-0.00045	127	106	13	0.038559	26	0.033392	101
			-0.00010	127	297	18	0.047942	26	0.027834	101
			-0.00150	127	49	5	0.054549	22	0.034060	105

Recurrent Neural Network

- RNNs are good for sequence problems because their connections form a directed cycle i.e. they can retain state from one iteration to the next by using their own output as input for the next step.
- But a simple recurrent network suffers from a fundamental problem of not being able to capture long-term dependencies in a sequence.
- LSTMs are powerful enough to learn the most important past behaviors and understand whether or not those past behaviors are important features in making future predictions.

Our Solution: Rolling Window Regression

- **Independent Lasso regression** models trained for each month offset
- Ranges of past signals used as features
- Hyperparameters optimized for each model:
    - Look-back range
    - Regularization coefficient
    - Train/test proportion
    - Upsample distribution shape (weight more recent data during training using nonuniform interpolation)

Price	PriceLog	DirectElecUse*PriceExp	ElecNetImports++Price
3.114551	1.136085	57.245103	6.544049
3.160991	1.150885	60.049758	6.512692
3.247678	1.177940	71.407899	6.745247
3.027864	1.107857	58.340342	6.580538
3.179567	1.156745	60.069562	6.391884

→

Price	Price-1	PriceLog-1	DirectElecUse*PriceExp-1	ElecNetImports++Price-1	...	HydroGen/PriceExp-18	HydroStorageGen*Price-18	IndepGen/PriceSquare-18
3.114551	3.160991	1.150885	60.049758	6.512692	...	0.114276	8.463106	0.356414
3.160991	3.247678	1.177940	71.407899	6.745247	...	0.130202	7.432454	0.397318
3.247678	3.027864	1.107857	58.340342	6.580538	...	0.115091	8.382287	0.353784
3.027864	3.179567	1.156745	60.069562	6.391884	...	0.110844	9.076551	0.332916
3.179567	3.402477	1.224504	70.922539	6.760662	...	0.070796	12.801406	0.233193

24 month Rolling Window

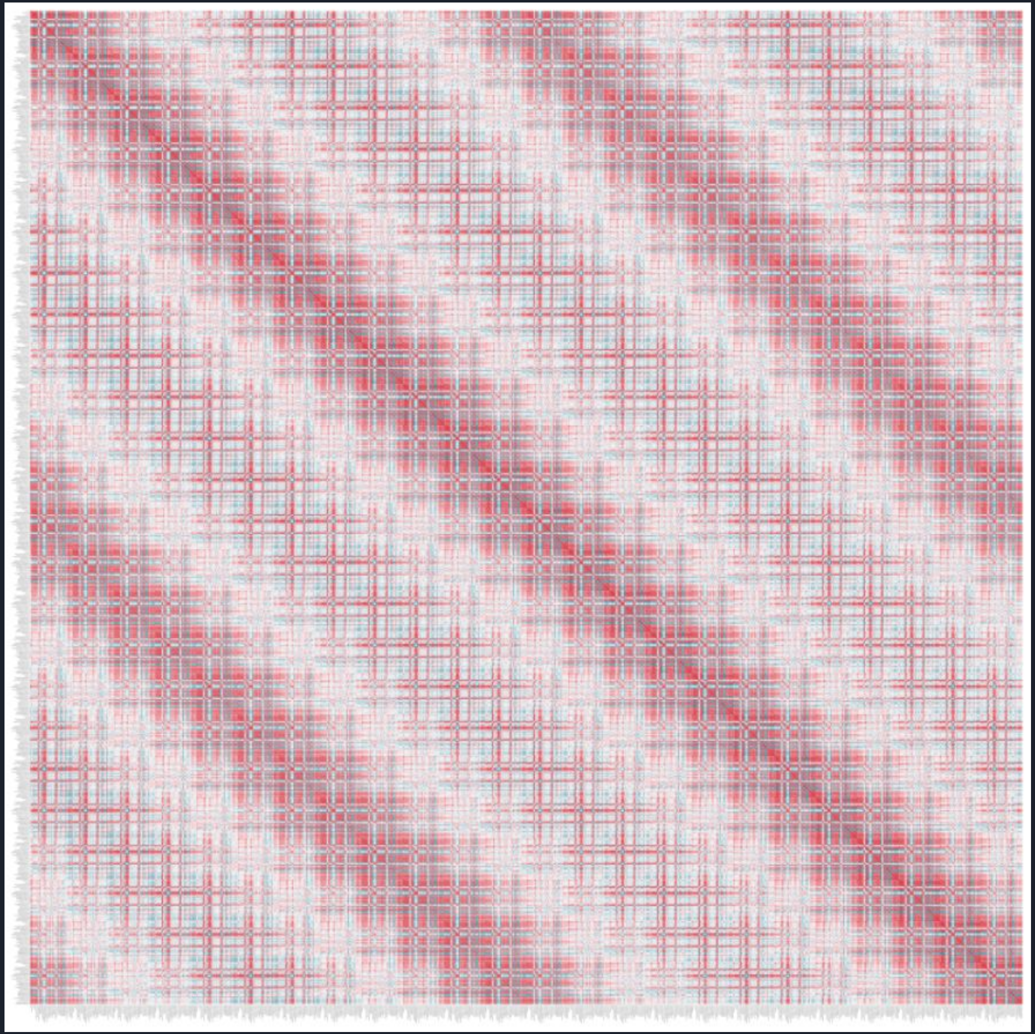
This is for a 1-month ahead prediction

- e.g. predicting Jan 2018 using Jan 2016 - Dec 2017 data

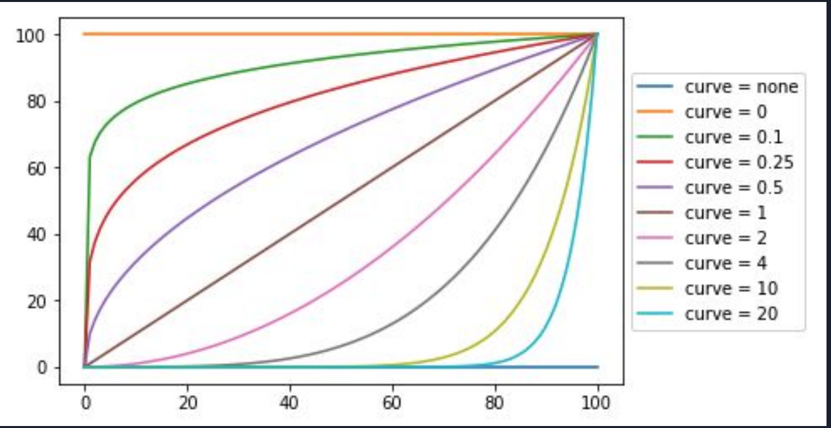
Presents some features that are highly correlated with price, but not with each other

Similar correlation heat map for future month offsets

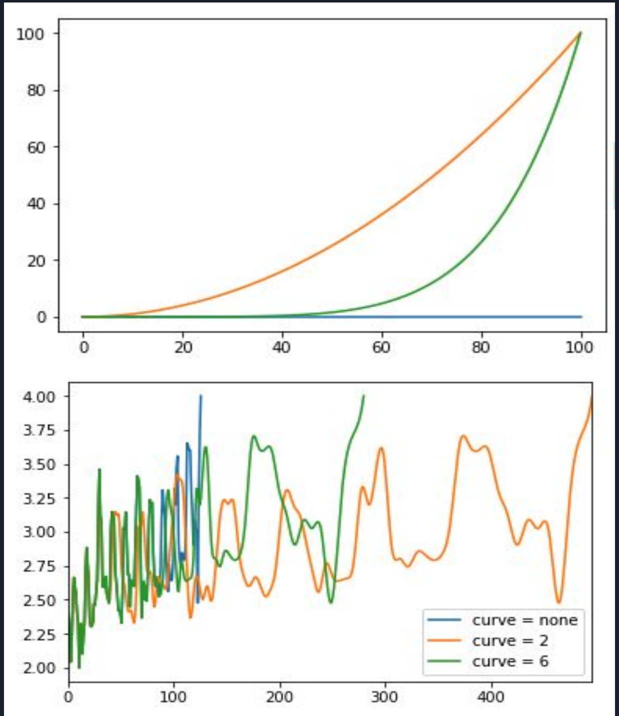
- e.g. predicting Dec 2018 using Jan 2016 - Dec 2017 data



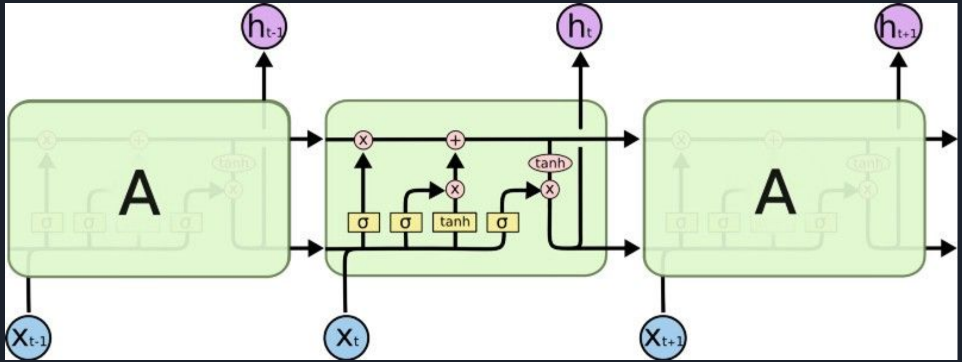
Weighted interpolation density



Hyperparameter tunes shape of interpolation density function to increase frequency of training samples from more recent data



Our Solution: Recurrent Neural Network



Forget gate takes the output at t-1 and the current input at time t and concatenates them into a single tensor and applies a linear transformation followed by a sigmoid

Input gate takes the previous output and the new input and passes them through another sigmoid layer. The internal state is then updated with the product of this weight and output of the candidate layer

Output gate controls how much of the internal state is passed to the output

Our Results

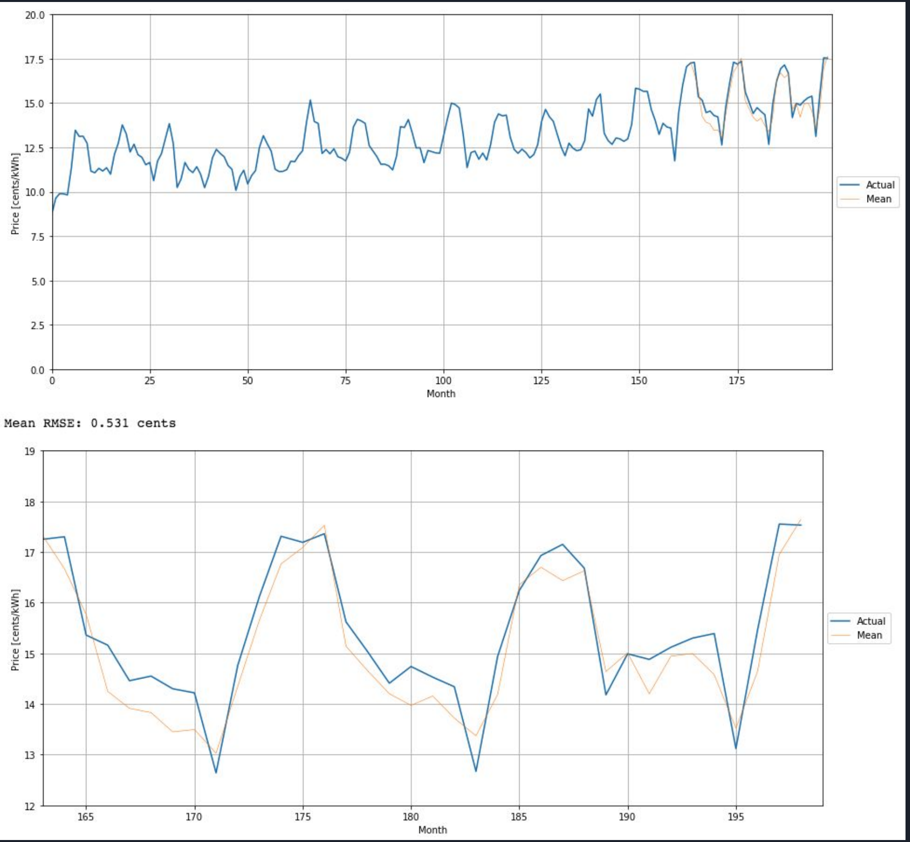
Predictions

Next Steps

Our Results: Predictions

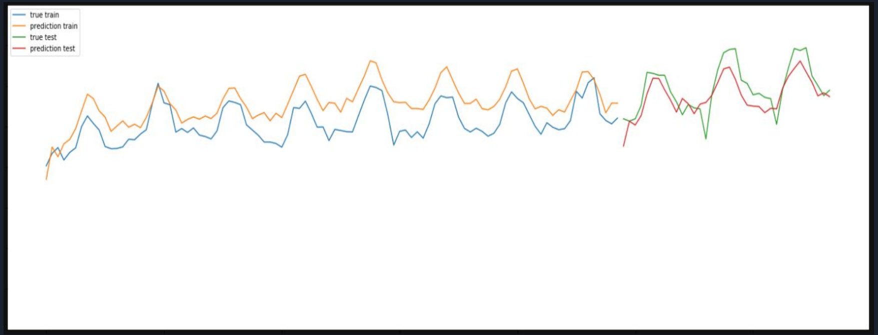
Rolling Window

RMSE **0.531**



RNN

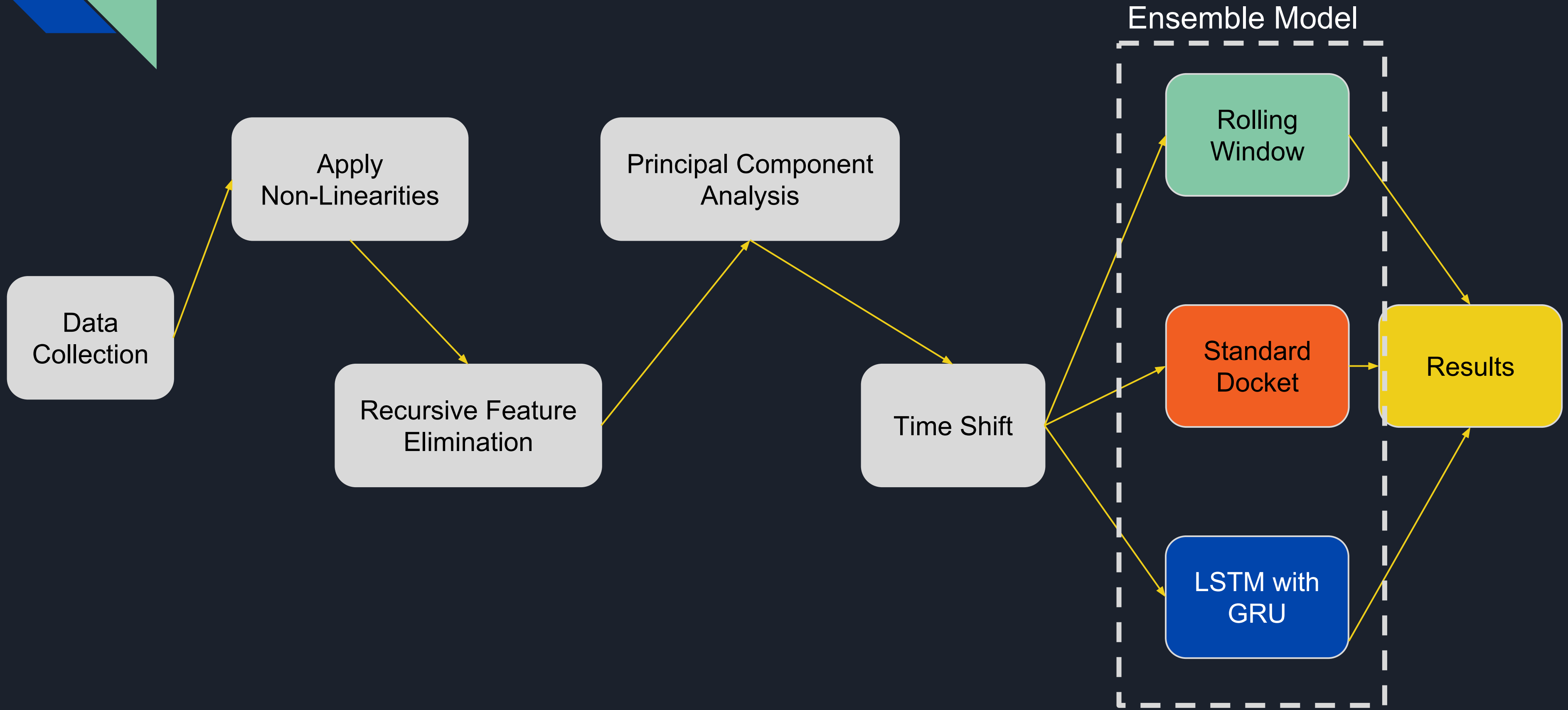
RMSE **0.877**



Our Results: Next Steps

- Combine work since a lot of our workflows were individual, and while there was some passing-off, not everything was completely integrated between our different feature selection methods and models
- Automated feature construction optimization
    - Compare performance of different feature combinations to optimize for best ones
    - Use PCA between feature construction steps to reduce dimensionality and speed up process
- Run models on servers with more computing power to optimize speeds
- Continue to add more features and retrain model on additional samples
- Try out the pipeline on other forecasting problems… stocks?

Learning Process: An Overview



Our Results: Next Steps

- Combine work since a lot of our workflows were individual, and while there was some passing-off, not everything was completely integrated between our different feature selection methods and models
- Automated feature construction optimization
    - Compare performance of different feature combinations to optimize for best ones
    - Use PCA between feature construction steps to reduce dimensionality and speed up process
- Run models on servers with more computing power to optimize speeds
- Continue to add more features and retrain model on additional samples
- Try out the pipeline on other forecasting problems… stocks?

What Did We Learn?

What Did We Learn?

- Your prediction is only as good as your features
- Minimizing computation time is a must
- Coordination between different parts of the pipeline is essential
- Data collection never ends!
- Interpolation/extrapolation of data needs care (in terms of number of datapoints/type of data etc)
- Saving results and parameters long-term, in an organized way prevents rework and allows for reproducibility.

Thanks!
Any Questions?

GITHUB REPO:
https://github.com/ericyehl/EPP-Feature-Modeling