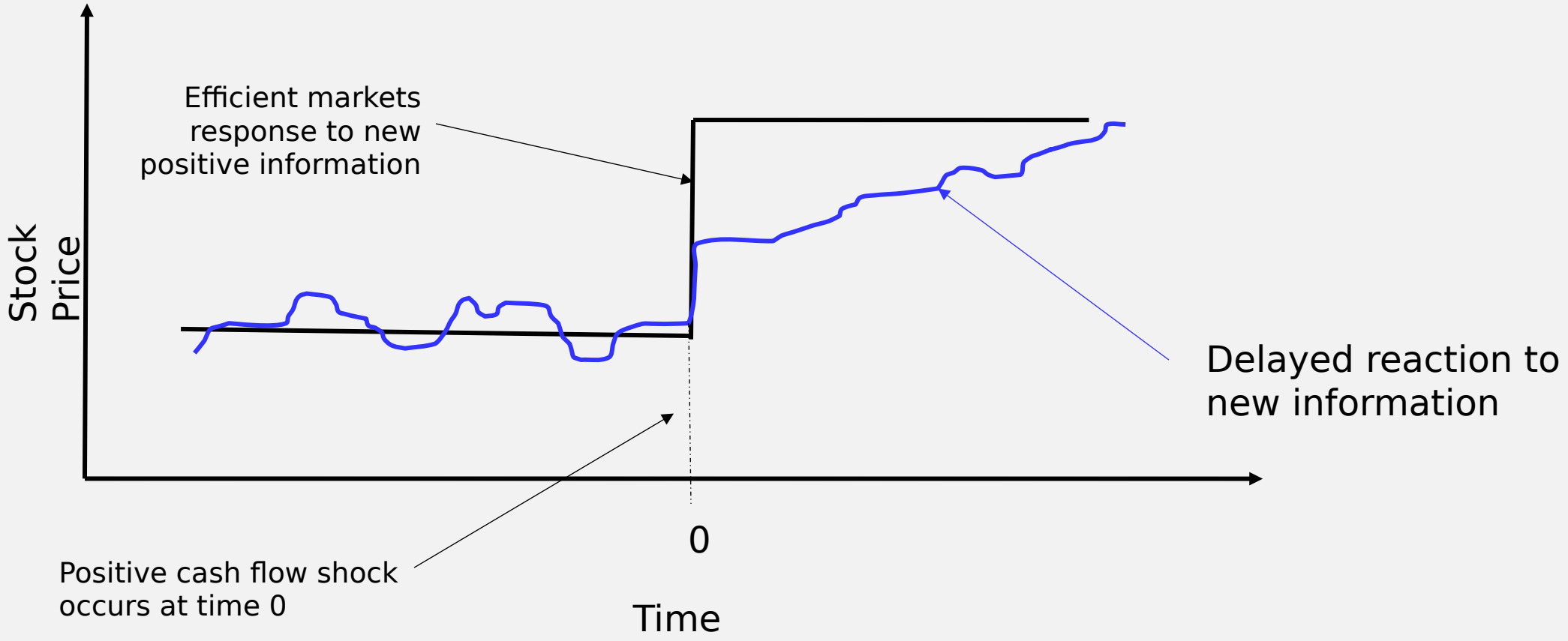
CASE STUDY:
PREDICTING INDUSTRY RETURNS USING MACHINE LEARNING

EVIDENCE OF SLOW-DIFFUSION OF INFORMATION ACROSS INDUSTRIES

- Hong, Torous, and Valkanov (2007):
  1. Investors with limited information-processing capabilities specialize in specific market segments.
  2. When a cash flow shock arises in a particular industry, information-processing limitations prevent investors specializing in other industries from understanding the full implications of the shock.
  3. Information diffuses gradually across industries
  4. The resulting delayed adjustment in equity prices gives rise to cross-industry return predictability.

GRADUAL DIFFUSION OF INFORMATION MIGHT GIVE RISE TO PROFITABLE TRADING OPPORTUNITIES

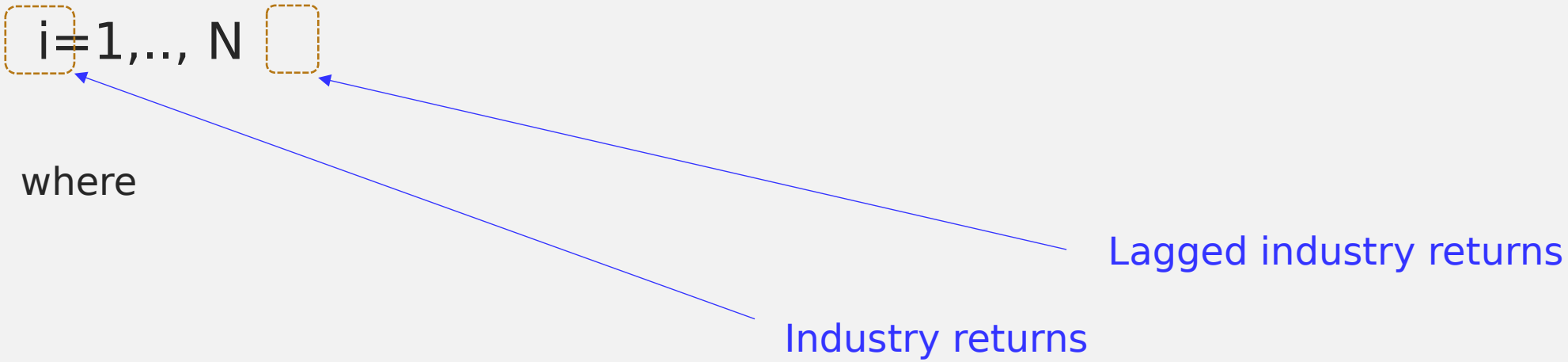


RAPACH ET AL. (2018) "CROSS-INDUSTRY RETURN PREDICTABILITY: A MACHINE LEARNING APPROACH"

- Use machine learning techniques to identify undervalued/overvalued industries

- The long/short trading strategy generates returns of over 8.4% per year with very little volatility!

PREDICTIVE REGRESSION FRAMEWORK



- $r_{i,t}$ is the month-t return on industry portfolio i in excess of the one-month T-bill return
- ι_T is a T-vector of ones,
- T is the usable number of monthly observations,
- N is the number of industry portfolios, and $\varepsilon_{i,t}$ is a zero-mean disturbance term.

ESTIMATION VIA ORDINARY LEAST SQUARES (OLS)

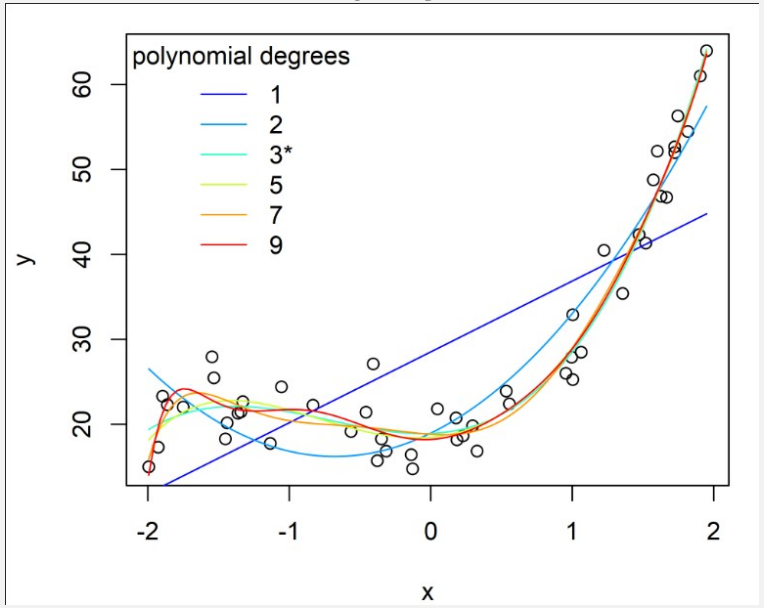
- We can estimate the predictive regression via OLS
- There are 30 lagged industry returns, so overfitting is a real concern
  - We may also want to add some additional independent variables
- An overfit regression model is too complex for the data
  - Overfitted models describe noise instead of any underlying relationship.
  - They generally have poor predictive performance on test data.

OVERFITTING/UNDERFITTING ILLUSTRATIVE EXAMPLE IN R

- Consider a 3rd degree polynomial of the form:

$$y = 3*x^3 + 5*x^2 + 0.5*x + 20 + \text{random noise}$$

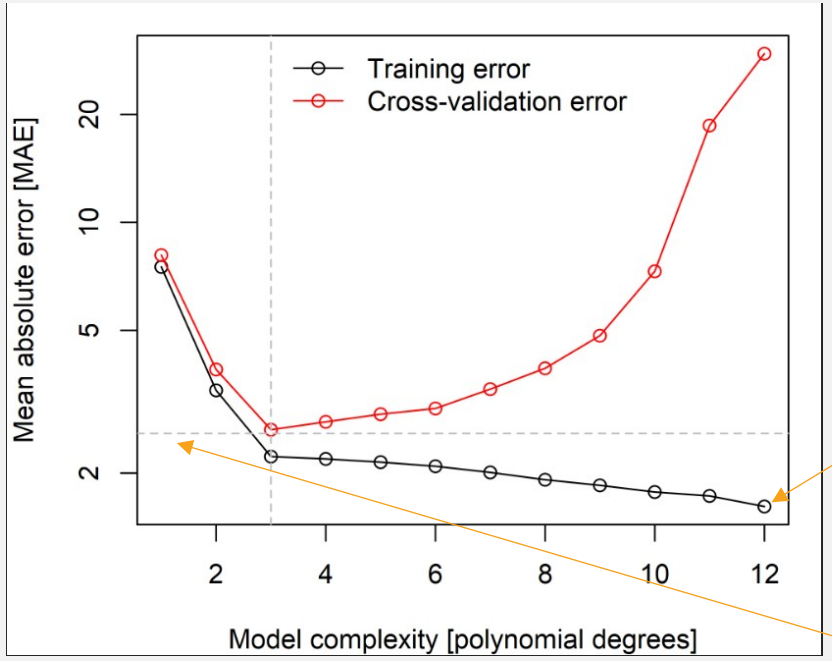
- Let's fit various polynomial models to this data



Notice that the model fit improves as we consider higher order polynomials

*Adapted from an R-bloggers post from 5/3/2014

OVERFITTING: TRAINING VERSUS TEST SETS



Lower MAE in the training set with the 12th degree polynomial, but much higher error in the test set

Not surprisingly the best model in the test set is the 3rd degree polynomial

THE LASSO

- Least Absolute Selection and Shrinkage Operator

$$(\hat{\alpha}, \hat{\beta}) = \arg\min \{ \sum_{i=1}^{N} (y_i - \alpha - \sum_j \beta_j x_{ij})^2 \} + \lambda \sum_j |\beta_j|$$

$$= \arg\min \| Y - X\beta \|_2^2 + \lambda \| \beta \|_1$$

Loss Penalty

LASSO, CONTINUED

- The tuning parameter λ controls the strength of the penalty
  - If λ=0, we get the OLS estimates $(\hat{\alpha}, \hat{\beta})$
  - If λ=∞, $\hat{\beta}$ are just 0
- For 0 < λ < ∞, we are balancing two ideas:
  1. fitting a linear model of y on X
  2. shrinking the coefficients.
- The nature of the penalty causes some coefficients to be shrunken to be zero exactly
- LASSO is able to perform variable selection in the linear model.
  - As λ increases, more coefficients are set to zero

LASSO OLS

- Fan and Li (2001): LASSO estimates of the coefficients for the selected predictors are downward biased
  - The LASSO penalty term overshrinks the coefficients for the selected predictors,
  - Effect is more severe for the most relevant predictors.

- Solution proposed by Belloni and Chernozhukov (2011, 2013):
  - Re-estimate the coefficients for the LASSO-selected predictors via OLS

DATA DESCRIPTION

- The data is from Ken French's data library:

  http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html

- Monthly data for 30 industries from July 1926 to present

- Consistent with Rapach et al. (2018), estimate the model using data from 1960 to 2018

DATA SNAPSHOT

Subset of the 30 Fama-French industries

Date	Food	Beer	Smoke	Games	Books	Hshld	Clths	Hlth	Chems	Txtls	Cnstr	Steel	FabPr	ElcEq
1/29/1960	-4.16	-5.38	-1.72	1.54	-5.14	-7.51	-8.20	-6.35	-9.70	-4.44	-6.34	-9.05	-4.09	-11.97
2/29/1960	3.64	-1.85	2.56	4.52	2.68	9.60	1.73	0.27	-0.45	0.61	3.36	-1.74	-1.84	5.40
3/31/1960	-1.32	-2.59	0.17	-0.30	2.53	-0.21	-2.24	1.61	-2.40	-6.44	-0.43	-5.23	-3.23	-0.54
4/29/1960	1.36	-1.97	1.54	6.65	-0.98	-1.08	0.40	1.68	-5.34	-0.91	-1.26	-5.34	-3.91	-0.92
5/31/1960	8.47	-0.25	2.71	7.55	11.94	8.01	2.01	13.77	3.67	2.37	3.45	2.62	1.19	6.75
6/30/1960	5.63	0.71	4.97	2.48	0.26	6.62	-1.35	-0.16	0.69	4.28	2.78	1.56	1.87	0.04
7/29/1960	-1.98	-0.66	4.73	-4.59	0.36	-0.47	-0.97	-3.86	-6.67	-3.01	-5.04	-2.25	-3.76	-6.83
8/31/1960	4.74	3.41	5.37	7.33	3.80	5.26	3.51	2.46	1.34	-0.67	0.74	0.80	0.97	-0.87
9/30/1960	-3.72	-4.84	-1.93	-2.17	-6.04	-9.02	-4.07	-8.71	-6.54	-5.09	-4.87	-8.08	-5.53	-10.30
10/31/1960	1.24	0.76	4.09	0.33	2.60	6.70	-3.28	-3.49	-1.37	-2.84	-1.19	-2.13	-1.20	-3.36
11/30/1960	9.59	6.70	5.57	14.04	10.24	9.26	3.28	4.04	4.38	2.17	6.75	1.40	8.41	5.35
12/30/1960	4.67	-0.15	3.70	7.93	7.57	1.92	3.44	6.22	3.01	0.68	4.43	4.95	4.95	1.18
1/31/1961	4.89	5.42	8.96	0.75	9.66	4.55	6.13	6.05	6.65	11.40	5.85	8.49	6.14	-0.40
2/28/1961	4.35	8.30	5.55	22.47	2.29	6.04	7.98	5.19	2.27	6.95	4.93	5.88	9.17	-0.15

Monthly return data

LASSOOLS

- LassoOLS in R:
  - LassoOLS computes the two-stage estimator Lasso+OLS (default) or the Lasso estimator (if OLS=FALSE).
  - Part of the HDCI library
  - Format:
  - LassoOLS(x, y, OLS = TRUE, lambda = NULL, fix.lambda = TRUE, cv.method = "cv", nfolds= 10, foldid, cv.OLS = TRUE, tau = 0, parallel = FALSE, standardize = TRUE, intercept = TRUE, ...)
  - Use mypredict(obj, newx) to generate the forecast
- LassoOLS in Python: sklearn-Lasso, LassoCV

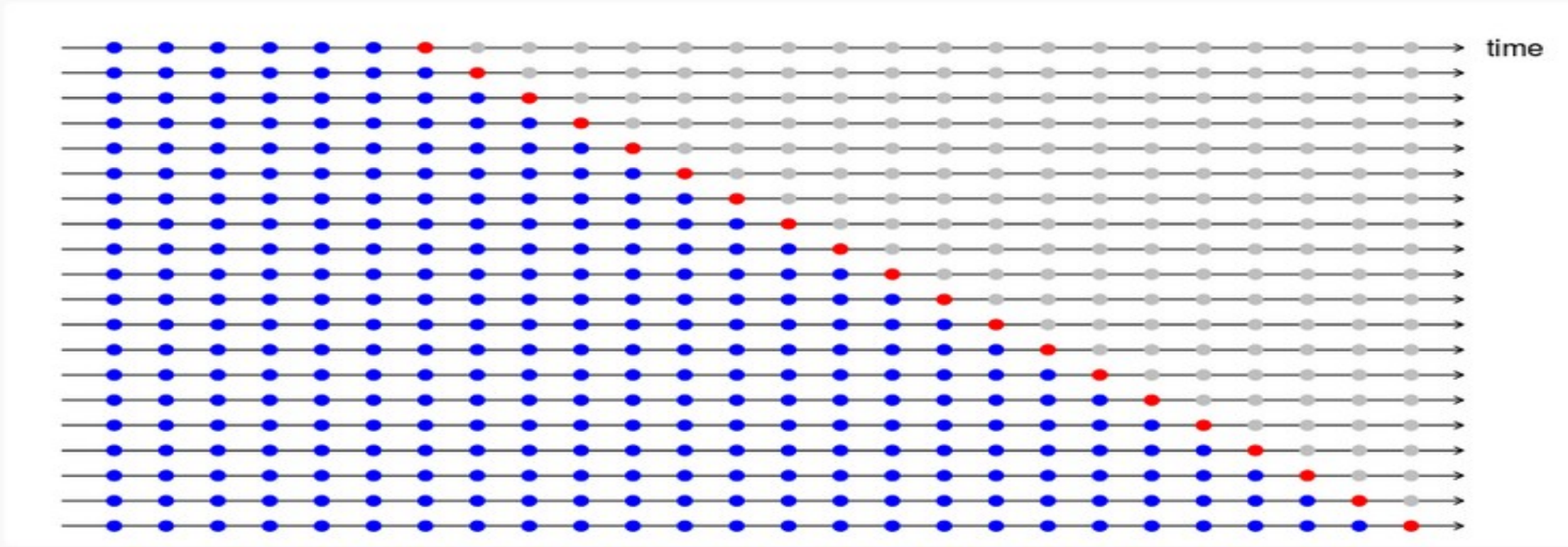
ESTIMATING THE MODEL: TIME-SERIES CROSS VALIDATION

- The approach in Rapach et al (2018):
  - Use data from 1960:01 through 1969:12 to estimate the predictive regression for each industry via Lasso
  - Generate a set of 30 out-of-sample industry excess return forecasts for 1970:01.
  - Sort the industries in ascending order according to the excess return forecasts and form equal-weighted quintile portfolios
  - Create a zero-investment portfolio that goes long (short) the top (bottom) quintile portfolio.

- Repeat the process using data from 1960:01 to 1970:01 to compute an updated set of out-of-sample industry excess return forecasts for 1970:02
  - Create a zero-investment portfolio that goes long (short) the top (bottom) quintile portfolio.

- Continue this process through 2018:06

GRAPHICAL DEPICTION OF THE ESTIMATION STRATEGY



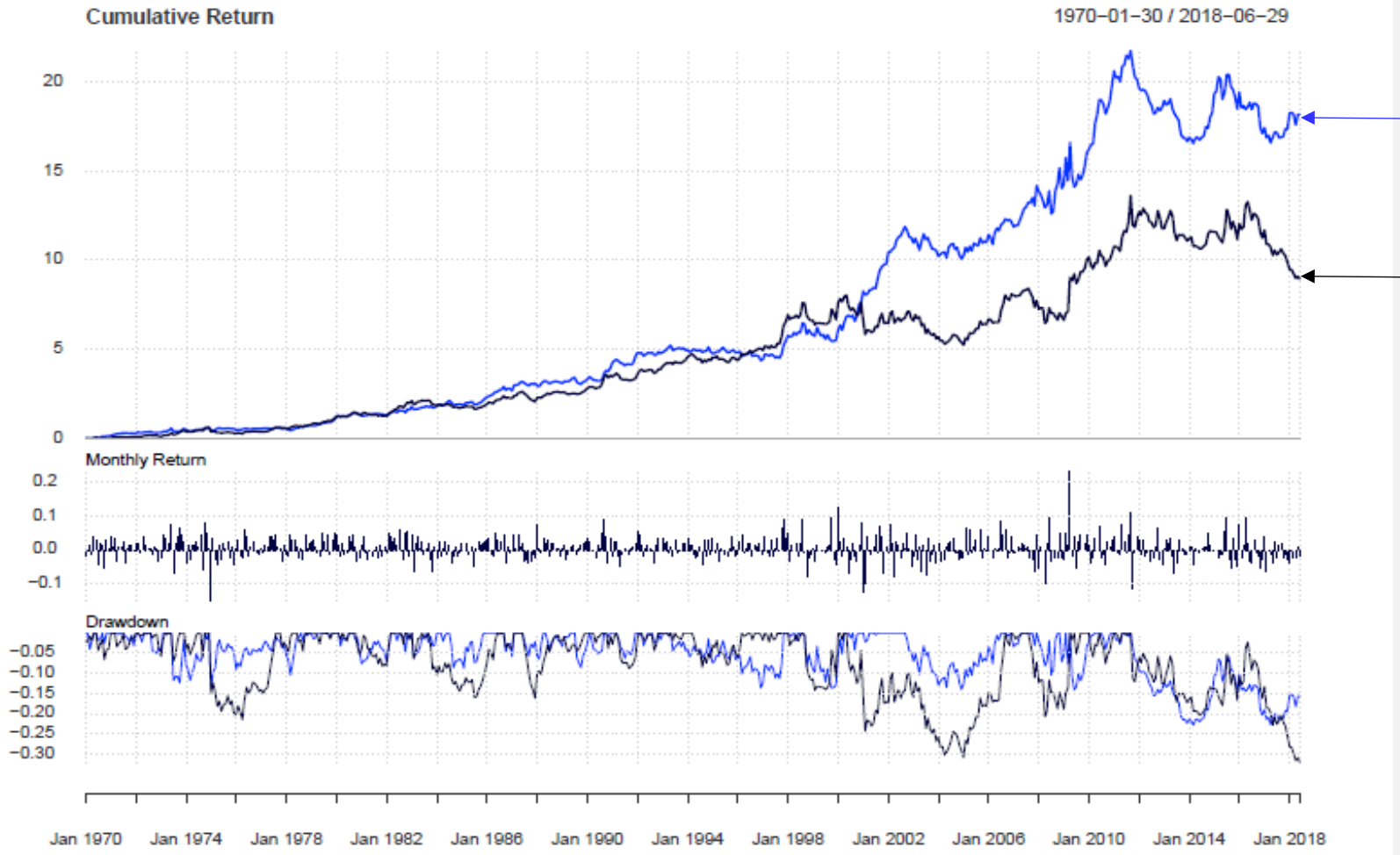
The blue observations form the training sets, and the red observations form the test sets.

THE INDUSTRY TRADING STRATEGY

- At the end of each month, sort the 30 industry portfolios according to out-of-sample forecasts of their excess returns for the subsequent month.

- Then form equal-weighted quintile portfolios based on the sorts, and each long-short industry-rotation portfolio is a zero-investment portfolio that goes long (short) the top (bottom) quintile portfolio.

TRADING STRATEGY PERFORMANCE: OLS VS. LASSO FORECASTS



LassoOLS Forecasts

OLS Forecasts

TRADING STRATEGY DESCRIPTIVE STATISTICS

Monthly descriptive data

	OLS Forecasts	Lasso Forecasts
Observations	582	582
Minimum	-15.00%	-13.80%
Quartile 1	-1.60%	-1.20%
Median	0.40%	0.40%
Arithmetic Mean	0.50%	0.60%
Geometric Mean	0.40%	0.50%
Quartile 3	2.20%	2.10%
Maximum	23.20%	13.80%
Stdev	3.40%	3.00%

Annualized performance measures

Variable	Formula	OLS Forecasts	Lasso Forecasts
Annualized return	Monthly return*12	6.00%	7.20%
Annualized SD	Monthly SD*sqrt(12)	11.78%	10.39%
Sharpe Ratio	Annualized return/Annualized SD	0.509	0.693

DO STYLE FACTORS EXPLAIN THE RETURNS TO THE STRATEGY?

	Dependent variable:	
	OLS Forecasts	LassoOLS Forecasts
	(1)	(2)
Mkt_rf	-0.065*	-0.121***
	(0.034)	(0.029)
SMB	-0.104**	-0.069*
	(0.047)	(0.041)
HML	-0.070	-0.145***
	(0.051)	(0.044)
Mom	0.006	0.032
	(0.033)	(0.029)
Constant	0.005***	0.007***
	(0.001)	(0.001)
Observations	582	582
R^2	0.020	0.055
Adjusted R^2	0.013	0.049
Residual Std. Error (df = 577)	0.033	0.029
F Statistic (df = 4; 577)	2.969**	8.408***
Note:		*p**p***p<0.01

Traditional style factors don't explain the anomaly. Adj. R^2 is less than 0.05

Alpha of 70 bps per month or 8.4% annualized!

WHY DOES LASSO DO SO MUCH BETTER THAN OLS?

- Prediction Accuracy
  - Assume $y = f(x) + \varepsilon$, and $E(\varepsilon) = 0 \quad \text{var}(\varepsilon) = \sigma^2$

    then the prediction error of the estimate $\hat{f}(x)$ is:

$$
\begin{aligned}
Err(x) &= E[(y - \hat{f}(x))]^2 \\
&= \sigma^2 + [E\hat{f}(x) - f(x)]^2 + E[\hat{f}(x) - E\hat{f}(x)]^2 \\
&= \sigma^2 + bias^2(\hat{f}(x)) + \text{var}(\hat{f}(x))
\end{aligned}
$$

  - OLS estimates have low bias but large variance (typically)
  - **Lasso improves the overall prediction accuracy by sacrificing bias to reduce the variance of the predicted value.**

A FEW CAVEATS ON THE RESULTS

- The results look good, but:
  1. The Fama-French industry portfolios are not tradeable
  - Could we actually capture these returns?
  2. We haven’t factored in transaction costs
  - Is portfolio turnover high in this strategy?
  3. Machine learning is short on economic theory
  - Why are returns in the construction industry impacted by lagged industry returns in clothing industry?

CAN WE IMPROVE UPON THESE FINDINGS?

- At least, two types of data could potentially improve the model performance:
  - Industry momentum
  - Macroeconomic data

RANDOM FORESTS

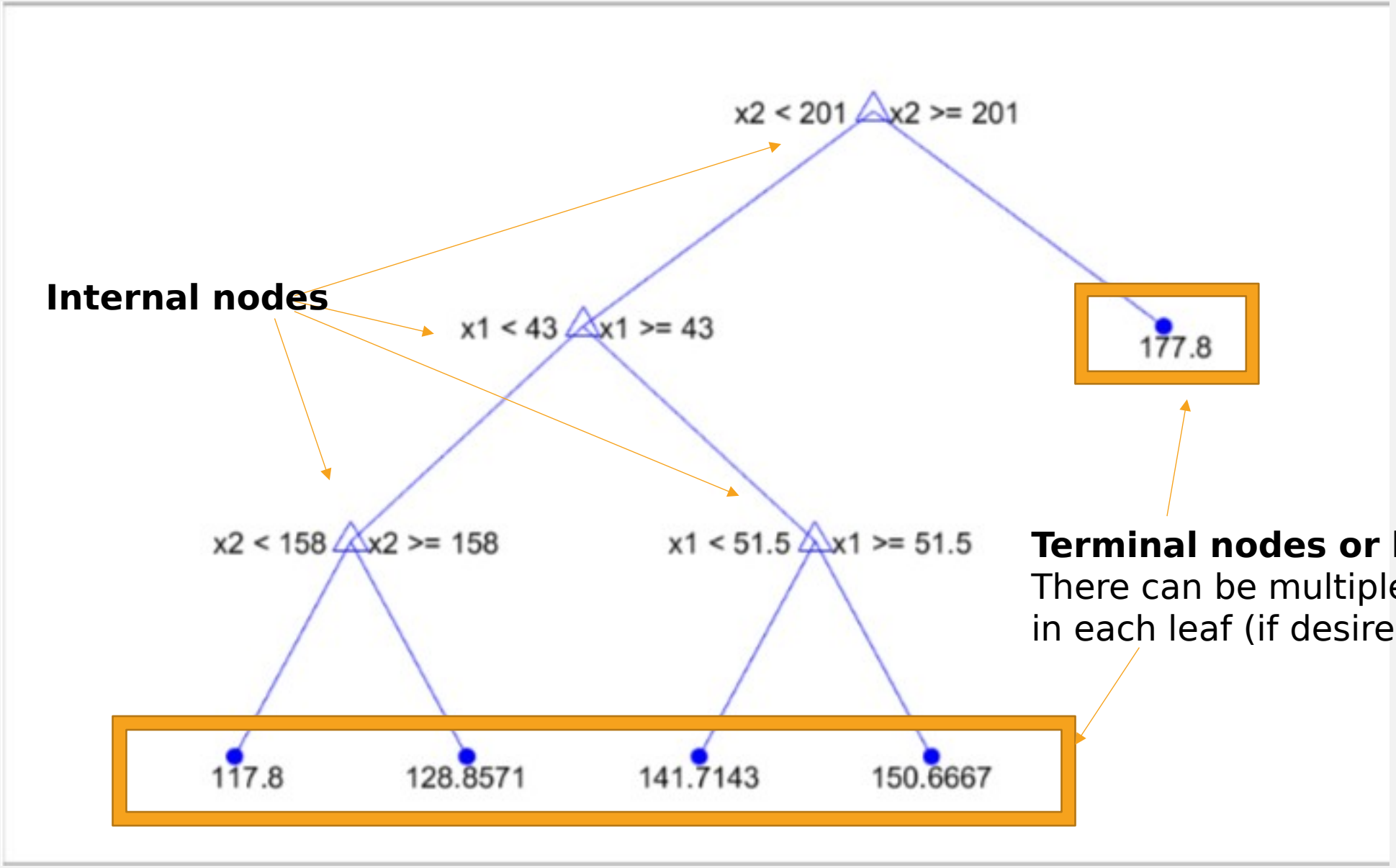
RANDOM FORESTS

- Random Forest Regressions are an “ensemble” algorithm that estimates a regression-type relationship between a numerical dependent variable and set of predictors.

- An ensemble of Regression Trees
  - Similar to taking the average over large number of randomly selected regression trees

REGRESSION TREE: BLOOD PRESSURE EXAMPLE

- Regression trees often used in biological and health sciences.
- As an illustrative example let’s look at predicting blood pressure:
- Regression: $y = X \beta$
  y = systolic blood pressure, X_1 = Age, X_2 = Weight
- Whereas linear regression linearly projects dependent variable onto the space spanned by predictors, regression trees iteratively split the data in order to minimize mean squared errors as follows:



Internal nodes

Terminal nodes or leaves
There can be multiple observation
in each leaf (if desired)

REGRESSION TREE

- Starting with the set of data (“the trunk”), search over all explanatory variables and split values to determine which (variable, value) combination minimizes mean squared error the most.
  - Error: the difference between fitted values (average within each partitioned group) and actual dependent variable values
- Once this optimal partition has been accomplished, the next iteration searches over all branches and chooses the single partition from among the entire set of branches to split just one of these branches in such a way that reduces the overall sum of squared errors the most.
- This process is repeated until the desired number of leaves is attained.

RANDOM DECISION FORESTS

- A regression tree tends to “over-fit” the data
  - Poor performance out-of-sample
  - Low bias, high variance

- Random decision forests:
  - 1. Bagging: select random samples of the training data (usually with replacement) and fit a decision tree
    - With sklearn the sub-sample size is the same as the original sample size
  - 2. At each split on the tree select a random subset of candidate features (sometimes called “feature bagging”)
    - Typically, with p features, randomly select
  - 3. We now have a “forest” of trees and can average over them to calculate predicted values
  - This reduces variance at the expense of a small increase in bias

MAKING PREDICTIONS

- How do we make a prediction $\hat{y}_i$ for an observation x_i?
  - 1. For each tree, we find the relevant leaf given the features of x_i
    - Each leaf is the set of observations that “survived” or “failed” the same splits
  - 2. Calculate the average y value for observations in that leaf $\hat{y}_i$
  - 3. Average these values across trees to get the predicted