MOOC Lab 5: Financial Market Regime Prediction with Machine Learning

Princeton University

February 2021

Outline

- Regime-based Models in Finance
- Regime Identification
  - Trend Filtering
- Machine Learning in Finance
  - Overview of Machine Learning Methods
    - Supervised Learning: Regression and Classification
    - Unsupervised Learning
  - Applications in Economics and Finance
  - Classification Methods
    - Logistic Regression, Decision Trees and Ensemble Methods
- Machine Learning Approach in Regime Prediction
  - Dataset description
  - Model framework for regime prediction
  - Results
- References

Why are Regime-Based Models used in Finance?

- Idea of regime change is **natural and intuitive**
- Regimes identified by econometric methods often correspond to **different periods in regulation, policy and other secular changes**
  - interest rate behavior markedly changed from 1979 through 1982
  - in equities, different regimes correspond to periods of high and low volatility, and bull and bear market periods
- Regime switching models can **capture stylized behavior of many financial series**
  - including fat tails
  - persistently occurring periods of turbulence followed by periods of low
  - skewness
  - time-varying correlations
- Regime switching models started to appear in finance literature in 1989
- For a recent overview of the concept
  - Ang and Timmerman (2011) "Regime Changes and Financial Markets"

Trend Filtering

- We are given a time series $\boldsymbol{y_t}, t = 1, \ldots, n$, assumed to consist of an underlying slowly varying trend $\boldsymbol{x_t}$ and a more rapidly randomly varying component $\boldsymbol{z_t}$, which is equal to $\boldsymbol{z_t} = \boldsymbol{y_t} - \boldsymbol{x_t}$
- **Goal:** To estimate the underlying trend in time series data, $\boldsymbol{x_t}$
- Optimization problem formulation for l_1- trend filtering for $y \in \mathrm{R}^n$

$$\hat{x} = \underset{\mathrm{x} \in R^n}{\operatorname{argmin}} \tfrac{1}{2}||y - x||_2^2 + \lambda||Dx||_1$$

where $\lambda \geq 0$ is the regularization parameter and $D \in R^{(n-1)\times n}$ is the first order difference matrix

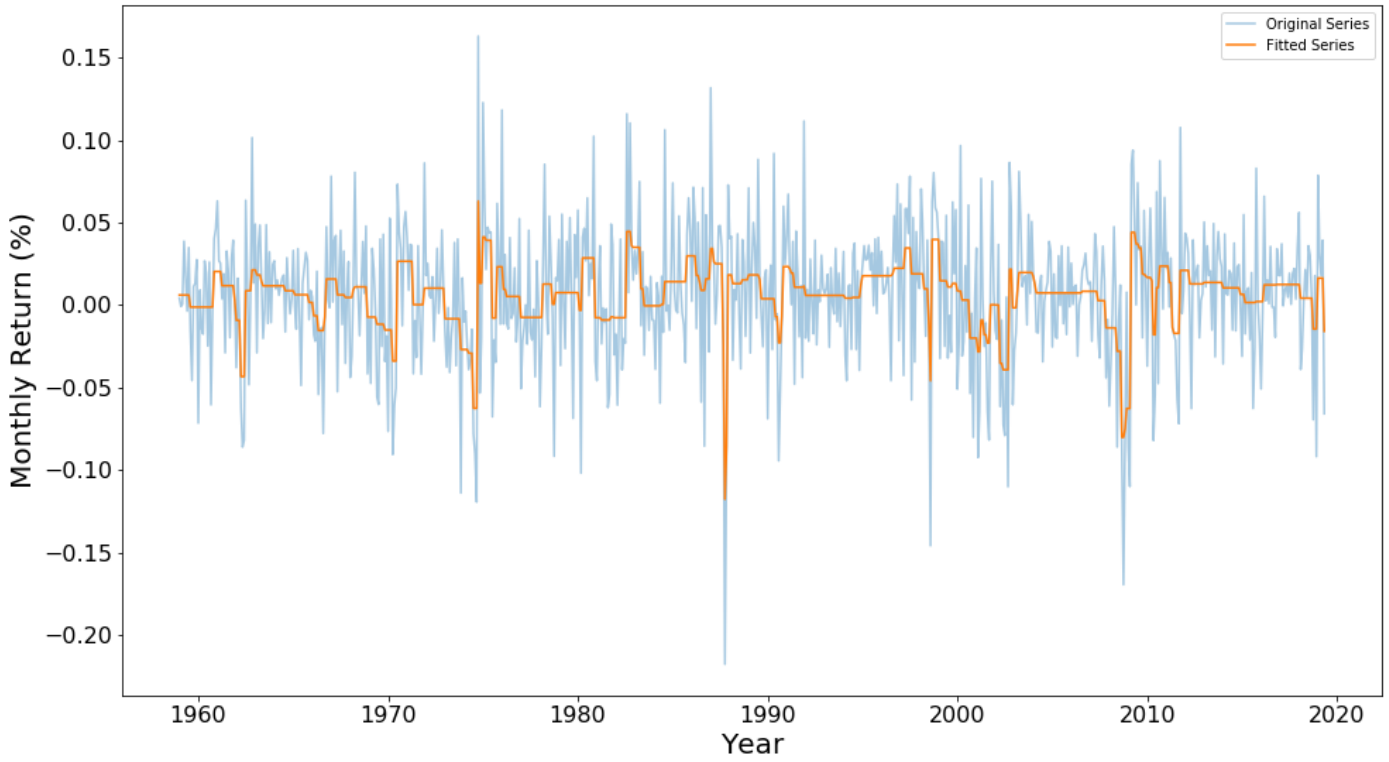
$$D = \begin{bmatrix} 1 & -1 & 0 & \ldots & 0 & 0 \\ 0 & 1 & -1 & \ldots & 0 & 0 \\ . & . & . & & . & . \\ . & . & . & & . & . \\ 0 & 0 & 0 & \ldots & 1 & -1 \end{bmatrix} \in \mathbb{R}^{(n-1)\times n}$$

Properties of l_1- Trend Filtering Estimator

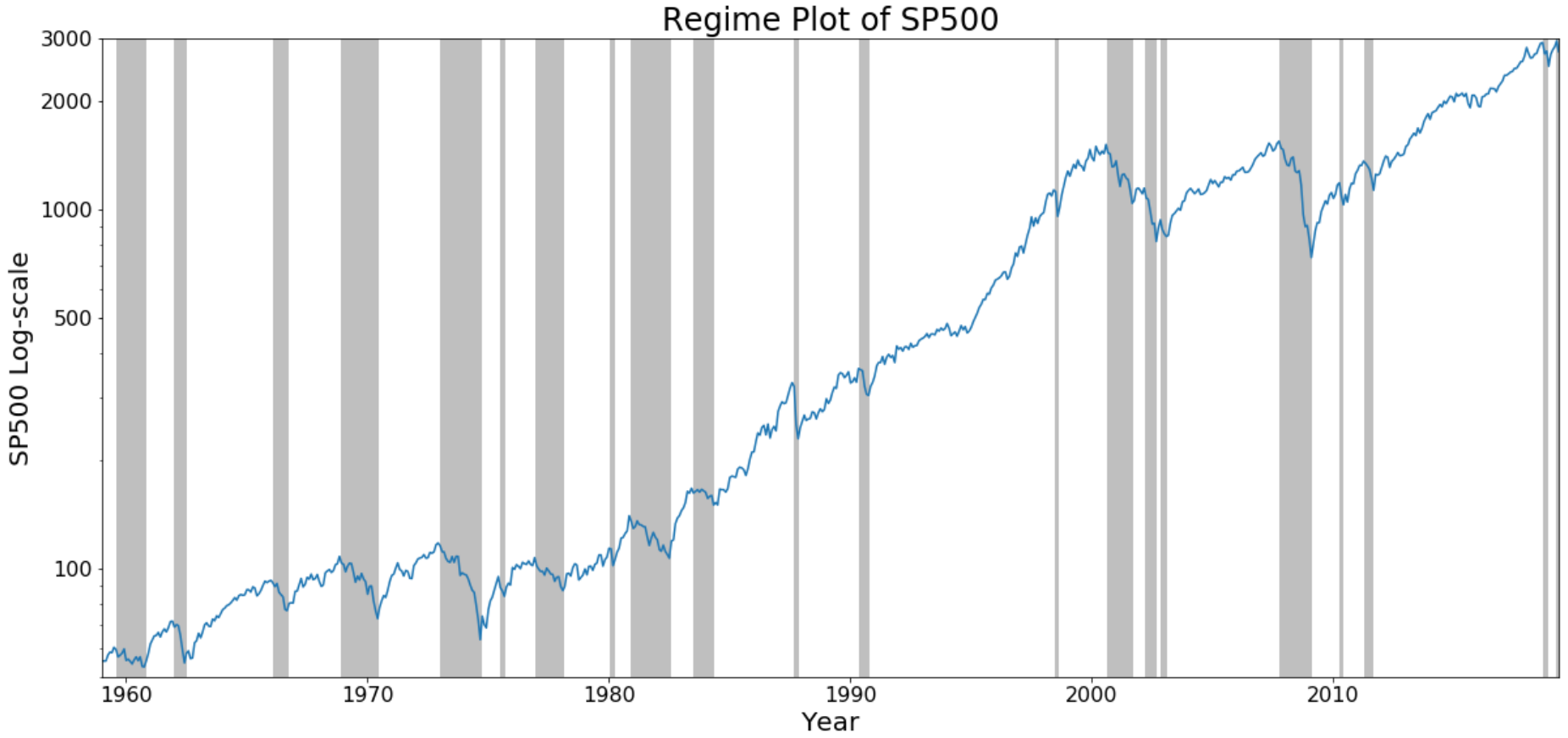
- **Convex optimization** problem
- The output will be a **piece-wise constant time series** which can be conveniently identified as a regime
- It is a **non-parametric regression model** which means it doesn't have a fixed structure of model and the nature of the parameters can be quite flexible.
- The estimator converges to **original data** y as $\lambda \to 0$, and converges to the **best constant fit** (i.e. the estimator is constant throughout the whole time) as $\lambda \to \infty$

l_1 - Trend Filtering on S&P500 Returns

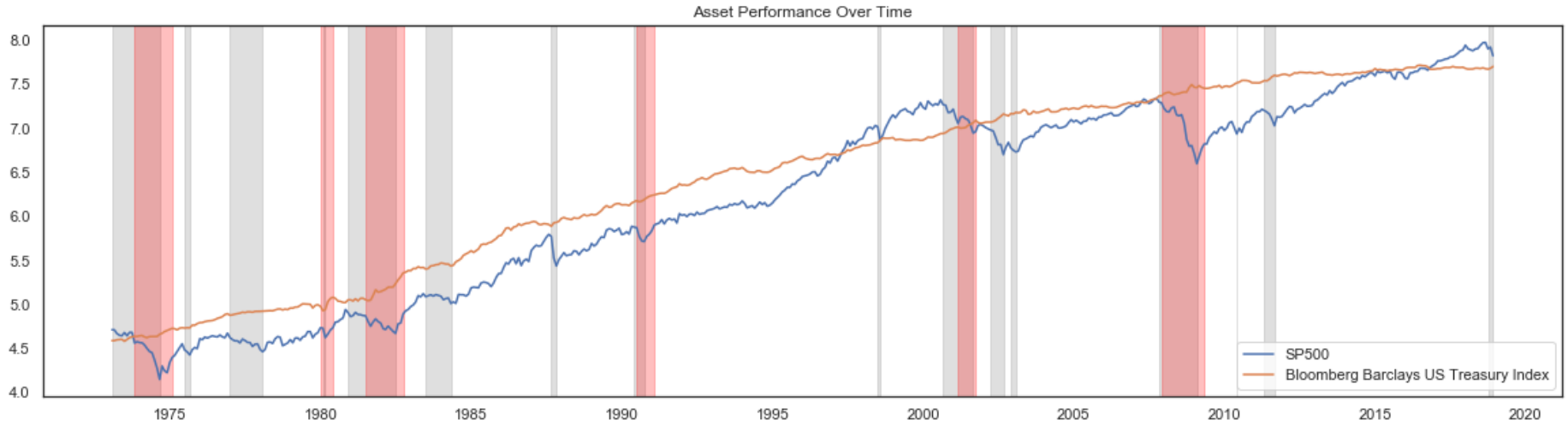
- Input: monthly S&P500 return data from 1959 until 2020



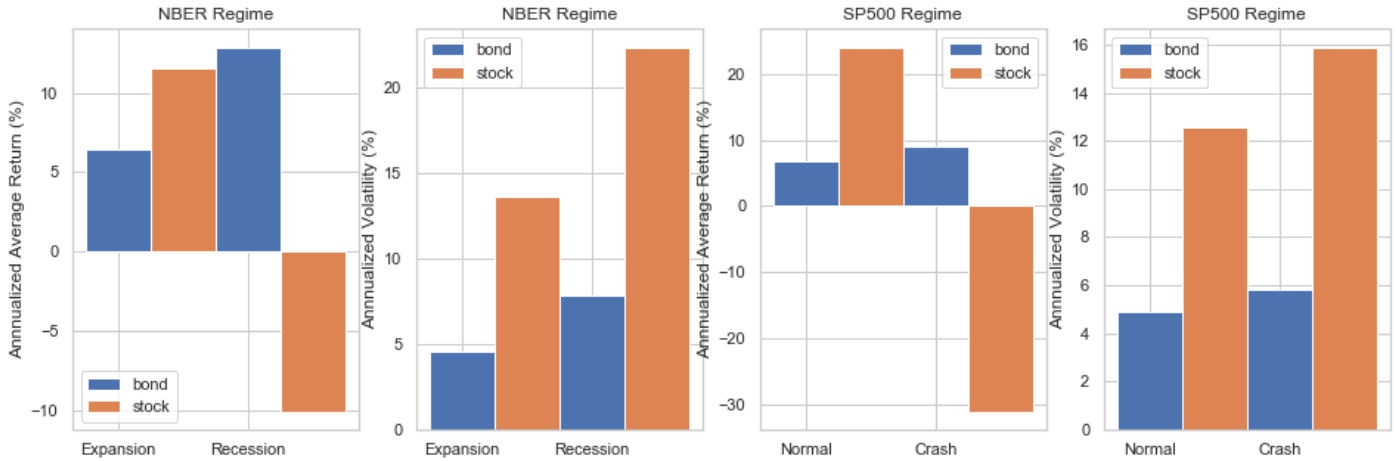
l_1 - Trend Filtering on S&P500 Returns



Asset Performances over Different Regimes



Red areas denote **recession periods** and gray areas show **stock market crashes** identified by trend filtering algorithm



Machine Learning - Overview

Supervised Learning

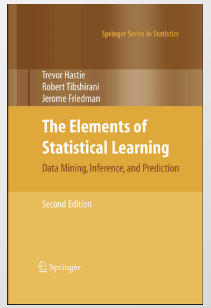
use an algorithm to learn the mapping function from the input (x) to the output (y)

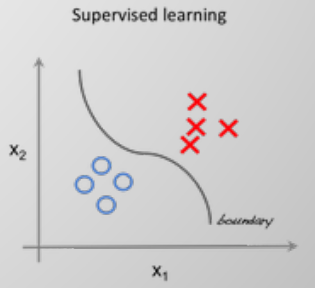
- Regression – continuous output
  - Linear Regression and it's cousins (LASSO, Ridge, ElasticNet, Best-subset)
  - Tree based models (Random Forest, Boosting Trees)
- Classification – categorical output
  - Logistic Regression
  - Tree based models (Random Forest, Boosting Trees)
  - Support Vector Machines

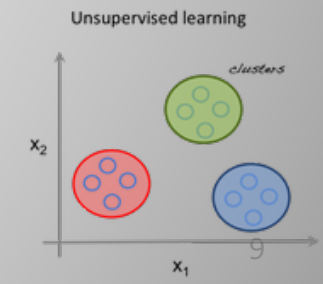
Unsupervised Learning

model the underlying structure or distribution in the data (x)

- Clustering – want to discover inherent groupings in the data
  - k-means
- Anomaly detection – fraud detection
- Principal Component Analysis







Machine Learning Applications in Finance and Economics

- Recent papers and books on machine learning in econ and finance

Journal of Economic Perspectives—Volume 28, Number 2—Spring 2014—Pages 3–28

Big Data: New Tricks for Econometrics[1]

Hal R. Varian

Computers are now involved in many economic transactions and can capture data associated with these transactions, which can then be manipulated and analyzed. Conventional statistical and econometric techniques such as regression often work well, but there are issues unique to big datasets that may require different tools.

First, the sheer size of the data involved may require more powerful data manipulation tools. Second, we may have more potential predictors than appropriate for estimation, so we need to do some kind of variable selection. Third, large datasets may allow for more flexible relationships than simple linear models. Machine learning techniques such as decision trees, support vector machines, neural nets, deep learning, and so on may allow for more effective ways to model complex relationships.

In this essay, I will describe a few of these tools for manipulating and analyzing big data. I believe that these methods have a lot to offer and should be more widely known and used by economists. In fact, my standard advice to graduate students these days is go to the computer science department and take a class in machine learning. There have been very fruitful collaborations between computer scientists and statisticians in the last decade or so, and I expect collaborations between computer scientists and econometricians will also be productive in the future.

■ *Hal Varian is Chief Economist, Google Inc., Mountain View, California, and Emeritus Professor of Economics, University of California, Berkeley, California. His email address is hal@ischool.berkeley.edu.*

[1] To access the Appendix and disclosure statements, visit http://dx.doi.org/10.1257/jep.28.2.3

doi=10.1257/jep.28.2.3

Journal of Economic Perspectives—Volume 31, Number 2—Spring 2017—Pages 87–106

Machine Learning: An Applied Econometric Approach

Sendhil Mullainathan and Jann Spiess

Machines are increasingly doing "intelligent" things: Facebook recognizes faces in photos, Siri understands voices, and Google translates websites. The fundamental insight behind these breakthroughs is as much statistical as computational. Machine intelligence became possible once researchers stopped approaching intelligence tasks procedurally and began tackling them empirically. Face recognition algorithms, for example, do not consist of hard-wired rules to scan for certain pixel combinations, based on human understanding of what constitutes a face. Instead, these algorithms use a large dataset of photos labeled as having a face or not to estimate a function $f(x)$ that predicts the presence y of a face from pixels x. This similarity to econometrics raises questions: Are these algorithms merely applying standard techniques to novel and large datasets? If there are fundamentally new empirical tools, how do they fit with what we know? As empirical economists, how can we use them?[1]

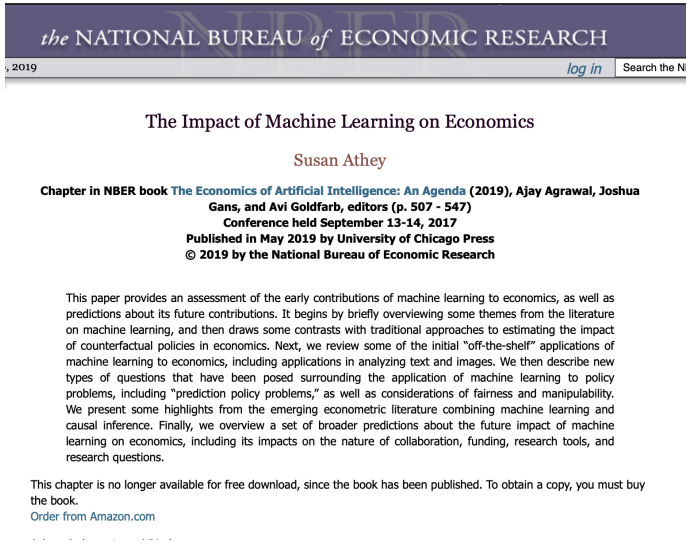
We present a way of thinking about machine learning that gives it its own place in the econometric toolbox. Central to our understanding is that machine learning

[1] In this journal, Varian (2014) provides an excellent introduction to many of the more novel tools and "tricks" from machine learning, such as decision trees or cross-validation. Einav and Levin (2014) describe big data and economics more broadly. Belloni, Chernozhukov, and Hansen (2014) present an econometrically thorough introduction on how LASSO (and close cousins) can be used for inference in high-dimensional data. Athey (2015) provides a brief overview of how machine learning relates to causal inference.

■ *Sendhil Mullainathan is the Robert C. Waggoner Professor of Economics and Jann Spiess is a PhD candidate in Economics, both at Harvard University, Cambridge, Massachusetts. Their email addresses are mullain@fas.harvard.edu and jspiess@fas.harvard.edu.*

[1] For supplementary materials such as appendices, datasets, and author disclosure statements, see the article page at https://doi.org/10.1257/jep.31.2.87

doi=10.1257/jep.31.2.87



the NATIONAL BUREAU of ECONOMIC RESEARCH

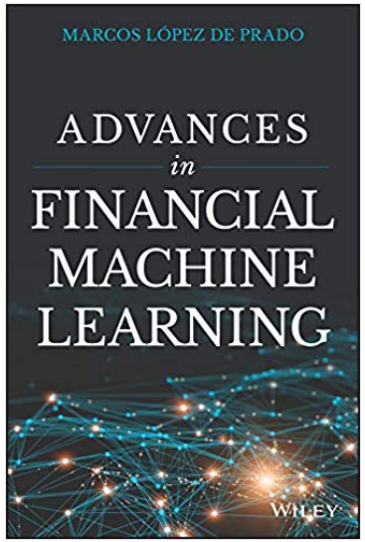
2019 log in Search the N

The Impact of Machine Learning on Economics

Susan Athey

Chapter in NBER book The Economics of Artificial Intelligence: An Agenda (2019), Ajay Agrawal, Joshua Gans, and Avi Goldfarb, editors (p. 507 - 547)
Conference held September 13-14, 2017
Published in May 2019 by University of Chicago Press

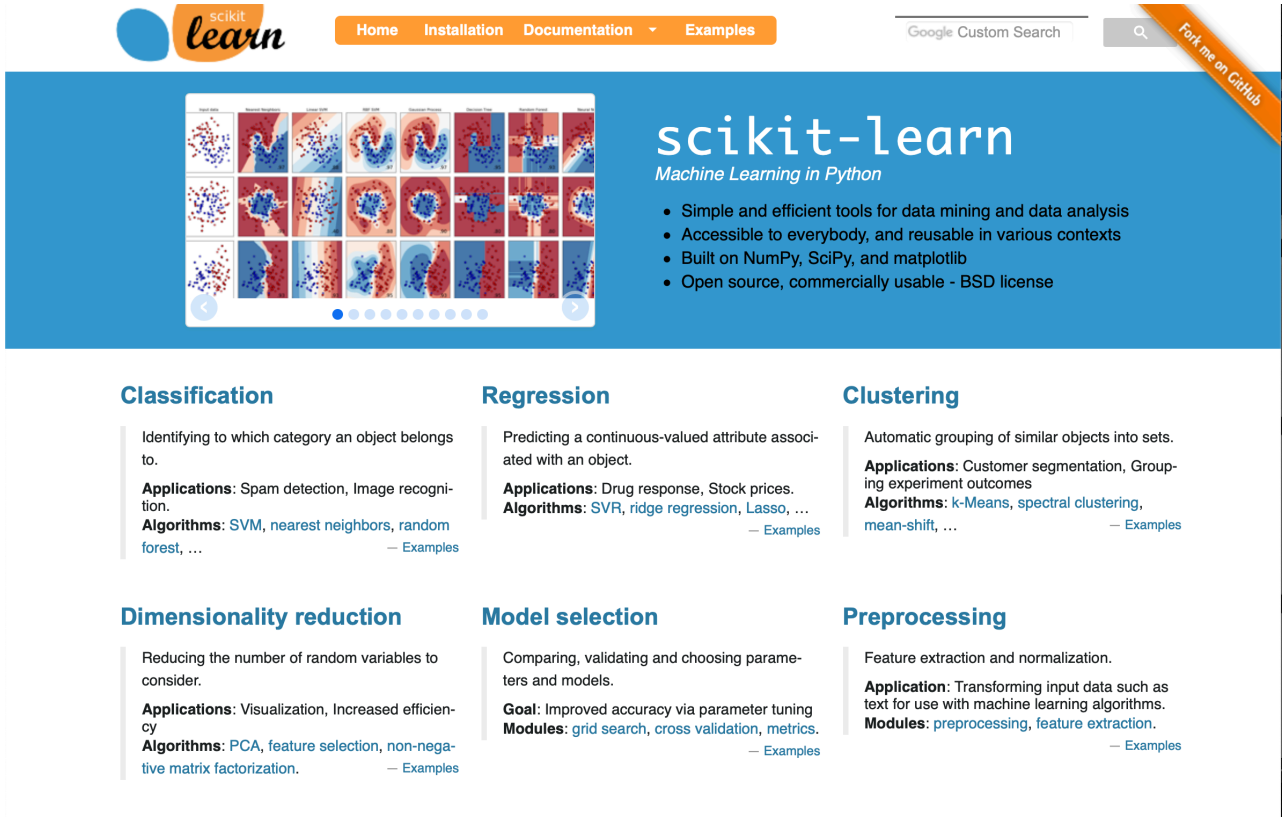
This paper provides an assessment of the early contributions of machine learning to economics, as well as predictions about its future contributions. It begins by briefly overviewing some themes from the literature on machine learning, and then draws some contrasts with traditional approaches to estimating the impact of counterfactual policies in economics. Next, we review some of the initial "off-the-shelf" applications of machine learning to economics, including applications in analyzing text and images. We then describe new types of questions that have been posed surrounding the application of machine learning to policy problems, including "prediction policy problems," as well as considerations of fairness and manipulability. We present some highlights from the emerging econometric literature combining machine learning and causal inference. Finally, we overview a set of broader predictions about the future impact of machine learning on economics, including its impacts on the nature of collaboration, funding, research tools, and research questions.



- Various applications in finance
  - factor models, asset pricing, regime prediction, credit default prediction, news sentiment analysis

Machine Learning – Software Package in Python

- scikit-learn is the machine learning library in Python
- Includes many regression and classification algorithms
- sklean.datasets package contains some small toy datasets



Example: Linear Regression in Scikit-Learn

- Diabetes dataset
- n=442 diabetes patients
- 10 features: age, sex, body mass index, average blood pressure, and six blood serum measurements
- Target: a quantitative measure of disease progression

In [4]:
```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from sklearn import datasets, linear_model
from sklearn.metrics import mean_squared_error, r2_score
```

In [49]:
```
# Load the diabetes dataset
diabetes = datasets.load_diabetes() #returns a dictionary-like object
type(diabetes)
```

Out[49]:
```
sklearn.utils.Bunch
```

In [50]:
```
X=diabetes.data
y=diabetes.target
df=pd.DataFrame(diabetes.data,columns=diabetes.feature_names)
df['Target']=y
df.head()
```

Out[50]:

	age	sex	bmi	bp	s1	s2	s3	s4	s5	s6	Target
0	0.038076	0.050680	0.061696	0.021872	-0.044223	-0.034821	-0.043401	-0.002592	0.019908	-0.017646	151.0
1	-0.001882	-0.044642	-0.051474	-0.026328	-0.008449	-0.019163	0.074412	-0.039493	-0.068330	-0.092204	75.0
2	0.085299	0.050680	0.044451	-0.005671	-0.045599	-0.034194	-0.032356	-0.002592	0.002864	-0.025930	141.0
3	-0.089063	-0.044642	-0.011595	-0.036656	0.012191	0.024991	-0.036038	0.034309	0.022692	-0.009362	206.0
4	0.005383	-0.044642	-0.036385	0.021872	0.003935	0.015596	0.008142	-0.002592	-0.031991	-0.046641	135.0

In [76]:
```
# Use only one feature - bmi
diabetes_X = diabetes.data[:, np.newaxis, 2]
# Split the data into training/testing sets
diabetes_X_train = diabetes_X[:-20]
diabetes_X_test = diabetes_X[-20:]

# Split the targets into training/testing sets
diabetes_y_train = diabetes.target[:-20]
diabetes_y_test = diabetes.target[-20:]
```

In [80]:
```
print('Size of Training Set: ' +str(diabetes_X_train.shape[0]))
print('Size of Test Set: ' +str(diabetes_X_test.shape[0]))
```

```
Size of Training Set: 422
Size of Test Set: 20
```

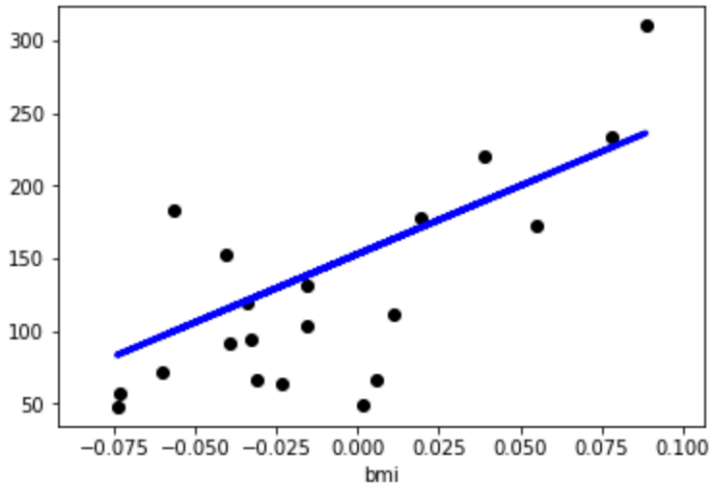
- Split dataset into training and test sets
- Training set is used for model training
- Test set is used for out-of-sample prediction to evaluate model performance

- sklearn.linear_model.LinearRegression(*fit_intercept=True, normalize=False, copy_X=True, n_jobs=None)*
- .fit is used for training model the model on training set
- .predict is used for predicting on test set
- Blue line shows fitted regression line and black points denote data points from test set

In [89]:

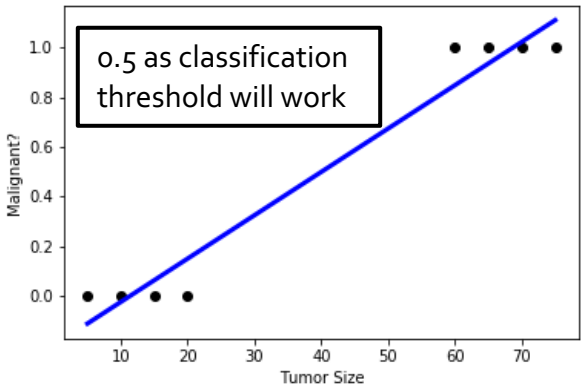
```
# Create linear regression object
regr = linear_model.LinearRegression()

# Train the model using the training sets
regr.fit(diabetes_X_train, diabetes_y_train)

# Make predictions using the testing set
diabetes_y_pred = regr.predict(diabetes_X_test)

# The coefficients
print('Coefficients: \n', regr.coef_)

# the mean-squared error
print("Mean squared error: %.2f"
      % mean_squared_error(diabetes_y_test, diabetes_y_pred))
# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % r2_score(diabetes_y_test, diabetes_y_pred))

# Plot outputs
plt.scatter(diabetes_X_test, diabetes_y_test,  color='black')
plt.plot(diabetes_X_test, diabetes_y_pred, color='blue', linewidth=3)
plt.xlabel('bmi')
plt.show()
```

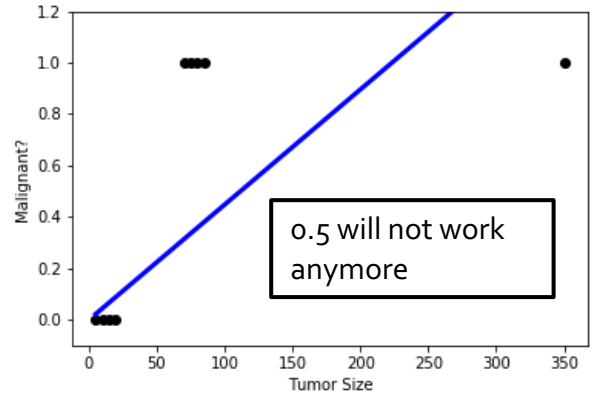
```
Coefficients:
 [938.23786125]
Mean squared error: 2548.07
Variance score: 0.47
```

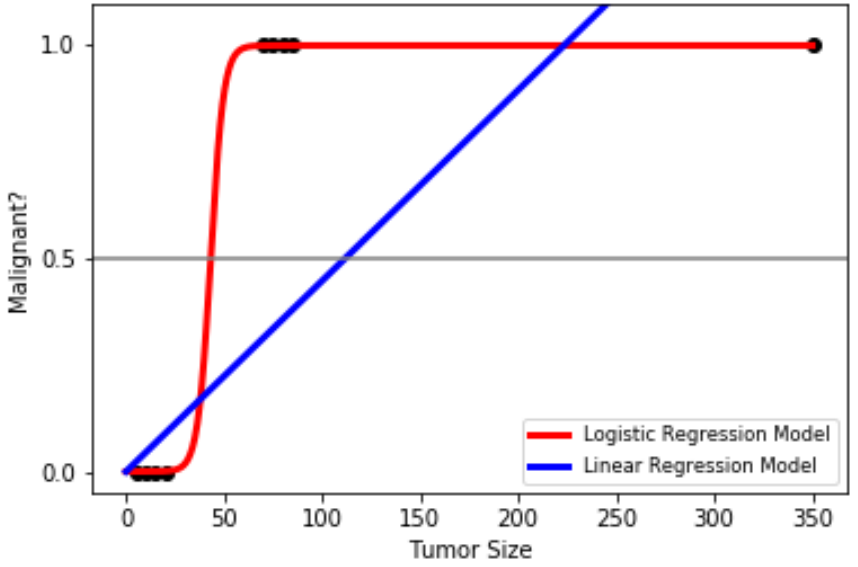


What is Wrong with using Linear Regression for Classification?

- You can label one of the classes with 0 and other with 1 and use linear regression but there are few problems
  - It simply interpolate between points so does not output probabilities
  - Does not extend to classification problems with multiple classes
- A solution for classification is **logistic regression**



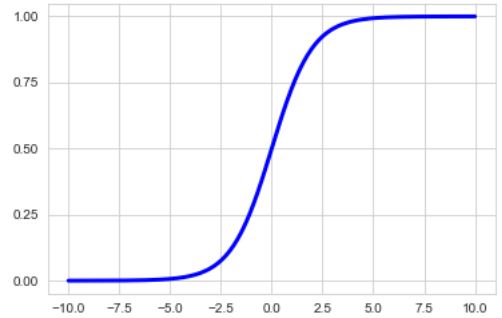




Logistic Regression

- Logistic regression model uses the logistic function to squeeze the output of a linear equation between 0 and 1.

$$logistic(x) = \frac{1}{1+e^{-x}}$$



- From Linear Regression

$$y^{(i)} = \beta_0 + \beta_1 x_1^{(i)} + \cdots + \beta_p x_p^{(i)}$$

- to Logistic Regression

$$P\left(y^{(i)} = 1\right) = \frac{1}{1 + e^{-\left(\beta_0 + \beta_1 x_1^{(i)} + \cdots + \beta_p x_p^{(i)}\right)}}$$

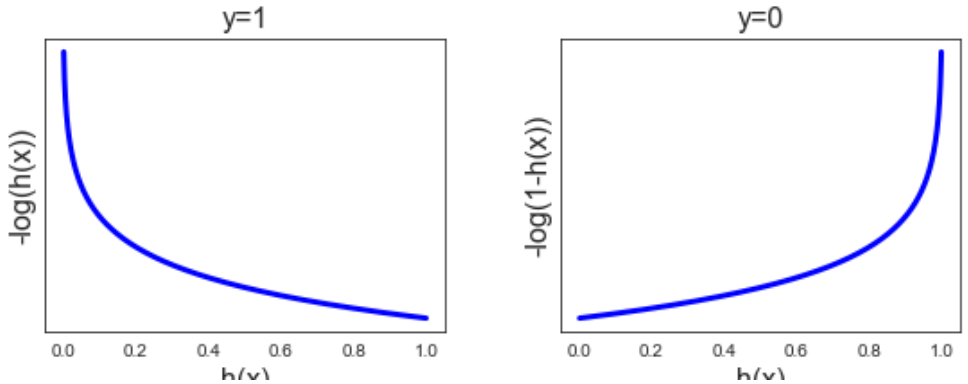
Logistic Regression Learning

- Logistic Regression for binary classification

$$P(y=1|x;\beta) = h_\beta(x) = \frac{1}{1+e^{(-\beta^T x)}} \Rightarrow \begin{cases} y=1 & if\ h(x) \geq 0.5 \\ y=0 & if\ h(x) < 0.5 \end{cases}$$

- Cost function – logistic loss function

$$cost\left(h_\beta(x), y\right) = \begin{cases} -\log\left(h_\beta(x)\right) & if\ y=1 \\ -\log\left(1-h_\beta(x)\right) & if\ y=0 \end{cases}$$



- Set up optimization problem as cost minimization

$$\min_{\beta \in R^p} J(\beta) = \frac{1}{n}\sum_i cost(h_\beta\left(x^{(i)}, y^{(i)}\right)$$

$$= \frac{1}{n}\sum_{i=1}^{n} -y^{(i)} \log\left(h_\beta\left(x^{(i)}\right)\right) - \left(1-y^{(i)}\right) \log\left(1-h_\beta\left(x^{(i)}\right)\right)$$

Regularized Logistic Regression

- Regularization terms can be added
  - in high dimensional problems to avoid overfitting
  - if you have prior assumptions on data – e.g. sparsity

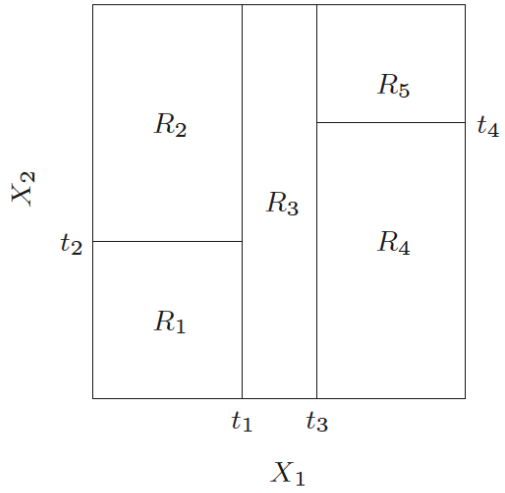
- For l_1 regularized logistic regression $\min_{\beta \in R^p, \lambda \geq 0} J(\beta) + \lambda ||\beta||_1$

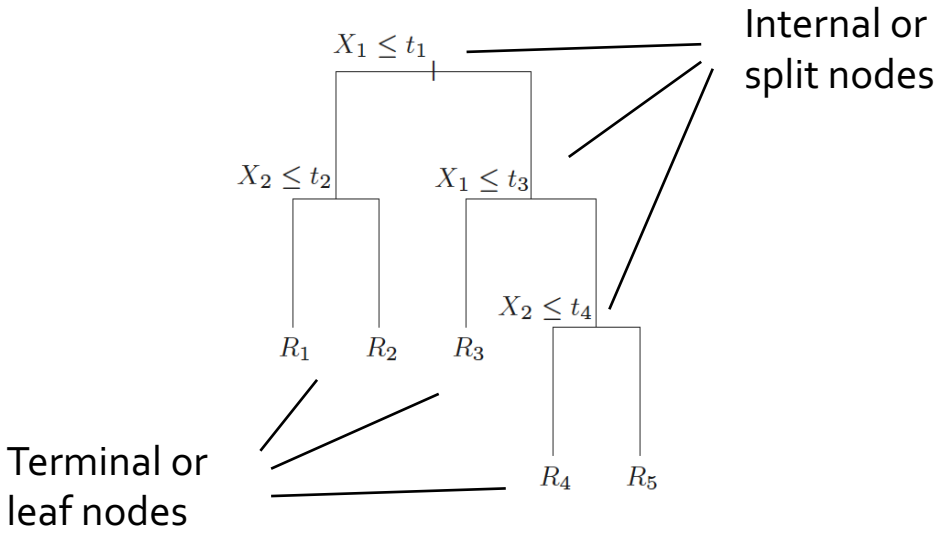
- For l_2 regularized logistic regression $\min_{\beta \in R^p, \lambda \geq 0} J(\beta) + \lambda ||\beta||_2^2$

Decision Trees

- Linear and Logistic Regression models fail in situations where
  - the relationship between features and outcome is **nonlinear**
  - **features interact** with each other
- Tree based models **split the data multiple times** according to certain cutoff values in the features.
- To predict the outcome in each leaf node, the **average outcome of the training data in this node** is used. Trees can be used for **classification and regression.**

Partition of a two-dimensional feature space by recursive binary splitting





Decision Tree Learning

- The classification and regression trees **(CART)** algorithm is probably the most popular algorithm for tree induction
- The goal is to find boxes $R_1, \ldots, R_J$ that minimizes residual sum of squares, given by

$$\sum_{j=1}^{J} \sum_{i \in \mathrm{R}_j} \left(y_i - \hat{y}_{R_j} \right)^2$$

where $\hat{y}_{R_j}$ is the mean response for the training data in j^{th} box

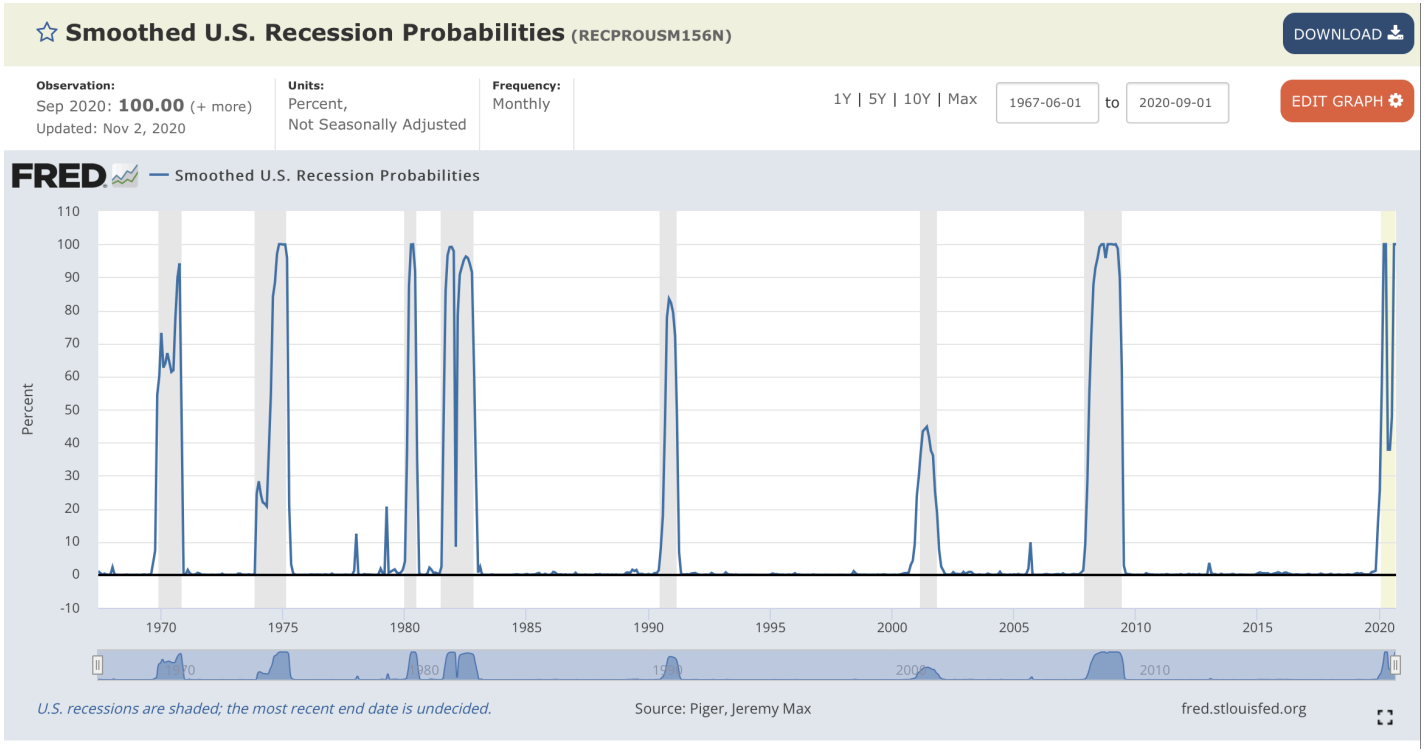
- Decision trees are **easy to use, interpretable, flexible** but a single decision tree is quite **unstable** and usually does not have good prediction accuracy
- However, by **aggregating many decision trees**, the predictive performance of trees can be substantially improved

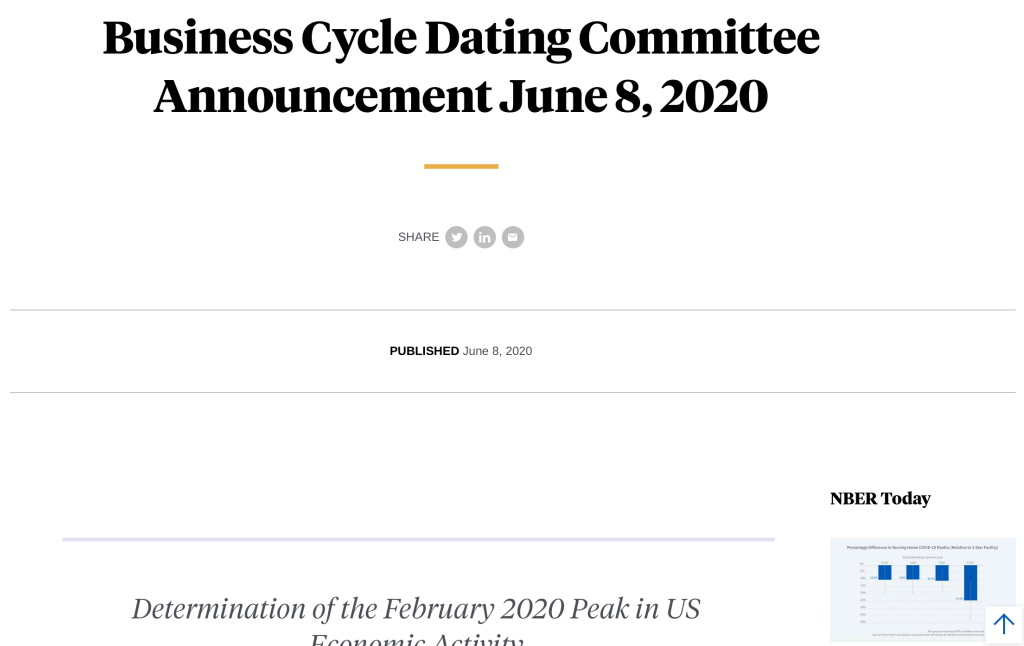
Ensemble Methods

- **Bagging**
  - Generate B different bootstrapped training datasets.
  - Train the learner on the b^{th} bootstrapped training set and get $\widehat{f^b}(x)$, the prediction at point a point x
  - Then average all predictions to obtain $\hat{f}_{bag}(x) = \frac{1}{B}\sum_b \widehat{f^b}\,(x)$
- **Random Forest**
  - Provides an improvement over bagged trees by decorrelating trees
  - This reduces the variance when we average the trees
- **Boosting**
  - "Boosting is one of the most powerful learning ideas introduced in the last twenty years" – Hastie et.al (2009)
  - Trees are grown sequentially: each tree is grown using information from previously grown trees

Business Cycle Prediction

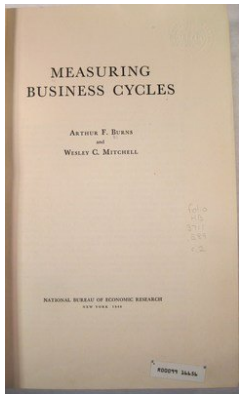
- Months and quarters of macroeconomic time series are separated into periods of **recession and expansion**
- NBER provides a chronology of business cycle dates but its methodology is **not explicitly formalized**,





Literature Overview

- Initial work


MEASURING
BUSINESS CYCLES

ARTHUR F. BURNS
and
WESLEY C. MITCHELL

NATIONAL BUREAU OF ECONOMIC RESEARCH

James H. Stock

KENNEDY SCHOOL OF GOVERNMENT
HARVARD UNIVERSITY
CAMBRIDGE, MASSACHUSETTS

Mark W. Watson

DEPARTMENT OF ECONOMICS
NORTHWESTERN UNIVERSITY
EVANSTON, ILLINOIS

New Indexes of Coincident and Leading Economic Indicators

1. Introduction

During six weeks in late 1937, Wesley Mitchell, Arthur Burns, and their colleagues at the National Bureau of Economic Research developed a list of leading, coincident, and lagging indicators of economic activity in the United States as part of the NBER research program on business cycles. Since their development, these indicators, in particular the leading and coincident indexes constructed from these indicators, have played an important role in summarizing and forecasting the state of macroeconomic activity.

A Comparison of the Real-Time Performance of Business Cycle Dating Methods

Marcelle CHAUVET
Department of Economics, University of California, Riverside, CA 92521 (*chauvet@ucr.edu*)

Jeremy PIGER
Department of Economics, University of Oregon, Eugene, OR 97403 (*jpiger@uoregon.edu*)

We evaluate the ability of formal rules to establish U.S. business cycle turning point dates in real time. We consider two approaches, a nonparametric algorithm and a parametric Markov-switching dynamic-factor model. Using a new "real-time" dataset of coincident monthly variables, we find that both approaches would have accurately identified the NBER business cycle chronology had they been in use over the past 30 years, with the Markov-switching model most closely matching the NBER dates. Further, both approaches, and particularly the Markov-switching model, yielded significant improvement over the NBER in the speed with which business cycle troughs were identified.

KEY WORDS: Dynamic-factor model; Markov-switching; Recession; Turning point; Vintage data.

1. INTRODUCTION

There is a long tradition in business cycle analysis of separating periods in which there is broad economic growth, called expansions, from periods of broad economic contraction, called recessions. Understanding these phases and the transitions between them has been the focus of much macroeconomic research over the past century. In the United States, the National Bureau of Economic Research (NBER) establishes a chronology that the NBER established the correct turning point dates in real time, thus making the NBER chronology the standard for accuracy.

Why are we interested in the speed with which business cycle turning points can be identified? The NBER is likely more concerned with establishing the correct turning point dates than establishing these dates quickly, which breeds additional caution. This caution comes at a low cost if the primary objective is to

- Recent work with machine learning methods

BOOSTING RECESSIONS

Serena Ng

Department of Economics
Columbia University*

August 2013

Abstract

This paper explores the effectiveness of boosting, often regarded as the state of the art classification tool, in giving warning signals of recessions three, six and twelve months ahead. Boosting is used to screen as many as 1500 potentially relevant predictors consisting of 132 real and financial time series and their lags. Estimation over the full sample 1961:1-2011:12 finds that there are fewer than ten important predictors and the identity of these variables change with the forecast horizon. There is a distinct difference in the size and composition of the relevant predictor set before and after mid-1980. Rolling window estimation reveals that the importance of the term and default spreads are recession specific. The Aaa spread is the most robust predictor of recessions three and six months ahead, while the risky bond and 5yr spreads are important for twelve months ahead predictions. Certain employment variables have predictive power for the two most recent recessions when the interest rate spreads were uninformative. Warning signals for the post 1990 recessions have been sporadic and easy to miss. The results underscore the challenge that changing characteristics of business cycles pose for predicting recessions.

Keywords: business cycle chronology, recession probabilities, classification.

JEL Classification: C5,C6, C25, C35.

International Journal of Forecasting 35 (2019) 848–867

Contents lists available at ScienceDirect

International Journal of Forecasting

journal homepage: www.elsevier.com/locate/ijforecast

Recession forecasting using Bayesian classification

Troy Davig, Aaron Smalter Hall *

Research Department, Federal Reserve Bank of Kansas City, Kansas City, MO 64198, USA

ABSTRACT

We demonstrate the use of a Naïve Bayes model as a recession forecasting tool. The approach is closely connected with Markov-switching models and logistic regression, but also has important differences. In contrast to Markov-switching models, our Naïve Bayes model treats National Bureau of Economic Research business cycle turning points as data, rather than as hidden states to be inferred by the model. Although Naïve Bayes and logistic regression are asymptotically equivalent under certain distributional assumptions, the assumptions do not hold for business cycle data. As a result, Naïve Bayes has a larger asymptotic error rate, but converges to the error rate more quickly than logistic regression, resulting in more accurate recession forecasts with limited data. We show that Naïve Bayes outperforms competing models and the Survey of Professional Forecasters consistently for real-time recession forecasting up to 12 months in advance. These results hold under standard error measures, and also under a novel measure that varies the penalty on false signals, depending on when they occur within a cycle; for example, a false signal in the middle of an expansion is penalized more heavily than one that occurs close to a turning point.

Machine Learning for Recession Prediction and Dynamic Asset Allocation

ALEXANDER JAMES, YASER S. ABU-MOSTAFA, AND XIAO QIAO

ALEXANDER JAMES is a co-head of research at Paramic Technologies US, Inc. in New York, NY. alex@paramic.com

YASER S. ABU-MOSTAFA is a professor of electrical engineering and computer science at the California Institute of Technology in Pasadena, CA. yaser@caltech.edu

XIAO QIAO is a co-head of research at Paramic Technologies US, Inc. in New York, NY. xiao@paramic.com

ABSTRACT: *The authors introduce a novel application of support vector machines (SVM), an important machine learning algorithm, to determine the beginning and end of recessions in real time. Nowcasting, forecasting a condition in the present time because the full information will not be available until later, is key for recessions, which are only determined months after the fact. The authors show that SVM has excellent predictive performance for this task, capturing all six recessions from 1973 to 2018 and providing the signal with minimal delay. The authors take advantage of the timeliness of SVM signals to test dynamic asset allocation between stocks and bonds. A dynamic risk budgeting approach using SVM outputs appears superior to an equal-risk contribution portfolio, improving the average returns by 85 bps per annum without increased tail risk.*

TOPICS: *Big data/machine learning, financial crises and financial market history, portfolio construction, tail risks**

Real-time business cycle dating is of central importance in

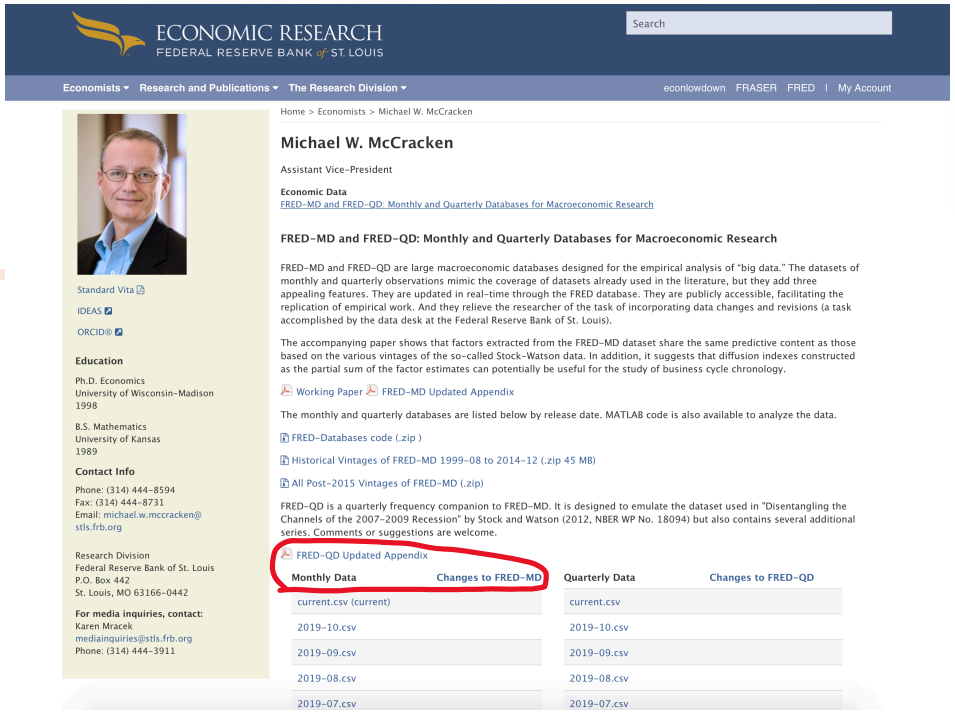
Investors with enough resources to use this information in their investment process may change their portfolios as the economy turns from growth to contraction.

The National Bureau of Economic Research (NBER) provides the official dating of expansions and recessions. The NBER's Business Cycle Dating Committee periodically assesses the prevailing conditions in the macroeconomy and determines if the economy is in an expansion or a recession.[2] The committee releases announcements about the dates of the turning points. Because of the reliance on macroeconomic data, which may be revised or released with a lag, and the committee's preference for accuracy over timeliness, the NBER has historically announced turning points with a delay of 4 to 21 months (Giusto and Piger 2017).

Is it possible to identify business cycle turning points in a more timely manner? This is the focus of a large body of literature going back to Burns and Mitchell (1946) and greatly expanded in a series of papers

Dataset Description

- Output
  - NBER business cycle dates used for recession periods
- Features: 135 macroeconomic time series
  - output and income,
  - labor market
  - housing
  - consumption, orders and inventories,
  - money and credit,
  - interest and exchange rates,
  - prices
  - stock market
- Sample period: 1959-2020



Appendix

The column TCODE denotes the following data transformation for a series x: (1) no transformation; (2) Δx_t; (3) $\Delta^2 x_t$; (4) $log(x_t)$; (5) $\Delta \log(x_t)$; (6) $\Delta^2 \log(x_t)$; (7) $\Delta(x_t/x_{t-1} - 1.0)$. The FRED column gives mnemonics in FRED followed by a short description. The comparable series in Global Insight is given in the column GSI.

Some series require adjustments to the raw data available in FRED. We tag these variables with an asterisk to indicate that they been adjusted and thus differ from the series from the source. A summary of the adjustments is detailed in the paper https://research.stlouisfed.org/wp/2015/2015-012.pdf.

Group 1: Output and income

	id	tcode	fred	description	gsi	gsi:description
1	1	5	RPI	Real Personal Income	M_14386177	PI
2	2	5	W875RX1	Real personal income ex transfer receipts	M_145256755	PI less transfers
3	6	5	INDPRO	IP Index	M_116460980	IP: total
4	7	5	IPFPNSS	IP: Final Products and Nonindustrial Supplies	M_116460981	IP: products
5	8	5	IPFINAL	IP: Final Products (Market Group)	M_116461268	IP: final prod
6	9	5	IPCONGD	IP: Consumer Goods	M_116460982	IP: cons gds
7	10	5	IPDCONGD	IP: Durable Consumer Goods	M_116460983	IP: cons dble
8	11	5	IPNCONGD	IP: Nondurable Consumer Goods	M_116460988	IP: cons nondble
9	12	5	IPBUSEQ	IP: Business Equipment	M_116460995	IP: bus eqpt
10	13	5	IPMAT	IP: Materials	M_116461002	IP: matls
11	14	5	IPDMAT	IP: Durable Materials	M_116461004	IP: dble matls
12	15	5	IPNMAT	IP: Nondurable Materials	M_116461008	IP: nondble matls
13	16	5	IPMANSICS	IP: Manufacturing (SIC)	M_116461013	IP: mfg
14	17	5	IPB51222s	IP: Residential Utilities	M_116461276	IP: res util
15	18	5	IPFUELS	IP: Fuels	M_116461275	IP: fuels
16	19	1	NAPMPI	ISM Manufacturing: Production Index	M_110157212	NAPM prodn
17	20	2	CUMFNS	Capacity Utilization: Manufacturing	M_116461602	Cap util

Model Set-up

Notations

- y_{t+h}: regime at period $t + h$ and $y_{t+h} \in \{0,1\}$ (1 for *crash* and 0 *normal* regime)
- Model results include both nowcasting($h = 0$) and forecasting ($h = 1,3,6,12$)
- X_t: input variables at period t which includes both current values of variables as well as lags
- $\hat{y}_{t+h} = P(y_{t+h} = 1|X_t)$ crash regime prediction probabilities
- 0.5 threshold can be used for binary classification
- **Linear classifiers** (Logistic Regression with various penalty terms) and **non-linear classifiers** (tree-based ensembles) are used in predictive model framework

Classification Tree in Regime Prediction

- Recursive splits partition the sample space into M non-overlapping regions
  - A_m^* denotes region m for $m = 1, \ldots, M$
  - T_m^* training sample observations in each region m for $m = 1, \ldots, M$
- Prediction for y_{t+h}^c is equal to within region proportion of class c

$$P_{A_m^*}^c = \frac{1}{T_m^*} \sum_{X_t \in A_m^*} I(y_{t+h} = c)$$

- The CART classifier is then:

$$\hat{y}_{t+h}^c(X_t) = \sum_{m=1}^{M} P_{A_m^*}^c I(X_t \in A_m^*)$$

How is the recursive portioning implemented to arrive at the regions A_m^*?

- Splits are based on binary condition of the form $X_{j,t} < \tau$ and $X_{j,t} \geq \tau$, where j and τ can differ across splits
- From unsplit node A, for a given j and τ^j group the data into two regions

$$A_L = \{X_t | X_{j,t} < \tau^j, X_t \in A\} \text{ and } A_R = \{X_t | X_{j,t} \geq \tau^j, X_t \in A\}$$

- To determine splitting variable, j , and the split threshold, τ^j we want to **maximize a measure of homogeneity of class outcomes** in A_L and A_R
- A common choice Gini Impurity

Gini Impurity

$$G_L = \sum_{c=1}^{C} P_{A_L}^c (1 - P_{A_L}^c) \text{ and } G_R = \sum_{c=1}^{C} P_{A_R}^c (1 - P_{A_R}^c)$$

- Gini impurity
  - is **bounded between 0 and 1**
  - Value **0 indicates a pure region** where only one class is present
  - **Higher values** of Gini impurity indicates **greater class diversity**
- The average Gini impurity for the new proposed regions is

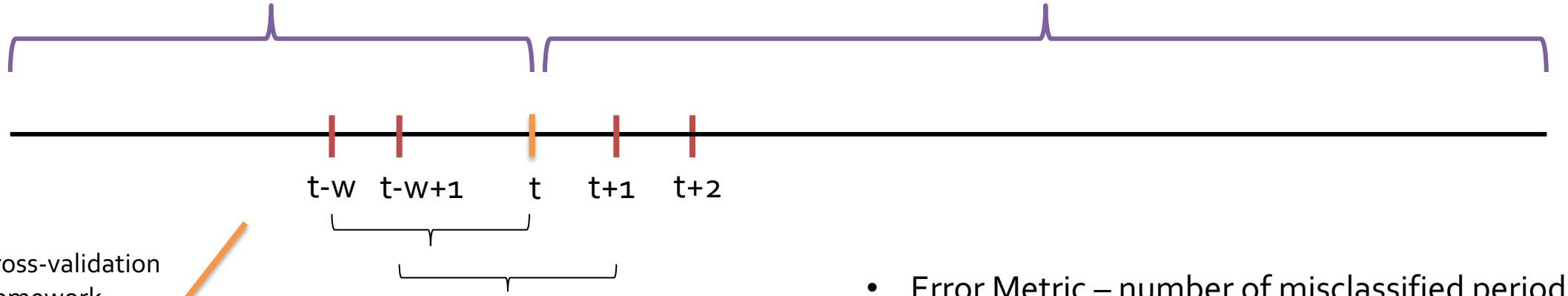
$$\bar{G} = \frac{T^L}{T^L + T^R} G_L + \frac{T^R}{T^L + T^R} G_R$$

- j and τ^j are chosen to create regions A_L and A_R that **minimize the average Gini impurity**

Predictive Model Framework

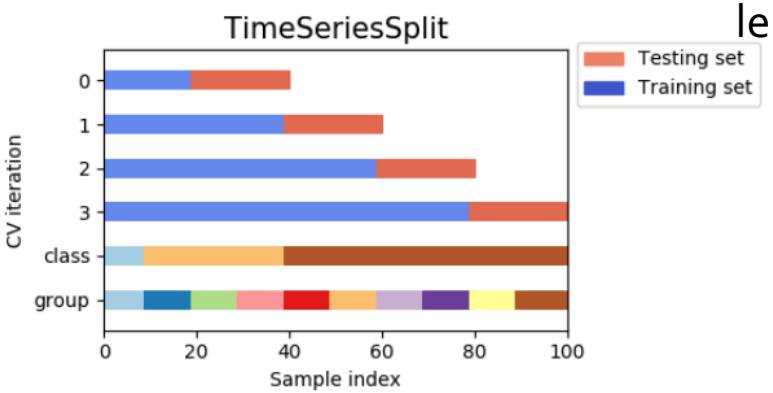
Training data (in-sample)
Cross-validation based on time series blocks

Testing data (out-of-sample)
Out-of-sample predictions with rolling window length w



Cross-validation framework

Window length w



- Error Metric – number of misclassified periods

$$error = \frac{\#\ incorrect\ observations}{\#Total\ observations}$$

- Benchmark
  - recession occurrences frequency in the sample **13.15%**

Error Metrics

- Quadratic Probability Score (QPS)
  - evaluates regime predictions in terms of probabilities
  - In perfect prediction setting QPS will be o
  - Lower QPS -> better model performance

$$QPS = \frac{1}{T}\sum_{t=1}^{T}(\hat{y}_t - y_t)^2$$

- Classification Accuracy (ACC) for a given threshold *c*
  - Natural classification metric
  - ACC should be at least better than benchmarks for each regime

$$ACC = \frac{1}{T}\sum_{t=1}^{T}\left[\mathbb{1}_{\{\hat{y}_t \geq c\}} y_t + (1 - \mathbb{1}_{\{\hat{y}_t \geq c\}})(1 - y_t)\right]$$

Class Imbalance Sensitive Metrics

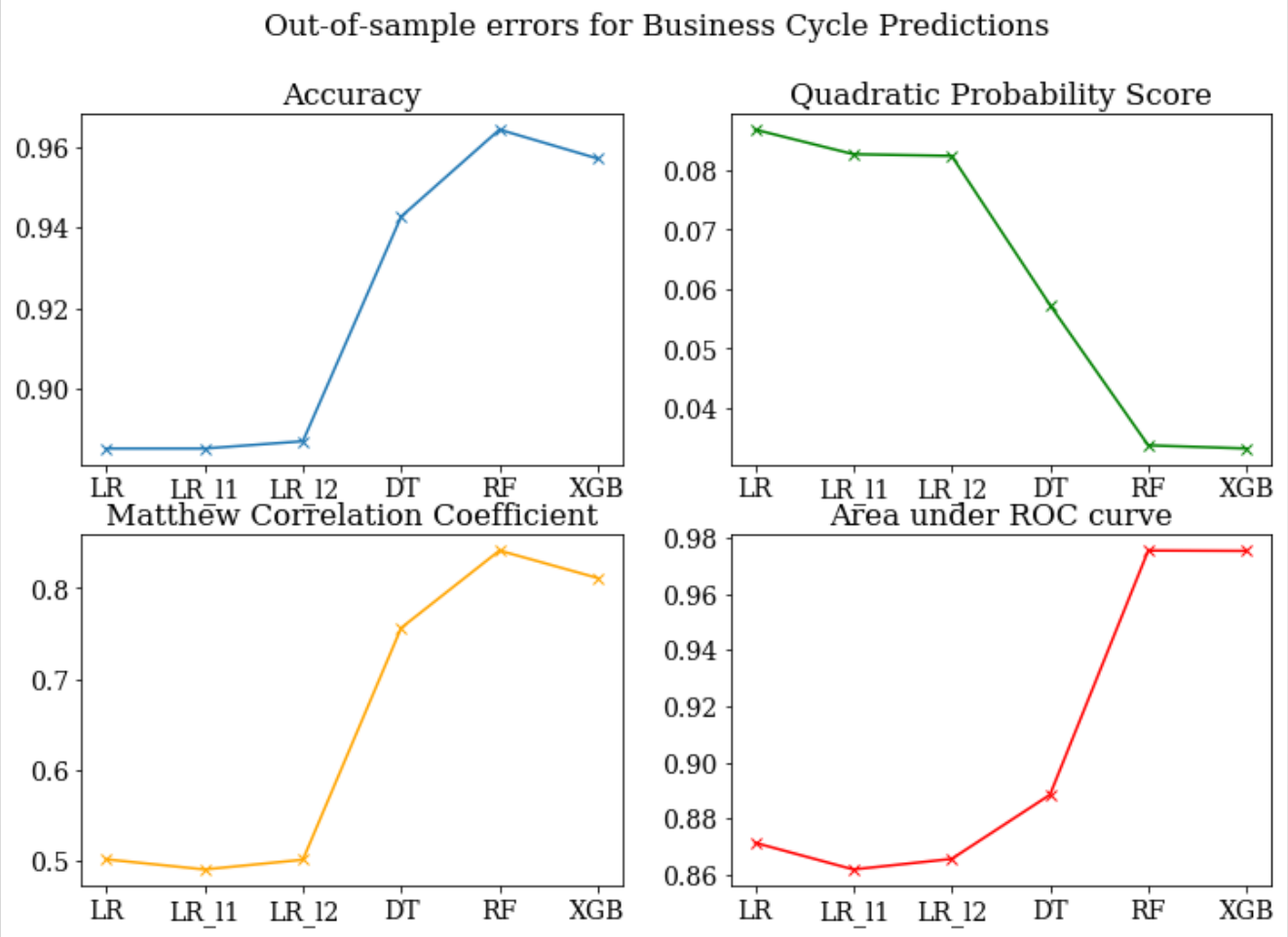
To avoid inflated error metrics, we will consider 2 other error metrics

Actual/Prediction	P	N
P	TP	FN
N	FP	TN

- Receiver Operating Characteristic (ROC)
  - Common metric for imbalanced classification problems
  - ROC curve plots true positive rate against false positive rate
  - Area under the curve (AUC) is used to generate summary statistics for ROC metric
  - Higher AUC -> better model performance
- Matthew's Correlation Coefficient (MCC)
  - Used widely in computational biology
  - Can be calculated from confusion matrix
  - Higher MCC -> better model performance

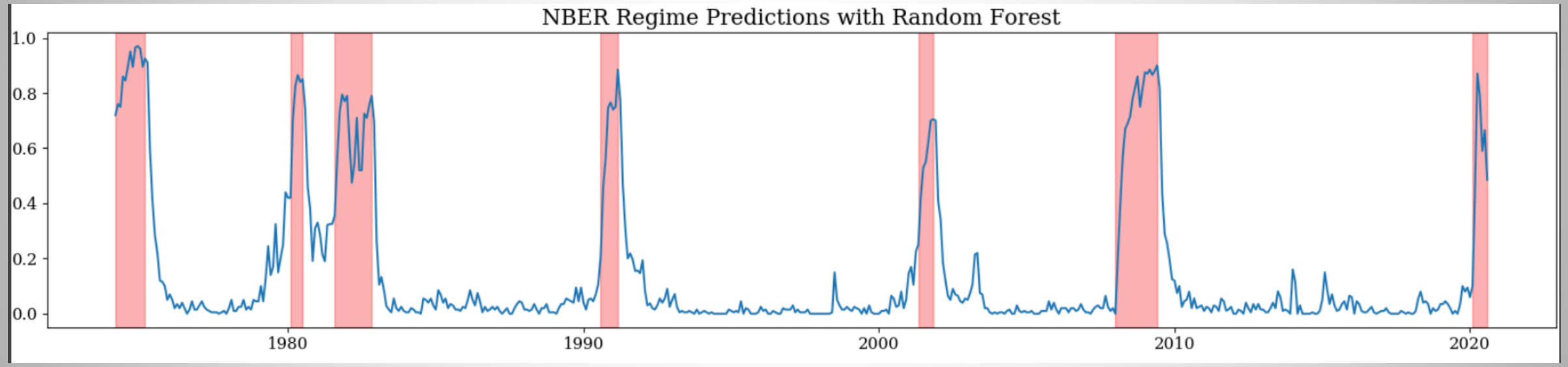
$$MCC = \frac{tp \times tn - fp \times fn}{\sqrt{(tp + fp)(tp + fn)(tn + fp)(tn + fn)}}$$

Regime Prediction Error Metrics (1975-2020)



LR: Logistic Regression
LR_l1: Logistic Regression with l1 penalty
LR_l2: Logistic Regression with l2 penalty
DT: Decision Trees
RF: Random Forest
XGB: Gradient Boosting Trees

Recession Prediction Results (1975-2020)



References

- Ang, A. and Timmerman, A. (2011) *Regime Changes and Financial Markets*
- Mulvey, J.M. and Liu, H. (2016) *Identifying Economic Regimes: Reducing Downside Risks for University Endowments and Foundations.* Journal of Portfolio Management
- Hastie, T., Tibshirani, R. and Friedman, J. (2009) *The Elements of Statistical Learning*
- Varian, H. (2014) *Big Data: New Tricks for Econometrics.* Journal of Economic Perspectives
- Mullainathan, S. and Spiess, J. (2017) *Machine Learning: An Applied Econometric Approach.* Journal of Economic Perspectives
- Athey, S. (2018) *Impact of Machine Learning in Economics* NBER
- Lopez De Prado, M. (2018https://arxiv.org/abs/1903.10075) *Advances in Financial Machine Learning*
- *Piger, J.* (2019) *Macroeconomic Forecasting in the Era of Big Data – Chapter 18*

References

- Burns, A. F., and W. C. Mitchell. (1946) *Measuring Business Cycles*. NBER
- Stock, J. H., and M. W. Watson. (1989). *New Indexes of Coincident and Leading Economic Indicators.* NBER Macroeconomics Annual 4: 351–394.
- Chauvet, M., and J. Piger. (2008) *A Comparison of the Real-Time Performance of Business Cycle Dating Methods.* Journal of Business and Economic Statistics 26: 42–49.
- Ng, S. (2013) *Boosting Recessions.* Canadian Journal of Economics
- Davig, T. and Hall, A.S. (2019) *Recession Forecasting using Bayesian Classification.* International Journal of Forecasting
- James, A., Abu-Mostafa, Y.S. and Qiao, X. (2019) *Machine Learning for Recession Prediction and Dynamic Asset Allocation.* The Journal of Financial Data Science
- Molnar, C. (2019) *Interpretable Machine Learning*
- McCracken,M.W. and Ng, S. (2015) *FRED-MD: A Monthly Database for Macroeconomic Research.* Working Paper