Predicting Fama-French Factors Using Machine Learning Techniques

Jing Li
University of Toronto
Department of Computer Science
lijing@cs.toronto.edu

David Veitch
University of Toronto
Department of Statistical Sciences
david.veitch@mail.utoronto.ca

Xuling Wang
University of Toronto
Department of Computer Science
shirlywang@cs.toronto.edu

Abstract

This paper applies a variety of machine learning models to the task of predicting two of the Fama-French factors, SMB and HML. For each year of predictions the models are trained on the previous four-hundred months of data. Predictions are made for the period of 1988-2018. The performance of trading strategies based on these predictions are assessed. Our results suggest that machine learning models do possess predictive ability for these factors.

1 Introduction

While much of the machine learning finance literature has been focused on the prediction of single-stock, or stock-index returns, our paper investigates the ability of machine learning models (logistic regression, support vector machines, random forests, AdaBoost, Gradient Boosted Trees, Multilayer Perceptrons, and GRUs) to predict Fama-French factor returns.

In this paper we assess the performance of a variety of machine learning models in creating trading strategies around the Fama-French factors. Specifically we aim to predict whether the monthly returns of the two most common factors (SMB and HML) will be positive or negative, and which one will be higher, and create trading strategies (long or short each individual factor, and long the factor with the highest predicted return) using these predictions. Unlike many machine learning tasks the dataset is small (801 rows) and extremely noisy; despite this, our results show that these models still possess a degree of predictive accuracy, and the trading strategies derived from them have been successful over time.

2 Background and Related Work

Factor investing involves investing in a portfolio of securities that share common characteristics in order to achieve excess returns [Bender et al., 2013]. Two of the most well known factors which have been shown to persist over time in the equity market, are the size (SMB) and value factors (HML) put forward by Fama and French [Fama and French, 1993]. The SMB factor represents the average returns of a portfolio of small firms (by market capitalization) over large firms, while the HML factor represents the average returns of a portfolio of value stocks over growth stocks.

$$\boxed{SMB = \frac{1}{3}(SmallValue + SmallNeutral + SmallGrowth - BigValue - BigNeutral - BigGrowth)}$$

$$\boxed{HML = \frac{1}{2}(SmallValue + BigValue - SmallGrowth - BigGrowth)}$$

While the SMB and HML factors have produced positive cumulative returns over time (Figure 1), there have been stretches where the factors produced negative returns. Understanding when the SMB or HML are prone to produce positive or negative returns can help portfolio managers better time these factors. One explanation for this is the sensitivity of SMB and HML to macroeconomic variables [Winkelmann et al., 2013]. This sensitivity to macroeconomic variables, variables which evolve slowly over time, coupled with most investors' short time-horizons, have been hypothesized to be reasons why returns associated with these factors have fluctuated, and persisted, over time.

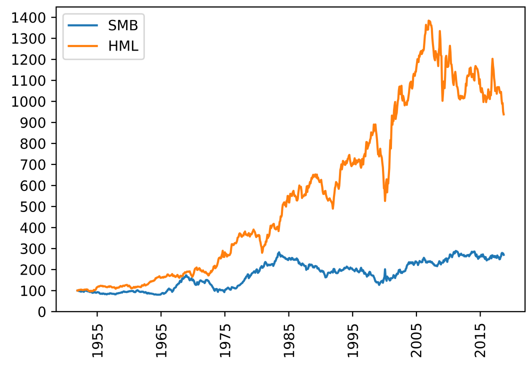


Figure 1: Cumulative Return of SMB & HML Factors (1952=100)

A number of papers in the finance literature explore the predictability of the Fama-French factors using macroeconomic variables. Tsuji [2012] demonstrated that industry returns contain predictive information for one-month-ahead SMB factor returns. Cooper et al. [2002] examined active SMB and HML strategies based on macroeconomic risks. These papers make their predictions using simple linear regression.

With the rising popularity of machine learning, the finance community has begun using these techniques to predict financial time series. Krauss et al. [2016] and Fischer and Krauss [2018] used a variety of machine learning models to create trading strategies for S&P 500 stocks. In particular, Fischer and Krauss [2018] focused on long short-term memory networks (LSTM), one of the most widely-used deep learning algorithms for time series prediction tasks. However, to-date there has been little work done on using machine learning methods to predict the Fama-French factors. Despite different financial time series of interest, these papers gave us significant insights in terms of problem formulation, methodology, and performance evaluation.

3 Data and Software

3.1 Data

Monthly Fama-French returns, risk-free returns, and market excess returns data for the United States was obtained from the online Fama-French data library [Fama and French, 2018]. The authors of this paper selected a number of explanatory variables that reflect the general economic and market conditions in the months between January 1952 and September 2018. One-year ahead forecasts of nominal GDP, CPI, and industrial production came from the Federal Reserve Bank of Philadelphia's Livningston survey [Federal Reserve Bank of Philadelphia, 2018]. These surveys occur twice annually (June and December); any month in the dataset between survey dates was assigned the value of the last survey date. Government deficit figures and corporate bond yields are sourced from FRED [Federal Reserve Bank of St. Louis, 2018]. S&P 500 valuation data is from Robert Shiller [Shiller, 2018], and data on S&P 500 returns, interest rates, business confidence, and industrial production is from GlobalFinancialData [GFD, 2018]. Further information about the features used can be found in Appendix A.

3.2 Software

The preprocessing of data occurred in Microsoft Excel. The machine learning models were built in Python using popular libraries such as Scikit-learn [Pedregosa et al., 2011], and PyTorch [Paszke et al., 2017].

4 Methodology

This section gives a formal description of our modelling approach. The flow chart in Figure 2 gives an overview of this process.

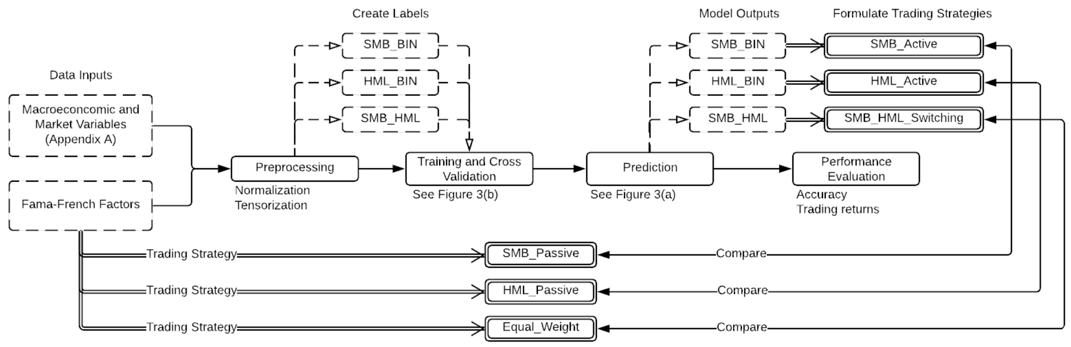


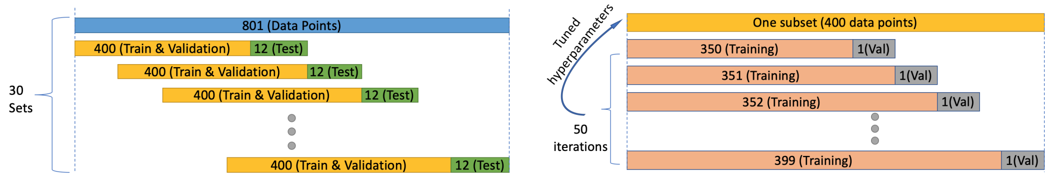
Figure 2: Modelling Process

4.1 Cross-Validation

Cross-validation (CV) is a popular technique for tuning hyperparameters and producing robust measurements of model performance. Two of the most common types of cross-validation are k-fold cross-validation and hold-out cross-validation. Traditional cross-validation should not be used when dealing with time series due to temporal dependencies. Instead, nested cross-validation should be used as it has been proven to provide an almost unbiased estimation of the true error [Varma and Simon, 2006].

We adopted a nested cross-validation approach. First, the whole data set was split into 30 study periods (see Figure 3). We define a study period as a training-test subset, consisting of a four-hundred month training period, and a twelve month test period. We split the entire data set from 1952 to 2018 into thirty study periods, with each test set containing non-overlapping sets of twelve months of data.

In each study period, we further split the training set into fifty training and validation sets (See Figure 3). In each iteration of cross-validation the training set was expanded by one month and the subsequent month was used for validation. Once cross-validation was completed, the validation accuracy for each model was averaged and the models parameters with the highest accuracy were selected and used to predict the test set.



(a) Train Test Split (b) Nested Cross-Validation Split

Figure 3: Cross-Validation Procedure

4.2 Performance Evaluation

The machine-learning finance literature suggests that using machine-learning methods to perform classification, as opposed to regression, performs better in predicting financial time series [Krauss et al., 2016]. Our initial trials on fitting a regression model on SMB and HML factor returns confirmed this observation. As a result, we formulated our predictions as a classification problem with three targets:

- SMB_BIN: A binary classification to indicate if SMB is predicted to be non-negative in the next month
- HML_BIN: A binary classification to indicate if HML is predicted to be non-negative in the next month
- SMB_HML: A binary classification to indicate if SMB is predicted to be greater than or equal to HML in the next month

In addition to reporting prediction accuracy on the test data, we also formulated three simple active trading strategies:

- SMB_Active: Long the SMB portfolio if SMB_BIN=1 and short the portfolio otherwise. Its return is compared to the passive strategy of always holding a long position in SMB portfolio (i.e. SMB_Passive).
- HML_Active: Long the HML portfolio if HML_BIN=1 and short the portfolio otherwise. Its return is compared to the passive strategy of always holding a long position in HML portfolio (i.e. HML_Passive).
- SMB_HML_Switching: Long the SMB portfolio if SMB_HML=1 and long the HML portfolio otherwise. Its return is compared to the passive strategy of investing 50-50 in SMB and HML portfolios (i.e. Equal_Weight).

We compared our active trading strategies to the corresponding passive trading strategies rather than the market return (such as the return on the S&P 500). This is because our strategies are market-neutral.

4.3 Models

4.3.1 Logistic Regression

Logistic regression with L2 regularization was used. We implemented logistic regression using Scikit-Learn and used cross-validation to tune regularization strength.

4.3.2 Support Vector Machines (SVM)

Support Vector Machines [Cortes and Vapnik, 1995] make use of non-linear mappings from an input space to a high-dimensional feature space and subsequently separate the data in the high-dimensional space using hyperplanes. These linear hyperplanes in the high-dimensional space ultimately map back to non-linear hyperplanes in the input space, which separate the data allowing for classification.

Cross-validation was used to tune the kernel (radial basis function, linear), gamma, and the classifier's margin C (0.01,0.1,1,10,100).

4.3.3 Random Forest

Random Forest algorithms [Breiman, 2001] produce classification estimates by taking the most popular classification from a collection of tree-based classifiers. Two important features of random forests are the use of bagging (training each tree on a random sample of the dataset), and training trees using a randomly chosen subset of the input features.

Cross-validation was used to tune the splitting criteria (gini, entropy) as well as the maximum tree-depth (1,2,3,4,unlimited).

4.3.4 AdaBoost

The AdaBoost algorithm [Freund et al., 1996] works by repeatedly running a given weak learning algorithm on various distributions over the training data, and then combining the classifiers produced by the weak learners into a single composite classifier.

Cross-validation was used to tune the learning rate as well as number of features.

4.3.5 Gradient Boosted Trees (GBT)

The algorithm for Gradient Boosted Trees [Friedman, 2001] evolved from the application of boosting methods to regression trees. The general idea is to compute a sequence of very simple trees, where each successive tree is built for predicting the residuals of the preceding tree. It can be shown as fitting a weighted additive expansion of simple decision trees.

Cross-validation was used to tune the learning rate as well as the number of features.

4.3.6 Multiayer Perceptron (MLP)

A Multilayer Perceptron [Rumelhart et al., 1986] consists of an input layer, multiple hidden layers, and an output layer. Each layer is composed of neurons which weigh the outputs of the previous layer (w_i), and adjust for bias (b), and use an activation function (f) to output a value. That is, for each neuron at each layer $Output = f(\sum_{i=1}^{n} w_i x_i + b)$.

Cross-validation was used to tune the activation function (tanh, ReLu) as well as architecture of the MLP. Similar to Krauss et al. [2016] the number of hidden units in the first layer matches the dimension of the input layer, and subsequent hidden layers have less or equal amounts of hidden units to the previous hidden layer (a form of dimension reduction). The architectures tested were: (28), (28,28), (28,16), (28,8), (28,4), (28,28,28), (28,16,8), (28,8,4).

4.3.7 Gated Recurrent Neural Network (GRU)

Similar to LSTMs, Gated Recurrent Neural Networks address the vanishing gradient issue in recurrent neural networks and are a good model for sequence data. However, due to the lack of a forget gate, GRUs require fewer parameters and are easier to implement. Chung et al. [2014] suggested that GRUs can perform better than LSTMs over small datasets.

For each target, we implemented a one-layer GRU using PyTorch. Models used a sequence length of twelve (i.e. one full year of data) and L2 regularization. To simplify implementation, we employed one-round cross-validation with a sequential 350/50 training/validation split to tune the learning rate and regularization strength. Five-hundred epochs were run for each model.

5 Results

5.1 Accuracy

In this section, we present the test accuracy on the three targets predicted by eight machine learning models: SVM, AdaBoost, MLP, Gradient Boosted Trees, Random Forest, Logistic Regression and GRU. Accuracy for each model for each target was calculated by averaging the predicted results over thirty years. Additionally, for each model we averaged the accuracy on the three targets to look at its overall performance.

Figure 4 illustrates the results. GRU and Logistic Regression achieved the highest average accuracy of 54%, and the third highest model is Random Forest with average accuracy of 53.8%. While an accuracy of 54% does not at first glance seem high, it is consistent with results in Krauss et al. [2016] who had average accuracies of 52%-54%. This speaks to the noisiness of the data, and the difficulty of inferring structure from it.

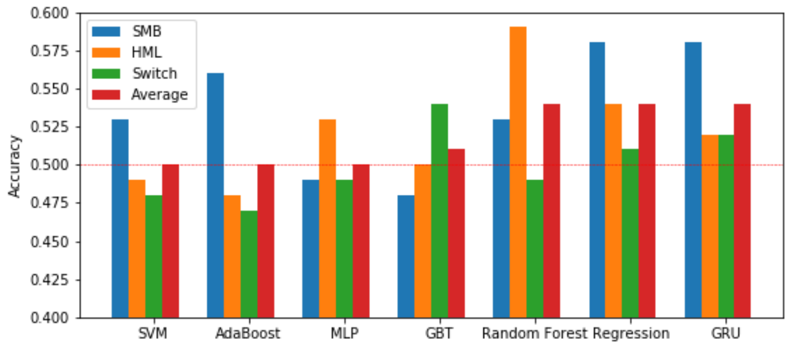


Figure 4: Accuracy on All Models

5.2 Returns

We adopted the top three models (GRU, Random Forest and Logistic Regression) based on accuracy to implement active trading strategies. A summary of returns can be found in Appendix B. Cumulative returns of these trading strategies are shown in Figure 5.

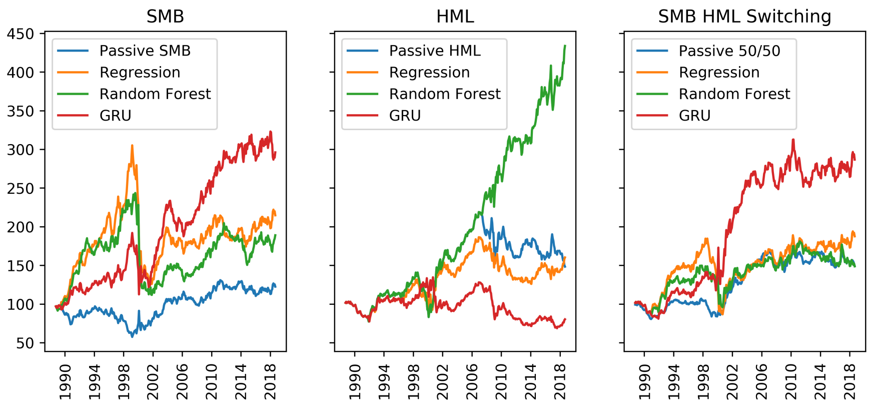


Figure 5: Returns Compared to Benchmarks (1988=100)

Figure 6 illustrates the performance in terms of average and standard deviation of monthly returns on each strategy using different models. Most of our trading strategies achieve higher mean returns as compared to the respective benchmark strategy, with the exception of GRU-modelled HML_Active. In addition, our SMB and HML active strategies do not introduce additional variability of returns. SMB_HML_Switching strategy introduces more noise. However, this is expected since the corresponding passive strategy averages two returns and therefore reduces variance.

Figure 7 illustrates model performance over time using monthly mean return. Green indicates out-performance and red indicates under-performance. One can see that model performance is generally worse in the period from 1998 to 2008. This might be related to high economic uncertainties during the dot-com bubble in early early 2000s and the global financial crisis of 2008.

In Figure 8, We plotted SMB and HML returns and indicated whether we predicted them correctly. As shown in the chart, our models largely reflect trends in the two returns, especially when the returns are further from zero. This also explains how our models achieve high returns even though the accuracies are not superb.

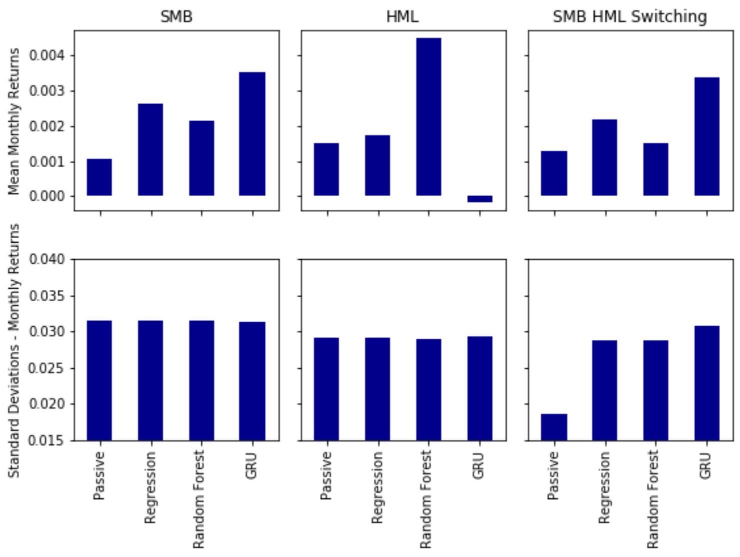


Figure 6: Return Summary Compared to Benchmarks

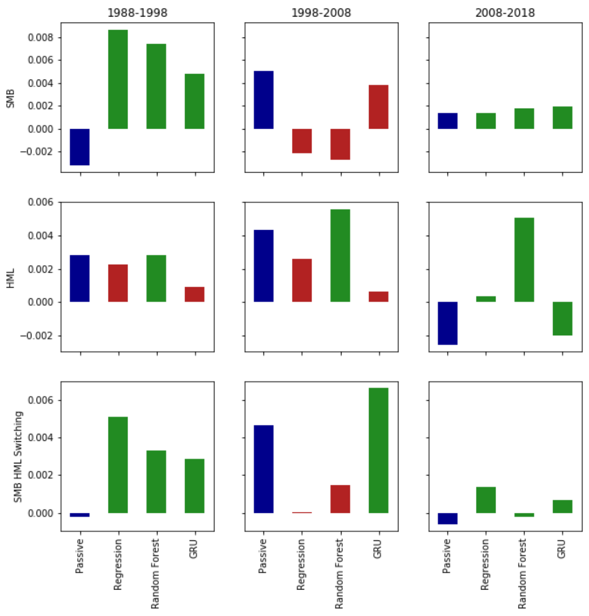


Figure 7: Average Monthly Return by Period

5.3 Feature Rankings

To get a sense of what features are driving the models' predictions, we observed the average feature importances (a measure similar to information gain) of the random forest and logistic regression models that were selected in cross-validation. For logistic regression, since all variables were scaled to range between 0 and 1, we could infer feature importance from absolute values of coefficients in the model.

The random forest and logistic regression models are consistent in terms of feature importances. One observation is that for all three time series it appears market-related features are very important.

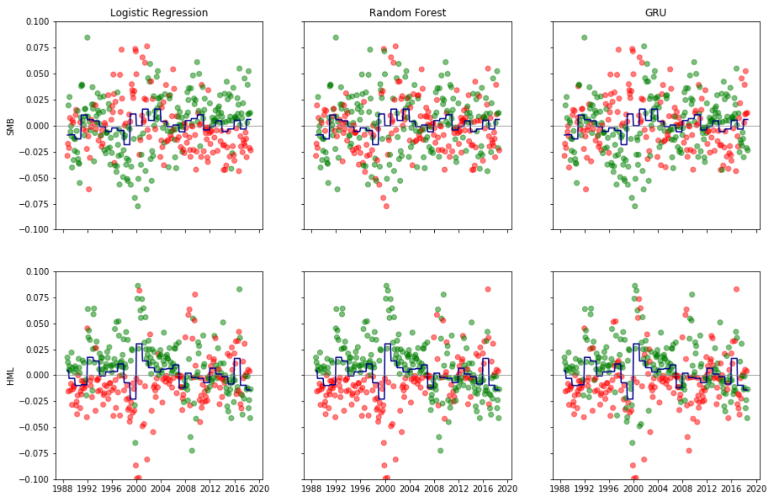


Figure 8: Predictions

This could be due to higher data frequency as many of the macroeconomic factors are only updated semi-annually. Detailed feature importances can be found in Appendix C.

6 Limitations

The limitations with respect to the conclusions is twofold – the first is the dataset being used, and the second is the ability to implement a trading strategy such as this in practice.

The first, and most important limitation, is the size of the dataset being used. In this paper, monthly Fama-French returns, risk-free returns, and market excess returns data since 1952 are used. In total this amounts to 801 rows. While this covers a long period of time by finance standards, the data set is very small with respect to machine learning standards. Given this, there is a definite concern that the model may be overfitting on the limited training data. In addition, some of the features used come with a lag (e.g. GDP forecasts from the Livingston survey are only updated once every six months). The staleness of some of the features could affect the models' ability to learn. Future work could look at expanding the dataset by including Fama-French factor returns for countries other than the United States, or simplifying the model using less features.

From a practical point of view, trading a strategy such as this will involve significant portfolio turnover and along with this transaction costs (associated with month-to-month changes in the constituents of the factor portfolios plus trades the model is suggesting). The inclusion of transaction costs would likely bias the returns of the benchmark, and our suggested trading strategies, down.

7 Conclusion

As demonstrated in the results section, machine-learning methods do appear to have a degree of accuracy in predicting Fama-French factors. This is demonstrated by a number of the models having over-50% accuracy, and the performance of simple trading strategies based on these models over time. The authors believe the models can be refined further to serve as a valuable tool for portfolio managers for factor investing.

References

Global Financial Data, 2018. Data retrieved from https://www.globalfinancialdata.com/.

Jennifer Bender, Remy Briand, Dimitris Melas, and Raman Aylur Subramanian. Foundations of Factor Investing. 2013. Available at SSRN: https://ssrn.com/abstract=2543990 or http://dx.doi.org/10.2139/ssrn.2543990.

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001. ISSN 1573-0565. doi: 10.1023/A:1010933404324. URL https://doi.org/10.1023/A:1010933404324.

Junyoung Chung, Caglar Gulcehre, Kyung Hyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Michael Cooper, Huseyin Gulen, and Maria Vassalou. Investing in size and book-to-market portfolios: Some new trading rules. *Manuscript, Purdue University*, 2002.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, Sep 1995. ISSN 1573-0565. doi: 10.1007/BF00994018. URL https://doi.org/10.1007/BF00994018.

Eugene Fama and Kenneth French. Fama/french 3 factors, 2018. Data retrieved from Kenneth French Data Library, http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html.

Eugene F. Fama and Kenneth R. French. Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, 33(1):3–56, February 1993. URL https://ideas.repec.org/a/eee/jfinec/v33y1993i1p3-56.html.

Federal Reserve Bank of Philadelphia. Historical Data: Livingston Survey, 2018. Data retrieved from https://www.philadelphiafed.org/research-and-data/real-time-center/livingston-survey/historical-data.

Federal Reserve Bank of St. Louis. FRED Economic Data, 2018. Data retrieved from https://fred.stlouisfed.org/.

Thomas Fischer and Christopher Krauss. Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2):654–669, 2018.

Yoav Freund, Robert E Schapire, et al. Experiments with a new boosting algorithm. In *ICML*, volume 96, pages 148–156. Citeseer, 1996.

Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232, 2001.

Christopher Krauss, Xuan Anh Do, and Nicolas Huck. Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. Technical report, 2016.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. chapter Learning Internal Representations by Error Propagation, pages 318–362. MIT Press, Cambridge, MA, USA, 1986. ISBN 0-262-68053-X. URL http://dl.acm.org/citation.cfm?id=104279.104293.

Robert Shiller. Robert Shiller Online Data, 2018. Data retrieved from http://www.econ.yale.edu/~shiller/data.htm.

Chikashi Tsuji. Do industries contain predictive information for the fama–french factors? *Quantitative Finance*, 12(6):969–991, 2012. doi: 10.1080/14697681003762271. URL `https://doi.org/10.1080/14697681003762271`.

Sudhir Varma and Richard Simon. Bias in error estimation when using cross-validation for model selection. *BMC Bioinformatics*, 7(1):91, 2006.

Kurt Winkelmann, Raghu Suryanarayanan, Ludger Hentschel, and Katalin Varga. Macro-Sensititve Portfolio Strategies. 2013. Available at: `https://www.msci.com/documents/10199/644d6abc-24ad-4d88-bb4f-89779edba408`.

Appendices

A Macroeconomic and Market Variables

Variable Name	Definition
Mkt-RF_prev_month	Previous month's market return minus the risk-free return
RF_prev_month	Previous month's risk-free return
SMB_prev_month	Previous month's SMB return
HML_prev_month	Previous month's HML return
SMB_prev_year	Previous year's SMB return
HML_prev_year	Previous year's HML return
12m_nom_gdp_forecast	Forecast nominal GDP growth over 12 months
12m_ip_growth_forecast	Forecast nominal industrial production growth over 12 months
12m_cpi_forecast	Forecast CPI growth over 12 months
12m_real_gdp_forecast	Forecast real GDP growth over 12 months
12m_real_ip_growth_forecast	Forecast real industrial production growth over 12 months
12m_unemployment_rate	Forecast unemployment rate over 12 months
12m_nom_gdp_forecast_chg_prev_month	Change in forecast nominal GDP growth
12m_ip_growth_forecast_chg_prev_month	Change in forecast nominal industrial production growth
12m_cpi_forecast_chg_prev_month	Change in forecast CPI growth
12m_unemployment_rate_chg_prev_month	Change in forecast unemployment rate
year_of_deficit_pct_gdp	Deficit as a percentage of GDP for a given year
shiller_cape	Previous month's Shiller PE ratio of S&P 500 index
seasoned_corp_bond_yield_prev_month	Previous month's long-dated corporate bond yield
ttm_pe_previous_month	Previous month's PE ratio of S&P 500 index
sp500_return_prev_month	Month-over-month change in the S&P 500 index
sp500_return_prev_year	Year-over-year change in the S&P 500 index
termstructure_slope_chg_prev_month	Change in the slope of U.S Treasury curve (3m/10y)
termstructure_level_chg_prev_month	Change in the level of U.S Treasury curve (average of 3m & 10y yield)
business_confidence_chg_prev_month	Change in the business confidence index
ip_chg_prev_month	Change in the industrial production index
spread_chg_prev_month	Change in spread between BAA-rated and AAA-rated bonds
Div_yield_chg_prev	Month-over-month change in the S&P 500 dividend yield

B Return Summary

B.1 SMB

	Passive	Logistic Regression	Random Forest	GRU
Mean (Monthly Return)	0.0011	0.0026	0.0022	0.0035
StdDev (Monthly Return)	0.0315	0.0314	0.0314	0.0313
Cumulative Return	0.2260	1.1485	0.8029	1.9622

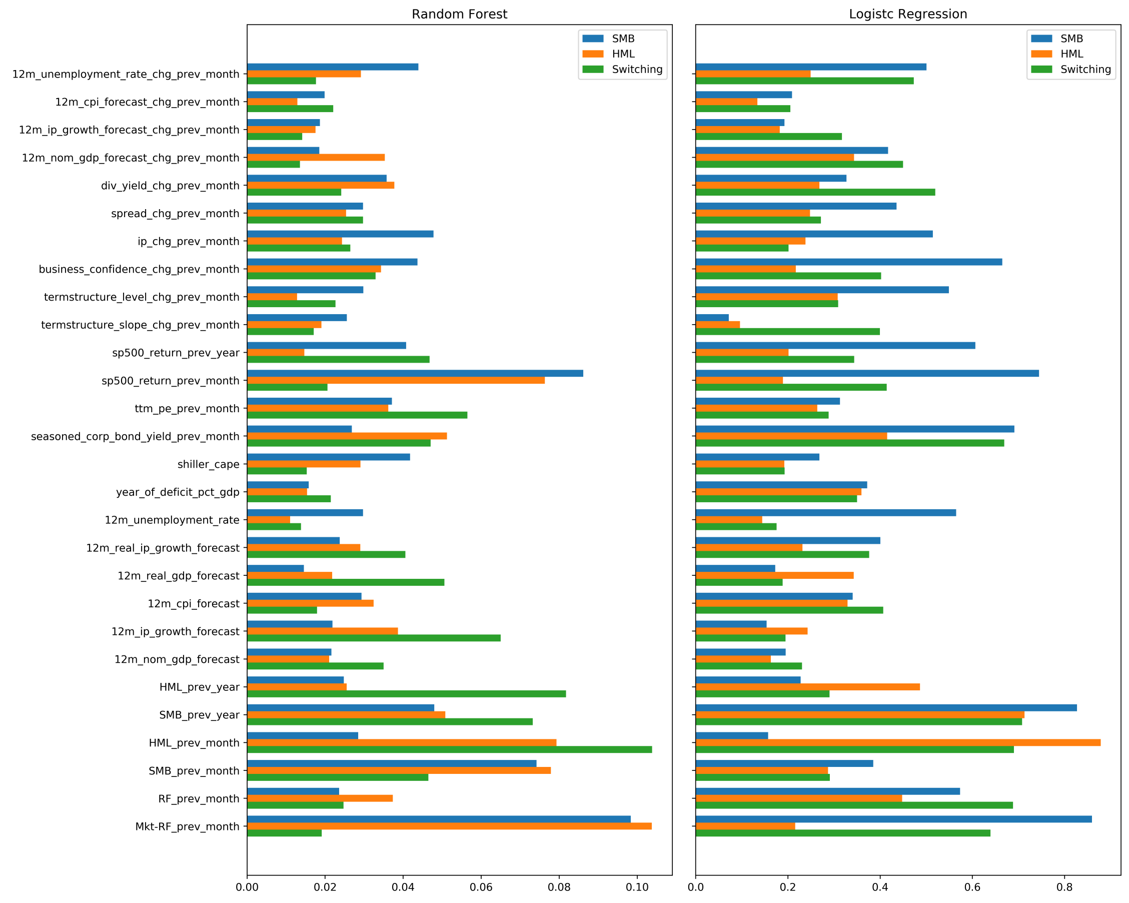
B.2 HML

	Passive	Logistic Regression	Random Forest	GRU
Mean (Monthly Return)	0.0015	0.0017	0.0045	-0.0002
StdDev (Monthly Return)	0.0292	0.0292	0.0289	0.0292
Cumulative Return	0.4845	0.6035	3.3386	-0.1963

B.3 SMB HML Switching

	Passive	Logistic Regression	Random Forest	GRU
Mean (Monthly Return)	0.0013	0.0022	0.0015	0.0034
StdDev (Monthly Return)	0.0185	0.0288	0.0288	0.0307
Cumulative Return	0.4955	0.8767	0.4886	1.8685

C Feature Importances


Random Forest
Logistc Regression
SMB
HML
Switching
12m_unemployment_rate_chg_prev_month
12m_cpi_forecast_chg_prev_month
12m_ip_growth_forecast_chg_prev_month
12m_nom_gdp_forecast_chg_prev_month
div_yield_chg_prev_month
spread_chg_prev_month
ip_chg_prev_month
business_confidence_chg_prev_month
termstructure_level_chg_prev_month
termstructure_slope_chg_prev_month
sp500_return_prev_year
sp500_return_prev_month
ttm_pe_prev_month
seasoned_corp_bond_yield_prev_month
shiller_cape
year_of_deficit_pct_gdp
12m_unemployment_rate
12m_real_ip_growth_forecast
12m_real_gdp_forecast
12m_cpi_forecast
12m_ip_growth_forecast
12m_nom_gdp_forecast
HML_prev_year
SMB_prev_year
HML_prev_month
SMB_prev_month
RF_prev_month
Mkt-RF_prev_month