Pages 1–6
December 2021

Security Selection Using the Fama-French Five Factor Model and Binary Classification

Heng-Chia Liu, Beibei Du, Dominic Ricci, and Sarah Kagzi
CSS 100
University of California - San Diego
December 5th, 2021

ABSTRACT

The age-old question in wealth management is how to generate a portfolio that provides investors higher return than the market. The Fama-French Five Factor model that is traditionally used to predict portfolio return can help us find insight on which security can outperform the market. Our goal is to create an algorithm that identifies the appropriate factor exposure for the next trading period and select stock accordingly. To do so, we will label data depending on if a given stock has exceeded the Sharpe Ratio of the SP&Y 500. This will classify each company as a winner or loser over each time interval. Then, we will use the Fama-French Five Factor model combined with Time-Series analysis using Machine Learning tools to attempt to identify the stocks that will beat the market in the next time period. This prediction is done by selecting the stocks that have similar factor exposures to the winners of the last n time periods. The companies will be selected from the SP&Y 500 Index to account for storage and time considerations. Data will be collected from the Yahoo Finance library and will be gathered in 2 minute intervals for testing. For the purposes of this project, in order to reduce run time, we will be specifically looking at one 2 minute interval dataset, downloaded from Yahoo Finance.

CONTENTS

1 INTRODUCTION

Wealth management is one of the most important factors for people to improve their life quality and stability. Obviously, financial assets are the most convenient and commonly used instruments that provide high liquidity and return on investment. However, after several shocks in the financial markets, "risk" has become the major concern that stops people from investing because a bad investment or speculation can harm the wealth value significantly. Since analyzing financial derivatives can be extremely complicated in this project, we will only focus on stock investments for our quantitative research. Our primary goal is to help investors make a better decision in their security selection and trading process.

In order to find securities that will outperform the market, we will first convert the daily log-return data to daily Sharpe Ratio (a risk/return metric commonly used in finance to assess the performance of an asset) for all S&P 500 components and classify the "market beaters". These market beaters are classified by comparing the Sharpe ratio for a particular stock versus the Sharpe ratio of the whole SP&Y for that time period. Since Fama-French five-factor model(FF5F) explains between 71% and 94% of the cross-section variance of expected returns for 5 fundamental factors[1], we will use the FF5F to estimate the factors exposure and relevant statistics for each of the S&P 500 components.

We concluded that the probability of any given company in the SP&Y to generate a Sharpe ratio higher than the market will eventually converge to 0.5. This is expected due to the law of large numbers. The figure below depicts this phenomena observing Apple's probability of beating the market over time.

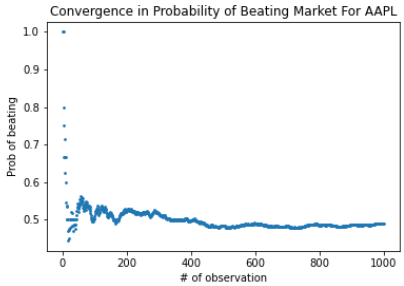

Figure 1. Convergence in Probability

We then incorporate the time-series possibility of beating the market into model construction. The features we construct are the factor exposures, statistical significance measurements, and time-series possibility of beating the market. Our target variable is the binary

outcome of the next trading period indicating whether a security beat the market in that time period. We will use precision score as accuracy metric for prediction because we don't want any losing company to show up in our predicted winner's club. In the project, we use the performance of the S&P 500, which contains the 500 largest publicly traded companies in the U.S. as the benchmark for market beaters. The reason for adopting the S&P as the benchmark is that if we can't select an investment that can outperform the market portfolio, we can just simply invest in the market portfolio (the S&P) instead.

2 FAMA FRENCH

Fama French five factor model captures the average return in a portfolio of securities, which was developed from the three factor Fama French model. The coefficients: s_i,h_i , h_i,r_i, and c_i are the five factor exposures. These exposures are extremely important for our time series analysis in section 4. This model is using the regression model to fit the factors into the model. And the factor number could be either 3 or 5. According to the paper, the three factors model is repetitive and the 5 factors model is more efficient on the average return. What we have used is to have 5 factors, in the Fama French model. In the 5 factor model, it includes "6 value-weight portfolios formed on size and operating profitability, and the 6 value-weight portfolios formed on size and investment". The formula for five factor Fama French model has been written in (1). The explanations of the factors will be below[1].

This is the five factor Fama French model equation:

$$R_{i_t} - R_F = a_i + b_i(R_{M_t} - R_{F_t}) + +s_i SMB_t + h_i HML_t + r_i RMW_t + c_i CMA_t + e_{i_t} \quad (1)$$

[1]

2.1 Fama-French Data Frame

Although there is a section reserved for feature engineering, we found it most sensible to include the engineering of the Fama-French data frame in this section. This data frame will be used as inputs into the iterative regression function. This data frame will help us separate companies based on certain metrics, which will then provide insight on which factors are the most important when deciding what stock will beat the market.

Datetime	SMB	HML	RMW	CMA	Rm-Rf
2021-10-11 09:32:00-04:00	-0.001462	-0.003179	0.001835	-0.000872	0.001503
2021-10-11 09:34:00-04:00	-0.005193	-0.002111	0.001227	-0.000469	-0.001025
2021-10-11 09:36:00-04:00	0.004055	0.001614	-0.001921	0.001213	0.000136

Figure 2. Fama French Factors Data Frame (4874 rows × 5 columns)

2.1.1 Data

We collected fundamental data from annual reports of all companies in the S&P 500 from Yahoo Finance through an unofficial API "yfinance". The financial information we extracted are annual Market Capital(Size), Price to Book Ratio(Value), Return on Equity(Profitability), and Asset Growth Rate(Investment). The companies that have higher market capital then the median will be classified as Big (B), and that have lower market capital than the median will be classified as Small(S). For Value, Profitability and Investment, the companies obtain values higher than top 25 percentile will be classified as High(H), Robust(R), and Aggressive(A), and those obtain values lower than bottom 25 percentile will be classified as Low(L), Weak(W), and Conservative(C), respectfully.

	mkt_cap	PB_ratio	ROE	Asset_growth
MMM	1.028818e+11	7.120830	0.46612	0.060122
ABT	2.090332e+11	6.184308	0.19658	0.068658
ABBV	1.959273e+11	15.586954	0.48855	0.689558

Figure 3. Information Used to Calculate Factors (505 rows × 4 columns)

2.1.2 $R_m - R_f$

The difference between the return of the market and risk-free rate. Regarded as the market factor. The estimated coefficient represents how sensitive a security's return is to market return movement.

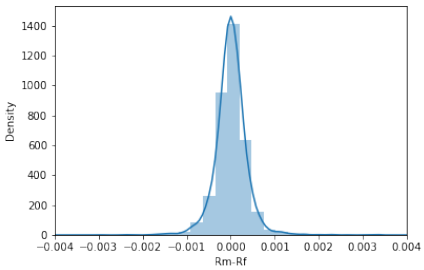

Figure 4. Rm-Rf Distribution Follows Standard Normal Distribution

2.1.3 SMB

Return difference between small stocks and large stocks, regarded as the size factor. We estimate this variable for an individual security by evaluating the market cap of the security specified. An example of where this is insightful is when the market at a certain time prefers small or large companies explicitly. If the s_i coefficient is negative this indicates that securities with smaller market caps perform better than those with larger market caps.Or for simplicity, we could interpret it as small minus big.

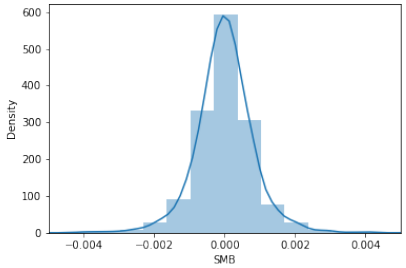

Figure 5. SMB Distribution Follows Standard Normal Distribution

2.1.4 HML

Return difference between cheap stocks and expensive stocks, regarded as the value factor. The coefficient of HML is how much of the security return can be explained by the value factor. To determine this variable we use the price-to-book ratio that compares a companies market value relative to its book value. This could be interpreted almost as the "true" value of a company, often used to identify under/over-valued securities. If the h_i coefficient is negative this indicates that undervalued companies perform better than others. Or for simplicity, we could interpret it as high minus low.

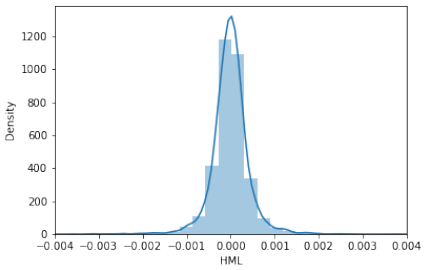

Figure 6. HML Distribution Follows Standard Normal Distribution

2.1.5 RMW

Return spread of profitable and less profitable firms, regarded as the profitability factors. Return on Equity is used as the metric to classify profitable companies and less profitable companies. The differences between the two are known as the RMW. Or for simplicity, we could interpret it as robust minus weak.

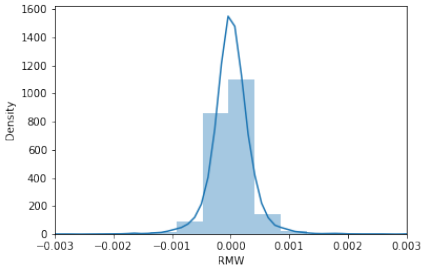

Figure 7. RMW Distribution Follows Standard Normal Distribution

2.1.6 CMA

Return spread of firms that invest conservatively minus aggressively. regarded as the investment factor. Asset growth rate is used to measure investment level. In short, the difference between the return on the conservative investments and the aggressive investments are known as CMA.

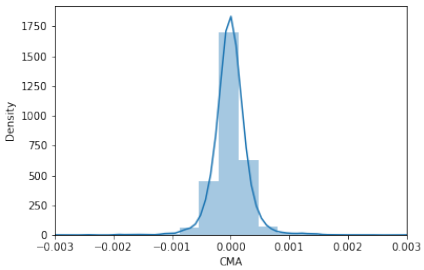

Figure 8. CMA Distribution Follows Standard Normal Distribution

2.1.7 Correlation Between 5 Factors

In the 2 minute return scale, RMW shows a strong correlation between CMA and HML. As we suspect multi-collinearity will exist in linear regression model, we will also implement Ridge, Lasso, and Elastic Net with optional polynomial features to estimate the factors exposures (Coefficient of the 5 Factors).

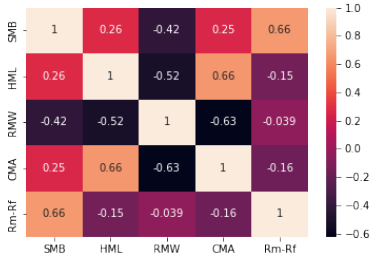

Figure 9. Correlation Between 2 min Fama French Factors

3 FEATURE ENGINEERING

3.1 Finding Market Beaters

Our first goal was constructing a data frame which identified what securities in the SP&Y 500 beat the market in a time interval. To do this we collected data from the yfinance library of stock return percentages over a 2 minute interval using the percent change formula below.

$$R_{t,i} = \frac{r_{t-1,i} - r_{t,i}}{r_{t-1,i}} \qquad Sharpe_{t,i} = \frac{R_{t,i} - R_f}{\sigma_i} \tag{2}$$

This formula is used to compute security returns and sharpe ratios. Each 'i' index denotes an individual security, while each 't' denotes a specific time interval. We then compute for the market as a whole over out 2 minute data frame when i=m. We then return dummy variables based on the comparison below.

$$\text{if } Sharpe_{t,i} \leq Sharpe_{t,m} \implies d_{t,i} = 0 \qquad \text{if } Sharpe_{t,i} > Sharpe_{t,m} \implies d_{t,i} = 1 \tag{3}$$

These dummy-variables will then be used in the time-series model in order to train and test our data frame.

3.2 Iterative Regression Methods

As described in the previous section the Fama-French model is extremely reliable when it comes to estimating return based on market factors. This is why our algorithm relies on an iterative regression function. This function attempts to find the values of the coefficients of $b_{t,i}$, $s_{t,i}$, $h_{t,i}$, $r_{t,i}$, and $c_{t,i}$ from the last 150 observations, then stores them in a data frame with their respective performance values. These coefficients can be determined by the four models below.

3.2.1 Linear Model We use scikit learn's linear model package to estimate the coefficients $b_{t,i}$, $s_{t,i}$, $h_{t,i}$, $r_{t,i}$, and $c_{t,i}$. To do so, we run the regression.

$$R_{t,i} - R_F = a_{t,i} + b_{t,i}(R_{t,m} - R_F) + s_{t,i}SMB_t + h_{t,i}HML_t + r_{t,i}RMW_t + c_{t,i}CMA_t + e_{t,i} \quad (4)$$

The coefficients and their respective p-values are stored in a data frame which will be used in section 4 to find the market winners.

3.2.2 Other Models: Lasso, Ridge, Elastic Net We also used other linear models to estimate coefficients, the Elastic Net model with interaction polynomial features transformation performs the best. We believe this result is due to following reasons.
1. Multi-colinearity exists between Fama-French return factors.
2. Linear relationship is not enough to explain return by its factors.

4 TIME SERIES MODEL

Our goal is to predict whether a given security or portfolio will outperform the market in the next trading period. To do so, we estimate factor exposure at a certain time(t_n) considering the most recent observations only, then use the exposures and relevant statistics to predict the outcome at t_{n+1}. By Figure 1, it is clear that the probability of a given stock beating the market will converge to 0.5 as time goes on. To account for this realization we include the realized probabilities that are observed from $[t_{n-10}, t_n]$, $[t_{n-20}, t_n]$, and $[t_{n-30}, t_n]$. We will roll the model and make prediction for at least 10 trading intervals, and will collect accuracy metrics of time-series prediction.

In order to take advantage of multiple classifiers, we use ensemble learning in order to use classifiers in combination. Since ensemble learning works best with a diverse set of classifiers, we chose the Stochastic Gradient Descent Classifier, the Logistic Regression Classifier, the Support Vector Machine Classifier, and the Random Forest Classifier [2]. We explain our choice and the specific advantage each of these brings to our project in the subsections below. Finally, we use the Voting Classifier and Bootstrap Aggregation to combine these classifiers into our final classifier (explained in Section 4.5).

4.1 Stochastic Gradient Descent Classifier

The SGD (Stochastic Gradient Descent) Classifier is one that takes the training set and calculates the gradient based off of one randomly selected instance at every iteration. We chose to use the SGD because unlike the Batch Gradient Classifier, it works well on large datasets, and unlike the Mini Batch Gradient Descent Classifier, it is good at avoiding local minima - this is a crucial aspect because of how textured the financial stock data can be.

4.2 Logistic Regression Classifier

Logistic Regression Classifier estimates the probability of one binary classifier belonging to a certain class. If the probability is greater than 0.5, it returns 1, and if not, it returns 0. This is extremely useful for our project because it helps us classify into different classes (which in our case is whether or not individual instances of stocks perform better than the average). Other than the fact that it's a binary classifier, we also value the fact that its classification is based on the minimization of the cost function, the "log loss". As mentioned in the textbook, "Hands-On Machine Learning", while this cost function does not have a specific equation for minimization, the partial derivatives (Equation 5) give a gradient which we can then use the SGD classifier with in order to get a more accurate classifier for the project.

$$\frac{\partial}{\partial\theta_j}J(\theta) = \frac{1}{m}\Sigma[\sigma(\theta^T x^i) - y^i)x_j^i] \quad (5)$$

4.3 Support Vector Classifier

SVM is a binary classifier that performs classifications on multiple binary classifiers. We chose to incorporate an SVM classifier because we wanted to take advantage of the powerful nature of the Gaussian RBF Kernel. Since we are working with a large and complex dataset, from the computational and run-time sense, it would be expensive and time consuming to use a similarity features method, but the Gaussian RBF allows us to obtain a similar result without adding those similarity features. Thus, for the purposes of our project, it was a powerful tool that we wanted to incorporate in our classifier.

4.4 Random Forest Classifier

Random Forest Classifier also classify binary variables to different classes by the hidden decision trees. [3](Pg 225). Through the bagging method, we have many sets of training set and put them into the Decision Tree Classifier, and the hyper-parameters will control how the algorithm ensemble itself.After running multiple Decision Trees, we will output one final result from the previous Decision Tree Classifiers. And this is the reason why this algorithm is called Random Forest Classifier. The Decision Tree Classifiers are multiple questions that require binary answers. And the max_depth is dependent on how many questions we have to ask. The bagging and random feature generated from the algorithm allows for the maximized randomness and be representative. A more accurate result will be generated by the multiple Decision Tree Classifiers.

4.5 Ensemble Learning

First, we use four bagging classifiers on the aforementioned four models, and then by utilizing voting, we combine the four classifiers with bagging and without bagging to get one final classifier. This is called the wisdom of the crowd according to the book [3].Thus we can see how multiple classifiers can benefit the final result and a better result is generated by ensemble some possible algorithms.

4.5.1 Bagging (Bootstrap Aggregating) In order to diversify our classifier, we decided to use Bagging to train the classifier on different subsets of the training set. We chose bagging over pasting because while both bagging and pasting permit that training instances be picked up in the sample several times for many predictors, bagging additionally allows for the same predictor to pick up training instances in its sample multiple times. The Bagging classifier aggregates the predictions of each of the predictors (SGD, Logistic Regression, SVM, Random Forest) by taking the statistical mode of the predictions. As mentioned in the textbook, "Hands-On Machine Learning", aggregating using the bagging classifier reduces variance compared to the original predictors by themselves[2]. Thus, with the bagging classifier, our model will generalize better than just the predictors alone. However, despite all these benefits, since bagging diversifies the subsets that each predictor is trained on multiple times, there is a risk that bagging increases the bias. Thus, in order to protect our model from this, we also use voting classifiers to choose the best option - with bagging or without bagging.

4.5.2 Voting Classifier We use the Voting Classifier to combine multiple models that we have used into one to predict the values. We decided to incorporate voting because as mentioned in the textbook, "Hands-On Machine Learning", this relies on the Law of Large Numbers to arrive at the result that even with weak predictors that are only slightly better at predicting than a random guess, by correctly predicting the class with the majority "votes", the model can still reach an accuracy as high as 75% . Since our initial classifiers are very different from each other in their methods and technique of classification, we can rest assured that they will not make correlated errors, thus preventing a majority vote

going to the wrong class. Moreover, to make our model more robust, we use *soft voting*, which selects the class with the highest probability since this adds more nuance to the voting classification procedure by giving more weight to the votes we know are accurate. In this way, we ensured that our ensemble's accuracy is high, and while calculating the precision and recall of our model, we got precision in the range of 0.61 - 0.65 (with the highest F1 score being 0.60).

5 CONCLUSION

To sum up, we found that the factors exposures are consistently useful determinants to predict the market beaters after we use 4 regression models and optional polynomial features transformation to estimate the factors exposures. While keeping the parameters and classification algorithm constant, all 6 different feature estimations result in at least 0.6 precision score on average in their 100 testing trials. Estimating features using Elastic Net with polynomial features provide us the highest average precision (0.645), whereas using plain Elastic Net provide us the lowest average precision (0.615).

	count	mean	std	min	25%	50%	75%	max
Linear	100.000000	0.626702	0.125047	0.125000	0.567441	0.643719	0.706163	1.000000
Lasso	100.000000	0.640537	0.129128	0.333333	0.563189	0.647059	0.731012	1.000000
Elastic	100.000000	0.614607	0.127436	0.000000	0.563590	0.631579	0.706196	0.818182
Ridge	100.000000	0.627760	0.114256	0.333333	0.571894	0.641893	0.709202	0.833333
Ridge_w_poly	100.000000	0.615665	0.140617	0.000000	0.535866	0.618615	0.704893	1.000000
Elast_w_poly	100.000000	0.645330	0.138658	0.000000	0.562158	0.663176	0.750619	1.000000

Figure 10. Summary of Precision Scores for different Feature Estimates

By looking at the precision scatter plot of all models, we didn't find any evidence of serial correlation between trials on all models, so we believe our accuracy score is robust over time.

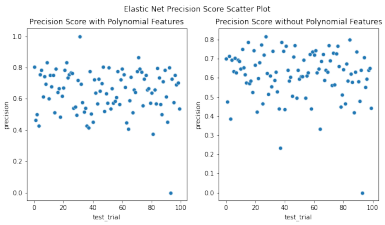

Figure 11. Precision Score Scatter (Best Performer and Worse Performer)

Although we observed a trade off between the level of precision score and its standard deviation, which means that well-performing estimators generally associate with instability, such an instability is mainly due to very small amount of outliers. Moreover, by comparing the best and worse performers, we concluded that the model using Elastic Net with polynomial features perform the best since its distribution is more centered and yield better accuracy while obtaining the same amount of significant outliers.

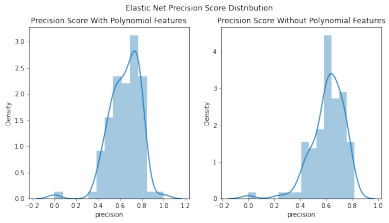

Figure 12. Precision Distribution of Both Models

REFERENCES

[1] Eugene F. Fama and Kenneth R. French. A five-factor asset pricing model. 2013.

[2] Géron and Aurélien. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*, chapter 4.1, 4.5. O'Reilly Media, 2 edition, 2019.

[3] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, 2019.