Machine Learning Winter 2020

Support Vector Machines on Mutual Fund Returns

Introduction

Individual investors struggle to grasp the concept on what strategy one should use to save for their retirement, house, and more. The common default strategy is to invest into a 401K or Roth IRA to save and grow your wealth and start to incrementally withdraw it when you reach a certain age, typically at 59 ½ years old. However, there is a dilemma for individuals who wish to save and grow their wealth on a shorter time frame, 5, 10, 15 years, and that timeframe would not be beneficial for a 401K. This can leave the financially uneducated investor puzzled and wondering what the various routes are they could pursue to achieve their goals. One solution to this problem is to instead invest into an index fund, mutual fund, real estate, bonds, and other markets where this is no heavy penalty for withdrawing your funds in a short time frame. The purpose of this is project is to dive into the details of what characteristics lead to successful mutual fund returns from 2016-2018 (3 years of annualized average returns) so that an investor can gain insight into which mutual funds will achieve the best results if they were to pursue a mutual fund investment route.

Related Work

Other traditional investing methods as below have addressed these similar questions:

Professional management - You don't have to keep track of every security your mutual fund owns. The fund is managed by experts who take care of that for you.

Convenience - You can buy and sell mutual fund shares online or by phone and set up automatic investments and withdrawals.

Intelligent Portfolio - Investing in mutual funds traditionally offers benefits that we will not get from trading individual stocks and bonds on our own. It has less risk through more diversification. One mutual fund can invest in hundreds, sometimes thousands of individual securities at once. So, if any one security does poorly, the others are there to help offset that risk. It also lowers costs. With a no-load mutual fund, we pay one expense ratio, instead of racking up the commissions we will pay when buying and selling individual securities ourselves.

Our method is focused on SVM algorithm whereas above traditional methods have wider depth and breadth of human intervention and professional experience.

Data & Experiment

The data that was used was provided by Kaggle Datasets and was a 25,308 x 125 dataset containing 25K+ number of mutual funds and 125 number of financial variables that was scraped from Yahoo finance in 2018. Due to the large number of variables, some were immediately deleted before preprocessing due to their non-relevant characteristics of the intended goal (fund name, fund family, year of inception) or due to that they were functions of returns (sharpe ratio, beta, and other statistical ratios).

Variables that only resembled data for the year 2018 were also dropped due to that they have no relevance from past returns. One exception, and reasonable assumption, was that the weighting values for different sectors in the market (healthcare, technology, etc) and asset allocation values (stocks, bonds, real estate) did not change too much. The reason as to why this assumption is made is because mutual fund managers will most likely have a strategy and would not deviate dramatically from it in a short time frame of three years. Variables were then further deduced through a correlation matrix, shown in table 1, where the top seven variables correlated with three-year mutual fund returns were abstracted to build a model. We believe correlated values with mutual fund returns will be useful in building an accurate model.

Table 1: Top correlated variables with three-year fund returns

Variable Selection	Pearson Correlation Coef
Technology	0.619
Portfolio Stocks	0.574
Investment Growth	0.412
Healthcare	0.401
Consumer Cyclical	0.326
Financial Services	0.24
Size Large	0.229

After extracting the top seven features, three year fund returns was then converted into binary classification (1,0) based on S&P market 500 returns during the same time frame plus additional fees associated with mutual funds that investors would have to pay out of pocket. In the histogram below, the average three year returns for mutual funds in the data set was 7%. During that time, the S&P 500 returned 7.57% meaning that S&P 500 index, on average, beat mutual fund managers. However as previously mentioned, investors would need much higher returns to be more profitable than the market after fund fees are accounted for. The average net annual expense ratio in 2018 for mutual funds was 1.12%, and since there was no data on load fees, we assumed an additional 3% load fee for all mutual funds. Thus, for a mutual fund to be considered “superior”, or labeled 1, they would need to return ~13% or more while those who were <13% would be considered non-superior, or 0, and would not be funds worth investing in.

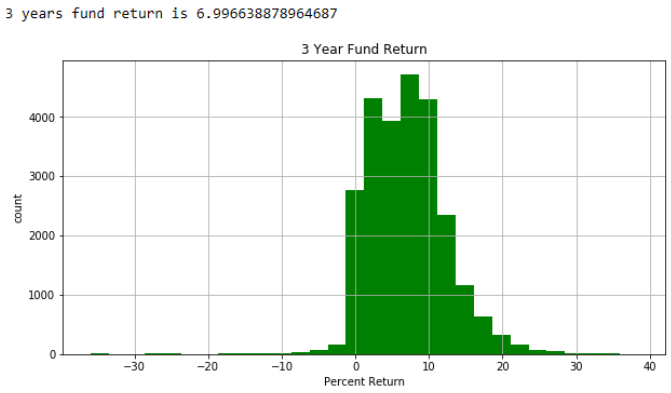

Figure 1: Histogram of three-year fund returns of all mutual funds

Once the conversion was complete, it was observed that 22,179 mutual funds would not be worth investing in (0’s) while only 3,012 mutual funds out of 25K returned >=13% (1’s). Due to this lop sidedness in classification, a random fraction of 13.5% was taken from the mutual funds labeled 0 so that the final data frame had equal 1’s and 0’s; in total 6,022 mutual funds in the final data frame.

Outlier Analysis

Outliers were analyzed by using box plots for features: three-year fund returns, technology, portfolio stock, consumer cyclical, financial services, and healthcare weighting. Outliers were eliminated from three-year fund returns for anything that returned over 30% or less than -20%. This was due to keeping realistic mutual fund returns or losses to investors. An assumption was also made to keep mutual funds that were not fully invested in one particular sector, i.e, some diversification.

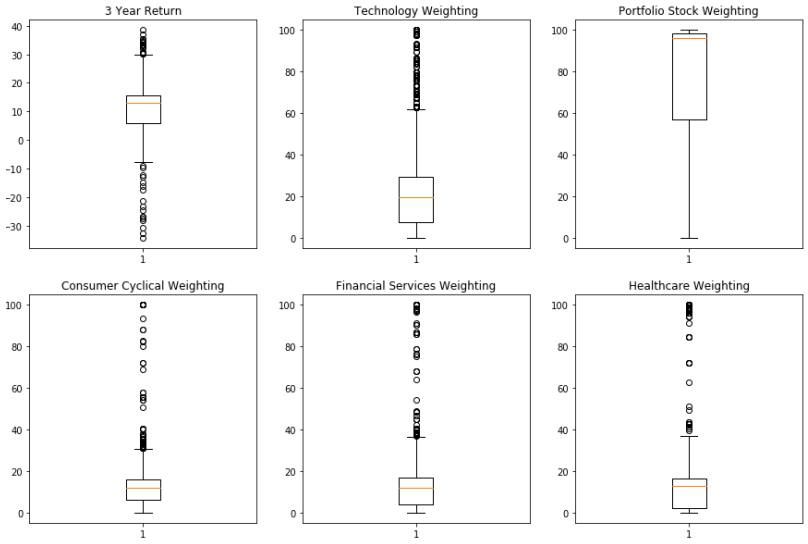

Figure 2: Box plots of features to depict ranges and outlier

Scale/Normalize

Scaling was done to keep all variables in the ranges from 0 to 1 by using sklearn preprocessing package and MinMaxScaler to scale. This was due to that variables investment growth and fund size were binary whereas the rest ranged from 0 to 100. Since investment growth and fund size were not used in the final model, scaling may not be necessary and may not affect the final outcome.

NA's Analysis

There were hardly few NA's in the data set, where column fund returns three-years had the most NA's that equate to 0.13% of the data. Since there was significant overlapping of NA's in the rows for various funds and that the amount of data that would be dropped was small, it would make sense to drop all rows with NA's from the dataset rather than imputing the median for all columns. This ended up being a 0.37% reduction in the data.

	fund_return_3years	technology	portfolio_stocks	investment_Growth	consumer_cyclical	financial_services	size_Large	healthcare	return_binary
5332	5.45	NaN	NaN	0	NaN	NaN	1	NaN	0
8528	NaN	NaN	NaN	0	NaN	NaN	0	NaN	0
16982	-1.29	NaN	NaN	0	NaN	NaN	0	NaN	0
17043	NaN	0.00	0.00	0	0.00	0.00	0	0.00	0
20237	14.92	NaN	NaN	1	NaN	NaN	0	NaN	1
17691	NaN	0.00	0.00	0	0.00	0.00	0	0.00	0
8429	NaN	14.57	48.22	0	13.04	16.66	0	11.90	0
2114	19.81	NaN	NaN	1	NaN	NaN	1	NaN	1
9509	NaN	1.27	2.22	0	31.25	0.00	0	6.63	0
22156	NaN	18.17	95.80	0	10.01	20.62	1	7.79	0
18212	NaN	NaN	NaN	0	NaN	NaN	0	NaN	0
1763	NaN	29.39	49.25	0	5.78	15.18	1	0.38	0
17051	NaN	NaN	NaN	0	NaN	NaN	0	NaN	0
21726	NaN	NaN	NaN	0	NaN	NaN	1	NaN	0
16138	NaN	0.00	0.00	0	0.00	0.00	0	0.00	0

Figure 3: NA's for various features

Solution/Method/Model

The final data frame consisted of scaled features: technology, portfolio stock, consumer cyclical, financial services, and healthcare weightings. The label used was return binary (1's for >= 13% return and rest 0's) and columns fund return 3 years, which contained the numerical fund return values, investment growth, and size large were dropped. Binary variables investment growth and fund size were dropped due to that mixing of binary and numerical weighted features may cause inaccuracy for this model. *It is important to note that same procedure was followed in spark, however all tables, plots, and graphs shown will be from scikit-learn.* Once the data frame was finalized, training and test data were then computed through sklearn.model_selection with a 70/30 training and testing split.

The model that was focused on for this project was Support Vector Machines (SVM), however Stochastic Gradient Descent (SGD) classifier and Random Forest ensemble was also used for comparison. SVM was initially ran on the five-feature model as well as on a two-feature model that was derived from a probabilistic importance feature measurement. This is calculated by calculating the increase in the model's prediction error after permuting the feature. A feature is "important" if shuffling its values increases the model error, because in this case the model relied on the feature for the prediction. Shown below, consumer cyclical and technology have a heavy influence on the model's prediction, where interestingly consumer cyclical is considered ~2.5x more important than any other feature. A similar procedure was analyzed via Random Forest to compare feature importance, which also showed to have the same results. Thus, the final two-feature model consisted for features: consumer cyclical and technology weighting.

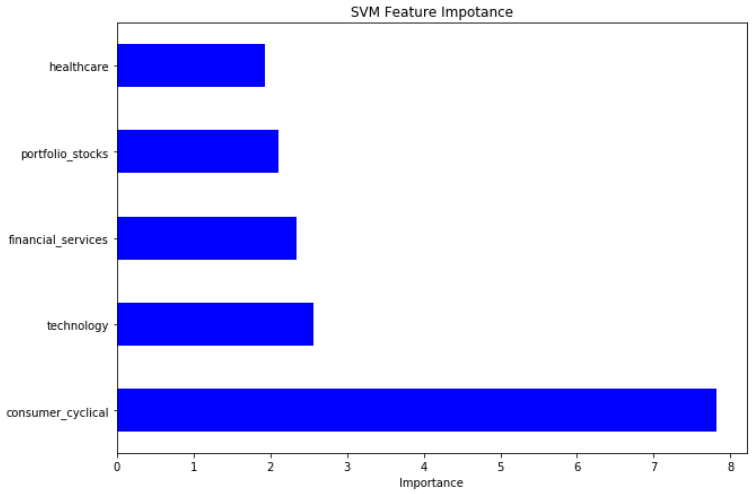

Figure 4: Probabilistic model to measure feature "importance"

Optimization was also attempted by using a method known as "C Grid Search" from the sklearn.model_selection library. This method will allow a user to specify ranges to test for various metric combinations, in this case for SVM, type of kernel, C parameter, and gamma parameter. It will then select a combination of variables that maximize recall score. Note that spark unfortunately did not have this option available for use. C grid search did show to improve both models where the optimized five-features model showed to have all around higher scores for accuracy, precision, recall, and AUC. Lastly, an attempt was done to improve accuracy on the two-feature model by removing areas, or "outliers", of misclassification in the model, shown below, and then utilizing a C grid search. This attempt did not end up leading to any improvement. Since spark did not have C grid search nor non-linear kernels to test, the only optimization attempted was by varying the C parameter.

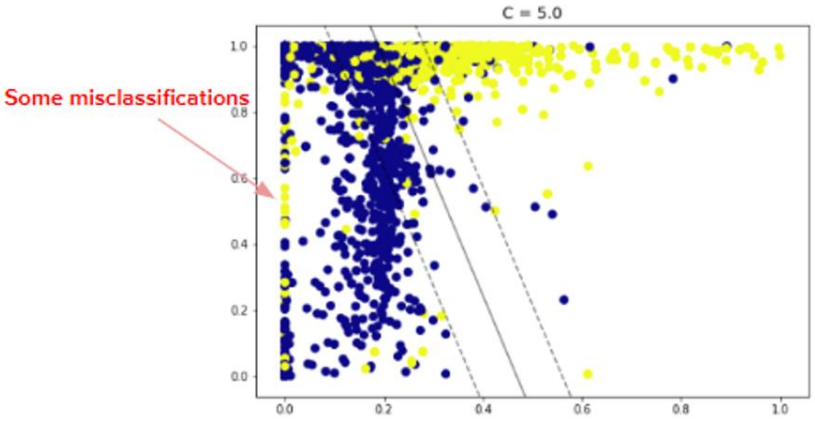

Figure 5: SVM plot for two feature analysis

Evaluation and Results

The results for SVM, SGD, and Random Forest were compared amongst one another and evaluated by sklearn.metrics library, which included: accuracy, precision, recall, and AUC. Initial attempt of SVM(kernel = 'Linear', C = 1) showed to have all metrics ~85%, which is fairly high. After utilizing C grid search, the SVM model suggested a non-linear kernel SVM(kernel = 'rbf', C = 10, gamma = 3) and

metrics improved to ~88%. The abstracted two-feature model was then tested with C = 5 and C = 0.1, as well as with C grid parameter tuning. Shown below, once can visually see how the cost of misclassification of C (low C being low bias, and high C being high cost for misclassification), and different kernels and can impact metric scores. Gamma, also highly important, is associated with non-linear kernels or radial kernels and will influence the decision boundary depending on the weight of gamma. However, after optimization the two-feature model was slightly les superior than the optimized five-feature model.

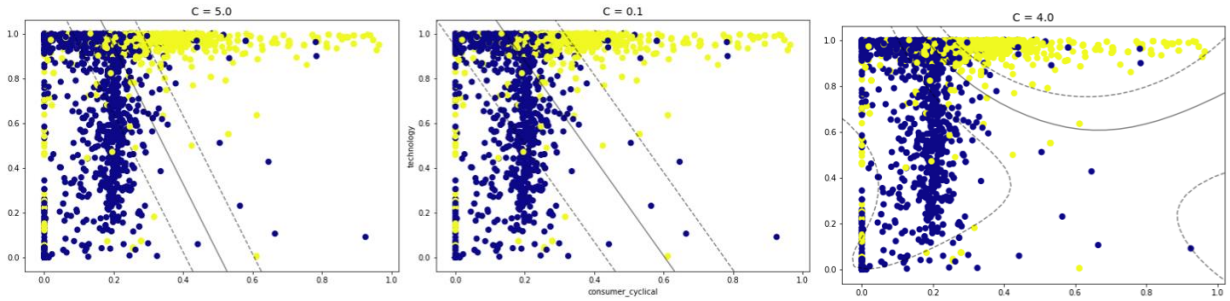

Figure 6: SVM Models for different C values and optimized C grid search for the last plot

When testing the spark SVM algorithm, a C parameter of 0.1 was used as well as various C values for the two-feature model. It is important to note that the cost of misclassification in spark is different than scikit, where C in spark is an inversed scaled relationship to C in scikit. Results showed that the five-feature model in spark scored ~87% for all metrics and was just slightly lower than the optimized five-feature model in scikit. When varying C, a C value in the range of 0.8-1 showed to score the highest metrics for the two-feature model, however this was not an improvement from the original five feature model, *results summarized below.*

Table 2: Metric scores for all scikit and spark models ran

Scikit Models

	Model 1	Model 2 (c grid)	Model 3 (2 features, c = 5)	Model 4 (2 features, c = 0.1)	Model 5 (c grid, 2 features)
Accuracy	85.9%	88.1%	86.3%	85.8%	85.2%
Precision	85.9%	87.3%	84.8%	82.1%	85.5%
Recall	85.4%	88.7%	88.1%	91.1%	90.4%
AUC Score	85.6%	88.1%	86.4%	85.8%	83.9%

Pyspark 5 Feature Models

regParma "C"	0.1	0.5
ROC	92.3%	92.1%
Accuracy	86.7%	87.0%
Precision	87.2%	87.3%
Recall	86.7%	87.0%

Pyspark 2 Feature Models

regParma "C"	0.01	0.05	0.1	0.8	1	2
ROC	90.6%	90.5%	91.0%	90.3%	90.0%	90.4%
Accuracy	84.8%	84.1%	78.6%	85.0%	85.1%	62.7%
Precision	84.5%	84.3%	80.4%	85.0%	85.1%	76.7%
Recall	84.8%	84.2%	78.6%	85.0%	85.1%	62.7%

When confirming scores of the optimized five-feature model with SGD and Random Forest, SGD scored at lower accuracy, AUC, and recall with a score of 85, 78, and 85% respectively. However, Random Forest showed to be more reliable than SVM scoring higher in every metric category (accuracy: 92%, precision: 91%, recall: 94%, AUC: 92%). SGD and Random Forest were not analyzed in spark. To gain further insight as to how the algorithms based their decisions, three-year fund returns were plotted

for superior (1's) and non-superior (0's) funds and a RandomizedSearchCV from scikit was applied to the Random Forest ensemble to abstract the "best" tree. From the three-year fund distribution, the distribution shows that superior funds had a center distribution of ~18% for consumer cyclical, 30-35% in technology, and heavily weighted in stocks whereas non-superior funds were distributed at much lower weightings. The Random Forest tree shows what level of weightings one should follow to explicitly see how to predict a superior fund vs non-superior. When closely looking at the two, similar weighting of the distribution means are used to in the decision tree.

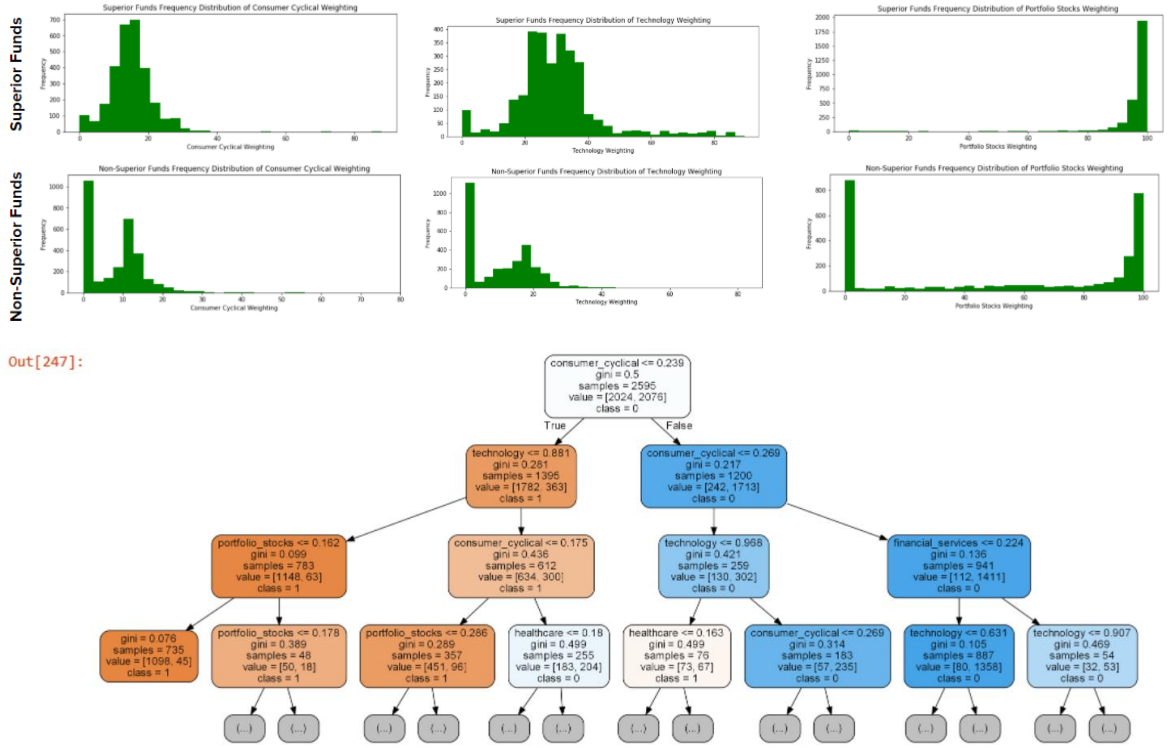

Figure 7: Distribution plots for superior vs non-superior funds for individual features. Best tree from Random Forest shown below

After analyzing the various distributions and weightings from the mutual fund data and Random Forest, one could conclude that funds that outperformed the market from 2016-2018 had characteristics of higher weight in consumer cyclical, technology, stock allocation, healthcare, and lower weighting in financial services. Although this is true, this model does not predict the future. This model can be useful and implemented if investors did their own research and conclude that these markets will continue to be growing and profitable into the future.

Conclusion

When analyzing mutual fund returns that the beat the market after fees, there was 3,012 funds out of 25K funds in America, or just 12% of funds in America. These odds are not comforting for investors, and they may be better off putting their savings in the S&P 500 index to grow over the long term. However, if investors believe that there will be continued success in technology, healthcare, and consumer cyclical, they may consider investing in mutual funds that are more weighted in these sectors. By using the SVM model investors could input specific weights for each feature and receive an answer if this fund will be superior or non-superior with ~88% accuracy.