Asset Management

DESIGN DOCUMENT

Team 13
Principal Global Investors - Client
Chinmay Hegde - Advisor

Carter Scheve - Communications Lead
Nathan Hanson - Project Progress Tracker/Manager
Caleb Utesch - Meeting Scribe
Jack Murphy - Research Analyst
Samuel Howard - Lead Engineer
Alex Mortimer - Project Manager

sddec18-13@iastate.edu
http://sddec18-13.sd.ece.iastate.edu/team.html

Revised: 04/16/2018 Version 3

Table of Contents

List of figures/tables/symbols/definitions

Tables:

Figures:

Definitions:

PGI: Principal Global Investors

BK_P: Book to Price

X12M_Ret: Investment's Twelve-Month Return

NaN: Not a Number - any non-numerical value in the dataset. Missing data can be considered this.

1 Introduction

1.1 Acknowledgement

Our clients at Principal have extensive experience in big data analysis and machine learning and have made it clear that they will are willing to help us in any way possible. They have already helped by providing access to learning resources such as tutorials and readings on topics like the ones mentioned above.

Our faculty advisor, Chinmay Hegde, will also be a valuable asset to developing the proposed product. As an expert in data processing and machine learning, he provides advice and guidance towards what we need to learn to deliver a successful product.

1.2 Problem and Project Statement

Investment analysis at Principal Financial Group currently relies on human calculation, using a variety of models and inputs. These statistical models are proven and effective, although the dependence upon human-given inputs and calculations is both inefficient and unreliable. Various steps of the statistical analysis process can be automated, which would remove most of the potential for human error, and reduce overhead costs by making accurate statistical modeling and prediction more accessible.

Our proposed solution makes use of our extensive background in computational sciences to implement a software approach to multi-factor statistical analysis. We aim to create a system which aids in the creation and management of a profitable investment portfolio based upon well-defined statistical models and machine learning algorithms. Such a system would not only increase profits for portfolio owners, but it would also reduce risk by eliminating erroneous human action and increasing decision-making speed in a volatile stock market. The requirements given to us for the project are quite open, so our solution goals are open as well, but we have enough to start towards a functioning deliverable.

1.3 Operational Environment

The product is expected to be run on a device using Python version 3.6 or higher. Additional libraries required include Scikit-learn, Pandas, and Numpy. No environmental hazards are expected beyond those inherent in running a computer.

1.4 Intended Users and Uses

Our product has two potential user groups. The first is the investment analysts at Principal, who have little to no experience with programming techniques. The other user group is the data analysts that are employed by Principal, who have a large amount of programming experience, along with statistical analysis and data mining.

The ideal intended use of our final product will be to provide a forecasting model that can predict with 50-60% accuracy whether or not a factor will outperform the current market. Another potential use of our final product could come even if our models are unsuccessful in predicting market behavior. Not being able to generate accurate models with the data that we were given is telling enough, and could be further analyzed to identify potential changes that would be beneficial to make in Principal's current forecasting model.

1.5 Assumptions and Limitations

Assumptions:

- Provided data will be of a consistent format
- Provided data will be valid and accurate
- Data will be open and accessible
- Users will have a working understanding of input and output data
- Models in chosen libraries are statistically valid so will not be tested

Limitations:

- Tools used to research and produce model shall not exceed $50
- The model shall be written primarily in Python
- Final predictive model must use machine learning techniques

1.6 Functional Requirements

- Models report processing time and accuracy
- Results are displayed in a human-readable format
- Models only use data from a certain time period to predict future behaviors
  - Example: The model does not use stock market data from 2005 to make predictions on the stock market for 2004

Summary of models gives concrete statistics for performance of each individual model, along with a comparison of each and a recommendation for which to use in similar future tasks.

1.7 Expected End Product and Deliverables

Table 1 End Product and Deliverables

Deliverable	**Description**
Software Models	The main deliverable of our project is a series of software models used to predict stock market factor performance. These models include a system to input data, train the model, and display their results in a human-readable manner.
Model Analysis	The secondary deliverable is an analysis of the model's performance throughout time. We will note how different market conditions affect the accuracy of the model. Additionally, special note will be made for performance for major events, such as the 2008 stock market crash.
Documentation	The last deliverable will be the documentation of the design process used for each model. Additionally, we will include the reasoning behind each model selection and tuning process. This is to help the client investigate methods for future growth and development.

All deliverables are expected by early December.

2. Specifications and Analysis

2.1 Proposed Design

As we discussed with our client, there are many solutions and many ways to come to a solution. Thus, to cover our bases and help our clients feel better, we are going to design a few solutions that differ a bit. Since the problem is open ended and our client supports us doing this, we are going to have 3-5 different analytic models. Each model will do the same thing: give predictions on the success of stocks in the near future. The differences of each model and possible solution are contained in the implementation. There are many types of models that are well known that we will use to our advantage. Each one will differ in what data we put into it, and how we adjust the internals. See Appendix 4.3.1 for the current list of models and feature selection techniques used.

One strength of using several different models is that it makes it easier to find potential errors in our predictions. Being able to analyze results from several different models at a time can greatly help us pinpoint where we might be going wrong, and what we can change in order to produce a final model that is as useful as possible. In short, there is really only one direction we can go about solving this, which is a constraint from our client. Our designs are not high level, but the designs are what we plan out when we are testing and analyzing data. There are almost no alternatives. Our client has given us enough creative room and time so that whatever we come up with will be a solution in some way. There is a possibility we will fail in designing any possible solution. In this case, we will still gain valuable insight from that and report those results.

2.2 Current Progress

2.2.1 Research

Much of the work our team has done so far this semester has been looking into the logic and mentalities of the predictive models that we have been assigned. This involves learning the techniques of implementation, reasons behind using each, and the benefits and drawbacks posed for each model. Applying our findings has led us to remove K-Nearest Neighbors and Support Vector Machine models as candidates for our final predictive model, although we are still utilizing both for feature selection.

2.2.2 Development

Obviously, some development has been associated with the research and introduction material we have been working with, but actual development toward a deliverable product for our client has been limited. We have each implemented a single, separate predictive model and presented the results to the client. These presentations have been focused on identifying the most accurate and applicable model for the specific dataset we are using, along with the benefits and drawbacks associated with using each. Once we have finished correcting our models and getting out-of-the-box results and clearly defined the advantages of each, we will start developing an actual model to use for investing.

Our development that has been done has been focused on starting toward the requirements that we set up in our Project Plan assignment. We have made sure our prediction functions track the accuracy of all predictions along with the time taken, which will be important metrics to report later on, as well as a large part of our testing process. In addition to these metrics, our weekly reports for both this class and the client will help us gauge the difficulty of each model for development, so we are able to make better predictions for continued development of our product after this ends.

2.2.3 Standards

There are no specific design standards for machine learning that we know of or that will benefit us. Since that is almost all of our design, this leaves us with only implementation standards. Since we are programming in Python, we need to have standards that deal with developing in Python. We will be using Python version 3. This impacts the specific syntax of our implementation and is needed to ensure common code is able to be compiled on all of our team's machines. Our other standard we will be using for the most part is PEP8. This is a naming convention and style guide used by many Python programmers. It is well developed and allows for a standard of clean and readable code. This ensures that common code retains readability and can be developed by all team members smoothly. IT will also ensure time and comprehension of code reviews by team members.

2.3 Design Analysis

Our design has mainly involved research with big data analysis and machine learning, as these are topics that most of us have not previously worked with. Currently, we have split up the research of various classification and regression machine learning models to each member of the group. Each meeting with the client has been beneficial for our group because of the insight our clients provide. We are starting to understand the necessary steps to properly perform exploratory data analysis as well as the steps necessary to develop a realistic model.

The initial models that were developed yielded too high of a prediction accuracy for a realistic estimation of the current financial market. We discovered that some of the testing data being used included variables that were technically future data. This obviously affected the prediction results of future output variables. Moving forward, we will most likely stop looking into developing a model with k-nearest neighbors or support vector machines. Our client has informed us that these models would most likely not scale well with the large amount of data involved in this project and often provide little to no information about how each decision was made. Alternatively, we have decided to push more towards the random forest and auto-regression models, since both our findings and the opinions of our clients show that they are the best models for this type of project.

3 Testing and Implementation

3.1 Interface Specifications

Our project is software based so we will not be implementing any hardware interfaces. As of now, our plan and our client's requirements also do not include a user interface. There are no externally facing software interfaces. The only internal ones are between our code and various libraries listed below.

3.2 Hardware and software

Since our project is entirely software based, all mentioned tests and tools involved will be software oriented. Most of these tools will be written as part of the project. These tools will monitor the training time and accuracy of each model. The others will be the scikit-learn and statsmodel libraries. These libraries will give us the false-positive/false negative charts along with other similar statistical utilities. Specifically, we have been using Python 3.6 with the scikit-learn and StatsModels libraries for our machine learning needs. NumPy and Pandas are being used for data processing.

3.3 Functional Testing

3.3.1 Unit Testing

The unit testing portion of this project will be focused on the importer and preprocessing utilities. These tests will ensure that all data is parsed into the internal format correctly through different cases. Examples include missing data handling. Preprocessing tests will ensure the results are mathematically valid and useful for predictions.

3.3.2 Integration Testing

The integration testing required for our project will contain two parts. The first and most important part will be giving our tool to the financial analysts at Principal and letting them use it. We will provide our documentation and instructions for use, and see if they are able to navigate the program. After that, we will conduct surveys that gauge the usefulness of the tool to these analysts, whether the results obtained have changed their outlook on the market or affected their decisions and suggestions. These surveys will be recorded and looked at by our team, with potential revisions to follow.

The second part of testing will be focused on transferring the development of our tool over to the data scientists at Principal. Our client has expressed interest in utilizing the product we make after we finish developing it, so we will need to ensure that the code is clearly written, documentation is thorough, and results are self-explanatory. Our testing for this will involve surveys from test users from Principal, along with observation, as we provide the tool and source code to some of their data scientists and observe the difficulty for them to continue development. This phase will take a back seat to the first stage of testing, since a functional product is the most important, but we are hoping to fully satisfy this section as well.

3.3.3 Acceptance Testing

Our client's definition for acceptance of the created system is very flexible up to this point. The ideal system we could create would be a model that is able to predict whether or not a stock will outperform the market with 50%-60% accuracy. However, they have stated that if we are not able to produce a model with this level of accuracy, this is also acceptable as it still provides them with useful information. Our current experimental models output their accuracy percentage when they are generated. Thus, no further testing is needed to determine their accuracy and know whether or not our current model is acceptable to the client's specified criteria.

3.4 Non-Functional Testing

3.4.1 Performance Testing

Testing for performance of the models will involve analyzing the prediction accuracy of each model. Analyzing the runtime will be another important aspect. As of right now, runtime is not an issue because the amount of data being processed is not too large. In the future, however, this could become more of an issue. The important aspect of performance that our client is most concerned about is the prediction accuracy.

3.4.2 Security Testing

Our project does not require any security testing. The only security required is to keep the source code that generates our models private. However, if we do end up implementing a user interface with a login system and other features, we will need to carry out security tests in order to try and ensure that the system can't be broken into by users with malicious intent.

3.4.3 Usability Testing

Usability at this stage of development still requires the user to have knowledge of python and general data science. In the future, we could potentially develop a web client that would make it easy for users to run tests on the models. For this project, the focus is on getting quality results and not as much on how usable running experiments on the models are to untrained users.

3.4.4 Compatibility Testing

For this project, compatibility with other systems or other software is not necessary. The models we develop will be able to run as stand-alone applications.

3.5 Process

Our process, while simple, has worked well for the early stages of our project. Since we had several different models thrown at us with expectations for quick results, we had to implement basic testing procedures to present the effectiveness of each. This involved finding the average runtime for each model for several different sizes of inputs, along with the average accuracy, highest accuracy, and ease of implementation. The accuracy and speed metrics are easy to record with python tools, while the ease of implementation is much trickier. We have been communicating a lot about the troubles of each, in order to more effectively suggest which to use in full for the second semester.

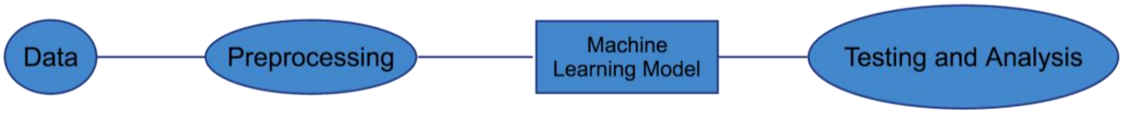

Figure 1 Graph of the project process

The figure above shows the basic parth our model takes. The data starts off in storage, and is then preprocessed for better results in our model. One example of preprocessing is resolving the NaN's in the dataset. Next, a portion of the processed data is fed into a tuned machine learning model to train it. This trained model is then tested using the other portion of the processed data. The model will try to make predictions from what it learned, which may or may not match up with the truth. These results are then analysed to further tune the models.

3.6 Results

Our project is a model for the stock market, so it would be redundant to make a model to simulate it. Initial tests are showing that our models need to be tuned more finely. Given the large number of variables we have at our disposal, we have found that we need to take care in choosing which data should be visible to our models. Many of our tests have yielded unrealistically high accuracy results when making predictions to compare against the testing data. We believe there are a number of contributing factors to this issue, ranging from models using future data to train, to an improper training window size. To address these issues, we have been meeting with our more statistically literate clients to adjust how our models handle input.

Figure 2 is a plot of the autocorrelation vs lag in weeks of the 12 month return. The orange line is the actual return relative to the market. The blue is a binary representation of the data from the orange line, with 1 signifying a return better than the market average with a 0 for all other cases. Lag is an indicator for how many weeks each data point is relevant. This graph shows that the 12 month return has very high correlation for a few select periods, meaning the past 12 month returns can be used to predict future ones.

Although promising, this model has shown to be unusable. Attempting to create a model with the required timeline are abysmally slow. Additionally, even if the speed were up to par, predicting a year in advance requires predicting every week in between. Then, each estimated week will be used to predict the next one. Making predictions off of those predictions is a great way to reduce accuracy as test have shown. Our current model progress can be seen in Appendix 4.3.

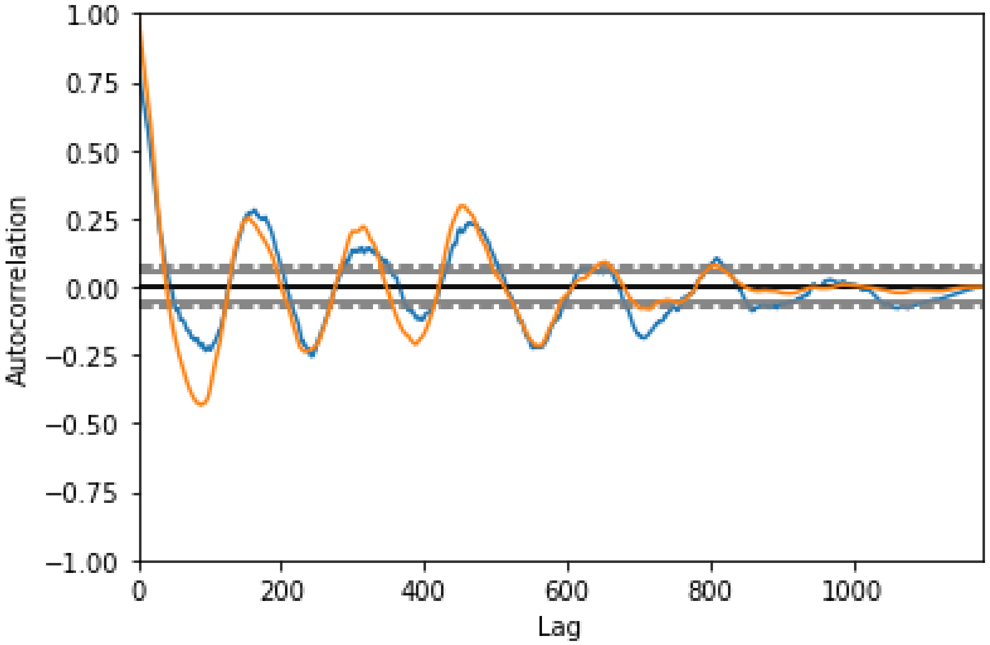

Figure 2 The autocorrelation vs lag in weeks of the 12 month return. This shows certain past 12 month return values can be used to predict future ones.

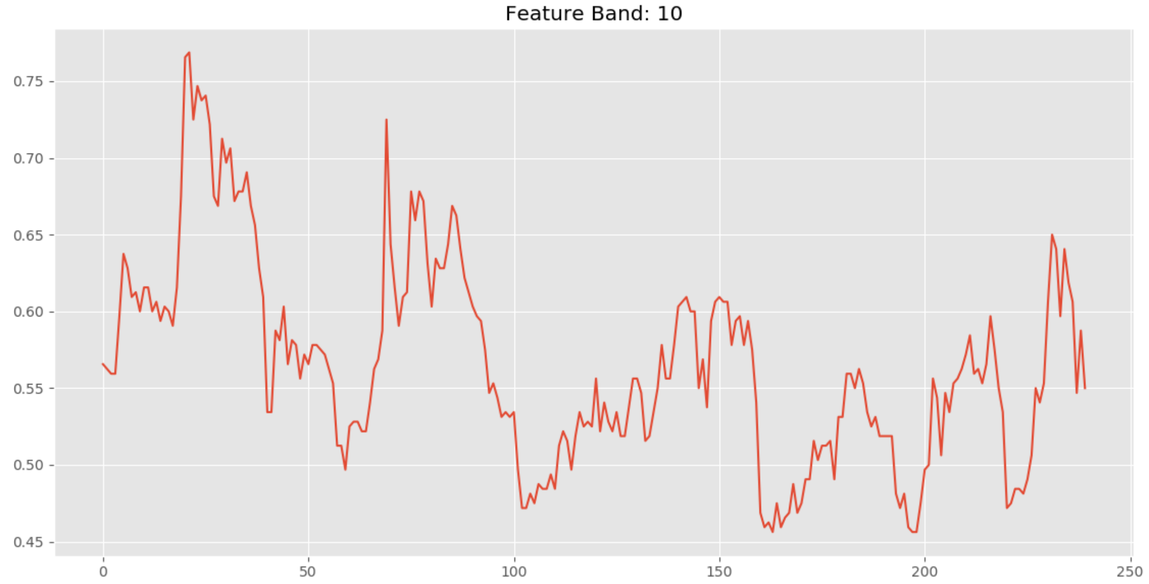

Figure 3 Selecting the correct features for models is the main part of developing them to achieve the highest accuracy. This graph shows a basic experiment to help determine what sets of features achieve the highest accuracy. The ith *data plot is the accuracy with* $\square\square\square\square\square\square\square\square_{\square} \rightarrow \square\square\square\square\square\square\square\square_{\square+10}$ *is selected as inputs to the model*

4 Closing Material

4.1 Conclusion

Our work that we have done so far is best separated into two parts: research and development. Our research consisted of learning and developing our skill base in data analytics, machine learning models, and testing and development processes for machine learning models. Our development that has been done has been focused on starting toward the requirements that we set up in our Project Plan assignment. More specifically, our work consisted of experimentation. As we were doing research into these topics, we were experimenting with them to get a better understanding and also work with our dataset specifically. We have done the basics of this part and this will be a long process in which most of our future progress will reside.

Our goal of this project is to create a system that aids in the creation and management of a profitable investment portfolio. This system will use well defined statistical models and machine learning algorithms to help make decisions about the state of these portfolios

The solution that was originally proposed, and since decided upon as our final design, is to use a complex machine learning and regression model. We have 4 different models that we are using as predictive analyzers. These models will all take in the same inputs, and will have the same outputs. We will use a complex “voting” system between these models to determine actions that will be taken on the aforementioned portfolios. We believe this solution is better than other proposed or analyzed designs because this combines multiple models for a more stable decision making process. We also have honed down the specific models we will be using in order to achieve the best results (i.e. no skewing of the data from bad models). It is easily testable and can exist on its own. All of these fulfill each requirement from the client while giving the highest rate of accuracy possible.

4.2 References

Sehgal, Manav. “Titanic Data Science Solutions.” Kaggle, Kaggle Inc, 1 Jan. 2017, www.kaggle.com/startupsci/titanic-data-science-solutions.

PEP8 Reference: https://www.python.org/dev/peps/pep-0008/

Baldi, P., et al. “Assessing the accuracy of prediction algorithms for classification: an overview.” Bioinformatics, vol. 16, no. 5, Jan. 2000, pp. 412–424., doi:10.1093/bioinformatics/16.5.412.

Robert, Christian. “Machine Learning, a Probabilistic Perspective.” *Chance*, vol. 27, no. 2, Mar. 2014, pp. 62–63., doi:10.1080/09332480.2014.914768.

4.3 Appendices

4.3.1 Table of Models

Table 2 - Table of Models past and present

Model	Usage - Description	Accuracy and # of features
Random Forest (classification)	Predictive model that creates decision trees based on training data to predict whether a factor outperforms or underperforms the market	● Best set of features ○ 'SALES_P_Volatility', ○ 'EBIT_MCAP_Bat', ○ 'X9M_RET_Volatility', ○ 'CFO_P_Volatility' ○ 'ROE_Bat', ○ 'DIV_YID_Bat', ○ 'SALES_EV_Bat', ○ 'TED_SEN_Volatility', ○ 'X6M_RET_Bat', ○ 'RANK_Bat' ● Current Accuracy ○ Range: 40%-60% ○ Based on features, amount of training data, etc. ● With PCA feature: ○ ‘RET_F12M_OP’ ○ ~57% accuracy

Random Forest (regression)	Predictive model that creates decision trees based on training data to predict the actual future value of a certain feature in the market	● Average r-squared value: -2.58 ● Raw model score: -0.79 ● Predictor: RET_F12M ● Features Used are the same as above
Naive Bayes (classification)	Predictive model	● Predictor: RET_F12M_OP ● Current Features ○ From Univariate Selection ○ X6MVT, X12MVT, BETA_1Y, NET_CFO_P, BK_P, SALES_P, AST_P, OIL_SEN, X12MVT_Med, BAA.AAA_Mkt ● Overall Accuracy: 73.4375% ○ Train Size: 75% ● Expanding Window: 7 Years ○ Avg Accuracy: ~52% ○ Range: 38%-70% ● Sliding Window: x Years ○ 1 Year test window ○ Avg Accuracy: ~62% ○ Range: 25%-95%
Tree-based selection	Feature selection	● 11 features currently ○ FCF_P_Bat ○ SALES_P_Volatility ○ EBIT_MCAP_Bat ○ X9M_RET_Volatility ○ CFO_P_Volatility ○ ROE_Bat ○ DIV_YID_Bat ○ SALES_EV_Bat ○ TED_SEN_Volatility ○ X6M_RET_Bat ○ RANK_Bat
Recursive feature elimination	Feature selection	Number of features varies ● 10 features of highest rank ○ D_E ○ SALES_AST ○ Val_SD_Mkt ○ X12MVT_Bat ○ X1SS_ERNQLT ○ X3IVH_CGBS ○ X3IVH_CGBS_Bat

		○ X6MVT_Bat ○ X6M_RET ○ X9M_RET
Auto-regression	Predictive model - Uses pattern of past output points to determine the future. Useful when future events can be preceded by the relatively recent past.	● The data is autocorrelated, with partial autocorrelation tests showing an optimal A value of 2. ○ Model completely nonviable, given we are predicting 52 weeks out, and the model needs to generate and use every intervening week for the next one
Univariate selection (with χ^2test)	Feature selection	● 10 features currently ○ X6MVT ○ X12MVT ○ BETA_1Y ○ NET_CFO_P ○ BK_P ○ SALES_P ○ AST_P ○ OIL_SEN ○ X12MVT_Med ○ BAA.AAA_Mkt
L1-based selection	Feature selection	● 18 features currently ○ X1,3,6,9,12M_RET ○ MCAP ○ D_E ○ ROE ○ SALES_AST ○ FY1_3MCHG ○ X3IVH_CGBS ○ X1SS_ERNQLT ○ BAA.AA_Mkt ○ ti.rank_Mkt ○ cor.rank_Mkt ○ Val_SD_Mkt ○ Crowd_Mkt ○ Earnings_Res_Mkt
Principal Component Analysis	Feature Selection - Creates a new set of orthogonal axis to explain the maximum amount of variance in the	● Over 99% variance using 15 components on normalized data ● Directions of maximum variance (decreasing order) ○ SALES_AST ○ MCAP

	dataset. Can be visualized as a rotation and translation of the axis.	○ Earnings_Res_Mkt ○ D_E ○ cor.rank_Mkt ○ Crowd_Mkt ○ Val_SD_Mkt ○ X1SS_ERNQLT ○ X3IVH_CGBS ○ ti.rank_Mkt