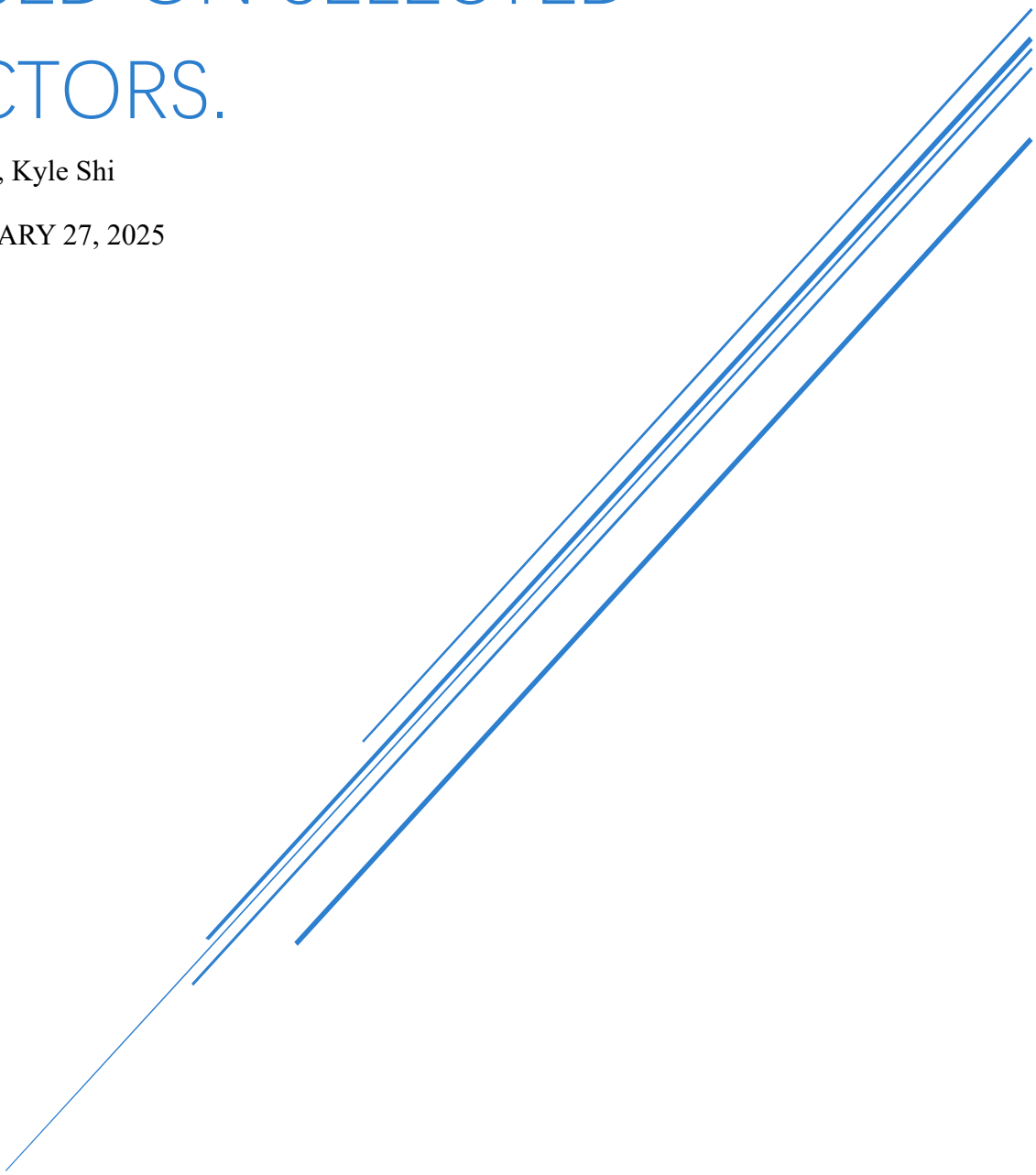
PREDICT STOCK PRICE THROUGH MACHINE LEARNING ALGORITHMS BASED ON SELECTED FACTORS.

Iris Gao, Kyle Shi

FEBRUARY 27, 2025



Contents

Abstract 2

Introduction 2

Background 2

Objective 2

Literature Review 3

Paper 1: Regression Shrinkage and Selection via the Lasso (Tibshirani, Jan 1995) 3

Paper 2: Sparse Principal Component Analysis (Hastie, Tibshirani, and Zou, May 2004) 3

Paper 3: Stock Return Prediction with Multiple Measures Using Neural Network Models (Cong Wang, 2024) 7

Relevance of Literature Papers to Our Topic 11

Connections between Literature Papers 11

Proposed Implementation Plan 12

Conclusion 14

References 14

Abstract

This report explores and analyzes various machine learning approaches in factor investing to construct portfolio with optimal return. In-depth review has been conducted on four literary papers that deploys different machine learning models to enhance the factor selection and portfolio construction process, which includes LASSO Regression, Neutral Networks, and Sparse Principal Component Analysis. Key models, methodologies and research results are summarized while identified gaps, critiques and limitations are discussed in detail. Our comparative analysis further highlights the strengths and weaknesses of each model, providing a foundation for designing and refining an implementation strategy in the second group project. In addition, a well-structured implementation plan is proposed for developing a machine-learning-driven strategy that integrates factor selection and stock return prediction.

Introduction

Background

As machine learning is developing so fast these years, the intersection of machine learning and factor investing has emerged as a promising research area to address persistent challenges in stock return prediction, such as high dimensionality, nonlinear relationship and unequal variance.

Objective

This report conducts a literature survey on the topic of machine-learning-based factor investing for optimal portfolio construction, which critically reviews current research and identifies strengths and weaknesses of existing studies. The goal is to evaluate their models and methodologies, obtain key insights, and design an implementation plan that leverages their strengths and improves their weaknesses. Our project aims to predict stock prices by applying ML algorithms with carefully selected financial factors. Ultimately, we want to design an ML framework that can extract signal from high-dimensional financial data and predict the future market return based on selected factors. Our approach will incorporate advanced data-preprocessing, factor selection (potentially via lasso-based shrinkage or sparse principal component analysis), and neural network models to capture both relationships among predictors.

Literature Review

Paper 1: Regression Shrinkage and Selection via the Lasso (Tibshirani, Jan 1995)

Summary: This paper introduces the Least Absolute Shrinkage and Selection Operator (LASSO), which is a method used for variable selection and regularization. Lasso minimizes the residual sum of squares subject to the sum of the absolute value of the coefficients being less than a constant (Tibshirani, Jan 1995). Studies have shown that Lasso products interpretable models with some coefficients converge to exactly zero.

Strengths:

- *Intuitive Explanation on Geometric Properties of LASSO*: this paper includes effective illustrations to compare L1 constraint used in Lasso and L2 constraint used in ridge regression. The contour plots provide a clear explanation of how Lasso encourages sparsity and why it sets coefficients to zero, which contrasts with ridge regression that only shrinks the coefficients.
- *Robust Empirical Validation*: simulations and real-world applications have both been conducted using Lasso. The empirical results show that Lasso outperforms the existing method.
- *Better Interpretability and Prediction Accuracy*: Lasso balances predictive accuracy and interpretability by selecting a sparse subset of variables.

Weaknesses:

- *Unable to Capture Non-linear Relationships*: Lasso is a linear model that does not capture non-linear relationships. However, Lasso can be extended to tree-based models, but this paper does not further discuss how to impose feature engineering on Lasso to solve non-regression problems.
- *Sensitivity to Correlated Predictors*: one of Lasso's inherent weakness is the high sensitivity to correlated predictors. Lasso arbitrarily selects one of the predictors and sets the others to zero when two or more predictions are highly correlated. This results in instability in feature selection since a small change in data can lead to different variable selection.

Paper 2: Sparse Principal Component Analysis (Hastie, Tibshirani, and Zou, May 2004)

Summary: Principal component analysis (PCA) is a widely used technique for dimensionality reduction. The primary drawback of PCA is that each principal component is a linear

combination of all the original variables with nonzero loadings, which hinders interpretability and poses challenges in identifying the most impactful features. This paper introduces a novel method of sparse principal component analysis (SPCA) that incorporates the *lasso (L1 penalty) and elastic net (L1+L2 penalties)* to modify principal components with sparse loadings, resulting in zero coefficients for many principal component loadings, thereby reducing the number of explicitly used variables.

The LASSO and Elastic Net: The lasso is a penalized least squared method that imposes a constraint on the L_1 norm of the regression coefficients and the estimated lasso beta is obtained by minimizing the lasso criterion below:

$$\hat{\beta}_{lasso} = \arg_{\beta} \min \left\| Y - \sum_{J=1}^{p} X_j \beta_j \right\|^2 + \lambda \sum_{j=1}^{p} |\beta_i|$$

The lasso shrinks the coefficients toward zero if λ is sufficiently large and would produce a sparse model if incorporated into PCA, making it an effective variable selection method.

One of the several limitations of LASSO is that the number of variables selected by the lasso is limited by the number of observations, meaning that the maximum selected variables is the number of samples, which might not be sufficient and accurate. This drawback can be mitigated using the elastic net, which generalizes the lasso to include more variables in the fitted model by setting a fixed λ_2 and estimates beta using the following equation:

$$\hat{\beta}_{elatic\ net} = (1 + \lambda_2) \left\{ \arg_{\beta} \min \left\| Y - \sum_{J=1}^{p} X_j \beta_j \right\|^2 + \lambda_2 \sum_{j=1}^{p} |\beta_j|^2 + \lambda_1 \sum_{j=1}^{p} |\beta_i| \right\}$$

The elastic net penalty is a convex combination of the ridge and lasso penalties where the lasso is a special case when $\lambda_2 = 0$. Therefore, when dealing with insufficient number of samples with respect to the number of variables, elastic net can be deployed to eliminate the lasso drawback and potentially include all the variables.

Methodology: To address the PCA drawback, SPCA is used with a quadratic penalty where the *lasso penalty via the elastic net* is directly integrated into the regression criterion, leading to a modified PCA with sparse loadings. The key modifications to traditional PCA are presented in detail below:

- *Method 1: Reformulation of PCA as a Regression*
  The first method to enforce sparsity on PCA is to approximate PCA using a regression-based approach. For traditional PCA, the exact PC loading is computed via eigen-decomposition; however, the PC loadings can be estimated using the connection

between PCA and ridge regression. The reformulated ridge regression formula is shown below:

$$\widehat{\beta_{ridge}} = arg_{\beta} min \, \|Z_i - X\beta\|^2 + \lambda\|\beta\|^2$$

Where $Z_i = U_i D_{ii}$ is the i^{th} principal component. The ridge regression can be used to estimate principal components, which shrinks the coefficients towards zero but not exactly to zero, then normalize the regression coefficients would result in a match with the PCA loadings. The second step is to add the L_1 penalty as shown below:

$$\hat{\beta} = arg_{\beta} min \, \|Z_I - X\beta\|^2 + \lambda\|\beta\|^2 + \lambda_1\|\beta\|_1$$

Where $\|\beta\|_1 = \sum_{j=1}^{p}|\beta_j|$. The large enough λ_1 in the above equation leads to a sparse approximation of PCA.

- *Method 2: Self-contained Sparsity Constraint*

  The second method integrates ridge and lasso penalties into the constraint equation, so this is a fully integrated optimization framework that directly optimizes for sparse loadings. The modified regression equal is shown below:$(\hat{A}, \hat{B}) =$

  $$arg_{A,B} min \sum_{i=1}^{n} \|x_i - AB^T x_i\|^2 + \lambda \sum_{j=1}^{k} \|\beta_j\|^2 + \sum_{j=1}^{k} \lambda_{1,j} \|\beta_j\|_1$$

  Where A corresponds to the ordinary principal components, B corresponds to the sparse loadings, λ is a ridge penalty and $\lambda_{1,j}$ is an L_1 penalty for sparsity. The first term minimizes the error, the second term is the ridge penalty that prevents overfitting, and the third term is L_1 penalty that shrinks coefficients to exactly zero.

Strengths:

- *Adjusted Total Variance*: for traditional PCA, the principal components are uncorrelated, and their loadings are orthogonal since PCA maximizes variance along each axis and are obtained by singular value decomposition. However, for SPCA, the principal components might be correlated. To solve the correlation issue, this paper introduces a new formula for computing total explain variance that accounts for correlation among sparse principal components as shown below: let $\hat{Z}$ be the modified principal components where $\hat{Z} = QR$ for Q being an orthonormal and R being an upper triangular, then the total variance is $\sum_{j=1}^{k} R_{jj}^2$

  where $R_{jj}^2 = \|\hat{Z}_{j,\dots,j-1}\|^2$.

This adjusted variance formula accounts for the overestimation of explained variance due to potential correlation issues.

- *Computationally Feasible*: In terms of computation complexity, since PCA is computationally efficient for both n > p and p >> n data, then SPCA algorithm can sufficiently handle data with a very large n if p is small (i.e. p < 100), where n is the number of samples and p is the number of predictors. For the dataset that will be used in group project 2, the number of observations is much larger than the number of predictors, therefore SPCA will be computationally feasible for our implementation. The details and source of the dataset will be discussed later in the implementation plan.
- *Improved Interpretability*: SPCA enables the shrinkage of coefficients to zero, resulting in sparse loadings that can be more easily interpreted. The importance of each predictor can be identified.
- *Efficient Algorithm*: this paper introduces an alternating optimization algorithm to minimize the SPCA constraint from method 2. This iterative method ensures the computed principal components remain orthonormal via Procrustes rotation, as well as increasing computational efficiency for various types of datasets.
- *Model Flexibility*: the degree of sparsity in SPCA can be adjusted by tuning the λ_1 parameter. If the goal is to fit the model with more predictors, then lower λ_1 should be chosen to yield a less sparse solution; on the contrary, if the goal is to model with less predictors, then higher λ_1 should be chosen to yield a sparser solution. Therefore, this model is quite flexible and enables users to adjust the parameter to adapt to different applications.
- *Empirical Validation*: this paper conducted experiments of SPCA on real-world and synthetic data, validating the efficacy of SPCA.

Weaknesses:

- *Interpretability vs Accuracy Trade-off*: this paper focuses on discussing the dimensionality reduction capability of SPCA. However, the sparser the solution, the higher the interpretability, the lower the accuracy. This paper does not discuss the impact on predictive accuracy upon reduction of loadings and lacks guidelines on how to strike a balance between achieving high accuracy while maintaining interpretability.
- *Linearity Assumption*: traditional PCA's principal loadings capture the linear relationships in data, however the paper does not include whether SPCA is able to capture non-linear relationships.

- *Computational Inefficiency for Fewer Samples and High-Dimensional Data*: for small-sample and high dimensional data (p >> n), solve each elastic net can be consuming and expensive for large p. Nonetheless, this high computational cost can be solved efficiently using a special SPCA algorithm.
- *Lack Hyperparameter Tuning Discussion*: the methods of solving SPCA discussed in this paper requires setting the parameters λ and λ_1, which corresponds to the ridge penalty and lasso penalty, respectively. The inputs of these parameters have a significant impact on the model results, but the paper does not provide further guidelines on how to choose the optimal values for those parameters.
- *Potential Challenges in Alternating Algorithm Convergence*: the alternating algorithm introduced in the paper lacks discussion on whether there can be potential issues with model convergence. The speed of convergence is not mentioned.
- *Sparsity vs Explained Variance Trade-off*: the higher the sparsity, the lower the explained variance. This paper focuses on enforcing sparsity but does not provide any insights on how to optimize the trade-off between sparsity and explained variance.

Paper 3: Stock Return Prediction with Multiple Measures Using Neural Network Models (Cong Wang, 2024)

Summary: this paper investigates the performance of machine learning methods in predicting stock returns for various factor models and analyzes the interaction effects among firm-specific variables. This study focuses exclusively on the US stock market and employs 49 stock fundamental characteristic variables and 14 distinct macroeconomic variables, covering the period from 1971 to 2021.

***Methodology*:** Neutral Network (NN) models are used in this paper to predict stock returns due to the capability of the model to capture non-linear relationship and interaction effects. A simple feed-forward neural network architecture is employed to capture the relationships between stock feature variables and returns.

The NN model consists of three or more layers:

- The input layer comprised a set of firm characteristics and macroeconomic variables
- One or more hidden layers capture the interactive effects between different variables and perform non-linear transformations through activation functions on input variables
- The output layer aggregates all information from the hidden layers to generate an output

Stock returns can be effectively predicted using an additive prediction error model (Gu et al., 2020), with the equation shown below:

$$R_{i,t+1}^{abr} = E_t\left(R_{i,t+1}^{abr}\right) + \epsilon_{i,t+1}$$

Where i = 1, 2, …, N and t = 1, 2, …, T represent individual stock index and time periods, respectively. R represents abnormal return of stock i at time t+1, which is equal to the expected stock return based on the conditional information available at time t plus the error term. The conditional expectation of returns can be written as a function that represents a mapping function, which incorporates a set of predictors z and a set of parameters θ:

$$E_t\left(R_{i,t+1}^{abr}\right) = g(z;\theta) = \theta_0 + \sum_{j=1}^{n} x_j\,\theta_j$$

The parameters θ represent the weights and biases in each layer of the neutral network model. The way a feed-forward neural network model works is that the model iteratively adjusts weights of the nodes in each layer to establish a mapping between the input predictors and output returns. The goal is to minimize the errors on the training dataset and to improve its ability to predict stock returns.

Objective Function

The objective function calculates the difference between predicted and actual outcomes in a neutral network, and the model goal is to determine the optimal weights that leads to more accurate stock return predictions within the training dataset. Typically, the model uses a stochastic gradient descent (SGD) optimization algorithm with weights being adjusted using the backpropagation of error method. In the stock return prediction application, MSE is a preferred objective function choice due to concerns regarding large outliers. The MSE objective function is: $MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \widehat{Y}_i\right)^2$

Activation Function

The activation functions account for non-linearity of individual predictors and influence the transformation of the weighted sum of inputs into output from one layer to the next. There exist several types of activation functions such as soft-max, sigmoid and the rectified linear unit (ReLU). This paper adopts ReLU as the activation function for the hidden layers due to its computational simplicity: $ReLU(x) = \begin{cases} 0, & if\ x < 0 \\ x, & otherwise \end{cases}$

Learning Rate

Learning rate controls the speed of learning with a higher rate leading to potential over-adjustments and erratic performance (closer to value of 1), whereas a lower rate adjusts weights slower and with more precision (closer to value of 0). A commonly used learning rate is $learning\ rate = initial\ rate\ * \frac{1}{1 + decay * iteration}$

Modeling and Portfolio Construction

The methodology to conduct neural network model is to randomly split the sample into three equal-sized training, validation and test datasets. The neural network is first being trained using the training dataset, then hyperparameters are tuned using the validation set with the objective to maximize the R-squared value in the validation set. Finally, the fitted model and fixed hyperparameters are used to predict stock returns in the testing dataset. The model predictions are used to build value-weighted portfolios. Each month these stocks are sorted into deciles based on predicted returns, within each decile, the weights are assigned to each component stock based on market capitalization. Every month the portfolio is rebalanced without considering transaction costs.

Results

This paper explores prediction accuracy by using different return measures, namely, excess returns and abnormal returns. Multiple measures of stock returns are compared for different decile portfolios. Results show investing in the top decile can achieve nearly 12 times the excess return, while investing in the bottom decile would result in negative return. Furthermore, incorporating macroeconomic factors lead to substantial improvement in performance particularly for excess returns: a seven-hold increase in predictive power, which implies macro conditions can capture excess stock returns that was not captured by firm specific features alone. However, incorporating macro factors leads to smaller improvements for abnormal returns.

Strengths:

- *Predictive Accuracy:* Neural networks have become a promising approach for forecasting both excess and abnormal returns by combining firm-specific attributes and macroeconomic indicators. These models are particularly effective at identifying nonlinear patterns and complex interactions. For example, recent research by Wang (2024) demonstrates that incorporating macroeconomic variables can significantly enhance model accuracy.
- *Overfitting with Regularization*: this paper addresses one of the common challenges in neural network, overfitting, which leads to exceptional model performance on training data but underperforms on testing data. Two approaches for mitigating this issue are introduced, one approach is to enlarge the training dataset, which enhances the model's performance to maximum learning capacity. The other approach is to adjust the complexity of the model by using L1 penalties on network weights.
- *Guidelines on Setting Parameters and Functions*: this paper provides recommended functions and parameters to use based on the stock return prediction application, which can be used as a guideline for group project 2 because this is directly related to

our research topic.

- *Impact on Predictive Power:* this paper compares the model prediction performance for different return measures (excess versus abnormal) and highlights which type of factors can provide incremental values for neural network models. The research finds that excess returns benefit most from neural network models due to the incorporation of macroeconomic factors, while abnormal returns show smaller gains. This is helpful for us to determine which return measures to use for group project.
- *Comprehensive Dataset:* the dataset used in this paper consists of 49 firm characteristics and 14 macroeconomic variables, which aligns much better with the real world and provides a holistic view of return drivers.

Weaknesses:

- *High Variance Nature*: the neural network model has a low bias and high nature due to its high sensitivity to initial conditions. Different initial conditions can converge to different weights, resulting variations in predictions even for identical datasets. This is a great challenge for the development of a consistent predictive model. To address this challenge, multiple models can be trained to produce a more stable output after aggregating all predictions through ensemble learning.
- *Random Splitting May Cause Lookahead Bias*: the methodology used is to split the sample data randomly, meaning that later data might be in the training dataset while earlier data is allocated to the testing dataset, which will introduce lookahead bias that might overestimate real-world performance. For better practices, in our project, we will use an earlier period for training and a later period for testing.
- *Lack Model Comparison*: the model performance can be better interpreted if it is compared to a benchmark or other models. This paper only focuses on neural networks but there is no comparison on the performance of linear model or OLS. It would be helpful to see how neural network model compares to other models. In our project, we might consider implementing simpler models just as a baseline or benchmark for the neural network models so stronger conclusions can be made in terms of the predictive power of neural network model.
- *"Black Box" Nature and Complexity*: the inherent "black box" nature of neural networks complicates interpretability. Despite their robust predictive power, neural networks demand meticulous architectural planning, such as selecting the appropriate number of hidden layers and nodes and often rely on ensemble techniques to address instability across training iterations. The substantial computational expenses and tuning complexities must be weighed against the incremental improvements in performance.

Relevance of Literature Papers to Our Topic

***Paper 1*:** Paper 1 introduces the LASSO model, which is widely used in factor selection and highly relevant to our chosen topic. Lasso identifies the most important factors in predicting returns, ensuring most relevant predictors are included in later prediction. It outlines the characteristics and impact of Lasso, providing details on model applications. This paper offers another approach to consider when choosing a feature selection model for our project.

Paper 2: Paper 2 is highly relevant to our chosen topic. One of the steps required in our project is factor selection. Traditional PCA is one of the methods that can be used to extract impactful factors that will be used later for stock return prediction. However, principal components involve loadings on all the original variables, but too many variables are difficult to interpret. SPCA can overcome this drawback by incorporating L1 and L2 penalties to produce sparse loadings. These techniques proposed in this paper can be adopted in our project. When constructing portfolios, we can use the SPCA technique to select a set of meaningful factors that explain stock returns without overfitting. This is an effective method to reduce factor dimensionality while maintaining explainability. The strengths and weaknesses of SPCA discussed provide us with the pros and cons of adopting this method in our project and whether this is a suitable method for the dataset we use.

***Paper 3*:** Paper 3 studies how neural network models can be used in factor investing to capture complex relationships in financial data, which is extremely relevant to our project. The neural network model has demonstrated the ability to enhance return prediction and increase model accuracy by taking correlation and non-linear relationships into account, which is a suitable machine learning model that we can use in our project to predict stock returns. Furthermore, it highlights the importance in incorporating both firm-specific and macroeconomic factors, so a broader set of features can improve the predictive power of neural network models. Learning from the research findings, for our project, we will use both firm-specific and macroeconomic factors to predict stock returns.

Connections between Literature Papers

The connection between the first two papers is that the first paper uses the LASSO model and concept from the second paper. The SPCA model incorporates LASSO as a key technique to enforce sparsity. The keynote here is that LASSO and SPCA are not competing models, rather they are complementary.

The first two papers are closely related because they offer different models to reduce dimensionality. Since the dataset that will be used in our project contains close to 100 features, it is important to choose an appropriate model that can achieve the project objective. These two papers allow a direct comparison between LASSO and SPCA models, including

each of their strengths and weaknesses. SPCA is an extension of PCA that is useful when dealing with highly correlated features. It extracts latent factors with sparse loadings (only a few features contribute to each factor), meaning that it combines correlated predictors into fewer meaningful factors, which effectively handles feature redundancy. However, for SPCA, hyperparameter tuning is required for sparsity level. On the other hand, LASSO shrinks regression coefficients to exactly zero, directly picks the most important predictors for stock return. However, it does not account for correlations between factors. The variable selection process is unstable for LASSO when multiple features are correlated, it would arbitrarily ignore other correlated variables, potentially overlook important information. The selected features are also not guaranteed to be uncorrelated, which might lead to multicollinearity issues later.

Knowing the pros and cons of each model, the approach our project will take is to apply SPCA first and then use LASSO if necessary. The reason behind is that it is important to reduce dimensions because many of the features in our dataset are redundant and highly correlated. It is important to use a model that deals with strong collinearity among predictors. Applying SPCA first can reduce dimensionality by keeping a smaller set of sparse components that still explains most of variation in data. If these components are not all equally important, then apply LASSO on these components to further select the most important ones for prediction.

After obtaining selected components, factor scores for each stock are calculated. SPCA factor scores translate raw variables into a new feature set. The stocks can be ranked and selected based on factor scores. Rather than using raw variables, SPCA factor scores are used as input features in neural network models, which connects to the third paper. The third paper describes the methodology of using neural network models to predict stock returns, which will be implemented in our project.

Proposed Implementation Plan

A detailed implementation plan is proposed below outlining each step:

Data Wrangling

This project will use the data_ml.csv data set provided in this course. This dataset includes firm-specific features, macroeconomic features and stock forward returns with a total of 93 firm and macro features and 4 return labels. For this project, we will focus on the 1-month forward return (R1M_Usd). The data will be further wrangled to account for missing values. Clean, normalize, and preprocess the data to ensure consistency and remove outliers.

We will split the data based on time. For example, if the dataset used contains value from 2000 to 2019, then data from 2000-2014 can be training set, data from 2015-2017 can be

validation set and data from 2018-2019 can be testing set. The random splitting technique used in paper 3 will not be used in our project because we want to avoid having future data in the training dataset.

Dimensional Reduction

As discussed above in the connection section, SPCA will be used to reduce dimensions, then LASSO will be applied to select important components if necessary.

Factor Score Calculation

Compute factor scores to transform stocks original features into SPCA factor scores.

Stock Return Prediction Using Neural Network Model

We will use a feed-forward neural network model. The input layer is the reduced factor scores, and the output layer will be a single neuron of predicted 1-month ahead return for each stock. Between input and output layer, there will be one or more hidden layers that use ReLU activation as mentioned in paper 3, which enables non-linear feature interactions. The learning rate used in paper 3 will be adopted by our model because it is commonly used. Hyperparameter tuning will take place by conducting a grid search over required parameters such as hidden layer size, learning rate, penalty strength using the validation set. Select the set of parameters that yields the lowest validation loss.

Portfolio Construction and Monthly Rebalancing

Using the trained model, the neural network will generate predicted 1-month returns for each stock in the test period, meaning that for month t in the test dataset, the trained model takes in all the relevant features and predict the stocks' return for month t+1. The predicted stock returns will be used to construct a portfolio at month t to hold until month t+1, and the process repeats every month or also known as rolling forward.

Each month the portfolio will be rebalanced, where stocks need to be ranked based on predicted next month's return. We want to invest in stocks that have the top decile of predicted return (highest 10% of predicted return) and sell the stocks that have the bottom decile of predicted return (lowest 10% of predicted return), which is also known as the decile long-short strategy. Essentially, the portfolio for month t is based on the prediction for month t+1 that is used to construct the holding, then the rebalancing process continues for each month.

Back-testing and Validation

We will conduct out-of-sample backtest on the test dataset using the trained model. This simulates the performance of the model in real time. A graph of cumulative return over time will be plotted to visualize how this strategy generates return. We can compare the model

generated trend with the market index S&P500 since all the stocks in the dataset are listed in the US. The metrics used to evaluate this model's performance are annualized return and Sharpe ratio.

Conclusion

We have explored the integration of machine learning techniques with factor investing to address the challenges of stock return prediction and we can now develop a framework which can effectively select and reduce the dimensionality of financial predictors while maintaining interpretability. Moving forward, our focus will be actually implementing a machine learning algorithm following our plan to predict stock returns.

References

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1), 267–288. https://www.jstor.org/stable/2346178

Zou, H., Hastie, T., & Tibshirani, R. (2004). Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2), 265–286. https://doi.org/10.1198/106186006X113430

Wang, C. (2024). Stock return prediction with multiple measures using neural network models. *Financial Innovation*, 10, 72. https://doi.org/10.1186/s40854-023-00608-w