Numerical Industry Sentiment Analysis

Yining Gan, Jiang Du

Columbia University, School of Engineering, Industrial Engineering and Operations Research

1. Introduction

The energy industry, from upstream to middle stream and then to downstream, comprising production, distribution and refinement of oil, gas, and renewable energy, plays a vital role in the global economy. Fluctuations in energy prices as well as commodity changes can significantly impact various financial markets and investment portfolios. Understanding the energy sector's sentiment is crucial for risk management and investment optimization before market fluctuation.

Our project aims to enhance predictive accuracy in energy sector sentiment analysis using advanced data transformation and machine learning techniques. By simply deriving the cumulative returns of energy factors over a two-month period, we strive to provide reliable prediction results that aid in making investment decisions. We begin by detrending time-series data to remove linear trends, ensuring our models capture intrinsic market variability. Using robust rolling principal component analysis (R2-PCA), we perform dimensionality reduction to extract meaningful information while mitigating minor noise. We implement various classification models to predict binary outcomes of market movement. Additionally, we explore deep learning techniques, specifically Long Short-Term Memory (LSTM) networks. To further enhance model robustness and address non-stationary data issues, we incorporate an economic regime framework, improving the generalization ability of our models. Through this project, we demonstrate how advanced analytics and machine learning can provide valuable insights into energy sector sentiment, helping investors reduce risk and make better-informed portfolio decisions.

2. Background and Assumption

In this project, we define our label—factor of energy—as the energy sector returns uncorrelated with the market index S&P 500 over the next two months through linear regression. This method is inspired by the approach introduced by Two Sigma in their paper "Two Sigma Factor Lens." We operationalize this approach by regressing the monthly returns of the S&P 500 (ticker symbol: ^GSPC) against the monthly returns of the energy market index (ticker symbol: ^GSPE). In this simple linear regression model, we decompose the returns of the energy index as follows:

$$\text{Returns of energy index} = \text{Intercept} + \text{Error term}$$

Given that the intercept is not statistically significant, we exclude the constant term and consider the entire error term as the returns of the energy index.

The underlying assumption of this approach is that by removing the non-stationary component represented by the market benchmark, the residuals (error term), which are stationary and normally distributed, can be interpreted as the returns of the energy sector index that are uncorrelated with the broader market movements.

3. Method

This section will be divided into Data Transformation, R2-PCA, Machine Learning Prediction, Deep Learning Prediction, and Integrating Economic Regime.

3.1 Data Transformation

As the goal is to perform binary classification prediction on the cumulative return of factor of energy in the next two months in a moving window for time-series forecasting, we need to transform the data points to be more stationary but not perfect stationary. Specifically, we detrend the time-series data points by calculating the percentage change over the past 6 months of each observation. By focusing on the percentage change, we remove the linear trend component of the series. Detrending is important because it:

1. Helps stabilize the mean and reduces the impact of non-stationary behavior on the PCA results. This ensures that the principal components capture the intrinsic variability rather than the trend and
2. Mitigates the issue that the existence of trends in data can significantly affect the covariance matrix, leading to misleading results that will affect the accuracy of PCA results

Additionally, we avoid over-manipulation by refraining from applying specific transformations to address non-stable volatility and seasonality. Over-manipulation can artificially create a perfect pattern fit for predictive modeling, which we avoid because such patterns are unlikely to hold in real-world scenarios.

3.2 Dimensionality Reduction Through Robust Rolling PCA (R2-PCA)

Robust Rolling PCA (R2-PCA) is a refined methodology of standard PCA to reduce and extract meaningful signals from large time-series data-sets. The algorithm performs eigenvalue decomposition of the covariance matrix (which is scaled data, implying a correlation matrix) in a moving window, reorders the principal components by cosine similarity rather than by variance (eigenvalues), and flipping inconsistent sign of eigenvectors through screening cosine similarity over time. This approach ensures the consistency of the principal components across rolling windows.

3.2.1 Implementation of R2-PCA on specific sector datasets

To implement this algorithm into the energy sector data (now consisting of 40 columns of features), we

1. Set the window size to be 60 months, ensuring the datasets that we perform PCA are large enough to minimize the influence of market condition changes that might lead to drop in cosine similarity over every two windows' principal components. This makes it more certain that the cosine similarity analysis that we will conduct in the next step focuses on screening the principal components that capture noises over time.
2. Take a stepwise movement over each datapoint.
3. Store the percentage of variance captured to perform number of principal components selection.
4. Store the cosine similarities between the current and previous eigenvectors to perform cosine similarity analysis.

3.2.2 Cosine Similarity Analysis

In order to enhance the generalization accuracy of predictive modeling, we conduct cosine similarity analysis to perform reduced-dimensional features selection.

Figure 1 shows the cosine similarities between the current and previous eigenvectors over time when we reduce the dimensionality to 10 features. Figure 2 shows the total variance captured over time using these 10 features. The first 10 principal components capture an average of 80.74% variance and a minimum variance of 70.96%. We first choose 10 principal components to ensure that we capture an average of at least 80% of variance and a minimum variance of 70% over the timeframe. By visualizing the stored cosine similarities, we've got a general idea of how many eigenvectors experience significant drop in cosine similarities, thus implying the number of principal components that are potentially capturing noise.

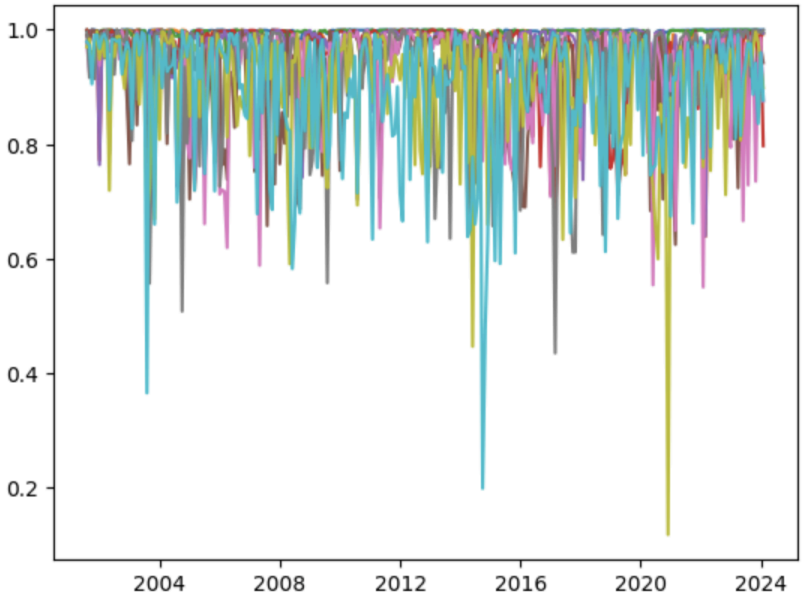

Figure 1. Cosine similarities between the current and previous eigenvectors (# PC = 10)

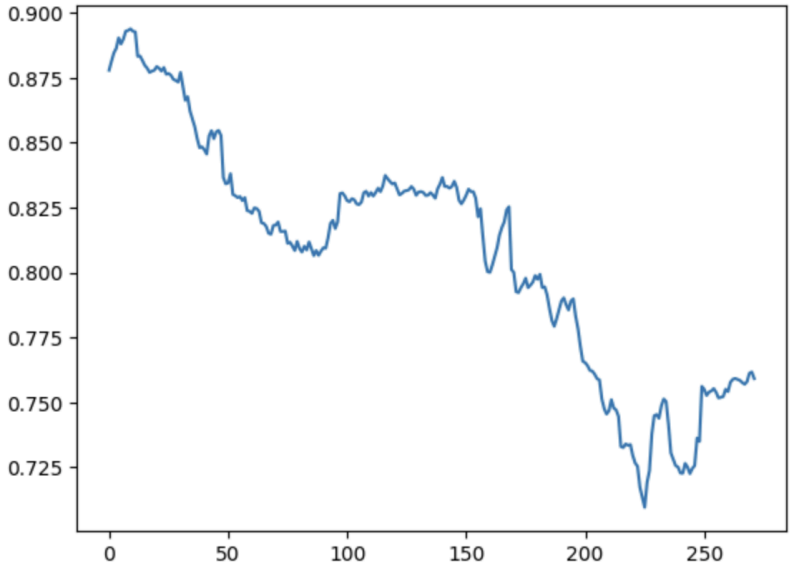

Figure 2. Total variance captured from first 10 principal components

The threshold of minimum cosine similarity we choose is 0.55 and the principal components that have a cosine similarity below this threshold are considered potentially capturing noises. The value 0.55 is chosen by experimenting with a different number of principal components we drop. This threshold ensures that we not only drop the principal components that potentially capture noises but also capture as much variance as we could. By setting the threshold as 0.55, we drop the last three principal components and retain a total of 7 principal components. The first 7 principal components capture an average of 72.27% variance over time and a minimum variance of 62.56%.

Figure 3 shows the cosine similarities between the current and previous eigenvectors over time when we reduce the dimensionality to 7 features. Figure 4 shows the total variance captured over time using these 7 features.

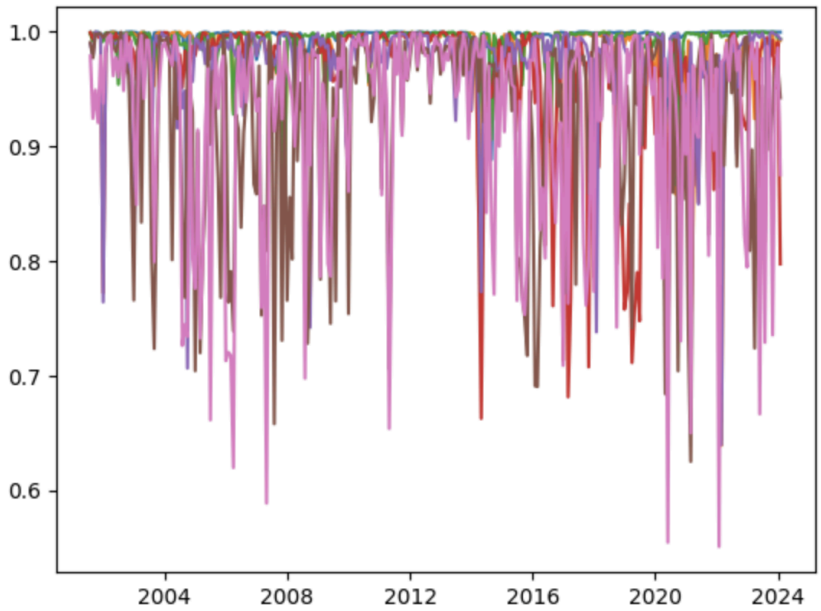

Figure 3. Cosine similarities between the current and previous eigenvectors (# PC = 7)

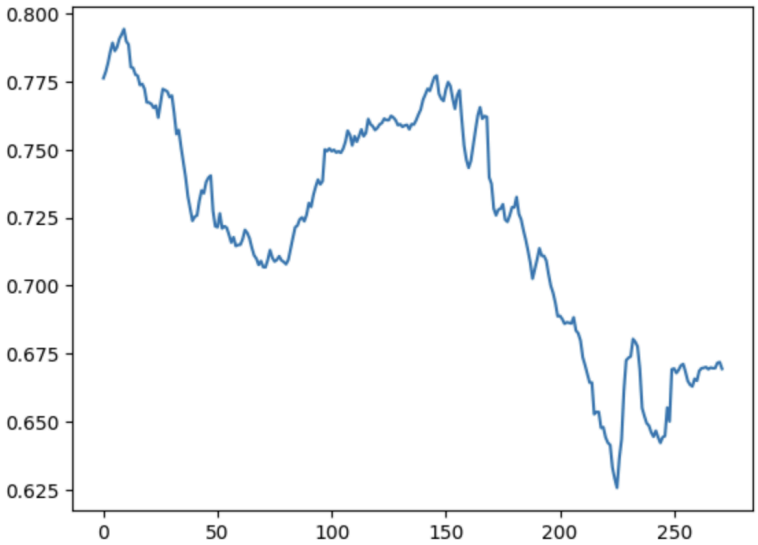

Figure 4. Total variance captured from first 7 principal components

3.3 Machine Learning Prediction

In order to perform prediction on the binary classification for the next two months' cumulative return of factor of energy, we implement various classification models including Logistic Regression, Naive Bayes, Support Vector Machine, and Random Forest. The input datasets are paired data points of (X, Y), where X is a keep-appending dataset that includes all observations from the first date to the current looping date, and Y is a single label corresponding to next two months' binary classification result of cumulative return changes. We start performing predictions when there are at least 15 data points in the keep-appending dataset of features. After experimenting with the 4 classification models we select, we find out that Support Vector Machine with 'rbf' non-linear kernel shows the best classification accuracy of 63.28%. Random Forest, Naive Bayes, and Logistic Regression all with default parameters show accuracy of 57.42%, 56.64%, 57.42% respectively.

The classification accuracy shows a significant improvement from detrending the features before conducting R2-PCA and dropping 3 principal components that are potentially capturing noises. Without these two analyses, all the four classification models show accuracies of around 50%, which is roughly the same as a random guess. The significant improvement in the model's predictive accuracy shows the importance of moderate manipulation to improve the stationarity of data and feature selection in machine learning modeling.

Figure 5 shows the confusion matrix generated from SVM with 'rbf' non-linear kernel.

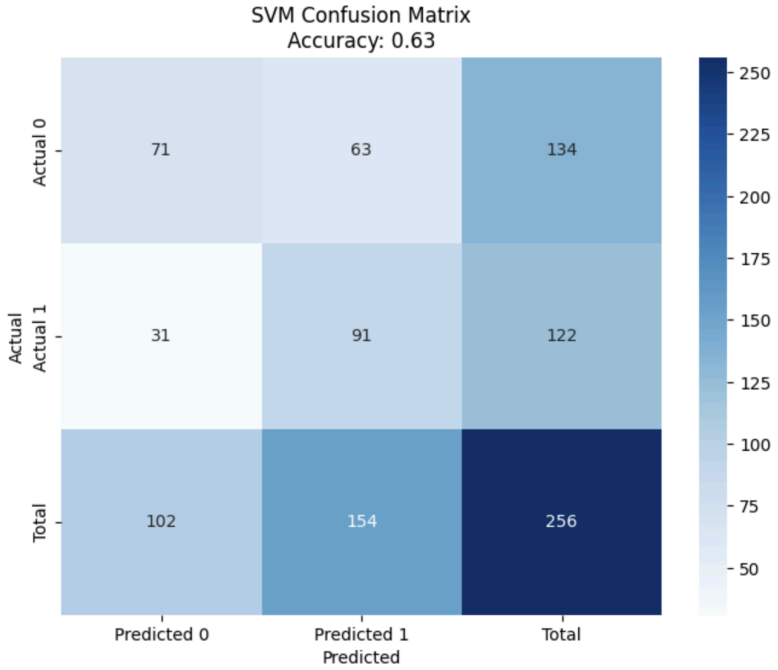

Figure 5. SVM Confusion Matrix

3.4 Deep Learning Prediction - LSTM

In order to experiment with more classification models, we attempt to construct LSTM from scratch and customize the structure of this recurrent neural network to evaluate the predictive accuracy of deep learning models on the exact same datasets of features and labels.

The datasets for the LSTM model are generated by creating fixed-size sequences of features, with corresponding binary labels representing the cumulative return of the energy factor for the next two months. We use a window size of 12 months to capture a year's worth of raw industry data movement to predict the binary label. This approach differs from our machine learning model setup, where the training dataset continuously appends new data points as we progress through the dates. In contrast, the LSTM model maintains a fixed window size.

Through careful hyperparameter tuning, we determined the optimal network structure for our LSTM model as follows: we used Binary Cross Entropy Loss with a Sigmoid activation function in the output layer for classification. The hidden dimension was set to 26 with 1 LSTM layer, and the learning rate was adjusted to 0.007. We trained the model using a batch size of 32, employing early stopping based on validation loss with a maximum of 30 epochs. The results of our tuning process yielded promising metrics: Precision of 0.67, Recall of 0.86, and an F1-Score of 0.75.

Although the deep learning neural network shows improvement in prediction accuracy, we do admit that the model has limited ability to generalize to new data. This is because we have very limited datasets to train. In this project, we only have access to 272 rows of monthly observations, this might lead to high risk of overfitting and ineffective hyperparameter tuning.

3.5 Economic Regime

Provided by Ask2Ai company, a possible approach to manage non-stationary problems is the classification of distinct economic regimes, providing a clearer perspective on market behavior in different periods. An economic regime is probability weights which presents market conditions. We use the economic regime framework to enhance predictive modeling and solve the data insufficiency problem.

Previous work uses a dictionary data structure to store the probabilities at a specific timeframe, in detail, we could derive our formula as below:

$$y_{\text{pred}} = \sum_{i=1}^{n_{\text{regimes}}} \left(P_{\text{prob}}(i|x_{\text{test}}) \times \text{probability\_of\_regime}_i \right)$$

- n_{regimes} is the number of regimes.
- $P_{\text{prob}}(i|x_{\text{test}})$ is the predicted probability of the positive class from the model for regime i.
- $\text{probability\_of\_regime}_i$ is the probability weight of regime i.

1. For each regime, we train a model using the sample weights corresponding to that regime. We first extract the regime probabilities for each date using a zig-zag way, stored as current_regime_probabilities.
2. In the perform_grid_search function, we perform a grid search to optimize the parameters of a Support Vector Classifier (SVC) for each regime. This involves training models with different hyperparameters and selecting the best-performing model based on cross-validation accuracy. For

each regime i, we train a model using the sample weights *Pi* corresponding to that regime, which ensures the model prioritizes learning from the most relevant data points.

3. Outside the gridsearch function, we multiply each model's probability prediction for the positive class by the corresponding regime probability. These weighted probabilities are then summed to compute the overall prediction score *y*pred. If this score exceeds 0.5, the final prediction is classified as the positive class.

In this way, we create a general frame to apply the macroeconomic regime as a special weights to our time series prediction.

4. Data

4.1 Data Collection

Data collection: For this part, we obtain a wide range of data containing crude oil production data as well as integrated oil data across different platforms, from EIA to Bloomberg. To collect our diversified dataset, we

1. For crude oil production data, we collect oil gas production,average rig count by play, Canada oil production, Mexican oil production, rig productivity, crude oil supply, and geographical information, etc.
2. For integrated oil data, we collect prices of crude oil index price benchmarks such as WTI (Western Texas Intermediate) and Brent price, also we collect offshore, frac spreads, refining margins
3. We collected the majority portion of data from the U.S. Energy Administration on a monthly basis. As a result, we saved a csv file with 42 features.

4.2 Data Preprocessing

We perform data cleaning and fetch the data which has the smallest proportion of Null value by visualizing the number of Nanl values against time. Our original dataset starts in the 1970s and we decided to pick data after the 1990s, which only allows approximately 10% of Null Values. Doing this allows us to predict models with a reasonable proportion of Nan values.

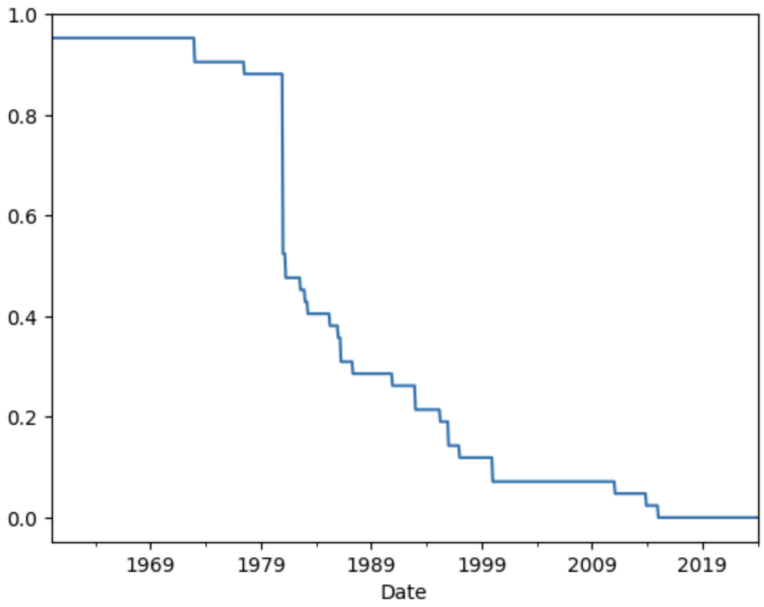

Figure 6. Total proportion of Nan Values versus timestamp (percentage)

5. Results and Evaluation

Through meticulous data transformation and feature selection, including detrending, standardization, and cosine similarity analysis, we optimized the performance of our predictive models. With careful tuning, the Support Vector Machine (SVM) outperformed other machine learning models, achieving a prediction accuracy of 63.28%.

This project focuses on developing and refining a comprehensive framework for predicting specific-market index returns to provide risk management signals for portfolio asset allocation. Our approach successfully completes and refines detailed and accurate steps for this framework. The framework encompasses data preprocessing and transformation, dimensionality reduction, cosine similarity analysis, feature selection, implementation and fine-tuning of machine learning and deep learning models, and ultimately, enhancing the model's ability to capture stationarity patterns in the financial market by integrating economic regime probabilities into the existing predictive models.

In evaluating our results, a primary challenge in optimizing model prediction accuracy is the limited dataset available. The scarcity of monthly industrial data hampers the grid search hyperparameter tuning process, resulting in relatively low prediction accuracy across various machine learning models and limited generalization ability of deep learning neural networks. Despite these challenges, the comprehensive and detailed framework we have developed provides valuable insights into step-by-step prediction of uncorrelated market return movements using raw industrial time-series data.

References

[1] Kay, Bradley. “Thematic Research: Introducing the Two Sigma Factor Lens.” *Two Sigma*, 24 Sept. 2020, www.twosigma.com/articles/thematic-research-introducing-the-two-sigma-factor-lens/.

[2] Hirsa, Ali, et al. "Robust Rolling PCA: Managing Time Series and Multiple Dimensions." *Robust Rolling PCA*, March 25, 2023.

[3] Hirsa, Ali, et al. "Robust Rolling Regime Detection (R2-RD): A Data-Driven Perspective of Financial Markets." *SSRN*, February 16, 2024.