Performance Clustering and Revenue Prediction Using Financial and Macroeconomic Data

Thomas MacPherson (tmacphe@umich.edu), Peter King (kingpete@umich.edu), Yueyao Ren (yyren@umich.edu)

Introduction

Traditional approaches to company valuation rely on human analysts making forward estimates of revenue and earnings, which are then used to justify investment decisions. This process is laborious, time-consuming, subjective, and often produces forecasts with significant errors when actual results are released. The problem is compounded when moving from an individual investor portfolio to a multi-billion-dollar mutual fund. The uncertainty around earnings reports introduces risk and inefficiency into the portfolio construction process. The problem we wish to address is how to generate more accurate and scalable forecasts of revenue so that we can produce company valuation models that outperform traditional analyst estimates and provide a stronger foundation for investment strategies.

More precise predictions of revenue reduces reliance on costly analyst estimates, allowing investors to allocate capital with greater confidence. Even moderate improvements in the accuracy of financial forecasting have the potential to transform investment decisions. For portfolio managers, this means building strategies that have less earnings surprises, potentially reducing the risk-adjusted returns. This could also allow for an arbitrage of mispriced opportunities earlier than competitors. At scale, this could lower the general cost of equity research and allow more access to high-quality valuation insights, increasing the efficiency across financial markets, bringing Wall Street to your computer screen.

The motivation for our project comes from the opportunity to generate excess returns, known as ‘alpha’, in a way that was once reserved for Wall Street institutions. Government regulations ensure ‘retail’ investors (non-institutional) have access to the same core financial information at the same time as hedge funds, while advances in computing power, data availability, and open-source tools have leveled the playing field. With these resources, an ‘at-home’ investor can now build and run sophisticated models that rival those of major funds. This shift creates a unique chance to use machine learning not only to compete with traditional analysts but to potentially uncover mispriced opportunities in the market and turn them into better investment outcomes. A key challenge to this is that financial data sets exhibit lower signal-to-noise ratios than those used by other machine learning applications due to arbitrage forces and nonstationary systems (López de Prado, 2020, p.19).

In this analysis, we applied dimensionality reduction techniques and cluster analysis to develop a better understanding of financial performance data, with the end goal being to extract features that would aid supervised models in predicting our target variable: future quarterly revenue. A principal component analysis (PCA) of base financial statement data revealed that a single principal component (PC) was able to capture 65 percent of the variance in the original data, indicating a high degree of multicollinearity among base features. Using K-Means and agglomerative clustering, we found that companies clustered well into seven classes across a broad range of feature subsets. This concept of clustering companies into categories according to their financial performance, and then using that category to aid in a supervised prediction task, appears to be a novel contribution among unsupervised feature extraction methods for financial analysis (Htun, et. al., 2023). Using multidimensional scaling (MDS) and t-distributed stochastic neighbor embedding (t-SNE) to visualize the data in two dimensions, we could discern local subclusters but found no clear pattern in the seven major classes, other than the raw financial size of the company (i.e., market capitalization). We also used PCA to extract a variety of features from an expanded dataset that included manually engineered features based on financial research, but neither the cluster categories nor the extracted features were ultimately useful in predicting our target variable.

We assessed multiple families of supervised learning models as a means to predict quarterly revenue. Revenue is one of the most important financial metrics used to value a company in the US stock market. Simple linear models struggled to capture meaningful relationships between features, but ensemble techniques and deep learning performed markedly better. We used an extensive grid search of hyperparameters to identify a champion gradient boosting ensemble decision tree model, and then performed extensive evaluation.

This study demonstrates a supervised regression method that predicts corporate revenue with a higher coefficient of determination than a dummy regressor. We found that previous quarterly revenue was the driving force in all of our supervised learning models. This suggests that our supervised learning technique may benefit from a more traditional time-series based analysis like vector autoregressive integrated moving average (VARIMA) analysis.

Related Work

The study by Gu, Kelly, and Xiu (2020) provided the conceptual foundation for extending deep learning methodologies from return forecasting to the prediction of firm-level fundamentals. Their models demonstrated that neural networks are capable of capturing nonlinear interactions among firm characteristics. This insight motivated the development of one of our approaches for supervised learning, which applies a similar deep-learning framework. Building on their architectural principles, we recontextualize the model to forecast future quarterly revenue rather than equity returns, offering an empirically grounded alternative to analyst-based revenue projections.

Singh and Thanaya (2023) explored the effectiveness of Extreme Gradient Boosting (XGBoost) models in predicting a stock's future earnings per share (EPS) using preprocessed technical, fundamental, and analyst forecast data, along with feature engineering techniques to construct indicators. Our study improves on this work by comparing the performance of a wider variety of model families, and distinguishes itself by constructing a novel and unique dataset.

Kurylek (2024) evaluated the precision of EPS forecasts over a wide array of explanatory variables, including financial, market, and macroeconomic factors, across diverse model families, including gradient-boosted decision trees, multilayer perceptron, and convolution networks. Our study differs in its use of unsupervised techniques for feature extraction and in sourcing data from U.S. stock exchanges (Kurylek used data from the Warsaw Stock Exchange).

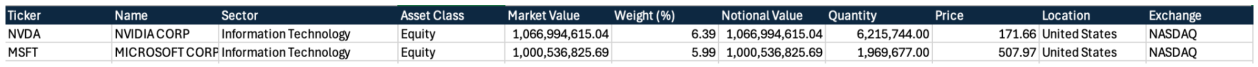
Data Sources

Data Acquisition

This project builds a custom dataset by integrating company-level financial fundamentals from Yahoo Finance (via the yfinance API) with macroeconomic indicators from the Federal Reserve Economic Data (FRED) API.

Cross-Sectional Data

The foundation of the dataset is derived from the Russell 3000 Index, downloaded directly from the iShares Russell 3000 ETF webpage. The CSV file contains 2,600 firms and 11 attributes, including Ticker, Company Name, Sector, and Market Value (examples in **Table 1**). These cross-sectional variables provide essential firm identifiers and classification features for subsequent data merging.



Ticker	Name	Sector	Asset Class	Market Value	Weight (%)	Notional Value	Quantity	Price	Location	Exchange
NVDA	NVIDIA CORP	Information Technology	Equity	1,066,994,615.04	6.39	1,066,994,615.04	6,215,744.00	171.66	United States	NASDAQ
MSFT	MICROSOFT CORP	Information Technology	Equity	1,000,536,825.69	5.99	1,000,536,825.69	1,969,677.00	507.97	United States	NASDAQ

Table 1: Example of data from the Russell 3000 Index.

Financial Statement Data

Using the Ticker column as the primary key, we retrieved company-level financial statement data through Yahoo Finance, covering the five most recent quarters (Q2 2024–Q2 2025). For each firm, we collected 18 key financial variables representing profitability, liquidity, leverage, and operational efficiency—such as Revenue, Net Income, Total Assets, Total Debt, and Earnings Per Share (EPS). This time-series component captures quarterly variations in firm fundamentals across the Russell 3000 universe. The integration of these data points added 90 new columns (18 variables × 5 quarters) to the dataset.

Macroeconomic Data

To contextualize firm performance within broader economic conditions, we incorporated five macroeconomic indicators from the FRED API, collected over the same five-quarter horizon: Gross Domestic Product (GDP), Inflation Rate (CPI), Unemployment Rate, Interest Rate (Federal Funds), and Industrial Production Index. These variables contributed an additional 25 columns (5 indicators × 5 quarters) to the dataset.

Alignment and Integration

All datasets were aligned by calendar quarter to ensure temporal consistency between company-level and macroeconomic information. The final integrated dataset contains 2,600 company observations and 126 feature columns, enabling a comparative and contextual analysis of firm performance in relation to macroeconomic dynamics.

Feature Engineering

Imputation of Missing Values

To address missing values in the dataset, we applied a sector- and market-cap–specific imputation strategy. For each variable with missing entries, we computed the median values within corresponding market cap tranches (**Figure 1**) and sector groups, then used these medians to fill in gaps. This approach preserves the relative structure of firms within similar peer groups while reducing bias compared to global mean imputation.

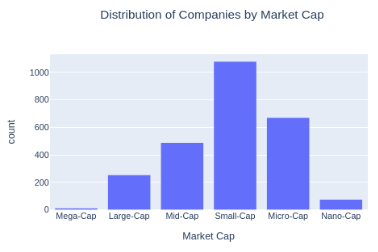


Figure 1: Market Cap distribution. *(note: all figures in this report are included at full size in Appendix C)*

In addition, we removed companies reporting zero revenue. This decision was made for two key reasons: First, from an analytical validity perspective, revenue serves as the denominator when calculating net profit margin, and a value of zero would distort the metric. Second, from business relevance, firms with no reported revenue are not considered representative of normally functioning operating companies. Furthermore, we restricted the sample to firms listed on major U.S. exchanges—specifically, the NASDAQ and the NYSE—to ensure comparability and market representativeness.

Nonlinear Transformation of Financial Statement Data

Histograms of the raw data from company financial statements revealed that much of our base data has a log-normal distribution. We observed that a log transformation resulted in more bell-shaped histograms, and had hoped to use this to reduce the distortion that significant outliers like Amazon, Nvidia, JP Morgan and other market giants introduced to the dataset. **Figure 2** shows distributions for Market Value.

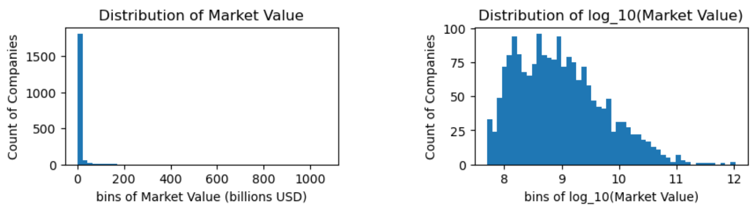


Figure 2: Distribution of $\log_{10}$(Market Value) suggests a log-normal distribution; we found similar results for most base variables.

Unfortunately, nearly a quarter of companies (459 out of 1910 in the training set) reported negative figures for Net Income, a key metric, making it impossible to apply a log-transformation to all base variables.

Key Performance Indicators (KPI)

Financial analysts typically assess the financial health of a company through a set of ratios derived from the company's quarterly and annual financial statements. In this project, we used a set of ten ratios recommended by Harvard Business School (Stobierski, 2020) as "key performance indicators" (KPI). Based on expert advice, these features were designed to enhance the base dataset and assist supervised learning algorithms in predicting company financial performance. **Appendix B** provides formulas.

Quarter-over-Quarter Performance (QoQ) and General Rates

A key challenge in developing our model was the restriction of free, publicly available data sources. Our source provided five quarters of historical financial statements. It was not possible to apply traditional time-series methods in this limited time window. To overcome this, we engineered features designed to capture temporal relationships within the data. Initially, we calculated the quarter-over-quarter rate of change calculated as $QoQ = \frac{\gamma_t - \gamma_{t-1}}{\gamma_{t-1}}$, where γ is a financial variable and t is the quarter for all temporal features. This allowed us to identify the rate of change in these variables over time and also allowed for direct comparison between companies. We extended this approach by examining each variable over the preceding quarters of the prediction period, fitting a line of best fit, and extracting the slope as a smoothed measure of trend. These engineered features allowed the model to incorporate dynamics of financial performance over time, despite the limited time horizon.

Feature Elimination and Extraction using Unsupervised Learning

The expanded training dataset contained only 1910 rows with 325 features per row. With relatively few training examples for data of such high dimensionality, we decided to use PCA to extract features that would capture most of the variance in fewer dimensions. In this way, PCA also served as a form of preprocessing to remove noise. We found that, for most subsets of features, the first 50 PCs captured roughly 85 percent of the variance in the data. A clustering analysis using both K-Means and agglomerative clustering found that data grouped well into seven clusters, and we used these clusters as class labels to augment the data for our supervised models.

Complete Machine Learning Feature Space

See **Appendix B** for a complete list of features contained in our finalized dataset.

Unsupervised Learning

Overview: Workflow, Choice of Methods

We used unsupervised learning methods to accomplish two primary objectives: 1) to develop a better understanding of the nature and structure of our data, and 2) to extract a smaller set of features for use in training our supervised models.

We began our analysis with a visual inspection of the data in lower dimensions using **Multi-Dimensional Scaling (MDS)** and **t-distributed Stochastic Neighbor Embedding (t-SNE)**. These two methods use different dissimilarity measures and stress functions to reduce high-dimensional data to a lower dimension while preserving some form of structure. They yield visual representations that are useful for different tasks. MDS tends to preserve global distances and is useful for interpreting the macro-structure of the data, while t-SNE was designed to "[reveal] structure at many different scales," and tends to preserve local structure (van der Maaten & Hinton, 2008). We used these visualization methods because we wanted to look for both global and local structure in the data -- for both large, macro-clusters, and smaller, local subclusters.

Next we applied two different clustering methods (**K-Means** and **hierarchical clustering**) that differ markedly in their approach to forming clusters. We chose hierarchical clustering because this method does not require the analyst to specify a number of clusters in advance, and we did not have enough domain knowledge to predict how many clusters we should expect to see. We chose K-Means for its simplicity and straightforward interpretability. If we found that the data clustered well, our intention was to use cluster labels to assist in training supervised models.

Finally, because the complete dataset consisted of over 300 features, we applied **PCA** to reduce the dimensionality of the data, reduce multicolinearity, and perform feature extraction. Our goal was to extract high-variance features (in orthogonal directions) that would help a supervised learning model distinguish between types of companies, with the idea that these differences would be important for predicting the target variable (Revenue) with high accuracy.

Description of Methods and Hyperparameter Tuning

Visualization method: t-SNE

The t-SNE algorithm defines a probability distribution over the set of all pairs of data points, and attempts to minimize the Kullback-Leibler (KL) divergence between the original high-dimensional distribution and a distribution of a lower-dimensional representation. The use of a Student's t-distribution for the lower-dimensional space alleviates the problem of crowding clusters in the visual representation, due to its heavier tails compared to a Gaussian (van der Maaten & Hinton, 2008). The resulting visualization tends to clearly separate local clusters, but the distance between clusters in the visualization has no meaning.

For hyperparameter tuning, we chose *cosine similarity* as the **dissimilarity measure** because the expanded data (including derived variables) was heterogeneous: base financial data is in thousands of dollars (USD), while rates of change (QoQ and others) are percentages over time, and most KPIs are unitless ratios. A comparison using *Euclidean distance* verified that cosine similarity did a better job of enhancing the visual separation of clusters in the data. To select **perplexity**, we first estimated an appropriate value and then tested a range of values around it to identify the setting that yielded the best visual separation. We also compared three major **subsets of features**: 1) *financial statement data* (base data only), 2) the *complete dataset* including all engineered features, and 3) a set of only the *engineered features*.

Visualization method: MDS

While MDS can use a variety of techniques and metrics, each is based on a geometric interpretation of the data and tries to preserve some form of global or macro-level dissimilarity between pairs of points. All versions of MDS seek a lower-dimensional representation of the data that preserves the pairwise dissimilarities in the original high-dimensional data.

For hyperparameters, we compared three **types of MDS** (*metric*, *non-metric*, and *classical*), using two **dissimilarity measures**: *Euclidean distance* and *cosine similarity*. We also explored four different **subsets of features**: 1) *base financial data*, *KPIs*, *rates of change*, and the *complete* dataset. Because our data was heterogeneous and contained extreme outliers, we believed non-metric MDS using cosine similarity would be the best combination, but we wanted to compare results using different measures.

Feature Extraction method: PCA

PCA finds the directions of greatest variance in the data through a spectral decomposition of the data covariance matrix. We chose this method because our dataset is small and high-dimensional (1910 rows x 325 features), and we believed we would see a high degree of multicollinearity between many of our features. PCA is useful in alleviating both of these concerns. It is able to condense the variation inherent in many features into a few extracted features.

We tried a variety of **scaling functions** to standardize the data before applying the PCA algorithm: the *StandardScaler*, *RobustScaler*, and *QuantileTransformer* from scikit-learn's preprocessing module. We also evaluated various **subsets of features**: 1) the *financial statement data*, 2) the *complete dataset*, and 3) various combinations of *engineered features* (KPIs, rates of change, and macroeconomic variables).

Clustering method: K-Means

K-Means assumes that data has been generated by a number (k) of multivariate Gaussian variables. The algorithm attempts to find the mean and variance of these Gaussians by iteratively determining cluster centers (initially at random), and then assigning nearby points to clusters. Our study design assumed that companies would naturally fall into different categories, and that these categories would be important for predicting the target variable.

The key hyperparameter in the K-Means algorithm is the **number of clusters** (the 'K' in K-Means). We also clustered using different **subsets of features:** the *base* features only, and various combinations of *engineered* features and *PCA-extracted* features.

Clustering method: Hierarchical Clustering (bottom-up)

The agglomerative clustering algorithm starts by computing a proximity matrix where each data point is considered a cluster, and then proceeds to iteratively 1) merge the two closest clusters, and 2) update the proximity matrix. The key operation is computing the proximity of two clusters, which can be accomplished using different cost functions. The most often used is Ward's distance, which compares the sum of squared distances of each point to its cluster center with the sum of squared distances from each point to a center defined by the merging of the two clusters. Specifically, Ward's distance defines the distance between two clusters as:

$$D_w(C_i, C_j) = \sum_{x \in C_i} (x - r_i)^2 + \sum_{x \in C_j} (x - r_j)^2 - \sum_{x \in C_{ij}} (x - r_{ij})^2$$

where each *r* is a centroid of its matching cluster *C*, and C_{ij} indicates that cluster C_i has been merged with cluster C_j. Estimating the number of clusters using this method can be subjective, and becomes especially challenging when domain knowledge is scarce. Since our purpose was to estimate a number of clusters for comparison with K-Means, we used *Ward's distance* as the **cost function** and *Euclidean distance* as the **dissimilarity measure**.

Unsupervised Evaluation

Overall Results

K-Means Clustering and Visualization using PCA

We used the Within-Group Sum of Squares (WGSS) score to evaluate our clusters. WGSS is the sum of the squared distance from each point in the dataset to its associated cluster center. A low WGSS score for a set of clusters indicates that the clusters are relatively compact, while a high WGSS score indicates that points lie far from their cluster centers.

Prior to clustering, we standardized data using scikit-learn's Quantile Transformer, which we found to be more robust to outliers but still sensitive to nuanced differences between companies. After scaling, we applied PCA to the complete dataset and kept only the top 50 PCs, which captured 85 percent of the variance in the data. **Table 2** shows that the KPI feature subset achieved the lowest WGSS score, indicating the tightest clustering. An elbow plot for all subsets showed a bend at seven clusters (**Figure 3**, KPI subset shown). We also found that this clustering resulted in meaningful visualizations.

Subset	WGSS at 7 clusters
complete	21,624
all engineered	14,318
rates of change	10,847
KPI	2,581
base financials	2,666

Table 2: WGSS scores at 7 clusters for various subsets of features.

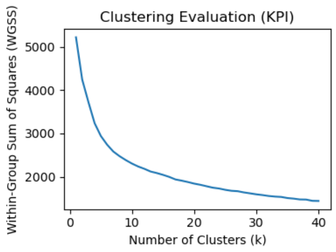


Figure 3: Elbow plot of WGSS for a range of clusters (K-Means)

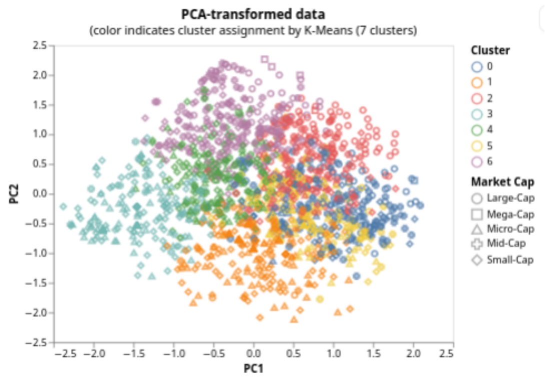


Figure 4: Plot of first two principal components (PCA using all features).

Figure 4 shows a plot of the first two PCs for the complete dataset after transformation using PCA. We can see that Mega- and Large-Cap stocks tend to move towards the upper-right quadrant, while Micro- and Small-Cap stocks tend to move towards the lower-left quadrant, and Mid-Cap stocks are more centrally located, suggesting that, while additional PCs capture more nuance and variation, the most salient difference between companies is still their raw financial size.

Hierarchical Clustering and Visualization using MDS

Most elbow plots of WGSS under K-Means had suggested that our data would group well into seven clusters. Inspection of the dendrogram for an agglomerative clustering of companies using base financial data (**Figure 5a**) confirmed that seven clusters was a good fit for our dataset. These clusters could easily be divided further into 12-15 subclusters, or merged into 3-5 major clusters, but seven clusters presented a good middle-ground and aligned well with the results we saw using K-Means.

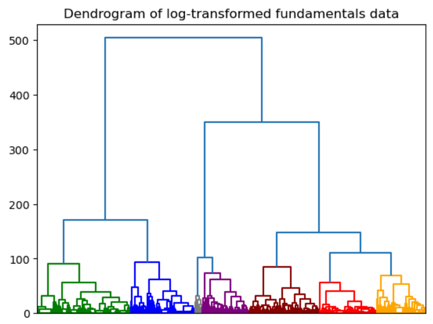


Figure 5a: Agglomerative clustering dendrogram highlighting 7 clusters.

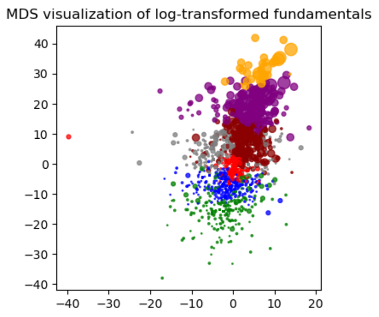


Figure 5b: Visualization of 7 clusters (via agglomerative clustering).

Despite varying dataset sizes and complexity, non-metric MDS using cosine similarity consistently achieved the lowest stress, demonstrating robustness and suitability for high-dimensional financial data. One useful visualization (**Figure 5b**), where colors correspond to cluster labels and size corresponds to Market Cap, revealed that a clustering analysis of base financial data essentially grouped companies according to their size: most Mega-Cap companies appear in orange at upper right, transitioning down to the Small- and Nano-Cap companies at bottom.

Visualizing subclusters using t-SNE

Clear clustering patterns emerged, showing that companies within the same sector and of similar size shared comparable financial characteristics (**Figure 6a**). This justified our choice to impute missing data using the median of groups based on Sector and Market Cap, since these groups captured meaningful similarities. Importantly, when additional engineered features were introduced, the t-SNE visualization reduced clustering by sector and market cap (**Figure 6c**). This lack of local separation indicated that the new features were improving cross-company comparability by emphasizing underlying financial performance rather than superficial categorical differences seen in sectors and market capitalizations, thereby strengthening the robustness of our supervised models.

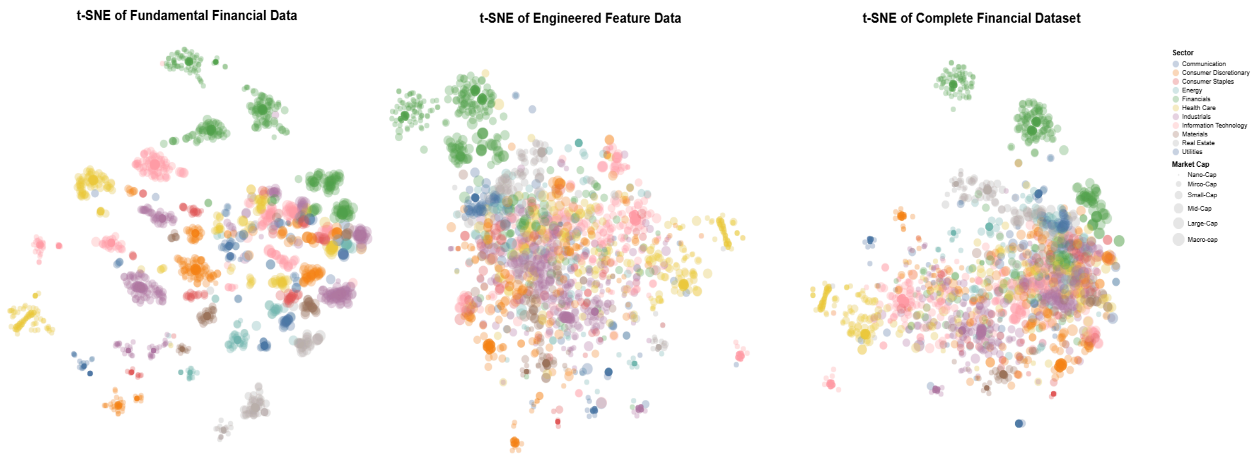


Figure 6 (a, b, c): t-SNE visualizations showing sector and market cap clustering based on a) raw financial data, b) engineered features, and c) complete dataset. Colors correspond to market sector and point size corresponds to market capitalization.

Sensitivity Analysis: Feature Extraction using PCA

Scaling Functions

We found that normalization using Standard Scaler proved too sensitive to outlier companies like Amazon, Google, and others to produce acceptable results using PCA: most companies were crammed together in a tight cluster, while a few Mega-Cap companies skewed the distribution significantly (**Figure 7a**). By contrast, the Robust Scaler appeared to compensate almost too much for the outliers, and was unable to capture nuanced differences between the smaller companies that constitute over 95 percent of the data. We found that the Quantile Transformer was able to enhance (i.e., spread out) the variance of smaller companies, while simultaneously mitigating the skewing effect introduced by large companies (**Figure 7b**). During the supervised learning evaluation, we found that model accuracy improved dramatically when data was scaled using the Quantile Transformer, as opposed to the Standard Scaler.

PCA using Standard Scaler **PCA using Quantile Transformer**

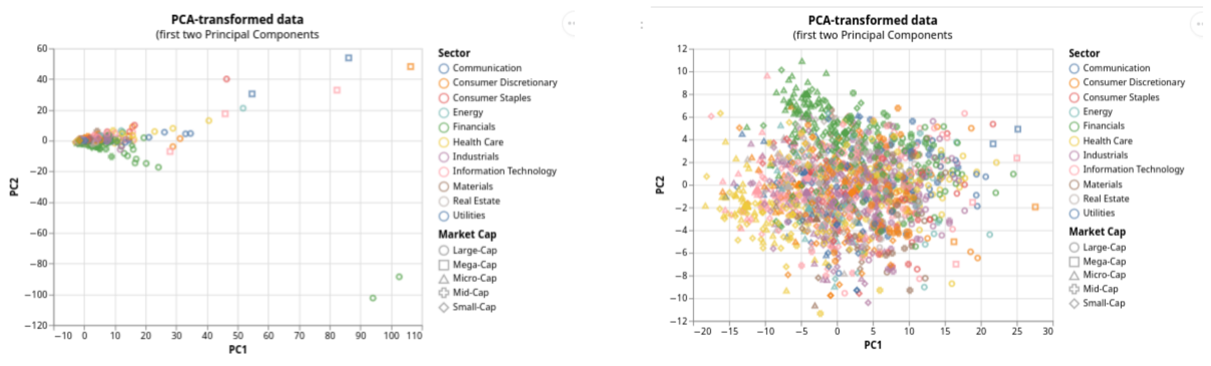


Figure 7 (a, b): Sensitivity of PCA to different scaling functions used to preprocess data.

Feature subsets

Subset	# PCs required
complete	49
all engineered	46
rates of change	40
KPI	11
base financials	6

Table 3: Sensitivity of PCA to various feature subsets

Table 3 shows that most feature subsets required 40-50 PCs to capture 85 percent of the variance in the data, with two notable exceptions: the base financial subset (where the first PC alone captured 65% of the variance) and the KPI subset. We interpreted these results to mean that we were correct in suspecting a high degree of multicollinearity in the base financial data, and that our engineered features helped to provide a richer, more nuanced representation of each company. We extracted 50 features using the complete dataset (to capture as much nuance as possible) and 10 features using the KPIs only. The decision to extract KPI-based features was based on the low WGSS score achieved by clustering with this subset.

Supervised models demonstrated far superior performance using the base features, or base features combined with engineered features, than with the engineered features alone. Model performance using solely extracted features (via PCA) was significantly worse compared to performance when base financials were included. We later discovered that this was due to over-reliance on a single type of feature: previous quarter revenue. A more detailed discussion follows in the supervised learning section.

Supervised Learning

Methods Description

We began our supervised learning phase by conducting a systematic comparison of our entire feature space as described earlier (**Appendix B)** across multiple families of supervised learning models, each tested with the baseline hyperparameters. The goal of this exploration was to identify a family of algorithms that showed the most promise given our dataset. This general evaluation allowed us to see whether the problem could be effectively addressed by simpler methods like linear modeling or if it required more complex approaches.

Linear models excel in cases where the relationships between predictors and outcomes are relatively straightforward and are often computationally inexpensive. However, other problems involve nonlinear interactions that need more complex approaches. Approaches like tree-based or deep learning, can automatically capture hierarchical patterns and subtle dependencies in the data. By contrasting performance across these model families, we were able to evaluate this trade-off directly and quantitatively without personal bias.

To ensure fairness in the comparison, we designed a consistent preprocessing pipeline that handled both categorical and quantitative features. Categorical variables were each one-hot encoded, and quantitative features were scaled using the QuantileTransformer from Scikit-learn, which is more robust to outliers. We then applied 10-fold cross-validation scoring to each supervised model, which allowed us to estimate out-of-sample performance while reducing the risk of overfitting to any particular fold. This process provided a rigorous, data-driven foundation for selecting the most appropriate model before proceeding to hyperparameter optimization and further refinement. After determining the nonlinear relationship in the data, we thought we might get better results using tuned deep learning, so we proceeded with the ensemble and deep learning techniques.

After selection of the appropriate models, we used scikit-learn's GridSearchCV function to perform a comprehensive grid search of hyperparameters using 10-fold cross validation evaluated using the coefficient of determination (R^2) value. Hyperparameters tuned in gradient boosted trees included learning rate, number of estimators, max depth, minimum sample per leaf, minimum samples per split, max features, sub sample, and two loss functions. This resulted in tens of thousands of models being tested and more than six hours of computation, however we felt that this was an important step to take to ensure we were obtaining a quality model to investigate further.

In parallel, we developed a deep learning regression model using TensorFlow's Multi-Layer Perceptron (MLP) architecture. The MLP employed multiple dense layers with GELU activations, dropout regularization, and the Huber (Smooth L1) loss function, optimized using AdamW with a warmup cosine learning rate schedule. We evaluated the model with metrics including R^2, MAE, RMSE, and a custom SuccessRate metric to measure prediction accuracy within specified error thresholds. Deep learning outperformed gradient boosting when measured by R^2 , however the improvement was not significant enough to overcome the lack of transparency in deep learning techniques, especially when performing more in-depth evaluations like failure analysis.

Once our champion gradient boosted model was selected based on the R^2 evaluation metric, we performed in-depth model evaluation to assess robustness. This included feature importance, feature ablation, learning curve, sensitivity, and failure/error analyses. Feature importance ranks the importance of a feature on its ability to separate instances to produce the best improvement in the R^2 value. Feature ablation involves the removal of features or 'sets' of features to see the impact on model performance. Learning curve analysis allows us to assess our training set size and determine if acquisition of more data may help improve the performance of our model. Sensitivity analysis allows us to look at the robustness of our model across a range of hyperparameters to increase our confidence in out of sample testing. Finally, failure analysis allows us to identify potential shortcomings in our model to help make decisions for future data acquisition and model tuning.

Supervised Evaluation

Overall results

In comparing the model families we used three major evaluation methods: the coefficient of determination (R^2), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE). Our primary model evaluation and selection was done using R^2, as it directly assesses the explanatory power of our model, not just the size of prediction error. It also allows us to have future comparisons across models and datasets, calculated as $R^2 = 1 - \frac{unexplained\ variance}{total\ variance}$. This value also compares the performance to a naive baseline (predicting the mean only) and helps us capture the variance in the model. RMSE and MAE were also recorded to see the absolute error values and to penalize values with significantly higher error. However, we found that due to the nature of the scale of our predictor variable (Revenue), our absolute errors tended to over emphasize companies with large revenue. For example, a 10 billion dollar miss on a company that is making 100 billion in revenue is still only a 10% miss. This led us to incorporate a 'SuccessRate' measure in determining our champion model that looked at how far the 'relative miss' was. A 'SuccessRate@10%' was the percentage of companies with an estimated revenue within 10% of actual quarterly revenue.

Model	R^2 Mean(±std)	RMSE Mean(±std)	MAE Mean(±std)
DUMMY (median)	-0.093 (0.041)	7515M (4143M)	2425M (862M)
Linear (LR)	-0.143 (0.608)	6808M (2741M)	4031M (370M)
Ridge (RIDGE)	0.191 (0.255)	6149M (3106M)	3488M (471M)
Lasso (LASSO)	-0.059 (0.485)	6665M (2783M)	3962M (398M)
ElasticNet (EN)	0.313 (0.079	6159M (3940M)	2521M (654M)
KNearestNeighbor (KNN)	0.327 (0.230)	6114M (4147M)	1550M (713M)
Decision Tree (DT)	0.801 (0.255)	3234M (4010M)	625M (723M)
Support Vector Machine (SVR)	-0.093 (0.041)	7515M (4143M)	2425M (862M)
Random Forest (RFR)	0.928 (0.082)	2055M (2469M)	351M (299M)
Extra Trees (ETR)	0.955 (0.061)	1705M (2288M)	329M (281M)
AdaBoost (ABR)	0.908 (0.096)	2304M (2813M)	1004M (303M)
Gradient Boosting (GBR)	0.916 (0.155)	1695M (2249M)	327M (297M)
Multi-Layer Perceptron (MLP)	-0.179 (0.089)	7732M (4114M)	2674M (876M)

Table 4: Supervised Learning family evaluation using coefficient of determination (R^2), root mean squared error (RMSE), and mean absolute error (MAE) in ten-fold cross validation reporting mean and standard deviation. Ensemble techniques clearly showed advantages over simpler linear based regression models leading us to explore the more complex machine learning families. With the complexity of deep learning, we assumed that we could still improve performance over a baseline MLP.

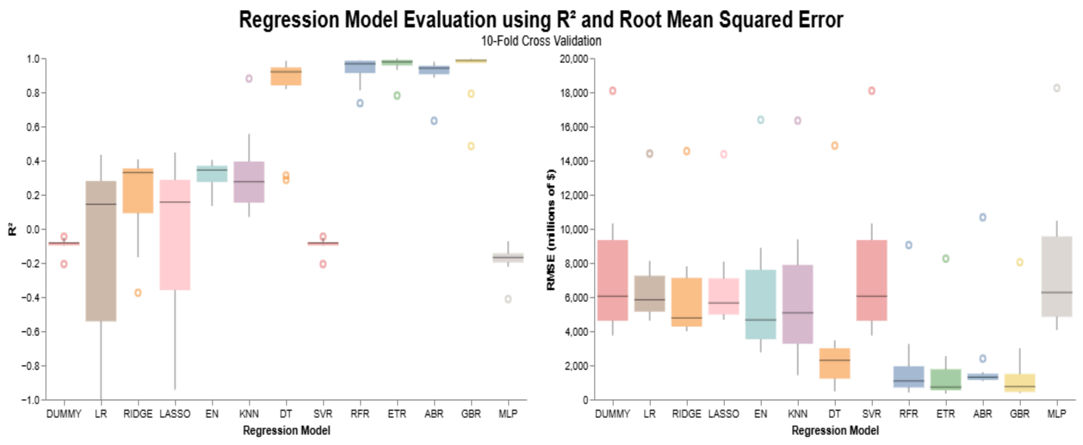


Figure 8: Supervised Learning family evaluation using R^2 and RMSE in ten-fold cross validation. Ensemble techniques showed advantages over simple linear based models leading us to explore the more complex supervised learning families.

Ensemble learning had a clear advantage over linear learning in all evaluation spaces **(Table 4, Figure 8)**. This signified a nonlinear relationship in our data that led us to investigate gradient boosted learning and a more comprehensive deep learning model. Ultimately, deep learning using multi-layer perceptrons achieved the highest scores after hyperparameter tuning with an R^2 of 0.973 ± 0.009 and a success rate of 78% of companies with predicted revenue within 20% of actual revenue in our out-of-sample test set. However, our gradient boosted learning model was very close with an R^2 of 0.967 ± 0.025 and a success rate of 75% of companies with predicted revenue within 20% of revenue in our out-of-sample test set.

Ultimately, due to explainability and in-depth model evaluation, we chose to use the more transparent gradient boosted learning with a learning rate of 0.15, using the squared error loss function, with a max depth of 3 and max features of 0.5. We had a minimum of 1 sample per leaf and a minimum of 50 samples per split with a total of 200 estimators. Initial analysis of our model shows that our predictive abilities weaken as quarterly revenue increases and there are fewer financially comparable companies **(Figure 9)**. Analysis of residuals showed that the variance of residuals tends to increase with the size of the prediction: a heteroscedastic relationship that tends to exaggerate R^2 values.

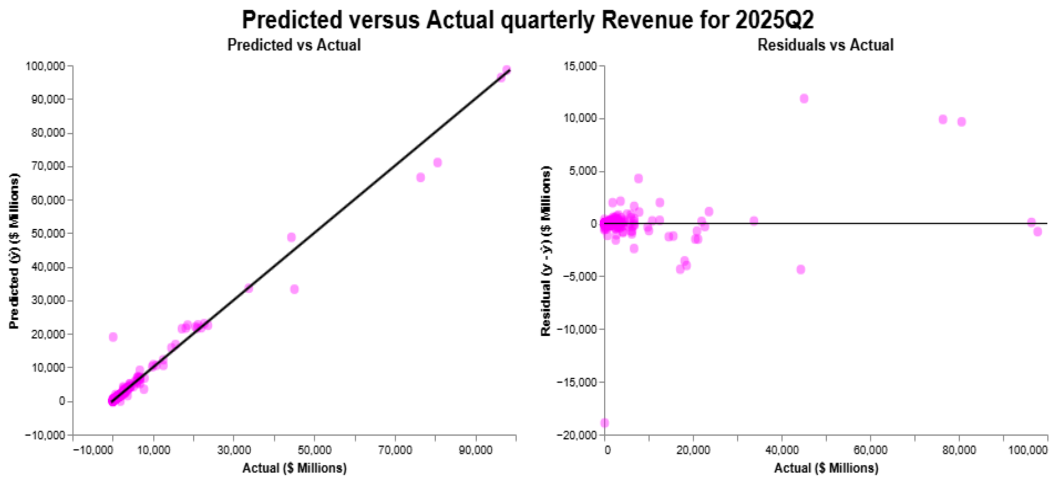


Figure 9: Correlation and residual plot of Gradient Boosted Model. a) Correlation between predicted and actual quarterly revenue of companies with diverse market capitals. b) Residual plot showing error between prediction and actual quarterly revenue with the most significant error coming from companies with high quarterly revenue and few financially comparable companies.

In-Depth Evaluation

Training Data Curve

Initially, we wanted to assess whether or not we had sufficient representative data to achieve stable model performance. We performed a learning curve analysis to determine how the change in training set size affected the RMSE. We can see a relative stabilization in 10-fold cross-validated mean RMSE around 800 samples, indicating that we have enough samples in our training set for a valid estimation. However, we can see that with increasing sample size we are continuing to reduce the variance as indicated by the standard deviation shadow on **(Figure 10).** This implies that we may benefit from including more companies over a more diverse timeframe.

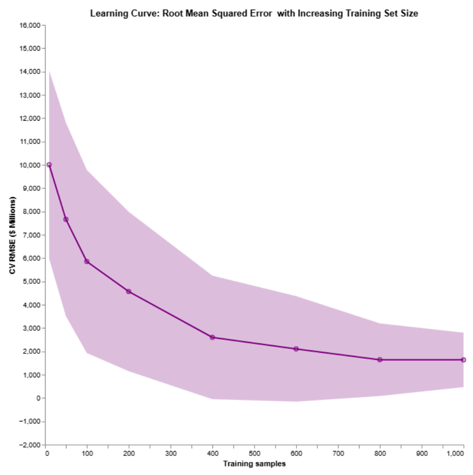


Figure 10: Learning curve analysis based on the size of our training dataset. While our error tended to plateau around 800 samples, we continued to see a reduction in variance up to our sample size indicating a further increase in the number of companies used to model may continue to improve performance.

Feature Importance and Ablation Analysis

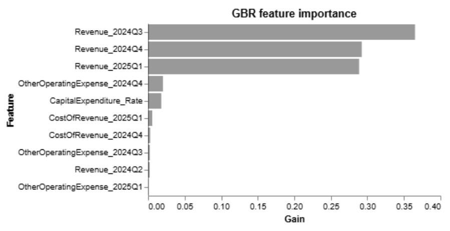


Figure 11: Feature importance as measured by Scikit-learn in the gradient boosted learning model. There was clear preference for revenue related features in our decision trees.

One of the large benefits of using tree based regression analysis is the ease at which we can assess feature importance. Ranking the features by importance allows us to reduce the feature space and significantly reduce the computational time using our model while keeping performance optimal. We ranked features based on importance and selected the top 10 features **(Figure 11)**. This showed a clear preference for revenue based features and only included one feature that we engineered ourselves. We expected that previous revenue would have a strong impact in our model forecasting but we underestimated the extent to which it would play a role. This led us to do further feature ablation analysis to determine how our model would perform using subsets of the data. These subsets included using only previous revenue, using the top 10 features in our feature space as ranked by feature importance, dropping only the most important feature, dropping the most recent quarterly revenue, and only using revenue itself as the independent feature **(Figure 12)**. Surprisingly, using revenue only led to the smallest standard deviation in both R^2 and RMSE and very similar performance compared to our model trained in the full feature space.

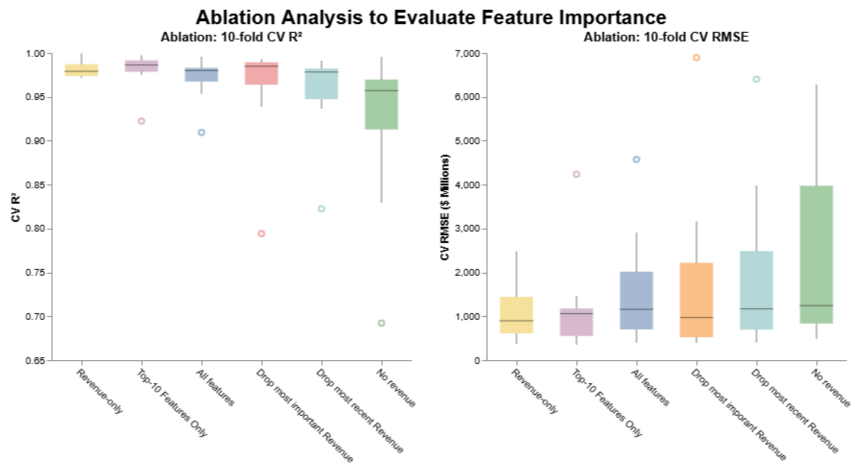


Figure 12: Ablation analysis comparing different feature space and the effect on model performance as measured by R^2 and RMSE. The most pronounced effect was seen on the variance in model performance as opposed to median values.

Removal of all revenue data however, led to a 5% drop in performance with a significant increase in variance as measured by a 0.09 standard deviation. Similar results were seen in the RMSE.

Sensitivity Analysis

Sensitivity analysis is another important evaluation of our champion model. This analysis shows how robust the model is to small changes in the hyperparameter space. If a model is very sensitive to changes in hyperparameter space, it is more likely to struggle with out-of-sample and changing data. Recall, we performed an extensive grid search in order to determine our champion model, however, we wanted to investigate the hyperparameter space around our selected values to determine the models robustness. Using RMSE, where lower values indicate less error and a better performing model, we can clearly see a valley of successful hyperparameters around those at which we chose. This indicates that although there is slight variation in error around our hyperparameter space, it is robust to small changes in the values.

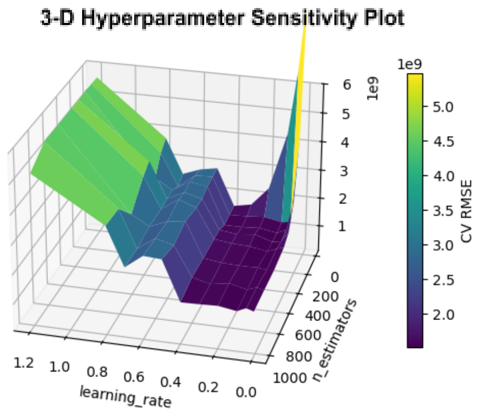


Figure 13: Sensitivity analysis across the learning rate and number of estimators hyperparameters. The lower the value of RMSE the better model performance.

Failure analysis

For our failure analysis, we randomly chose three companies where the prediction of quarterly revenue was more than 50% different from the reported quarterly revenue. Our goal was to identify three different categories of error. We looked at common features among these companies and compared them to companies that performed very well. Interestingly, companies of all market cap sizes performed both well and poorly, even though we thought Large- to Mega-Cap stocks would be hard to predict due to a more limited sample size (unbalanced data).

Ticker	Sector	Market Cap	Exchange	Revenue_2024Q2	Revenue_2024Q3	Revenue_2024Q4	Revenue_2025Q1	Revenue_actual	Revenue_predicted
TNET	Industrials	Small-Cap	NYSE	1243M	1237M	1326M	1292M	0.075M	1204M
WLK	Materials	Small-Cap	NYSE	3207M	3117M	2843M	2846M	1.313M	3253M
GLW	Information Technology	Large-Cap	NYSE	3251M	3391M	3501M	3452M	7.554M	3735M

Table 5: Three companies were chosen for failure analysis to determine what might be causing the most significant inaccuracies in our model.

Similarly, the Industrials and Information Technology Sectors were found in both the top three and bottom three predictions, indicating that sector volatility was not the likely culprit. NASDAQ listed stocks only made up 38% of our analyzed companies and 3 of the top 4 firms in quality of prediction were listed on the NASDAQ whereas all of the worst performing companies were listed on the New York Stock Exchange (NYSE). It was also interesting to note that companies who had consistently growing, or consistently shrinking revenues over the last 4 quarters all made it in the top predictions, whereas companies that had fluctuating revenues between quarters tended to perform poorly with larger residuals (random errors). If we had access to more historical data, many of these shortcomings would be addressed with a more traditional time-series approach to data analysis and modelling (Vector Autoregressive Integrated Moving Average; VARIMA). This could use autoregression of all line items in the financial statements to see if there were quarterly cycles and potential causal line items in

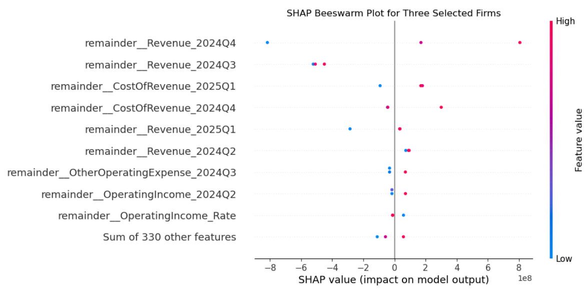


Figure 14: SHAP beeswarm plot to show feature importance for firms in which our model had poor forecasting performance. Large positive (negative) values caused the predicted value to be higher (lower) than the average predicted value. Colorbar indicates the $ value of the feature in the particular instance analyzed.

the financial statements, while making the data semi-stationary and smoothed with the moving average.

Far and away, the most significant failures in our model came from 'outlier misses' **(Table 5)** in the reported dependent variable, quarterly revenue of Q2 2025, and could be considered 'edge cases'. These were orders of magnitude off of previous quarterly revenues, and called into question our data validity. We used this information to verify the reported financial data from additional sources and found that the true reported revenue did not match the revenue obtained from yfinance (data quality). For instance, TriNet Group (TNET) reported an actual quarterly revenue of $1.24B in Q2 2025, indicating our actual error was only $40M, not ~$1.1B. Similarly, Westlake Corporation (WLK) reported a quarterly revenue of 2.95B, not 1.3M, reducing our residual to only $300M. The fact that our model is so reliant on previous revenue data features, the data quality and information contained in these features made up nearly all of our significant failures, with three examples shown against all top features in a SHAP beeswarm plot **(Figure 14)**. In this figure, we can see that the revenue features had large positive and negative impacts on the model output, driving our predicted target to substantially increase or decrease relative to the average prediction of the model. The color bar indicates the raw value of the actual feature showing we have both low and high values for Q4 2024 Revenue for a diverse comparison, for example. Comparing every reported revenue to revenue obtained by hand would not be a feasible task, however obtaining data from another free source and doing a large-scale cross-validation could benefit our model. Ultimately, obtaining data from a paid service that offers more reliable data may be the more time efficient method.

Tradeoffs

In order to achieve an understanding of the model, we made a few tradeoffs in the process. We chose the accuracy and quality of data at the expense of the quantity of data. We began with 2600 stocks, but at the time of modeling, had only ~1350 firms in our training set. We tried to eliminate highly volatile (nano-cap) stocks and companies that don't trade on the common U.S. stock exchanges as they may have different capital structures. We also removed any companies that had obviously incorrect data like a negative revenue which is not financially possible. We ultimately chose the coefficient of determination as our primary evaluation metric, even though forecasting would be in raw dollar values. We found that with revenue ranging from hundreds of thousands to hundreds of billions, and variance being heteroscedastic, absolute value evaluation metrics like RMSE and MAE biased toward the larger revenue companies, with squared errors in the billions of dollars ballooning quickly. We found alternate ways to use RMSE and MAE, for example in our in-depth model analysis, so we found it useful to track all three evaluation metrics. After in depth analysis, we found that only using 10 features was enough to have similar, if not better, results in our model. This came at the trade off of countless hours of feature engineering being removed from the model. As mentioned previously, we made the tradeoff from a deep learning model with minimal model performance improvements to an ensemble based technique for explainability and a deeper model analysis. Finally, and likely the most significant, was the tradeoff of using vector based data representation instead of a time-series data representation due to the limitations for free and public financial datasets for the project.

Discussion

Unsupervised Discussion

We used dimensionality reduction, visualization, and clustering methods to develop a better understanding of our data and to extract useful features that would aid supervised models in predicting our target variable. We found that data consistently clustered into seven categories, but we were surprised to find that these categories ultimately did not help in the task of predicting future revenue. Likewise, we engineered a variety of features based on research into financial analysis techniques, and we were able to extract salient aspects from these features -- but again, neither the manually engineered features nor the extracted features were very useful in the task of predicting future revenue. This experience highlights the twofold importance of problem formulation (having a clear goal in mind) and domain knowledge (understanding the domain well enough to select features that will be useful in achieving the goal). At a strategic level, our biggest challenge was a lack of expert knowledge in the domain of financial analysis.

On a more tactical level, other major challenges included data inconsistency (missing values, companies reporting values in billions USD vs. millions, etc.) and the skewing effects of very large, Mega-Cap companies. We dealt with missing values by imputation using a stratification technique that minimized the introduction of bias, but occasionally created nonsensical derived features (e.g., extreme KPI ratios). Instances where companies reported data in different units were harder to identify given our limited time and resources. For the most part, however, we overcame the skewing effects of the very large Mega-Cap companies using scikit-learn's Quantile Transformer.

With more time and resources, we could improve our domain knowledge (either through research or by bringing a domain expert onto the team), which would enable us to select a smaller set of company features that would be more appropriate to our prediction task. We could also develop a systematic method for identifying inconsistent data, or we could find a more reliable data source. If we can identify the *right data for the task* and ensure that it is of high quality, we will have a better chance of extracting useful features from that data.

Supervised Discussion

In the development of our project thesis, we discussed the preference to use time series based methods to model revenue, however the lack of free and accessible financial data made this task unlikely. We transitioned to a vector based data representation where we thought feature engineering would be able to capture some of the temporal relationships and help us predict quarterly revenue. What surprised us the most, through feature engineering, supervised modelling, and then evaluation was just how little feature engineering helped and how strong revenue was in feature weighting. This was strongly related to a lot of the challenges that we came across in our supervised learning portion. First and foremost, it was how to design features to capture financial ratios, temporal relationships, and rates of change. In this feature engineering, we used domain published mathematical transformations in order to provide our model with additional relationships.

Another challenge we experienced was determining the best evaluation metric for our model. In forecasting, RMSE and MAE prove to be invaluable as that tells us the absolute error from our prediction compared to the actual value. However, the scale of revenue was so large that errors from high revenue companies tended to overshadow those of smaller companies, especially in RMSE where these errors are further squared. This led us to use the coefficient of determination (R^2) as our primary evaluation metric as well as looking at a success rate of predicting revenue within a certain percentage of actual quarterly revenue. If given more time and resources, based on what we have learned from this modeling, a traditional time series analysis may be more appropriate for estimating company revenue. Supervised learning may be better used in our current manner to predict classifications, for example will this quarter be above or below analyst estimates for quarterly revenue? “Predicting the sign of an outcome is often much more important than predicting its size, and a reason for favoring classifiers over regression methods in finance” (López de Prado, 2020, p.21).

Ethical Considerations

Ethical issues that could arise from our unsupervised clustering model is if some clusters have relatively negative connotations like poor earnings or a lack of six month stock performance. If our model is publicly available it could limit potential investors stagnating corporate growth and development, perpetuating a negative cycle. One issue that could arise with our supervised model is that we are predicting a key metric used in valuing companies which inherently has risk. If we were to make this model publicly available, people could invest hard earned money or retirement funds based on the insights that they gain from our model. If there is co-variate drift or our model underperforms, this could have a significant monetary and emotional toll. A way that we can address this problem is through a clear and concise presentation of the data and caveats, along with risk disclosures related to investing that accompany the deployment of any model. This could include market risks that are out of the scope of our forecasting model, such as severe economic downturns.

Statement of Work

Yueyao Ren	Data Acquisition | Literature Review | Deep Learning Modeling | Writing
Peter King	Literature Review | Feature Engineering | Unsupervised Learning | Writing
Thomas MacPherson	Feature Engineering | Supervised Learning | Model Evaluation | Writing

APPENDIX A - References

Gu, S., Kelly, B., & Xiu, D. (2020). Empirical Asset Pricing via Machine Learning. *Review of Financial Studies 33*, 2223-2273.

Htun, H. H., Biehl, M., & Petkov, N. (2023). Survey of feature selection and extraction techniques for stock market prediction. *Financial Innovation, 9*(26). https://doi.org/10.1186/s40854-022-00441-7

Kuryłek, W. (2025). Is the inclusion of a broad set of explanatory variables relevant in EPS forecasting? Evidence from Poland. *Bank i Kredyt, 56*(3), 253-282. https://doi.org/10.5604/01.3001.0055.1459.

López de Prado, M. M. (2020). Machine Learning for Asset Managers. *Cambridge University Press*.

Singh, G., Thanaya, I. (2023). Predicting earnings per share using feature-engineered extreme gradient boosting models and constructing alpha trading strategies. *International Journal of Information Technology 15*, 3999-4012. https://doi.org/10.1007/s41870-023-01450-0

Stobierski, T. (2020). 13 Financial performance measures managers should monitor. *Harvard Business School Online*. https://online.hbs.edu/blog/post/financial-performance-measures

van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research, 9*(86), 2579-2605.

APPENDIX B - Data Schema

We extracted key financial and macroeconomic data using yfinance and FRED API's:

iShares Russell 3000 ETF download link:
https://www.ishares.com/us/products/239714/ishares-russell-3000-etf

FRED API
Gross Domestic Product (GDP) | FRED | St. Louis Fed
Unemployment Rate (UNRATE) | FRED | St. Louis Fed
Industrial Production: Total Index (IPB50001SQ) | FRED | St. Louis Fed
Federal Funds Effective Rate (FEDFUNDS) | FRED | St. Louis Fed
Consumer Price Index for All Urban Consumers: All Items in U.S. City Average (CPIAUCSL) | FRED | St. Louis Fed

Raw Features

Cross-sectional Data

Ticker - Company stock symbol
Name - Company name
Sector - Sector company belongs to
Asset Class - Used to filter out equities
Exchange - Exchange the security is traded on
Location - headquartered: 1 - US based equity 0 - Non-US based equity
Market Value - Total market value of company in US dollars (x1000)
Market Cap - Categorical representation of market value
Price - Stock price at time of data acquisition
Quantity - Outstanding shares at time of data acquisition
Weight (%) - Weight of stock in Russell 3000 index

Fundamental Data

CapitalExpenditure – Funds used by a company to acquire or upgrade physical assets such as property, industrial buildings, or equipment.
CashAndSTInvestments – Total cash, cash equivalents, and short-term investments available on the balance sheet.
CashFromOps – Net cash flow generated from a company's core operating activities.
CostOfRevenue – Direct costs attributable to the production of goods or services sold by the company.
CurrentAssets – Assets expected to be converted into cash within one fiscal year.
CurrentLiabilities – Obligations due within one fiscal year
EPS – Earnings per share; measures profitability on a per-share basis.
IncomeTaxExpense – Total income tax expense reported for the period.
InterestExpense – Total interest incurred on borrowed funds.
LongTermDebt – Total debt obligations with maturities longer than one year.
NetIncome – Company's total earnings after deducting all expenses, taxes, and costs.
OperatingIncome – Profit from core business operations, excluding interest and taxes.
OtherOperatingExpense – Miscellaneous operating expenses not categorized elsewhere.
Revenue – Total income generated from goods sold or services provided during the period.
TotalAssets – Sum of all assets owned by the company, representing total resources.
TotalDebt – Combined short-term and long-term debt obligations.
TotalEquity – Shareholders' residual interest in the company after liabilities are deducted from assets.
TotalLiabilities – Sum of all debts and financial obligations owed by the company.

Macroeconomic Data

GDP (in Billion) – Total value of goods and services produced within the U.S., indicating overall economic growth.
Unemployment Rate – Percentage of the labor force without employment, reflecting labor market conditions.
Interest Rate (Fed Funds) – Benchmark rate set by the Federal Reserve, influencing borrowing costs and monetary policy.
Industrial Production Index – Measures real output from manufacturing, mining, and utilities sectors, indicating industrial activity.
Inflation Rate (CPI) – Tracks changes in consumer prices, representing inflation and cost-of-living trends.

Quarterly Financial Data (YearQuarter - YYYYQQ)

CashAndSTInvestments_YYYYQQ - Cash and short term investments
CashFromOps_YYYYQQ - Cash from operations or operating cashflow
EPS_YYYYQQ - Earnings per share: net income / shares outstanding
LongTermDebt_YYYYQQ - Long-term debt
NetIncome_YYYYQQ - Revenue - expenses
OperatingIncome_YYYYQQ - Revenue - operating expenses
Revenue_YYYYQQ - Total revenue
ShortTermDebtOrCurrentLiab_YYYYQQ - Short term debt and current liabilities
TotalAssets_YYYYQQ - Total assets
TotalEquity_YYYYQQ - Total shareholders equity
TotalLiabilities_YYYYQQ - Total liabilities
CapitalExpenditures_YYYYQQ - Money spent on long-term assets (CAPEX)
CostOfRevenue_YYYYQQ - Similar to Cost of Goods Sold + delivery/services
IncomeTaxExpense_YYYYQQ - Taxable income * effective tax rate
InterestExpense_YYYYQQ - Cost of borrowing money (bonds, loans, line of credit)
OtherOperatingExpense_YYYYQQ - Any other operating expense

Quarterly Macroeconomic Data (YearQuarter - YYYYQQ)

GDP_YYYYQQ - Gross Domestic Product
GDPReal_YYYYQQ - Gross domestic product adjusted for inflation
Unemployment_YYYYQQ - Unemployment rate
IndustrialProd_YYYYQQ - Industrial production index
Inflation_YYYYQQ - Rate of inflation

Feature Engineering

Key Performance Indicators

Gross Profit Margin_YYYYQQ (GPM)
- A measure of profitability that does not account for the costs of production, sales, and other overheads
- *GPM = (Revenue - Cost of Revenue) / Revenue*

Net Profit Margin_YYYYQQ (NPM)
- A measure of profitability that accounts for the costs of production, sales, and other overheads
- *NPM = Net Profit / Revenue*

Working Capital_YYYYQQ (WC)
- A measure of operating liquidity; the ability to fund day-to-day operations

- *WC = Current Assets - Current Liabilities*

Current Ratio_YYYYQQ (CR)
- A liquidity ratio that measure the ability to pay short-term obligations (within one year)
- *CR = Current Assets / Current Liabilities*

Leverage_YYYYQQ (Lvg)
- A ratio that measures the use of debt to buy assets; high leverage is undesirable
- *Lvg = Total Assets / Total Equity*

Debt-to-Equity Ratio_YYYYQQ (DER)
- A ratio that measures how a company finances itself: whether through debt or equity primarily
- *DER = Total Debt / Total Equity*

Total Asset Turnover_YYYYQQ (TAT)
- An ratio that measures how efficiently a company uses its assets to generate revenue
- *TAT = Revenue / ((Beg Total Assets + End Total Assets) / 2)*

Return on Equity (ROE)_YYYYQQ (ROE)
- A profitability ratio that indicates how well a business uses equity to earn profit for investors
- *ROE = Net Profit / ((Beg Equity + End Equity) / 2)*

Return on Assets (ROA)_YYYYQQ (ROA)
- A measure of how well a company uses assets to generate net profits
- *ROA = Net Profit / ((Beg Total Assets + End Total Assets) / 2)*

Cash Flow_YYYYQQ (CF)
- How much cash a company has on hand to grow operations; if negative, a measure of how much financing a company requires to continue operations
- *CF = Cash From Operations*

QoQ Calculations

<feature>_QoQ_YYQQ_YYQQ - All quarterly features measuring QoQ changes as $QoQ = (\gamma_t - \gamma_{t-1}) \div \gamma_{t-1}$ with (t-1 and t) represented by YYQQ_YYQQ.

Rate Calculations

<feature>_QoQ_Rate - For all quarterly features, plotting a line of best fit between quarters and calculating the slope to determine a rate of change over the time period preceding our dependent variable.

APPENDIX C - Full Scale Images

Page 3

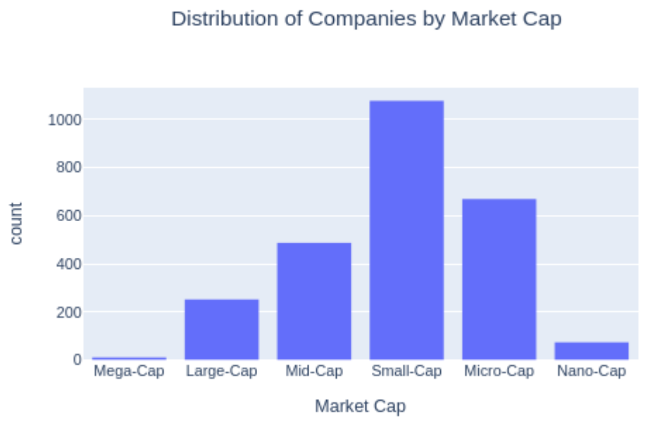


Figure 1: Market Cap distribution.

Page 4

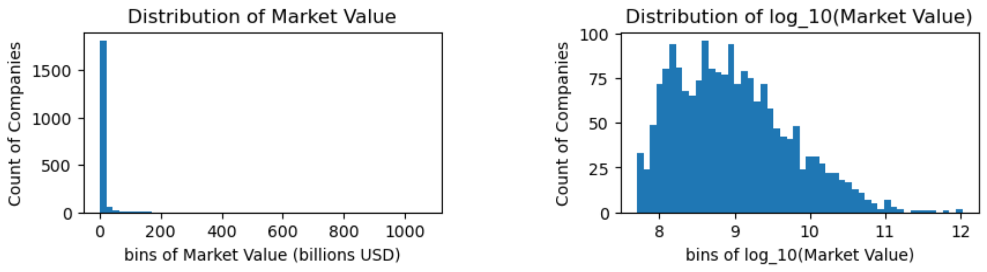
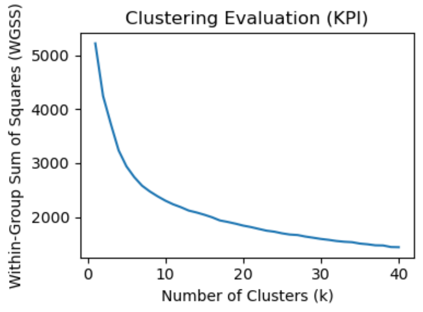


Figure 2: Distribution of Market Value compared to $\log_{10}$(Market Value).

Page 7



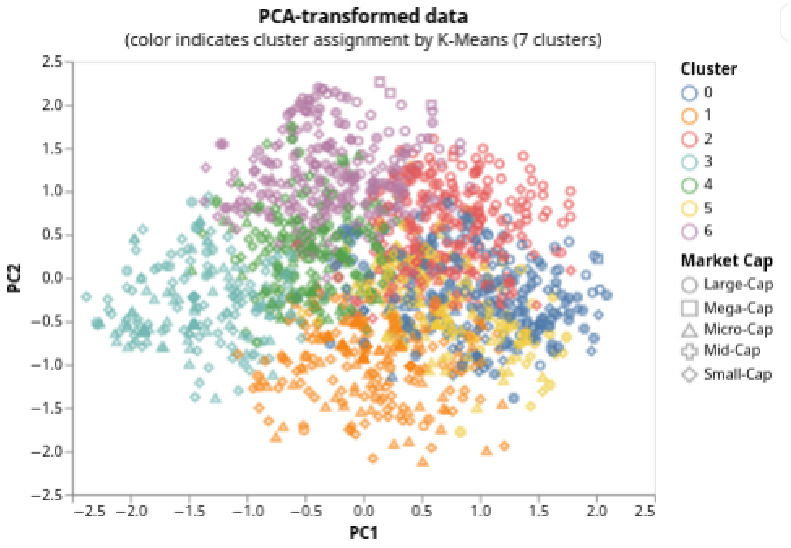


Figure 3: Elbow plot of WGSS for K-Means. **Figure 4**: Visualization of data using 2 principal components.

Page 7 (cont'd)

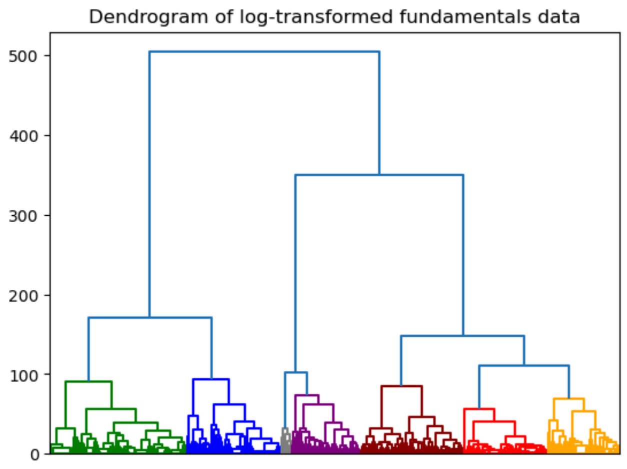


Figure 5a: Agglomerative clustering dendrogram.

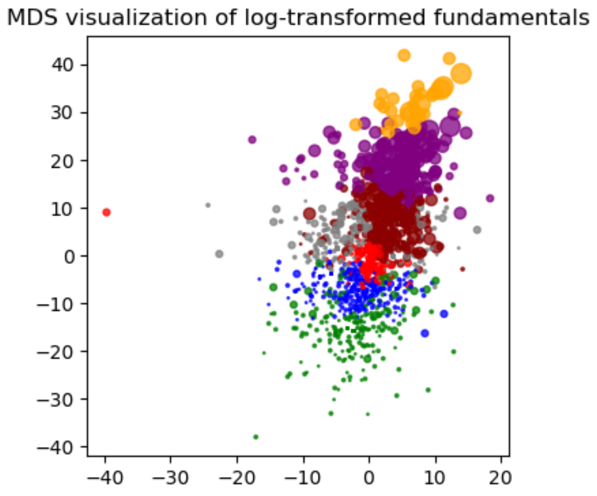


Figure 5b: MDS visualization of clustered data.

Page 8

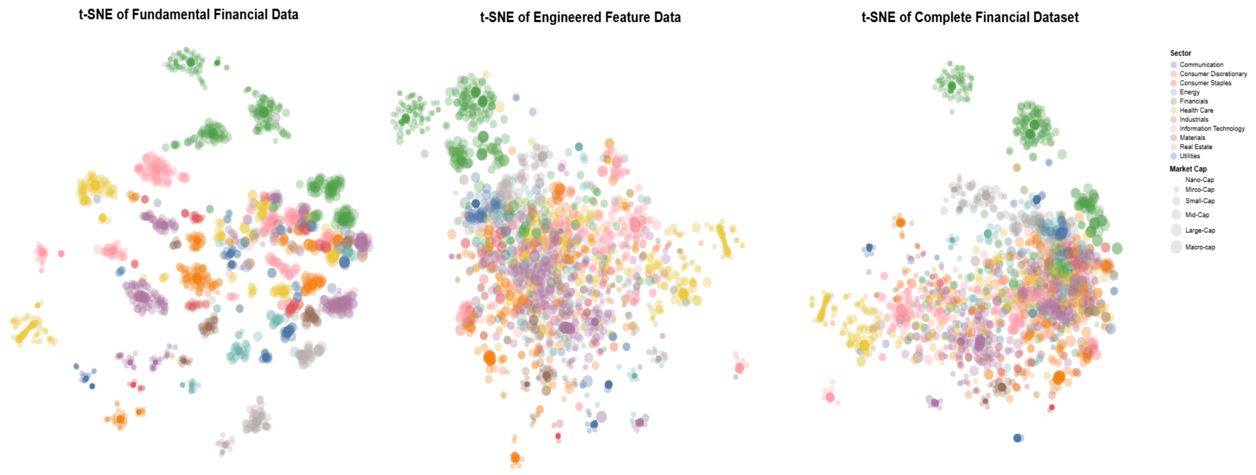


Figure 6 (a, b, c): t-SNE visualizations showing sector and market cap clustering based on a) raw financial data, b) engineered features, and c) complete dataset. Colors correspond to market sector and point size corresponds to market capitalization.

Page 8 (cont'd)

PCA using Standard Scaler

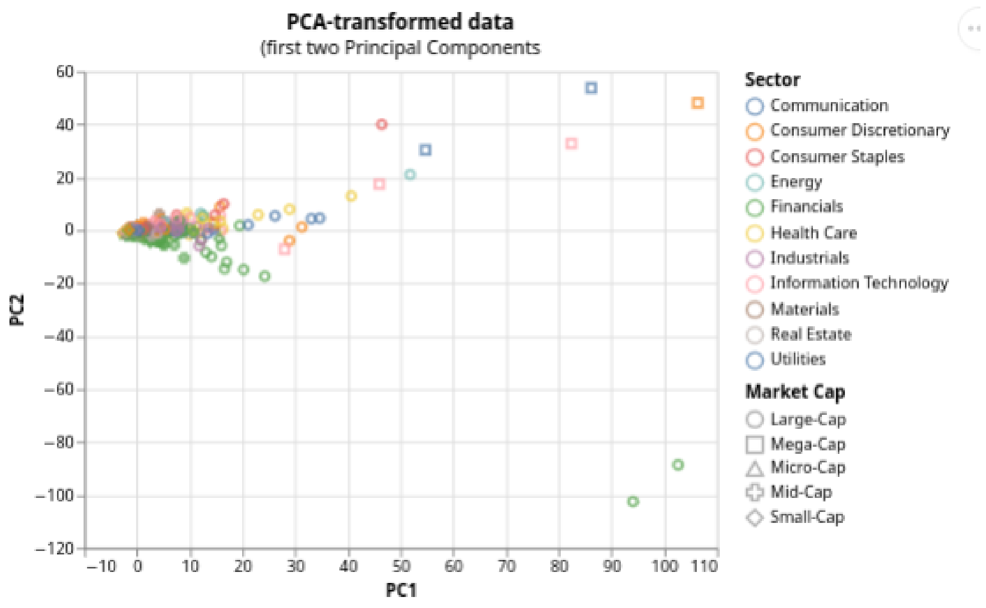


Figure 7a: PCA-transformed dataset (preprocessing using **Standard Scaler**)

PCA using Quantile Transformer

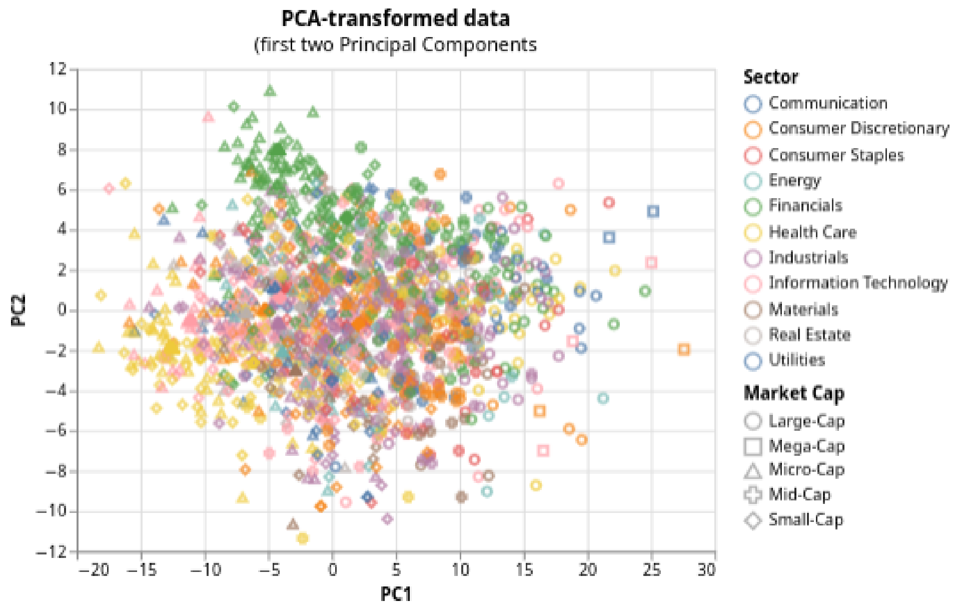


Figure 7b: PCA-transformed dataset (preprocessing using **Quantile Transformer**)

Page 11

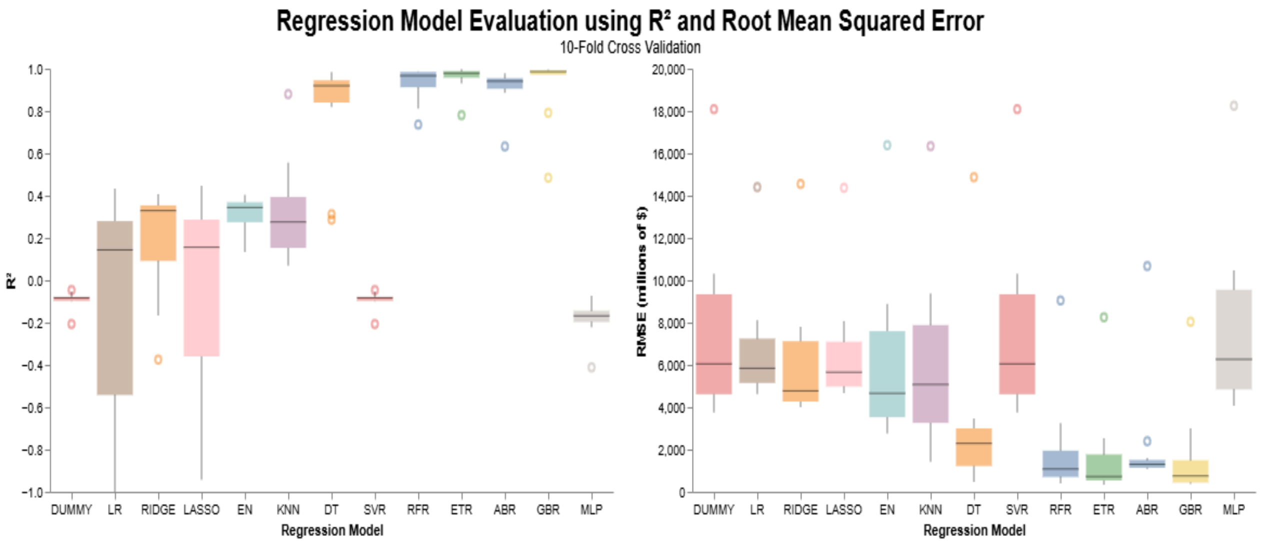


Figure 8: Supervised Learning family evaluation using R^2 and RMSE in ten-fold cross validation. Ensemble techniques showed advantages over simple linear based models leading us to explore the more complex supervised learning families.

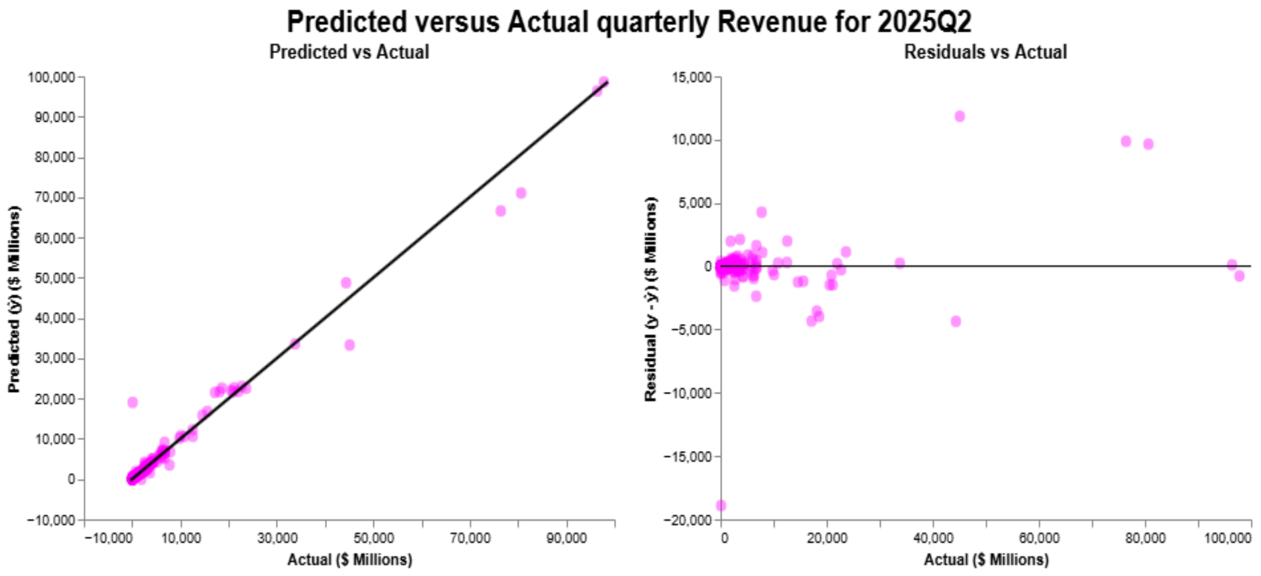


Figure 9: Correlation and residual plot of Gradient Boosted Model. a) Correlation between predicted and actual quarterly revenue of companies with diverse market capitals. b) Residual plot showing error between prediction and actual quarterly revenue with the most significant error coming from companies with high quarterly revenue and few financially comparable companies.

Page 12

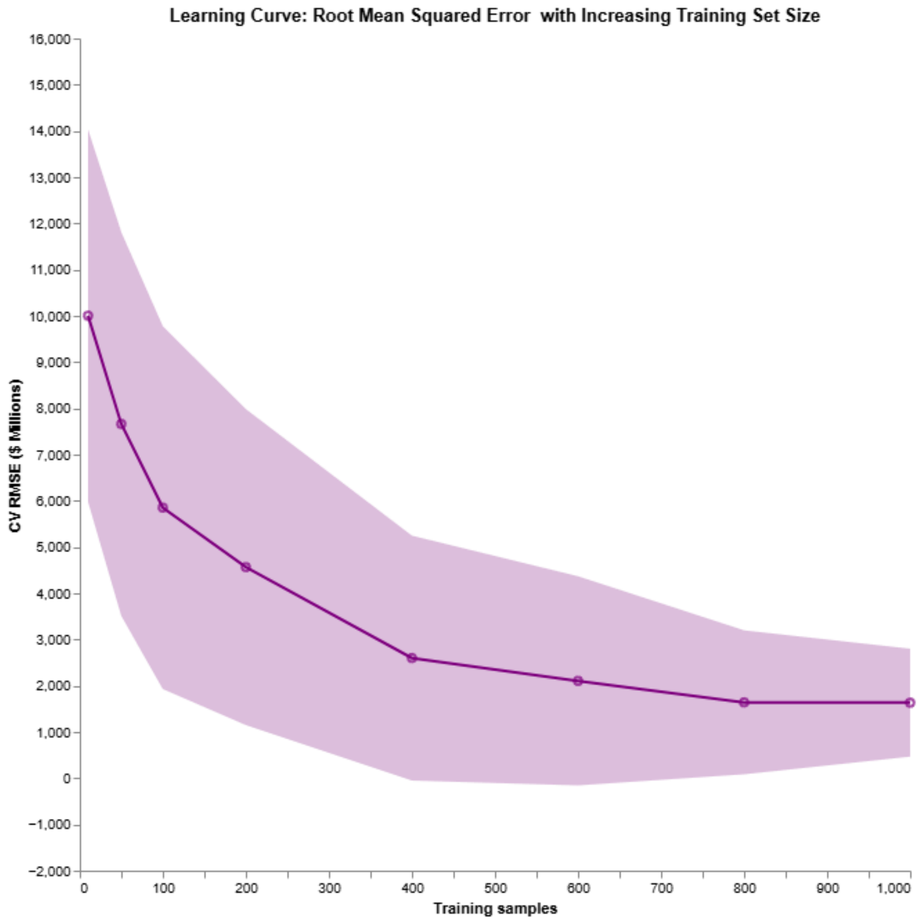


Figure 10: Learning curve analysis based on the size of our training dataset. While our error tended to plateau around 800 samples, we continued to see a reduction in variance up to our sample size indicating a further increase in the number of companies used to model may continue to improve performance.

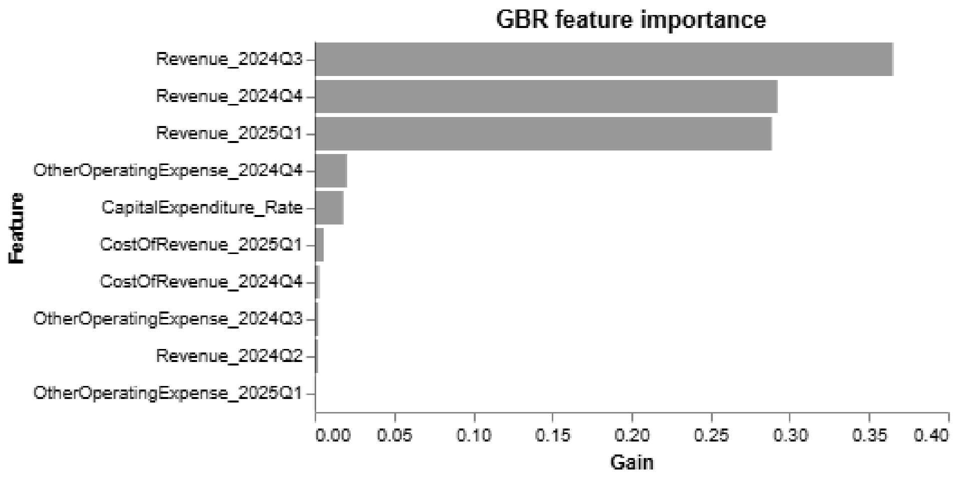


Figure 11: Feature importance as measured by Scikit-learn in the gradient boosted learning model. There was clear preference for revenue related features in our decision trees.

Page 12 (cont'd)

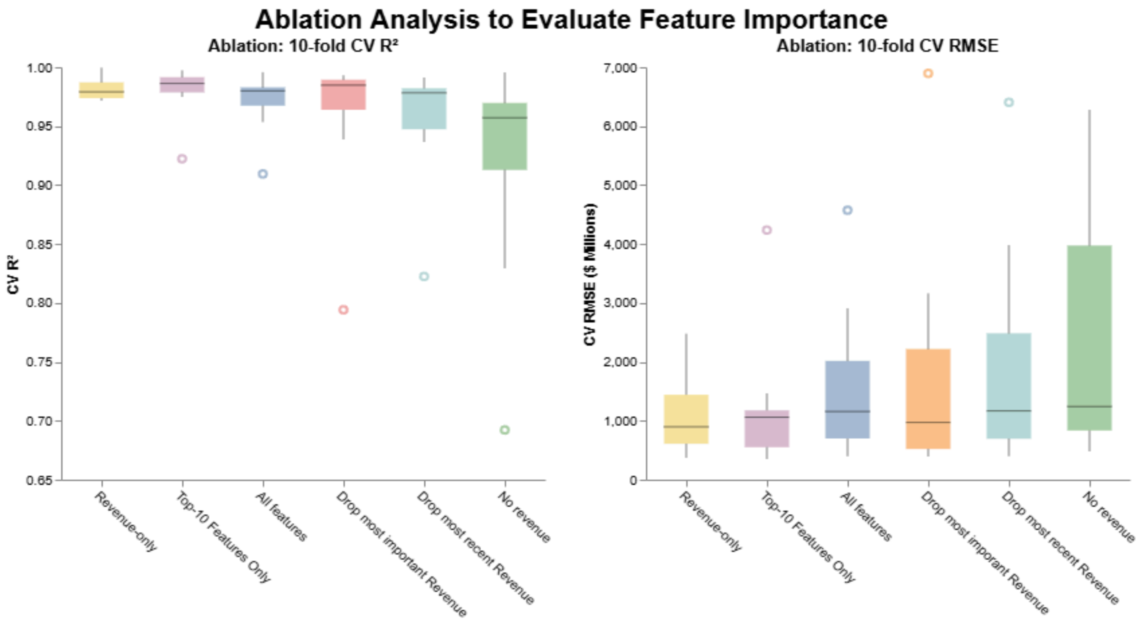


Figure 12: Ablation analysis comparing different feature space and the effect on model performance as measured by R^2 and RMSE. The most pronounced effect was seen on the variance in model performance as opposed to median values.

Page 13

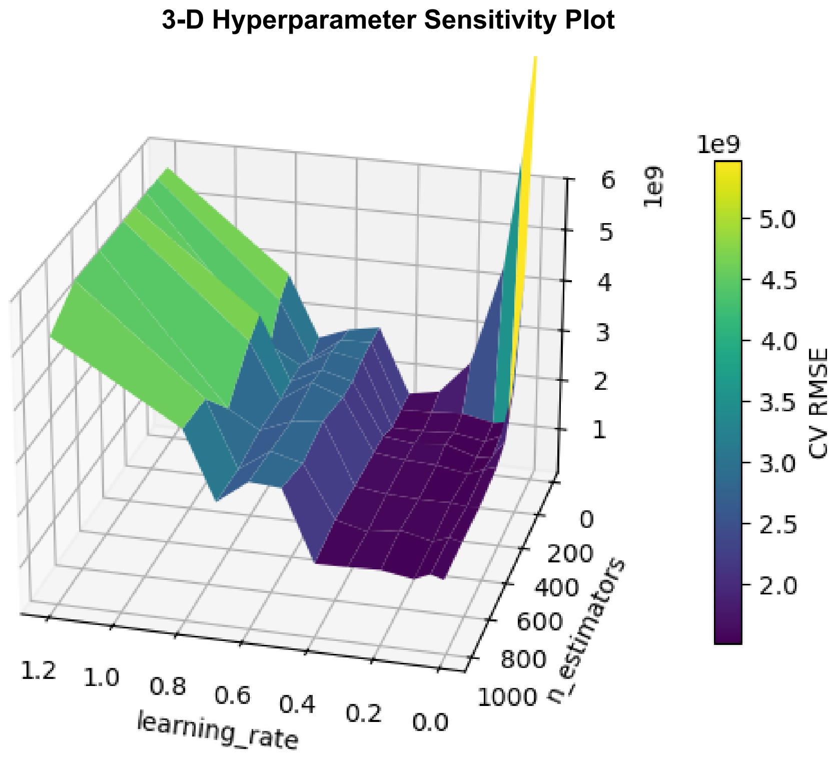


Figure 13: Sensitivity analysis across the learning rate and number of estimators hyperparameters. The lower the value of RMSE the better model performance.

Page 13 (cont'd)

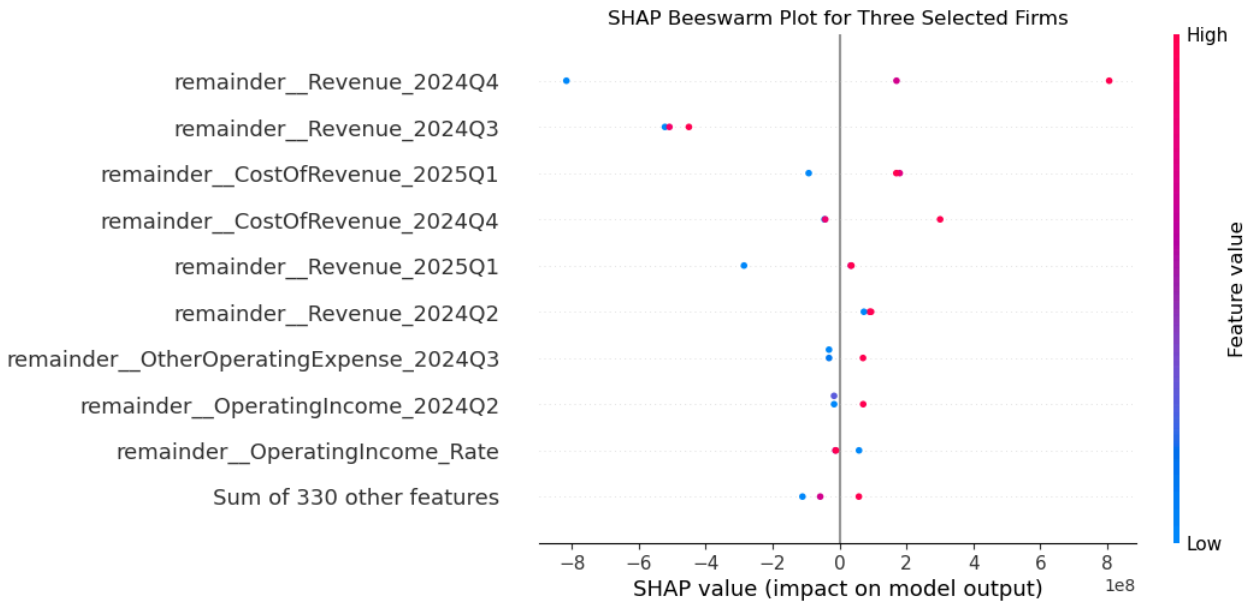


Figure 14: SHAP beeswarm plot to show feature importance for firms in which our model had poor forecasting performance. Large positive (negative) values caused the predicted value to be higher (lower) than the average predicted value. Colorbar indicates the $ value of the feature in the particular instance analyzed.