Springboard—DSC Capstone Project 3
Market Capitalization Regression
Garrick Skalski, November 2021

1 INTRODUCTION
Estimating the value of a business is fundamental to the functioning of capital markets generally, and public stock exchanges in particular. The goal of this project is to understand and predict the valuation of publicly-traded companies as measured by market capitalization (the target or response variable). Financial statement data will be used as predictor variables (features) to assess whether regression models of these fundamental variables can predict future market capitalization. The primary business use case is to predict next period (e.g., next fiscal quarter) market capitalization based on information known in the current period (e.g., current fiscal quarter).

The key findings are that (i) among a set of models assessed with two iterations of feature engineering, an xgboost decision tree model is the best predictor of next-quarter market capitalization, (ii) a variety of features appear to be important and are likely acting both linearly and nonlinearly in their associations with each other and the target variable, market capitalization, and (iii) the best models have an absolute percent error rate of 40%, thus a better understanding of how features are more or less effective in predicting market capitalization should improve future models and lead to higher accuracy.

Full details of the analysis with Python code can be found in Jupyter notebooks on GitHub (https://github.com/gskalski267/springboard/blob/main/Capstone_Project_3/notebooks/).

2 APPROACH
The machine learning approaches used here draw on the statistical learning presentations in Hastie et al 2009 and James et al 2021. The primary machine learning Python modules that I use are statsmodels (0.12.2), sklearn (1.0), xgboost (1.0.0), tensorflow (2.4.1), and keras (2.4.3).

2.1 Data Acquisition
All data are sourced from Alpha Vantage via their API (https://www.alphavantage.co/documentation/). I wrote Python wrapper code and functions to call the Alpha Vantage API and handle the resulting responses and any errors. Multiple API call types, with each handling about 30,000 records, were required to capture the raw source data. Full details are in a dedicated "Data Capture" notebook on GitHub.

2.2 Variable Definitions
I standardized the raw source data with respect to data types, duplicates, null values, and nonlogical (negative, zero) values. I then merged all required records into a single base data set with target and feature data for subsequent analysis and model fitting,

calculating required variables as needed, such as year-over-year metrics or ratio metrics (e.g., profit margins), for example.

The variables used in this analysis are defined as follows.

response variable (target)
market capitalization = [number of shares outstanding] x [share price]

predictor variables (features)
The features are metrics from financial statements such as revenue, revenue growth year-over-year, etc.

Features are measured with respect to a quarterly fiscal reporting date, 'fisc_date_t', and with respect to dates relative to that date. Financial statement metrics are measured on dates such as current quarter ('fisc_date_t'), next quarter ('fisc_date_tp1'), prior quarter ('fisc_date_tm1'), prior year ('fisc_date_tm4'), and prior quarter in the prior year ('fisc_date_tm5'). Stock prices are measured on measurement dates with respect to fiscal reporting dates, where the current measurement date ('meas_date_t') is the most recent trading date prior to the current fiscal reporting date ('fisc_date_t') and the next period measurement date ('meas_date_tp1') is similarly measured with respect to the next fiscal reporting date ('fisc_date_tp1').

The target, market capitalization, 'mktcap_sec_tp1', is measured on date 'meas_date_tp1', the most recent trading date just prior to the next fiscal reporting date, 'fisc_date_tp1'. The machine learning problem is to predict market capitalization ('mktcap_sec_tp1') just prior to the next fiscal reporting date ('fisc_date_tp1') using feature measurements from the current fiscal reporting date ('fisc_date_t').

The core features used in the analysis are:

clsadj_etf_t = adjusted close price of ETF ticker VTI on the most recent trading date prior to the current fiscal reporting date

rev = revenue for the fiscal quarter ending on the current fiscal reporting date
rev_yoy = year-over-year percent change in quarterly revenue for the current fiscal reporting date versus prior year
rev_yoy_qdq = current quarter rev_yoy minus prior quarter rev_yoy

gp_mgn_flg = 1 if gp_mgn = 1, 0 otherwise
gp = gross profit for the fiscal quarter ending on the current fiscal reporting date
gp_mgn = gross profit margin, gross profit divided by revenue, for the fiscal quarter ending on the current fiscal reporting date
gp_mgn_ydy = current quarter gross profit margin minus prior year gross profit margin

ni = net income for the fiscal quarter ending on the current fiscal reporting date
ni_ydy = current quarter net income minus prior year net income

ni_mgn = net income margin, net income divided by revenue, for the fiscal quarter ending on the current fiscal reporting date
ni_mgn_ydy = current quarter net income margin minus prior year net income margin

sh_iss_flg = 1 if total share issue amount > 0, 0 otherwise
sh_iss = total share issue amount in USD for the fiscal quarter ending on the current fiscal reporting date
sh_rprch_flg = 1 if total share repurchase amount > 0, 0 otherwise
sh_rprch = total share repurchase amount in USD for the fiscal quarter ending on the current fiscal reporting date
div_flg = 1 if total dividend amount > 0, 0 otherwise
div = total dividend amount in USD paid for the fiscal quarter ending on the current fiscal reporting date
div_ydy = current quarter total dividend amount minus prior year total dividend amount

cf_op = operating cash flow for the fiscal quarter ending on the current fiscal reporting date
cf_op_ydy = current quarter operating cash flow minus prior year operating cash flow
cf_op_mgn = operating cash flow margin, operating cash flow divided by revenue, for the fiscal quarter ending on the current fiscal reporting date
cf_op_mgn_ydy = current quarter operating cash flow margin minus prior year operating cash flow margin

The final data set used for modeling has

- 2,122 distinct stock tickers
- 9,723 observations
- 1,692 stock tickers with 5 observations (80% of all stock tickers)
- stock tickers with market capitalization in Sep 2021 ranging from $1.006 billion to $2.358 trillion
- fiscal reporting dates ranging from 2020-01-31 to 2021-06-04

2.3 Data Exploration

I initially analyzed the data in an exploratory data analysis in three main ways: (i) descriptive metrics and histograms to define the in-scope sample, (ii) regressions on a single feature variable with associated scatterplots, and (iii) regressions on pairs of feature variables with associated scatterplots. Full details are available on GitHub.

2.3.1 Records In Scope

I flagged records as ‘in scope’ for further analysis and modeling with respect to reporting currency, null/NaN values, and extreme or inconsistent values. Records not flagged as in scope were excluded from further analysis and modeling. Specifically,

- records with reporting currency USD are in scope
- records with NaN values for the core metrics are not in scope
- stock tickers with a median quarterly revenue of $5 million are in scope, all others are not in scope

- stock tickers with market capitalization values that are inconsistent across different Alpha Vantage data sets are not in scope
- very extreme values for revenue, gross profit, net income, and operating cash flow are not in scope based on examining percentiles for these metrics
- for gross profit, after comparing different calculations, I used the value provided by Alpha Vantage instead of other versions that I calculated as the former appeared more logically consistent when compared to revenue

This is a large and heterogenous data set that I am analyzing for the first time, hence I needed to make a few judgments in terms of identifying data as in or out of scope for further analysis and modeling. Overall, I tried to err on the side on including data instead of excluding data while at the same time excluding records that are obviously nonlogical. I also used filters on market capitalization and revenue as some of my initial filters with the expectation that these metrics are foundational, with the latter being a core GAAP (generally accepted accounting principle) financial metric, and hence presumably more reliable than some other measures which are calculated metrics or other metrics that are perhaps more applicable to some types of companies relative to others.

As an example, the companies in this analysis are $1 billion and above in market capitalization, so a firm with $20 million in annual revenue (a GAAP metric) at a valuation of $1 billion or more would typically be considered highly valued and quite speculative from an investing perspective. Hence, firms that are even more speculative with under $20 million in annual revenue (filtered via median quarterly revenue of $5 million) are considered out of scope for this analysis. For excluding extreme values, I excluded only the most extreme values, and I spot-checked some values against SEC filings as a check for reasonableness. Even with some records excluded as described here, the resulting in-scope data set is a large and heterogenous data set encompassing firms in a very large subset of the U.S. public equity markets.

Figure 1 shows histograms of the target variable, market capitalization (‘mktcap_sec_tp1’), and a core feature, revenue (‘rev). These metrics have a large range of values, ranging over several orders of magnitude with units in USD amounts.

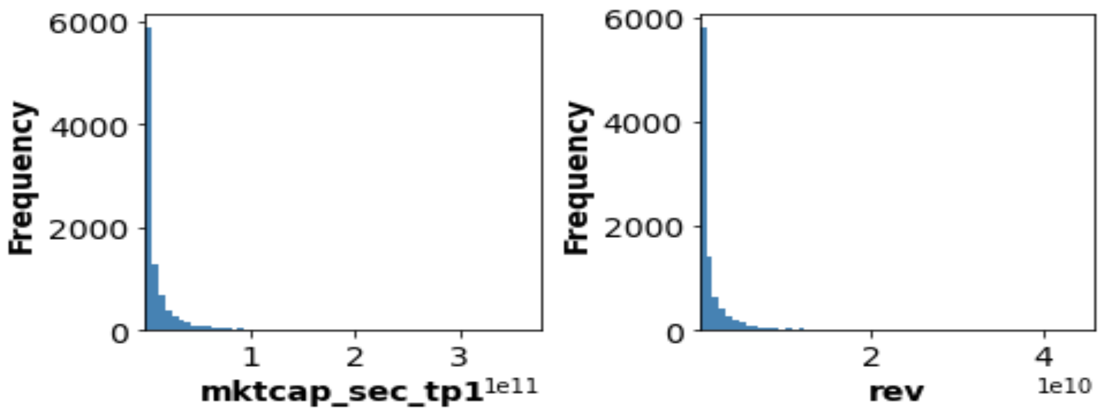

Figure 1 Histograms of the target variable, market capitalization (‘mktcap_sec_tp1’), and a core feature, revenue (‘rev’). Units are in USD amounts.

Figure 2 shows histograms of two features that are measured in units of percentages, net income margin (‘ni_mgn’) and net income margin year-over-year incremental change (‘ni_mgn_ydy’), and hence on much smaller scales than the examples in Figure 1, which are in units of USD amount.

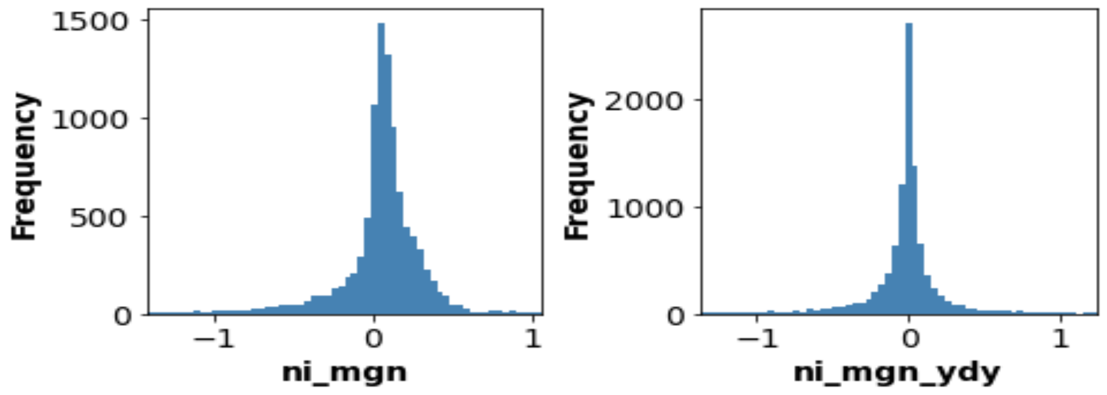

Figure 2 Histograms of two features, net income margin (‘ni_mgn’), and net income margin year-over-year incremental change (‘ni_mgn_ydy’). These features are measured as percentages.

2.3.2 Regression and Scatterplot Analyses

I implemented a set of regression analyses of the target variable, market capitalization, on various feature variables as predictors. Variables were transformed using the RobustScaler from module sklearn. This scaler transforms the data with respect to quantiles, centering with respect to the median and scaling with respect to a quantile range (the 10% to 90% quantile range was used here to scale the median-centered data). The transformations were fit on the training data and then applied to the training

and test data, so that transformation of the test data is based on a fit of the transformation to the training data.
First, I regressed market capitalization on each individual feature (univariate regressions) and sorted by the features explaining the largest amount of the variation in the target variable (in terms of percent of total sums of squares). Six individual feature variables explain from 47% to 18% of the variation in market capitalization: operating cash flow, share repurchase amount, gross profit, revenue, net income, and dividend amount. Figure 3 shows the top four features in terms of variation explained. The relationships between features and target are what would be expected based on the underlying business data, which is that market capitalization is positively related to financial metrics such as operating cash flow, share repurchase amount, gross profit, revenue, net income, and dividend amount.

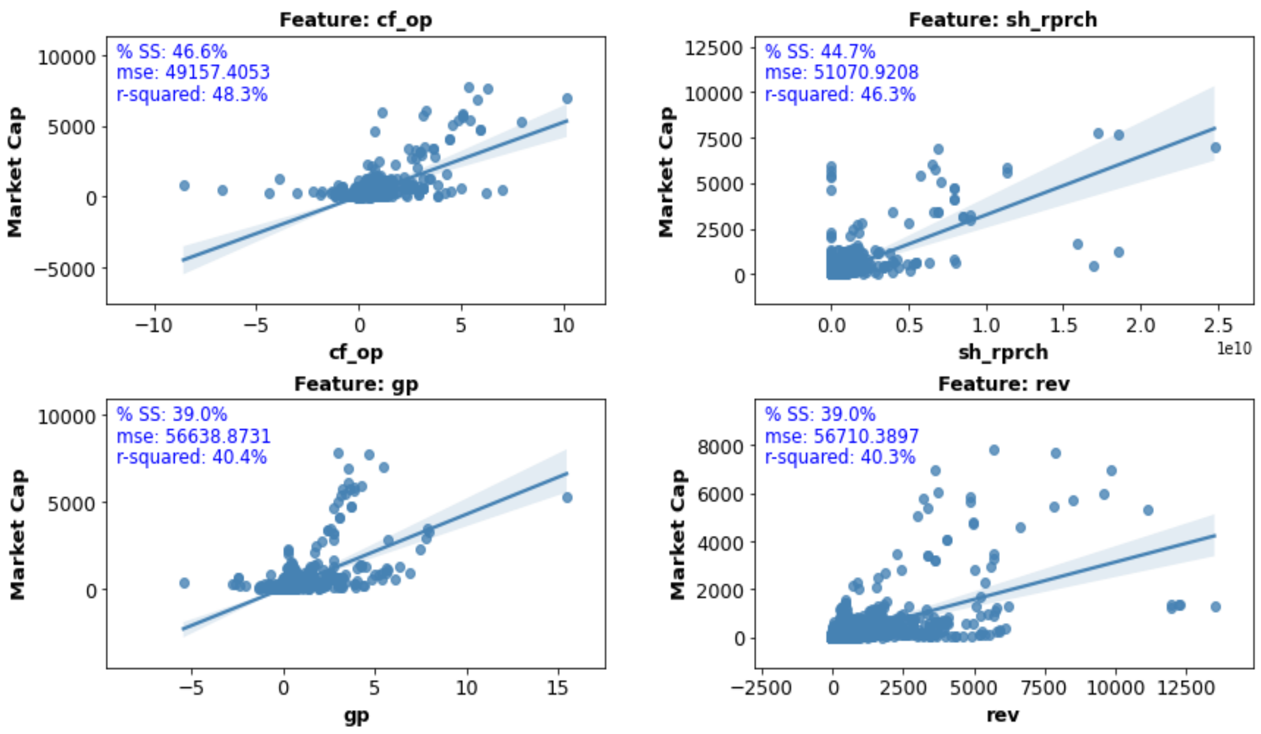

Figure 3 The top four features explaining variation in market capitalization in univariate regressions: operating cash flow (‘cf_op’), share repurchase amount (‘sh_rprch’), gross profit (‘gp’), and revenue (‘rev’).

I then extended the regression analysis by regressing market capitalization on every pairwise combination of features (bivariate regressions). These regression models include all 1st order and 2nd order combinations of the pairwise predictor variables in an attempt to identify important 1st order linear terms as well as nonlinear terms that act via 2nd order interactions among feature variables.

In these bivariate regression models, the most impactful 1st order linear terms are similar to those indicated in the single-feature/univariate regressions, so I do not repeat the display of these scatterplots. Regarding 2nd order features, the top 21 2nd order features explain between 16% and 4% of the variation in the target. Hence, there are many feature pairs that explain similar amounts of variation in the target. A few examples of the most impactful 2nd order terms are shown in Figure 4 in partial residual plots. The partial residual plots show the residual after subtracting all model terms except the 2nd order regressor of interest and this partial residual is plotted against the 2nd order term as a way to visually isolate and display the impact of the 2nd order term of interest. Again, the relationships between features and target are what would be expected based on the underlying business data. As one example, the quadratic term in net income explains 16% of the variation in the target, suggesting that market capitalization increases in an accelerating way as a function of net income in this specific bivariate regression (Figure 4). As a second example, the market capitalization is an increasing function of the product of revenue and operating cash flow margin, suggesting an interaction between these two variables with the positive effect of revenue increasing for companies with higher operating cash flow margins (Figure 4).

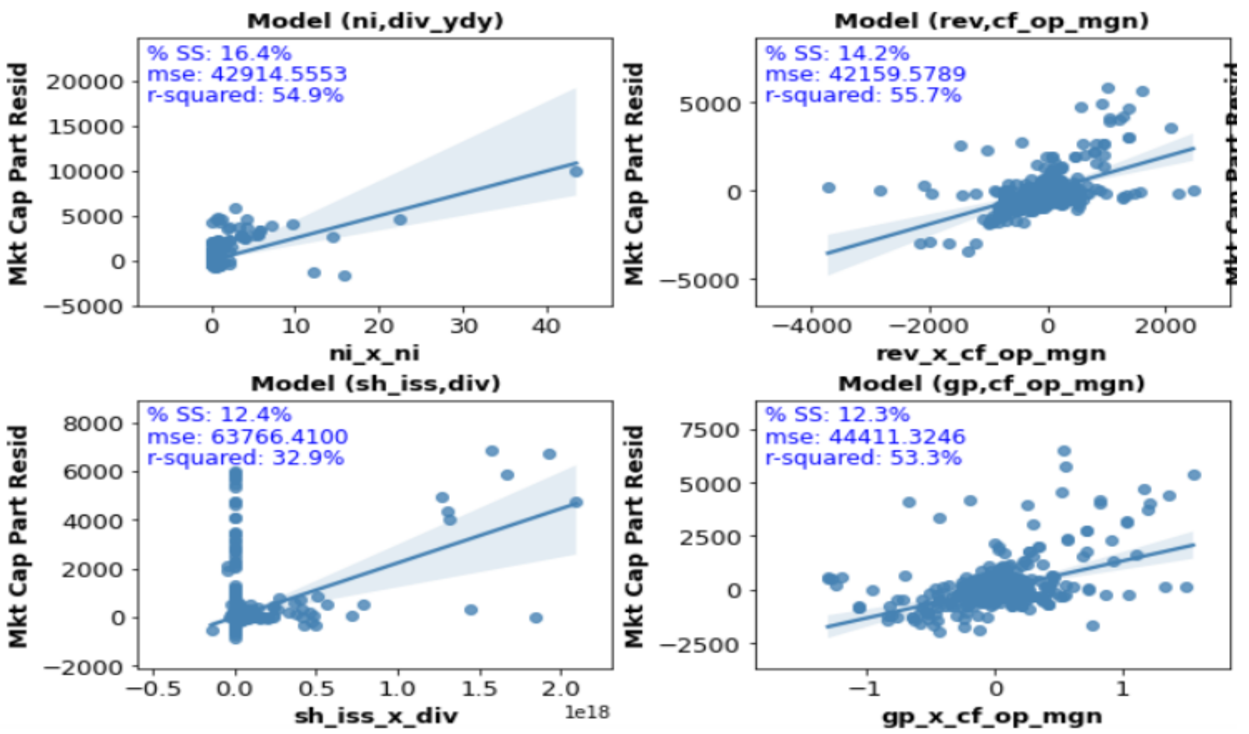

Figure 4 Partial residual plots from pairwise (bivariate) regressions showing the partial residual with respect to all terms except the 2nd order term of interest. The title denotes the pair of features used as regressors and the x-axis denotes the 2nd order regressor.

The exploratory data analysis indicates about five to eight features may individually act via 1st order terms to predict the target. An additional twenty or so features may act nonlinearly via 2nd order terms to predict the target, with many of these nonlinear effects explaining 5% or more of the variation in market capitalization in the bivariate regressions.

2.4 Regression Modeling

In my first iteration of modeling, I fit three different regression models to a training data set and evaluated the predictions on a separate test data set. The models assessed are a linear regression model (LR), an xgboost decision tree model (XGB) and a support vector regression model (SVR).

2.4.1 Training and Test Data, Cross Validation

Most of the stock tickers in the data set have five observations, with each observation corresponding to a fiscal reporting period. I defined the training data set as the first four fiscal periods and the test data set as the final, fifth fiscal period for each stock ticker. Hence, the latest approximately 20% of the data is used for testing and the prior approximately 80% of the data is used for training.

To reiterate, the machine learning problem is to predict market capitalization ('mktcap_sec_tp1') just prior to the next fiscal reporting date ('fisc_date_tp1') using feature measurements from the current fiscal reporting date ('fisc_date_t'). In terms of training and test data sets, the models are trained on earlier data in the historical sequence (the first four fiscal reporting periods) and tested on later, future data in the historical sequence (the fifth and final fiscal reporting period).

As in the exploratory data analysis, I transformed the data using the RobustScaler by centering on the median and scaling by the the quantile range 10% to 90%. However, I had difficultly identifying best-fit models for the support vector regression, which tends to require very scaled and normalized data. Hence, for the support vector regression, I transformed the data using the nonlinear QuantileTransformer from Python module sklearn which fits the data, on a quantile basis, to a normal distribution.

I fit all models to the training data using cross validation to select the best model for subsequent testing with a grid search over hyperparameters, as applicable. The linear regression model was fit by selecting over the k best linear predictors using the SelectKBest method from sklearn. For the xgboost decision tree model, I used the algorithm gbtree and I searched over the hyperparameters max_depth, learning_rate, and reg_alpha. For the support vector regression model, I used the nonlinear radial basis kernel and I searched over the hyperparameters C, gamma, and epsilon. I used mean absolute error (MAE) as the scoring metric during cross validation.

2.4.2 Model Fitting Results: Test Metrics and Feature Importance

Figure 5 shows scatterplots of observed versus predicted values for the test data (transformed back to the original scale) for the three models and Table 1 shows three metrics of fit on the test data, r-squared, mean absolute percent error (mean-APE), and median absolute percent error (median-APE). In many cases, within each of the three types of models, the top-ranking models across different hyperparameters fit similarly well. Full details are on GitHub.

All models (Table 1) have a relatively high r-squared (about 64% to 83%), but also quite high mean-APE and median-APE. Mean-APE is above 100% and as high as 235% and median-APE ranges from about 41% to 62%. The much larger values for mean-APE versus median-APE suggest some skewness in the values for the target, which is consistent with the empirical distribution for market capitalization which has some extremely large values (Figure 1). The xgboost decision tree model has the highest r-squared whereas the support vector regression model has the lowest median-APE.

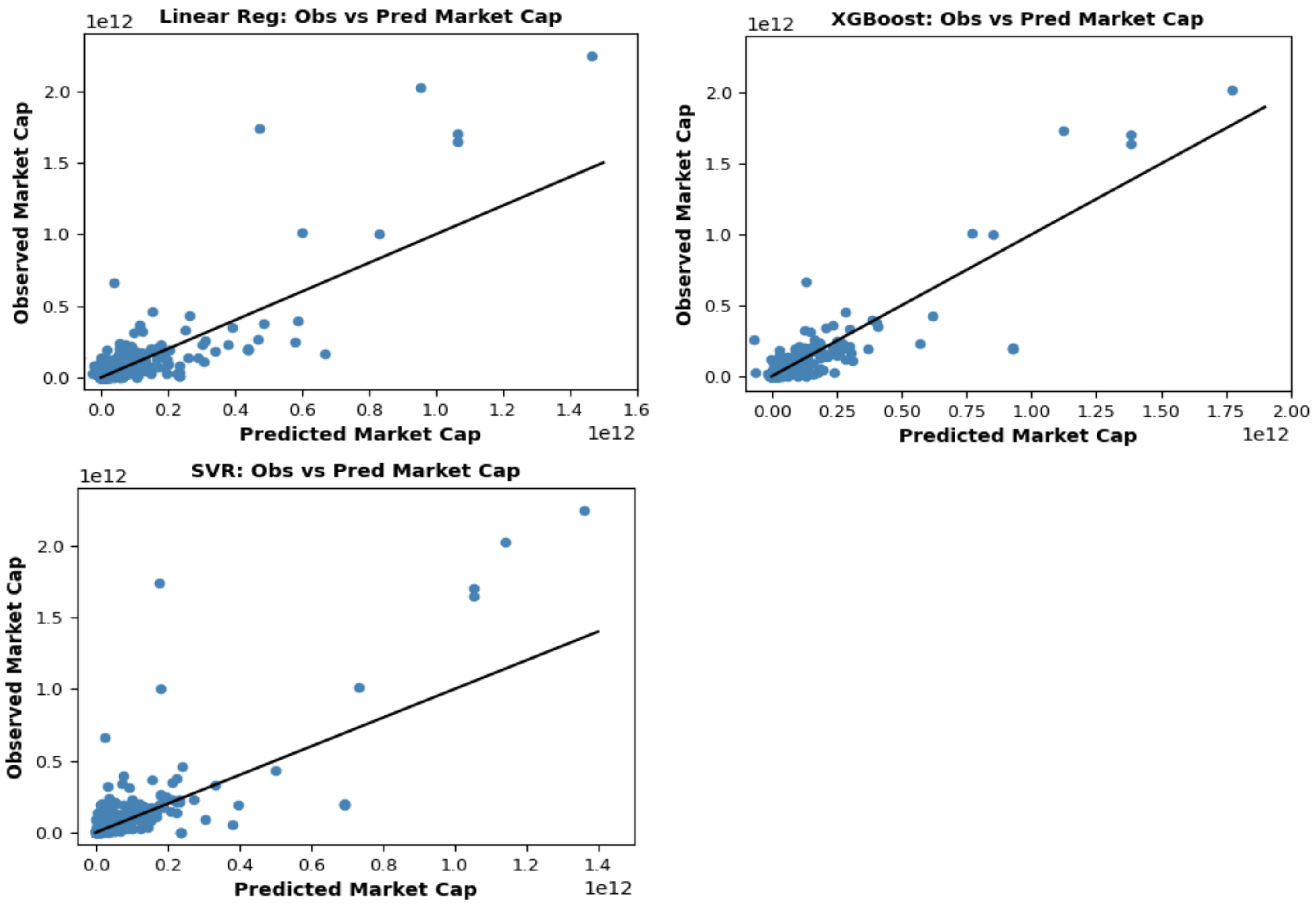

Figure 5 Scatterplots of observed versus predicted test data for the linear regression (LR), xgboost decision tree (XGB), and support vector regression (SVR) models. Data values have been transformed back to the original scale.

Table 1 Accuracy metrics for each of the three models on the original, untransformed test data: linear regression (LR), xgboost decision tree (XGB), and support vector regression (SVR).

Model	r-squared	mean-APE	median-APE
LR	70.0%	235%	59.7%
XGB	83.0%	218%	61.9%
SVR	64.2%	129%	40.8%

To help understand how different features are acting in these models to predict market capitalization, I calculated basic feature importance metrics for each model. For linear regression feature importance, I used the 1st order regression coefficients multiplied by one standard deviation of the corresponding feature and then scaled these values by their total absolute value to obtain a relative measure. For feature importance in the xgboost decision tree, I used the feature importance property from the fitted model attributes using parameter total_gain, which measures the cumulative improvement in fit attributable to a feature over all nodes in the tree. For the support vector regression model, I manually calculated feature importance as the sensitivity of the change in the target given a 1% change in each feature where features were perturbed by 1% over the set of support vectors. The sensitivities in the target were then averaged for each feature over the set of support vectors. Then, as in the linear regression model, absolute values of these sensitivities were scaled on a relative basis by dividing by their total absolute value.

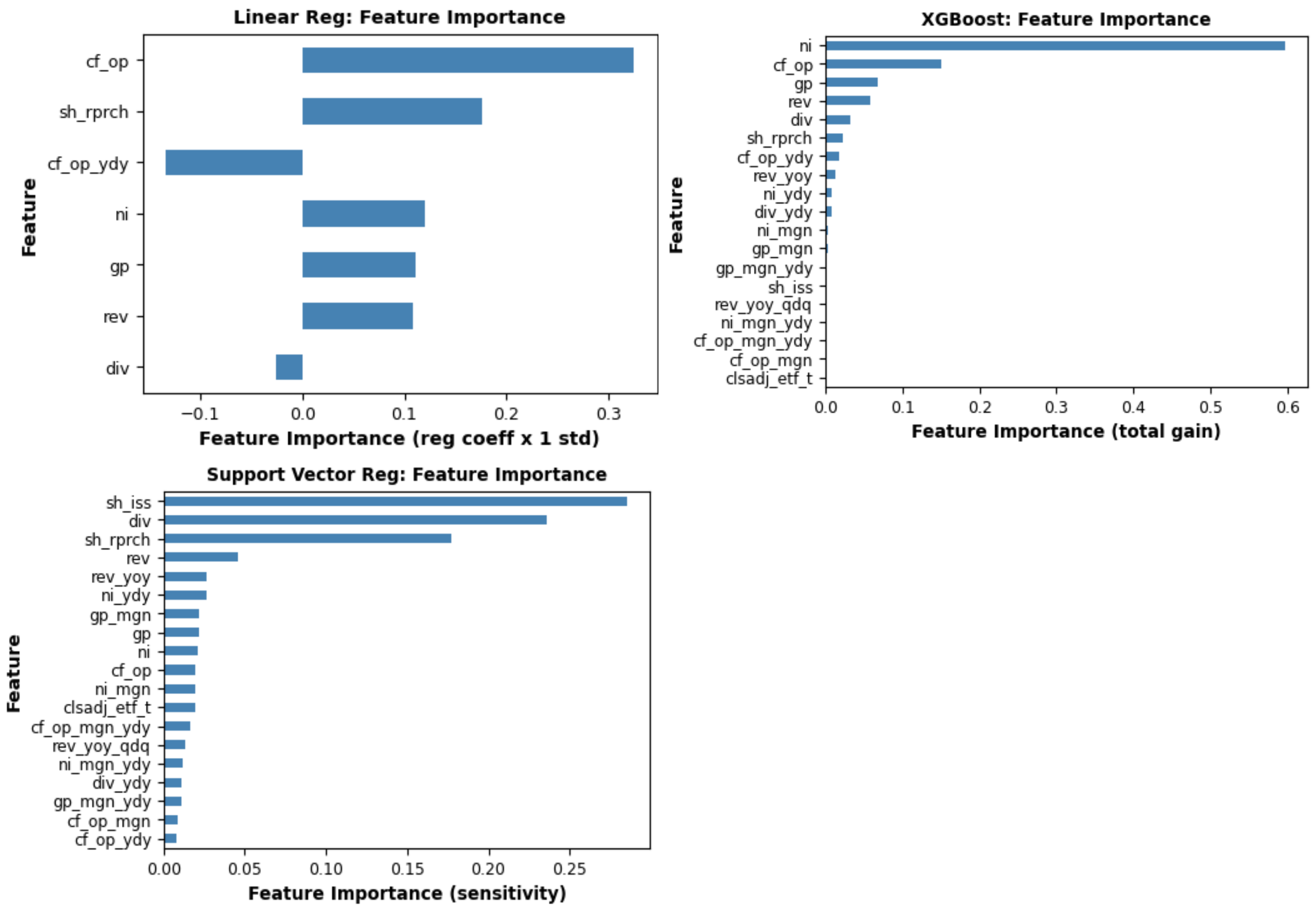

Figure 6 Relative feature importance on the fit to the training data for the linear regression, xgboost decision tree, and support vector regression models.

There are several interesting similarities and differences among the models with respect to feature importance (Figure 6) and I highlight a few observations here. First, the top seven features in the linear regression model are the same as the top seven features in the xgboost decision tree, except that the exact rank order differs between the two models. In the xgboost decision tree model, the top feature, net income, dominates and the other top six or seven features account for the remaining feature importance. In the support vector regression model, the top three features dominate but there is also a long tail of weakly-important features, which contrasts to the xgboost decision tree model in which the top handful of features dominates.

Finally, the linear regression and support vector regression models suggest the importance of nonlinear effects in the features. In the linear regression model, most coefficients have a positive sign, as expected, but two features, operating cash flow year-versus-year difference and dividend amount, have negative signs suggesting possible nonlinear effects that the linear model cannot accommodate. In the support vector regression model, any difference between the mean sensitivity for a feature, which can be positive or negative, and the mean of the absolute value of the sensitivity for a feature, which is strictly positive, indicates a nonlinearity. Several features show a difference in this regard (e.g., revenue, revenue year-over-year change, gross profit margin), suggesting a nonlinearity, whereas other features show no such difference (share issue amount, dividend amount, share repurchase amount), suggesting a pure linear effect.

2.5 Additional Feature Engineering

The prior exploratory data analyses and modeling results suggest the importance of scaling the data as appropriately as possible, particularly for the support vector regression and for variables that are on very different scales such as in units of USD amounts versus percentages. In an effort to extend the modeling, I performed additional feature engineering via a custom transformation with the aim to reduce the long tails in some distributions, to more normalize the data, and to scale the data to zero mean and unit standard deviation.

Four of the features (gross profit, share issue amount, share repurchase amount, and dividend amount) have distributions with many observations taking values zero or one. Dividend amount, for example, has many zero values for companies that do not pay a dividend and then a range of positive values for companies that pay a dividend. Accordingly, I defined four new indicator variables as features that take on values zero or one:

gp_mgn_flg = 1 if gross profit margin = 1, 0 otherwise
sh_iss_flg = 1 if total share issue amount > 0, 0 otherwise
sh_rprch_flg = 1 if total share repurchase amount > 0, 0 otherwise
div_flg = 1 if total dividend amount > 0, 0 otherwise.

As part of this additional set of feature engineering, I performed the following:

1. added four indicator variables as features, as defined just above,
2. transformed features with large values (USD amounts) using a square root transform on the absolute value (the sign of variables is preserved) with the aim of reducing the long tails in these distributions and making the data more normally-distributed (variables that are percentages or indicator flags are not square-root transformed, also the variable 'clsadj_etf_t' is not transformed),
3. regressed all USD amount variables (except revenue), including the target variable, on revenue, calculated the residuals for each regression and used these residuals (after scaling in the next step, below) as new target and feature variables to further scale the data, and
4. standardized all non-indicator variables by centering on the mean and dividing by the standard deviation.

The idea of the above transformations is to reduce the long tails in some of the distributions, to more normalize the data, and to scale the data to zero mean and unit standard deviation. I created custom transformation functions to implement the above steps and calculations were done in a way to preserve the sign of the data upon transformation and inverse transformation. Full details are on GitHub. An example of regressions on revenue are shown in Figure 7 and histograms showing example distributions after final transformation are shown in Figure 8.

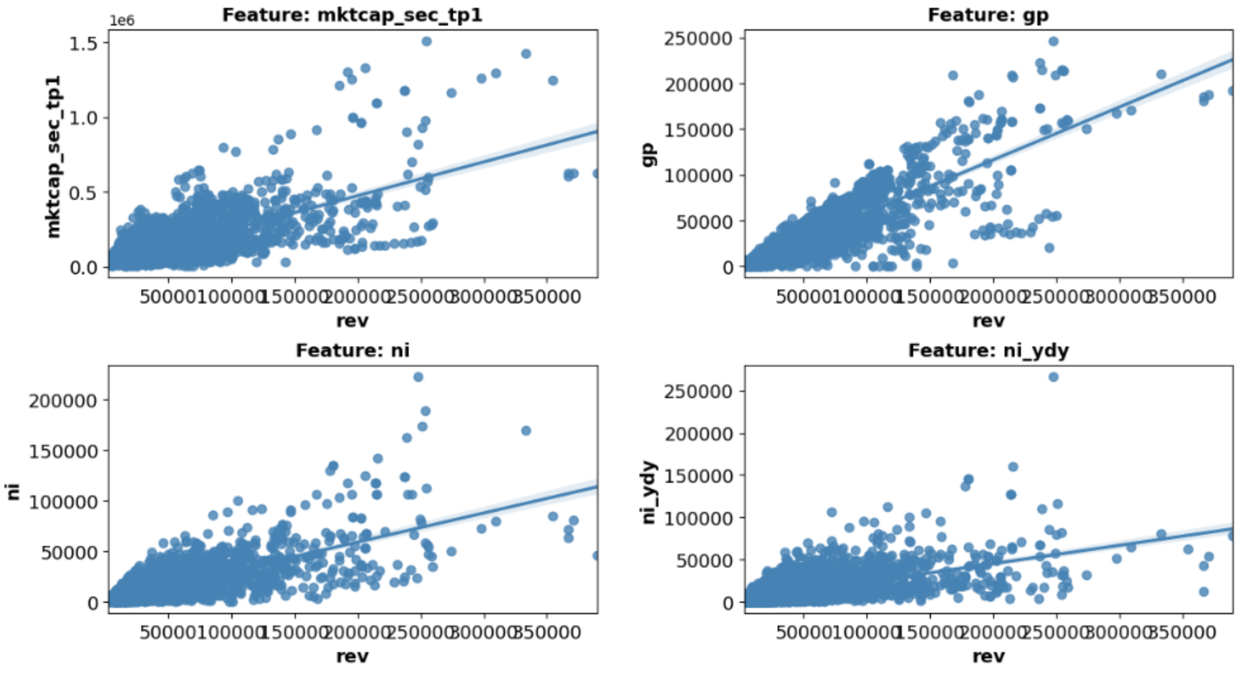

Figure 7 Example scatterplots and regressions of square-root transformed (after taking absolute value) variables regressed on revenue. The sign of the variables is preserved after transformation and inverse transformation.

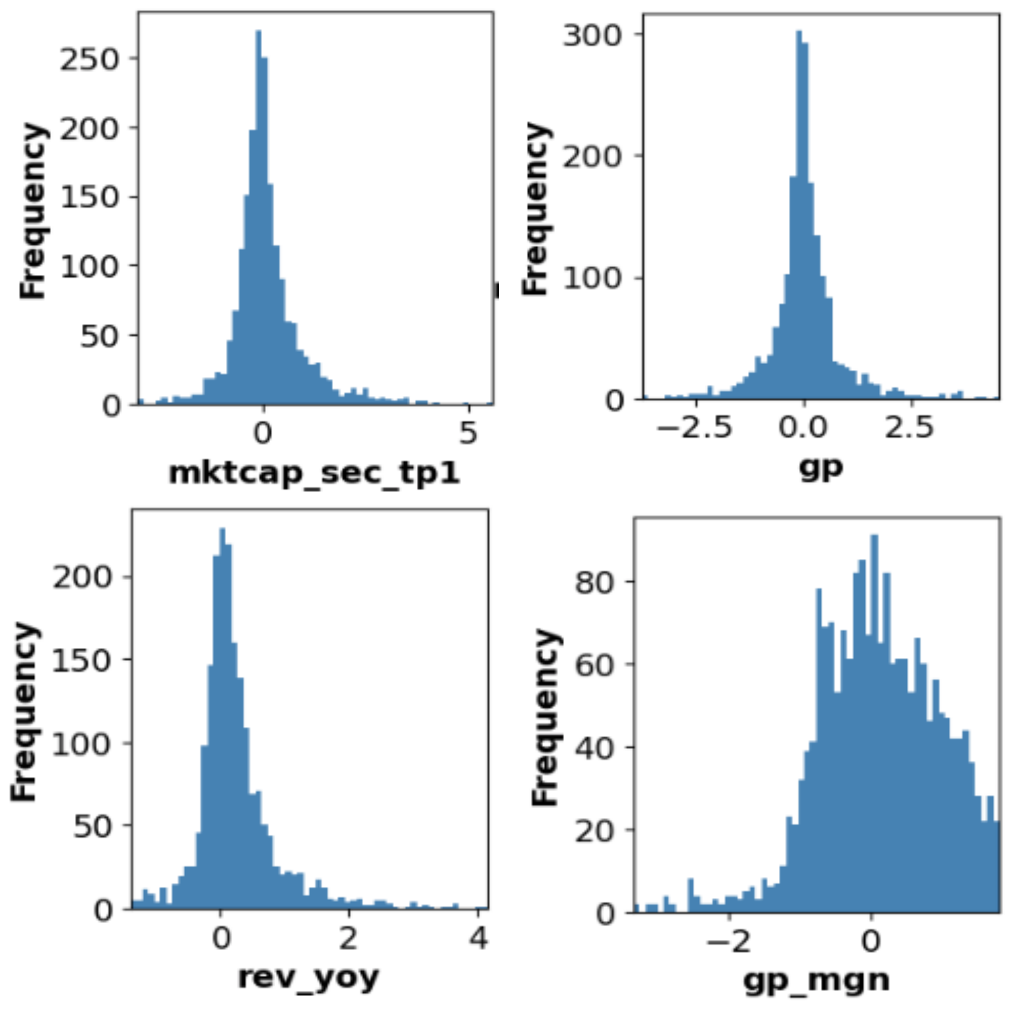

Figure 8 Example histograms of transformed variables: two variables having original units in USD amount (market capitalization, the target variable, and gross profit, a feature), and two feature variables having original units in percentages (year-over-year percent change in revenue and gross profit margin). These transformed variables have zero mean and unit standard deviation.

2.6 Additional Regression Modeling

Given the additional feature engineering and accompanying transformation explained in the prior section, I then re-fit the three models, the linear regression model, the xgboost decision tree model, and the support vector regression model using the same methodology described earlier.

To further extend the modeling in an attempt to better predict the target and better understand the effects of different features, I also developed a neural network model and fit two versions of this model to the data. The neural network model has two main components: (i) a linear layer that consumes features as inputs and then sends its outputs to the final output layers, and (ii) a nonlinear model of three sequential nonlinear layers that consumes features as input and also consumes the linear layer output as input before sending its outputs to the the final output layers. The outputs of the linear and nonlinear components are then concatenated into the final output node. Hence, the final node is a sum of the linear and nonlinear component models.

In this architecture, the model has an explicit linear component that feeds directly to the output as well as a nonlinear layer that consumes both features directly and the linear outputs directly. The idea behind this architecture is to include an explicit linear model to capture the linear effects of features and then supplement this linear model with a nonlinear model to capture any supplementary nonlinear effects.

The models have separate L1 regularization hyperparameters for the linear and nonlinear layers and I use these regularization hyperparameters in an attempt to build models that are more or less weighted towards the linear and nonlinear components of the model.

In the first neural network model (NN-1), I fit the model via cross validation across a broad range of L1 regularization parameters for both the linear and nonlinear weights. The idea with this parameterization is to build a flexible model and let the model fitting process determine the relative weights of the linear and nonlinear elements.

In the second neural network model (NN-2), I fit the model via cross validation across a more limited range of regularization parameters for the linear and nonlinear weights. In this case, I set the L1 regularization parameters to be larger for the linear weights thereby increasing the constraints on the linear effects and set the L1 regularization parameters to be smaller for the nonlinear weights thereby allowing the nonlinear effects to be more flexible in their fit to the data. The idea with this parameterization is to build a model that somewhat favors the nonlinear effects in an attempt to encourage the identification of more nuanced nonlinear features.

For each neural network model, I evaluate a set of models by cross validation with a grid search over the two L1 hyperparameters using mean absolute error (MAE) as the scoring metric. Hence, I fit a total of five models to the data, the same three models fit in the first iteration of model fitting plus two neural network models.

3 FINDINGS

3.1 Test Metrics

The additional feature engineering had mixed effects on the re-fitting of the three models, but did result in some improvements in fit. The linear regression and xgboost decision tree models show improvement in most fit metrics, but the support vector regression exhibits a decrease in performance (Table 2). In particular, the absolute percent error (APE) metrics improved by a meaningful amount for the linear regression and xgboost decision tree models. For example, median-APE improved by an incremental 13% and 22% in the linear regression and xgboost decision tree models, respectively. The r-squared metric improved by 6% from 83% to 89% for the xgboost decision tree model, but declined slightly by about 2.5% incrementally from 70% to

67.4% for the linear regression model. In contrast, all metrics show a decrease in performance for the support vector regression model, although median-APE only increased slightly by about 2%, increasing from 40.8% to 42.7%.

Focusing on median-APE as a test metric, the two neural network models are about as predictive as the linear regression model and thus did not improve upon the predictive accuracy of the first three models. The second neural network model, NN-2, shows much higher r-squared and lower mean-APE, suggesting perhaps that the nonlinear focus of this model can somewhat better predict the range of extreme values in the target variable.

Table 2 Accuracy metrics for each of the the five models on the original, untransformed test data using the second iteration of feature-engineered training data: linear regression (LR), xgboost decision tree (XGB), support vector regression (SVR), neural network 1 (NN-1), and neural network 2 (NN-2).

Model	r-squared	mean-APE	median-APE
LR	67.4%	176%	46.9%
XGB	89.1%	151%	39.8%
SVR	36.0%	150%	42.7%
NN-1	11.9%	237%	46.2%
NN-2	82.8%	193%	46.9%

3.2 Feature Importance

In this second iteration of model fitting, I calculated feature importance for the linear regression, xgboost decision tree and support vector regression models in the same way as described previously. For the neural network models, which have a linear component with coefficients analogous to the coefficients in a linear regression model, I also calculated feature importance for these linear coefficients using the same methodology as for linear regression feature importance.

Similarly to the feature importance results previously described, there are several interesting similarities and differences among the models with respect to feature importance (Figure 9) and I highlight a few observations here. The top features by importance in the linear regression model are very similar to the most important features in the xgboost decision tree model, including the approximate rank order. Revenue is an exception; revenue is not a top feature in the linear regression model, as would be expected given that the 1st order linear effect of revenue was explicitly modeled during feature engineering, but revenue is the top feature in the xgboost decision tree model. Interestingly, revenue is also the top feature in the support vector regression model. These results suggest that revenue has effects beyond the 1st order linear effect. The indicator variables have very low importance except that in the support vector regression model the indicator features div_flg and gp_mgn_flg rank fairly high.

Although its top features differ a bit more from the other two models, the support vector regression model has many top features in common with the other two models, including dividend amount, net income, gross profit margin, gross profit, net income margin, and year-over-year change in revenue. The difference between mean sensitivity and mean absolute value of sensitivity for the support vector regression features suggest that many of these features are acting in a nonlinear way. As in the first iteration of model fitting, the support vector regression model has a more even distribution of importance over features which contrasts to the xgboost decision tree model in which the top handful of features dominate somewhat more, although the xgboost decision tree has a more even distribution of importance over features in this second iteration of model fitting. The feature importance results for the linear coefficients in the two neural network models do not meaningfully differ from the other models, hence feature importance graphs are not shown here (these graphs are in GitHub). Many of the same features from the other models also rank high in the neural network models and the top two neural network features, revenue and dividend amount, are the same as in the xgboost decision tree and the support vector regression models.

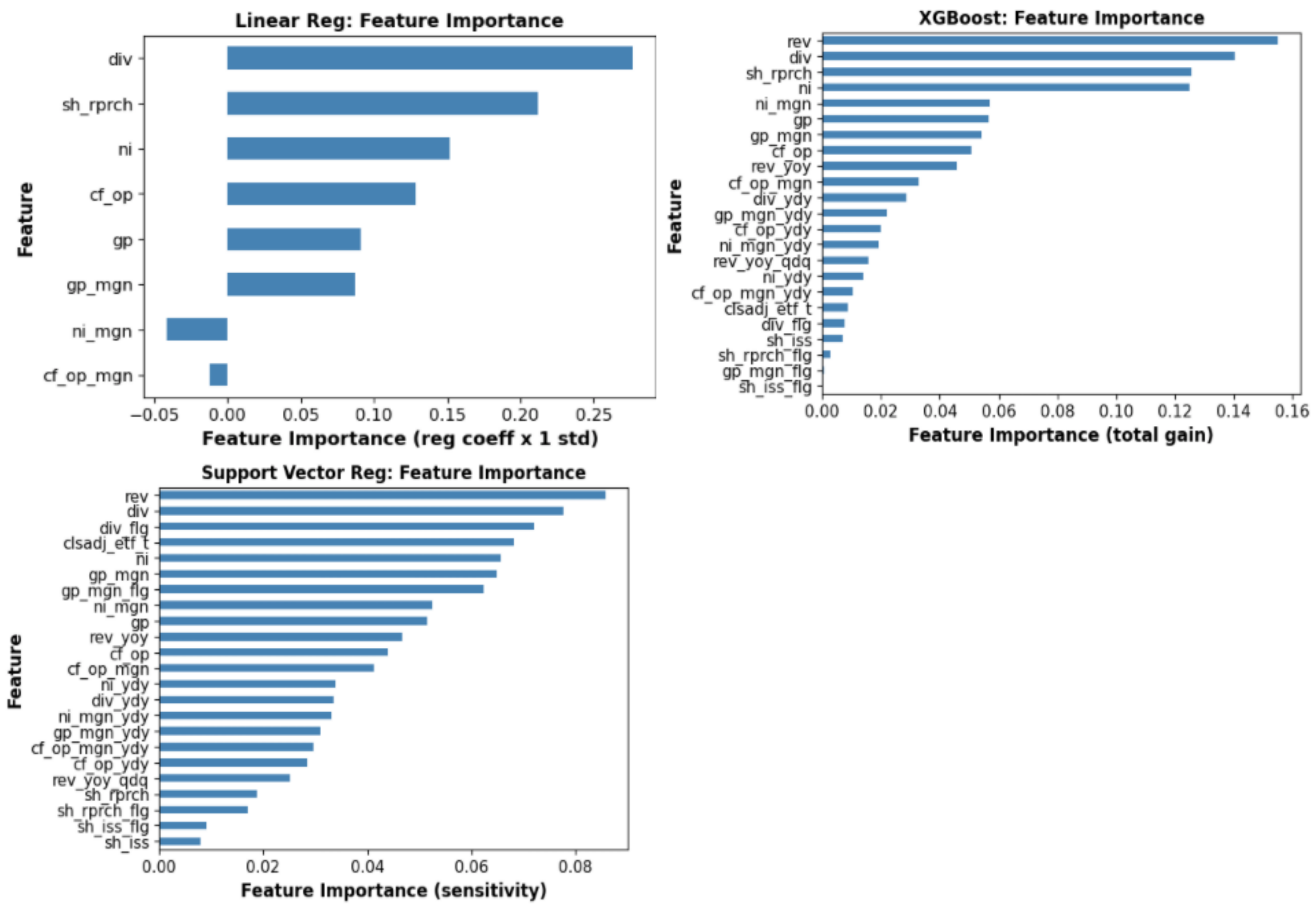

Figure 9 Relative feature importance on the fit to the training data using the second iteration of feature-engineered training data for the linear regression, xgboost decision tree, and support vector regression models.

4 CONCLUSIONS AND FUTURE WORK

The exploratory data analysis and the modeling suggest that this is a rich data set with many feature variables potentially acting to impact market capitalization. The best overall model in terms of predictive accuracy (40% median absolute prediction error on the test data) is an xgboost decision tree model trained on data with a custom transformation applied as part of the feature engineering process. The results from feature engineering and the analysis of feature importance across models suggest that the following features are among the top variables for predicting market capitalization: revenue, dividend amount, share repurchase amount, net income, net income margin, gross profit, gross profit margin, and operating cash flow. These are all core measures of a company's financial performance so it is not surprising that these features are important.

The best models fit here are not strongly predictive, so there is substantial opportunity for improving upon the accuracy of these models. There are several paths for potentially enhancing these models. One path for improving model performance is to analyze the models in more detail in terms of how features are acting via linear and nonlinear effects to predict market capitalization. A second path for further analysis is to better understand any heterogeneity in the data set: different features and models may be applicable to different kinds of companies, such as small versus large firms or firms in different sectors such as banking versus utilities. If there exists substantial heterogeneity in this regard, then a segmentation approach and/or mixture modeling may be appropriate. An analysis of model residuals to identify firms with accurate versus inaccurate predictions and any association with the feature values of these firms may help identify segmentations and features that could lead to alternative model structures. The xgboost decision tree model is 7% incrementally more accurate in terms of median-APE than the linear regression model, which is not a large difference. This result suggest that a combination of additional feature engineering and different model architectures may be required to meaningfully improve model accuracy.

5 BUSINESS RECOMMENDATIONS

The results of this analysis suggest the following business recommendations.

1. Use the xgboost decision tree model for one-quarter-ahead predictions of market capitalization. The median percent error rate is 40%, so the model predictions are approximate guidance.
2. The financial metrics revenue, dividend amount, share repurchase amount, and net income are the most important features for predicting market capitalization. These features have positive 1st-order (linear) effects on market capitalization, but may also have more complex relationships with market capitalization and other financial metrics.
3. In future analyses, consider segmenting the data by sector or business size or some other factor and then build predictive models specific to these segments. Combining business knowledge of these kinds of segments with the results from data science analyses in a collaborative way may be the best way to improve predictions. An analysis of specific companies that are accurately and inaccurately predicted by the models here may be a good starting point.

6 CONSULTED RESOURCES

T. Hastie, R. Tibshirani, J. Friedman. 2009. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd Edition. Springer

James, G., D. Witten, T. Hastie, R. Tibshirani. 2021 An Introduction to Statistical Learning with Applications in R, 2nd Edition. (https://web.stanford.edu/~hastie/ISLR2/ISLRv2_website.pdf\)

KERAS Python high-level neural network API (https://faroit.com/keras-docs/2.0.9/#keras-the-python-deep-learning-library)

SKLEARN Python package for machine learning (https://scikit-learn.org/)

STATSMODELS Python package for statistics (https://www.statsmodels.org/)

TENSORFLOW Python package for neural networks (https://www.tensorflow.org/)

XGBOOST Python package for boosted decision trees (https://xgboost.readthedocs.io/en/latest/install.html#python)