Data Description

We have five datasets from 2014 to 2018, each featuring over 200 annual financial indicators for publicly traded US companies. On average, there are about 4,000 listings in each dataset, compiled using Financial Modeling Prep API and pandas_datareader. The 'Sector' column sorts stocks into industry sectors, enabling analyses specific to each sector. 'PRICE VAR [%]' details the annual percentage change in stock price, with each year's dataset indicating the following year's price variation. A binary 'class' column indicates stock price movement, where '1' signals an increase and '0' a decrease, serving as potential buy or not-buy indicators for trading strategies. The datasets are equipped for classification tasks using the 'class' column and regression tasks using the 'PRICE VAR [%]' column to predict stock values.

Data Processing

1. Renamed all the column heads to follow the industry standard – lower case and '_' as separator.
2. Separated the predictors (all other features) and targets (class and price variation).
3. Did research on what factors influence stock prices. Came up with ways to reduce the number of predictors from 223 to 46(details see jupyter notebook attached). Might employ PCA again in the next step but for now this is good.
4. Examine missingness by using `missingno` library to visualize the patterns of missing data. Printed out the non missing ratio of each feature.
5. We have `class` and `sector` as our categorical variables. Summarized the distribution of `sector` and observed imbalance issues.
6. Printed out the mean of each numerical predictor and observed the need to scale the data.

Potential Data Issues and Solutions

- Data missingness
    - The nature of the missingness needs to be assessed to determine the appropriate imputation method.
    - High-missing-value columns across all years may be dropped.
    - Given the missing data seems not to be MCAR, mean or median imputation methods may not be suitable.
    - Mode imputation is also not applicable as most features are numerical.
    - Therefore, kNN imputation is currently the favored approach.
    - Missing indicators may be added for capturing missingness patterns and enhancing model interpretability.
- Imbalance issues
    - The sector distribution does show some imbalance, which is common in real-world datasets where some categories (in our case, sectors like Financial Services, Healthcare, and Technology) are more prevalent than others (like Utilities and Communication Services).

        - Financial Services, Healthcare, and Technology sectors are over-represented in comparison to other sectors across all the years provided.
        - Utilities and Communication Services sectors are under-represented, making up a smaller portion of the dataset.
    - Potential solutions
        - Resampling: Adjust the dataset to have more balanced sector representation. Oversampling methods (like SMOTE) can help to increase the representation of under-represented sectors. Conversely, undersampling can reduce the prevalence of over-represented sectors, though we lose data in the process.
        - Weighted Models: Use model algorithms that allow for weighting classes. Many algorithms have a class_weight parameter that can help balance the influence of each sector on the model training.
        - Stratification: Ensure that train-test splits or cross-validation maintain the original distribution of the sector variable using stratified sampling. This keeps the imbalance issue consistent across different subsets of your data.
- Data scaling
    - It looks like our features have very different scales. For instance, revenue is in the billions while eps_growth is close to zero. Such disparities can indeed cause issues with many machine learning algorithms that are sensitive to the scale of the data, particularly those that use distance measures (like k-NN, SVMs, and gradient descent-based algorithms like logistic regression).
    - We don't want to make any assumption about the distribution of our data. Hence, we will normalize our data (Min-Max Scaling). Normalization rescales the data to a fixed range, typically [0, 1].