Firm Analysis

Problem statement

A Supervision Manager has asked you to help in allocating scarce resources, and identify which firms their team should focus on.

Dataset

The dataset is divided into two sheets General and Underwriting with little under 500 firms. Some firms have both General and Underwriting data and few others have only one part of the data. All the features are the yearly aggregate between 2016 and 2020.

- General tab includes notable features like Gross Written Premium, Net Written Premium, SCR coverage ratio, etc
- The underwriting tab includes data on claims, BEL, expense, and combined ratio.

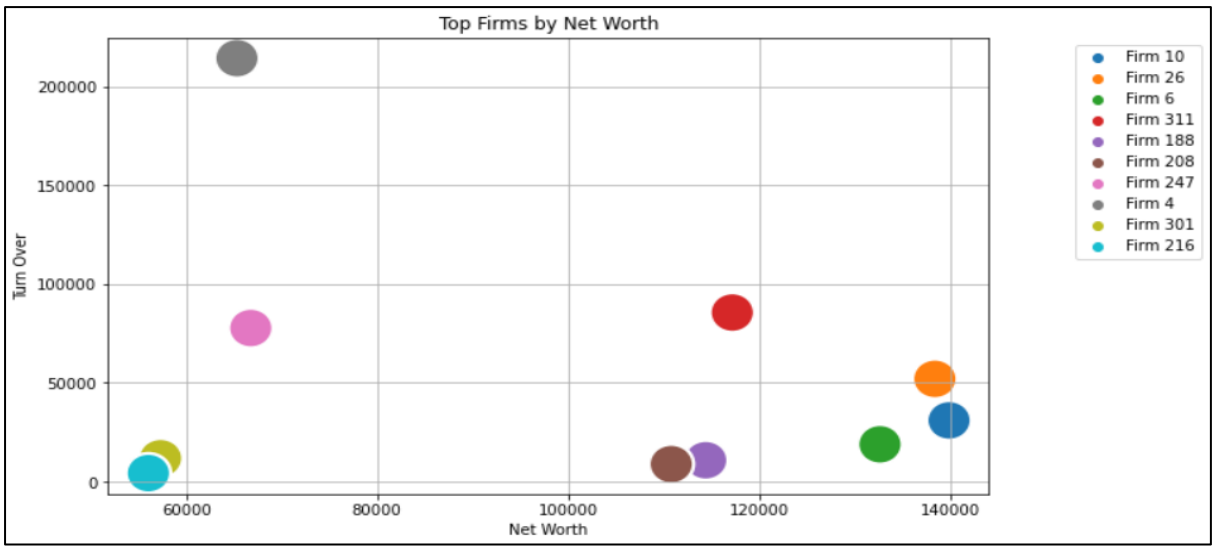
Programming language: Python

Code: github

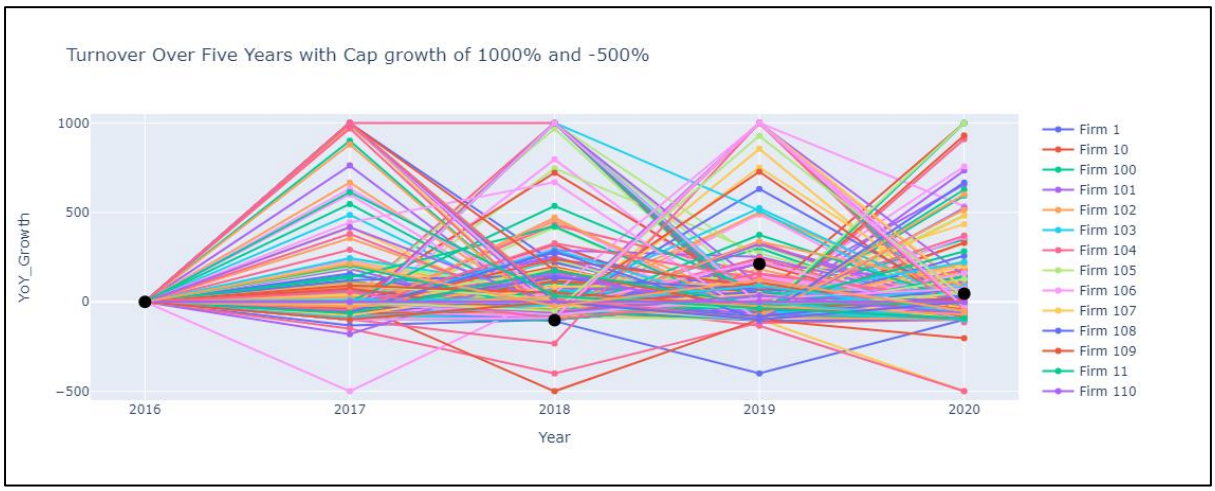
Task 1

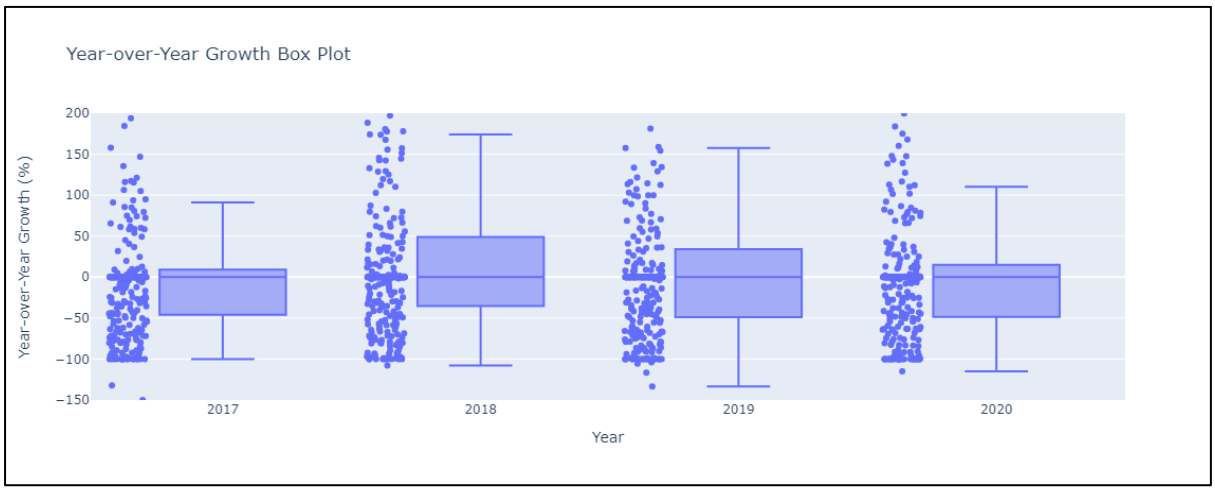
Given: Using the data provided, please analyse this data using a programming language of your choosing and produce a short report, including tables and charts, to highlight which firms should receive the most attention, according to the given metrics.

Method 1: Finding the high net worth firms. Net worth is the difference between total assets and liabilities across years. The below graph shows the top 10 net worth firms with their corresponding turn over the years.

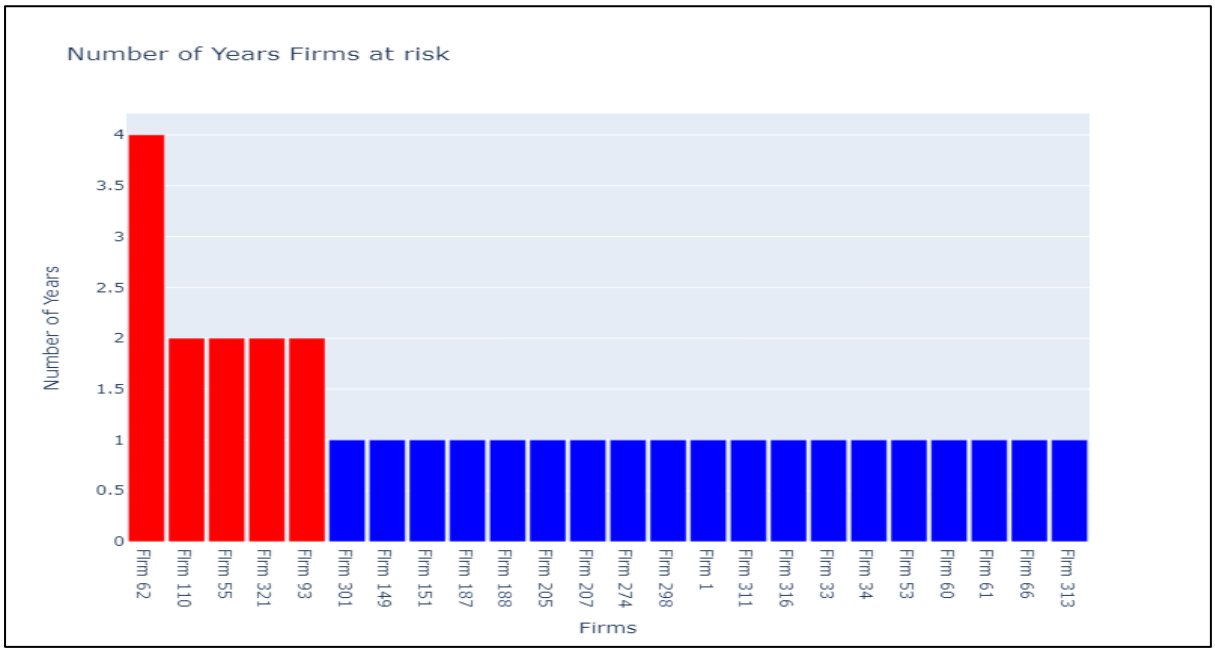


Method 2: Year-on-year growth based on the turnover for each firm. This gives the rate of growth over the 5 years. We see a huge surge in growth from 2016 to 2017 with a mean growth of over 1000% which is unusual. Hence using a box plot and bringing out the anomalies. It is found that in 2017 alone there were 55 (upper bound) and 4 (lower bound) outliers. Clearly, the data is not consistent here.

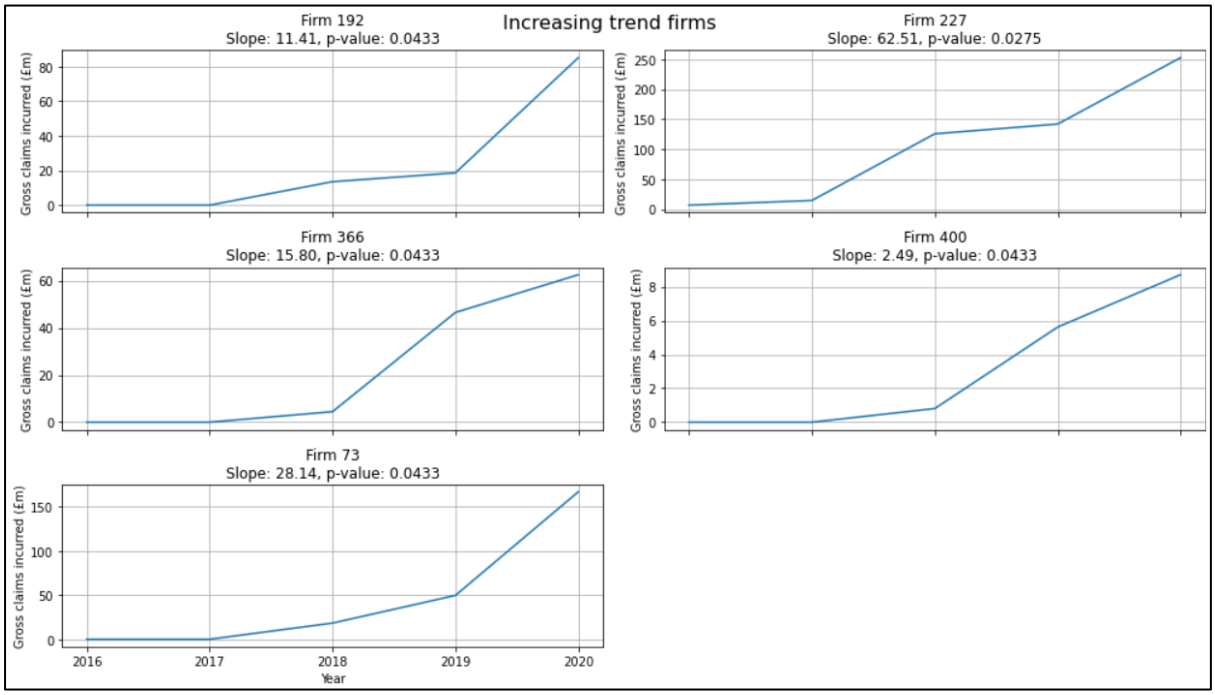


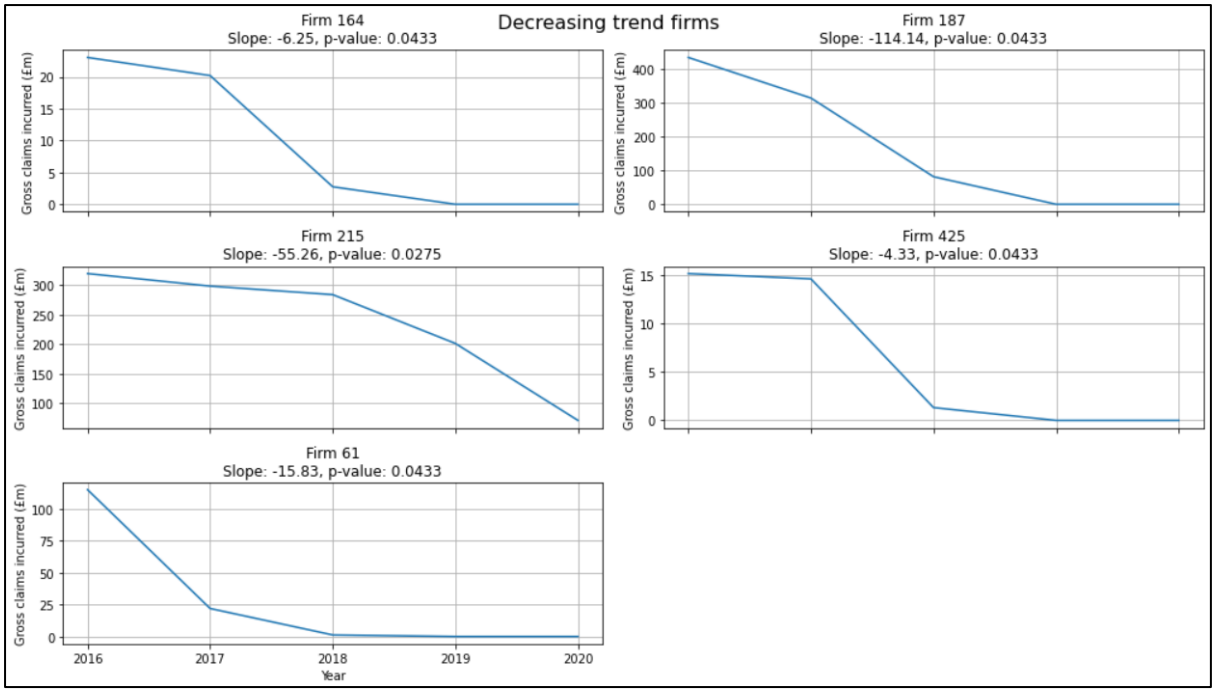


Method 3: Identified the list of firms that can pass risk on to the insurer. This is calculated using the ratio of Net Written Premium (NWP) to Gross Written Premium (GWP). Firm 62 is the riskiest firm among all with a ratio of less than 0 based on the below graph.



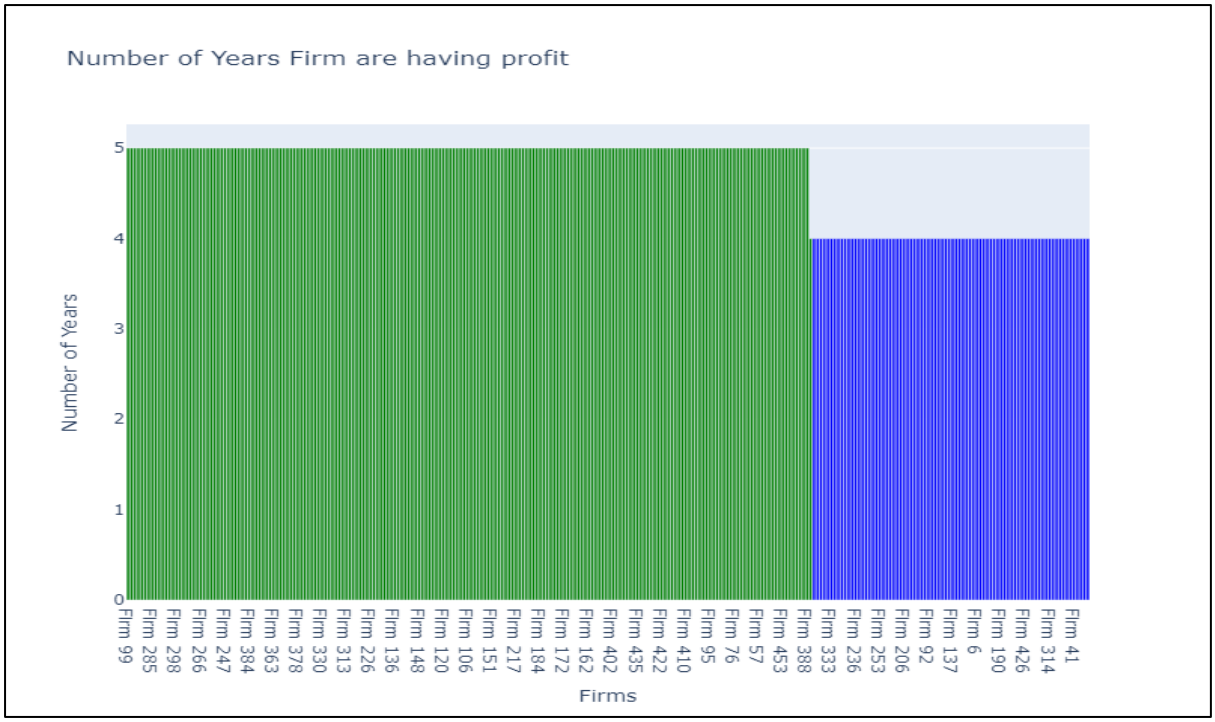
Method 4: Monitor the trend in Gross claims incurred using the mann kendall statistical test. Many firms are showing no trend with several firms showing increasing and decreasing trends. This will help in tracking the trend in cost to the insurer.



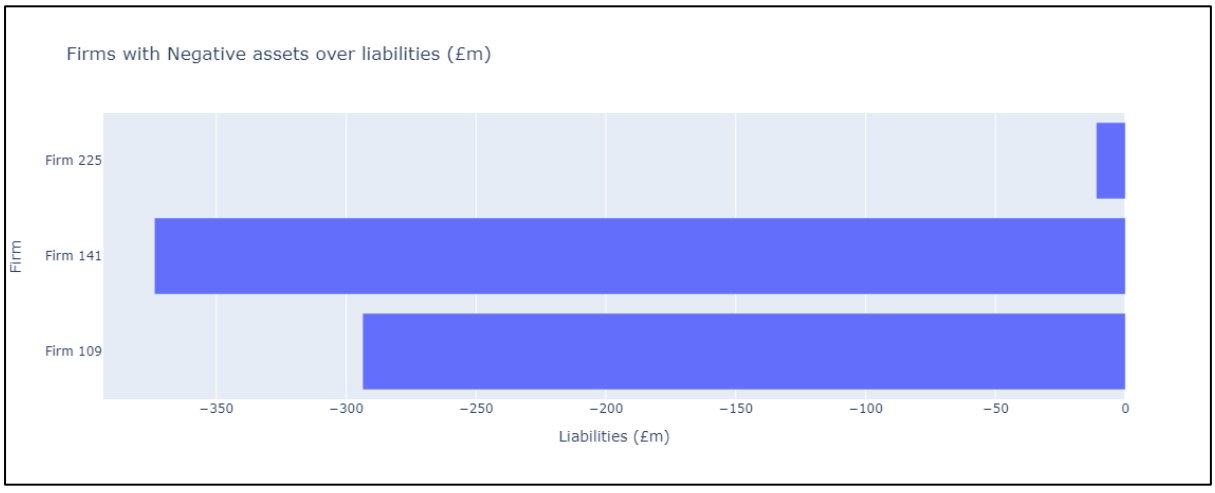


Method 5: Find companies that have enough capital gains to meet the requirements. There are about 15 firms as shown in the graph who had enough capital to stay in the market in all 5 years of data. These are the list of valuable companies.

Method 6: It is always essential to follow the companies that show profit each year. Hence the below graphs the list of 30 companies who had recorded profit in all 5 years. Hence, all these firms need attention.



Method 7: We need to be careful about companies excess of assets over liabilities (equity) less than zero. This means that they have more liabilities to be paid which shows the inefficiency of the company. The below graph shows 3 companies with a negative value and firm 141 tops with -370(£m).



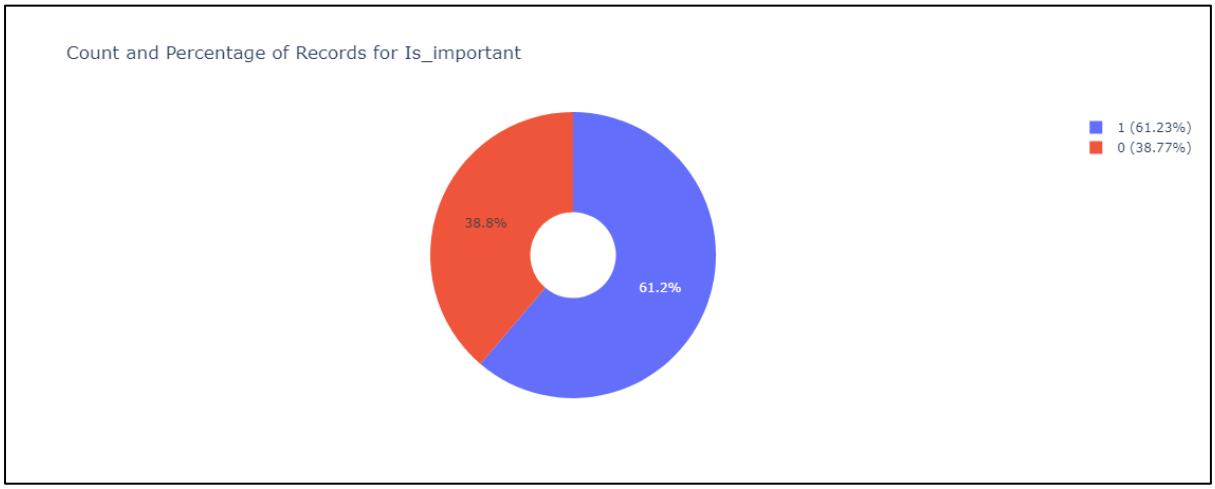
Task 2

In the data, it is essential to know given the features if the firm has enough capital requirements. This shows the potential of the company to grow. By definition, we know SCR coverage ratio of over 1(100%) has enough capital and else otherwise. So I change the

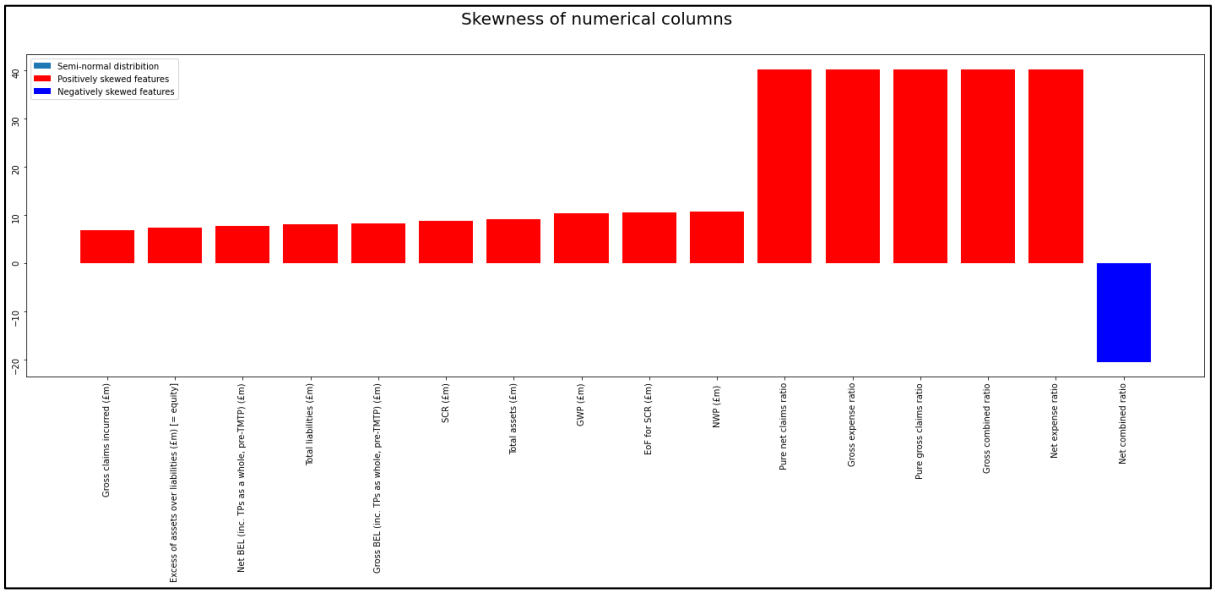
problem into a classification problem with 1 representing the firm with potential and 0 without enough capital. Below are the steps in model building.

EDA & Feature Engineering:

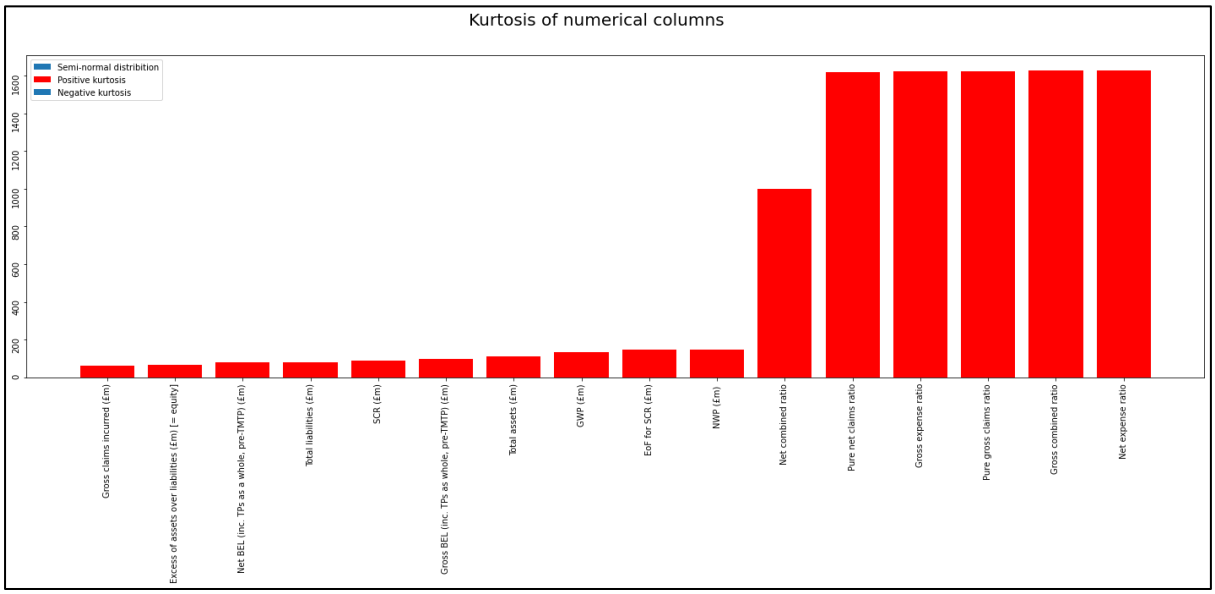
1. Merge data from both sheets and remove records without SCR coverage ratio(target).
2. Year data is pivoted so each firm has 5 records so we have more data to train.
3. Both firm and year columns are removed.
4. The target column has a split of 39% to 61% as shown in figure.



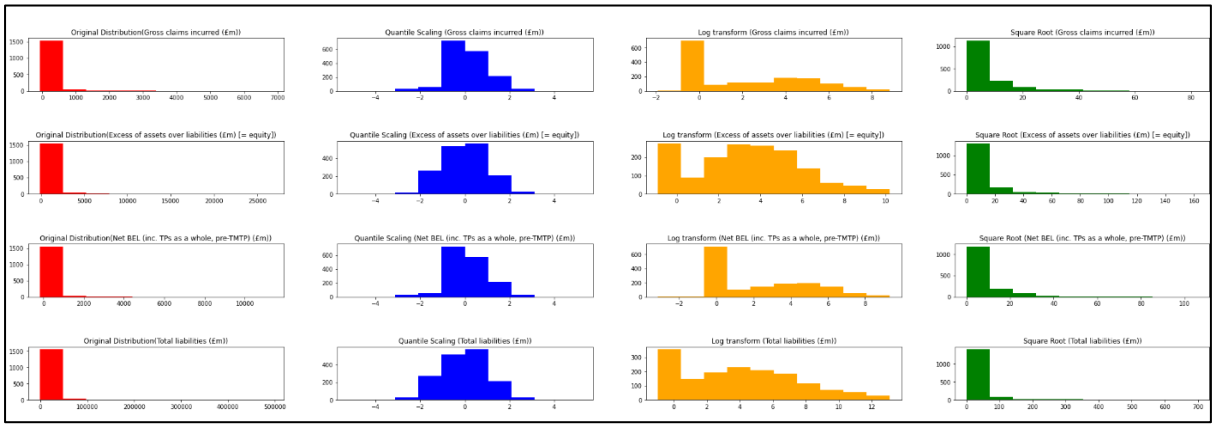
5. On checking the null values there are none.
6. Next we check for the skew and kurtosis in the numeric columns and understand the distribution.
7. The Below graph shows the skewness in each column. We can see they are highly skewed.



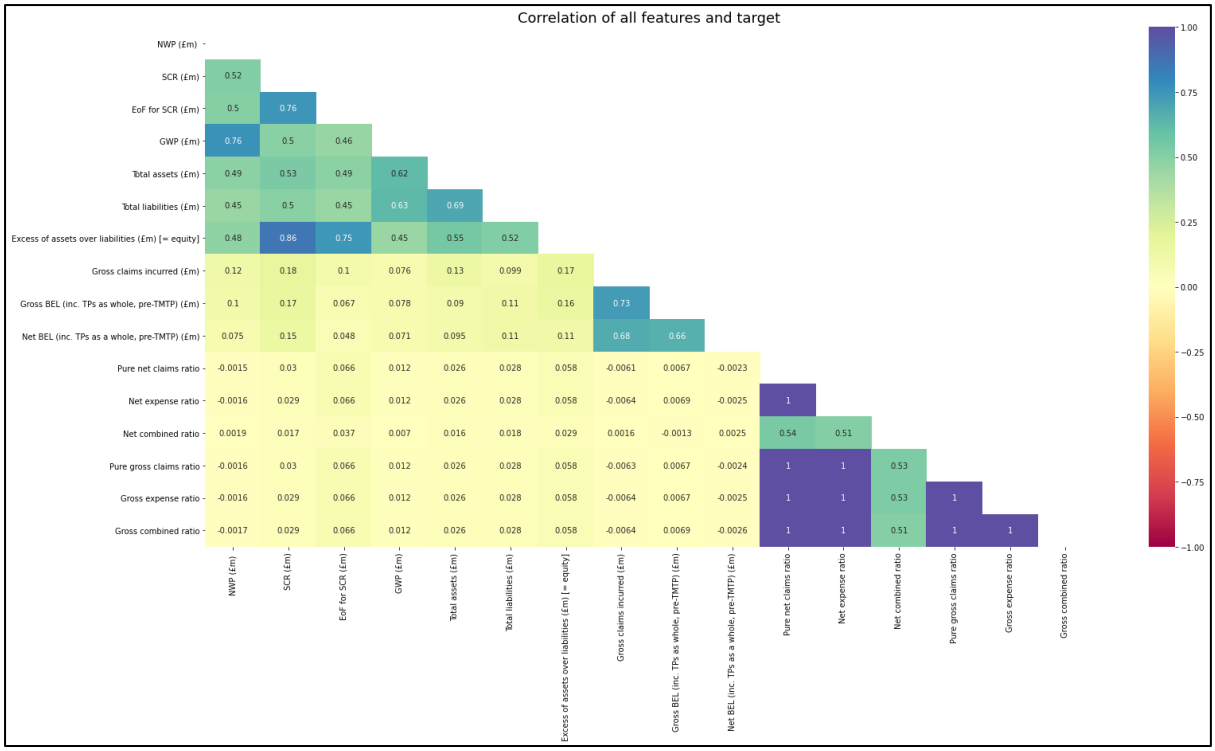
8. Similarly with kurtosis we see massive values in ratio columns.



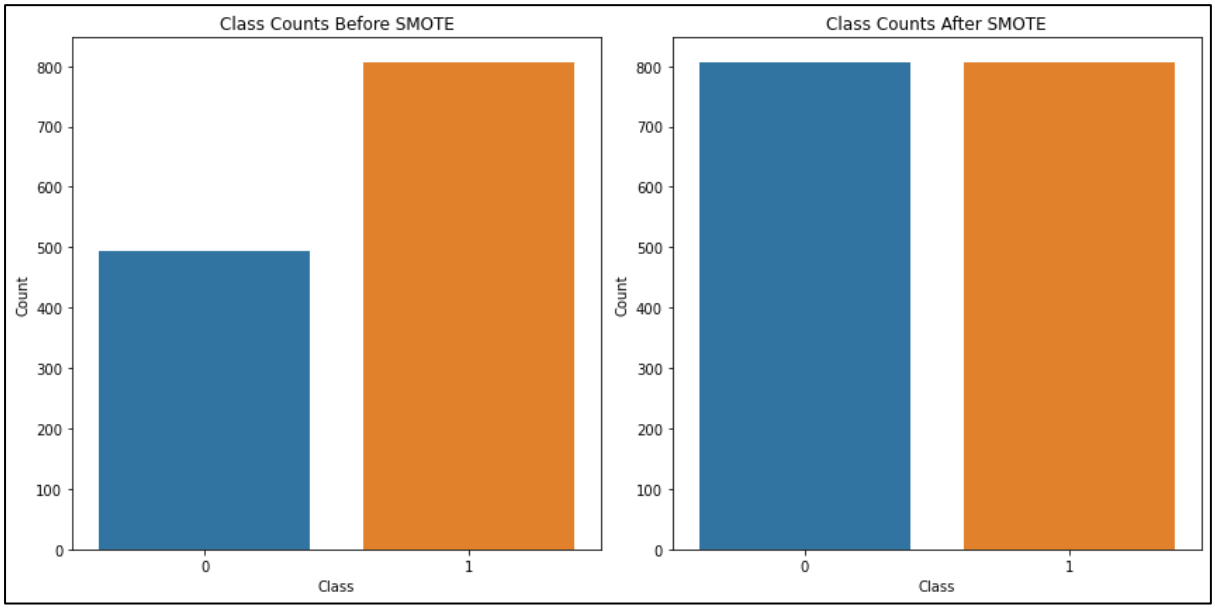
9. It is essential to scale this data effectively so that it does not affect the model performance. Hence we apply multiple transformations that could transform skewed data. Methods like quantile scaling, log transform, and square root of data are used to see its distribution. Below is the transformation of a few columns (For a full list refer to the notebook). Quantile scaling does a great job to normalize data as seen.



10. Also it is essential to check the correlation between variables to see intercolinearity between them. We can see ratio columns are highly correlated. Hence we need to remove it were possible.



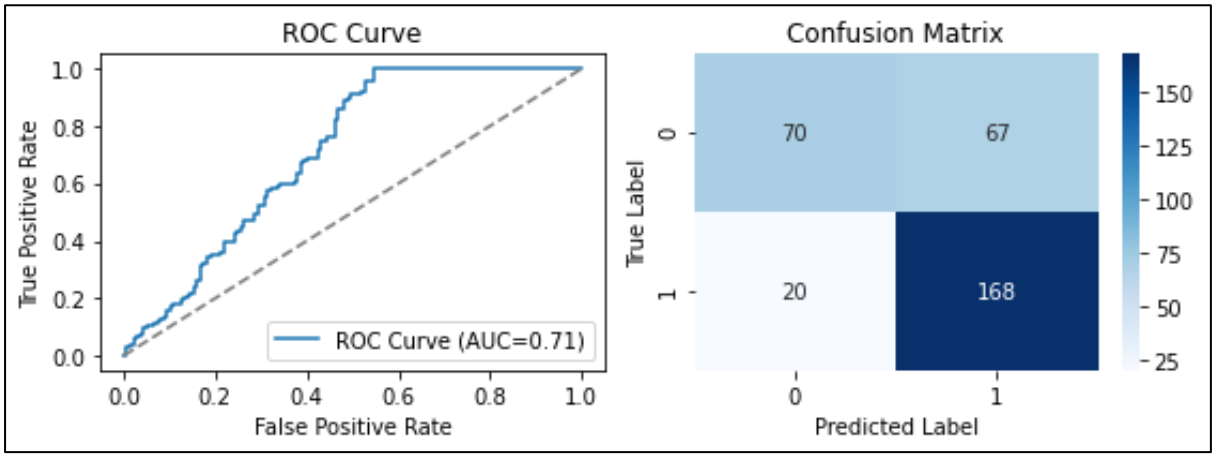
11. We split the data into train and test for model building.
12. Then we scale the data using quantile transform.
13. Apply PCA to reduce the less significant features. The n_components is set to be 0.95.
14. There is an imbalance in target data. More of class 1 will result in the model overfitting the class 1 case rather than a generalized model. Hence we generate synthetic data using a method called SMOTE.



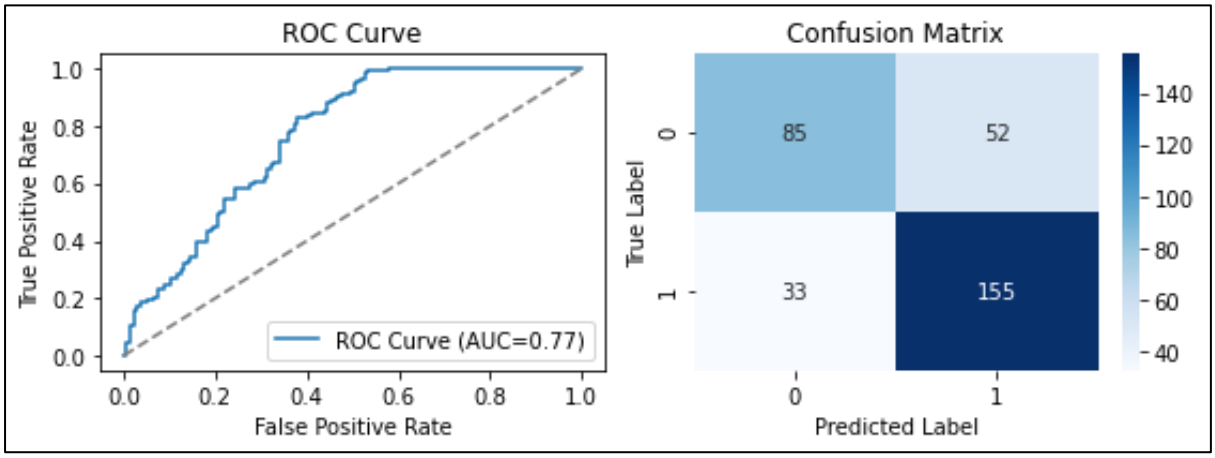
Model building:

We have cleaned, transformed, and removed the features that need to be modeled. I use two different algorithms to train the model that could to predict our classes.

1. **Logistic regression**: It is a process of modeling the probability of a discrete outcome given an input variable. The output is the probability score and we have set a cut of 0.5. In summary, any case getting a probability above 0.5 is set to 1 and 0 otherwise. On fitting our data we get an **accuracy of 73%** and **F1-score of 0.71**. It does a moderate job of predicting class 1 cases with an F1–score of 0.79. Below is the ROC curve and confusion matrix of the model in the test set.

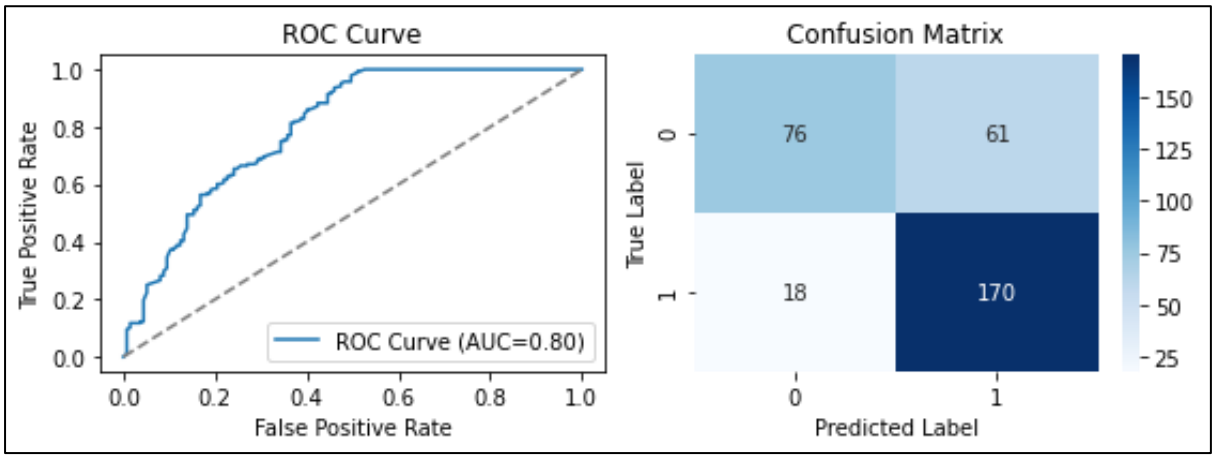


2. **XGBoost**: Next we use a more complex model which is of boosting type called XGBoost. In the boosting method n different weak models are trained on the completed dataset. The data points that are wrongly classified with the previous model are provided with more weights. Here we get a slightly improved **accuracy of 74%** and **F1-score of 0.73**. But the F1-score for class 1 has not changed.



3. **Hyperparameter Tuning**: Boosting is one of the powerful ML algorithms. Hence we try to do hyperparameter tuning for XGBoost to find the best combination of parameters for the model. I use a library called hyperopt. Some important features that are tuned are max_depth, learning_rate, and n_estimators. After 100 iterations

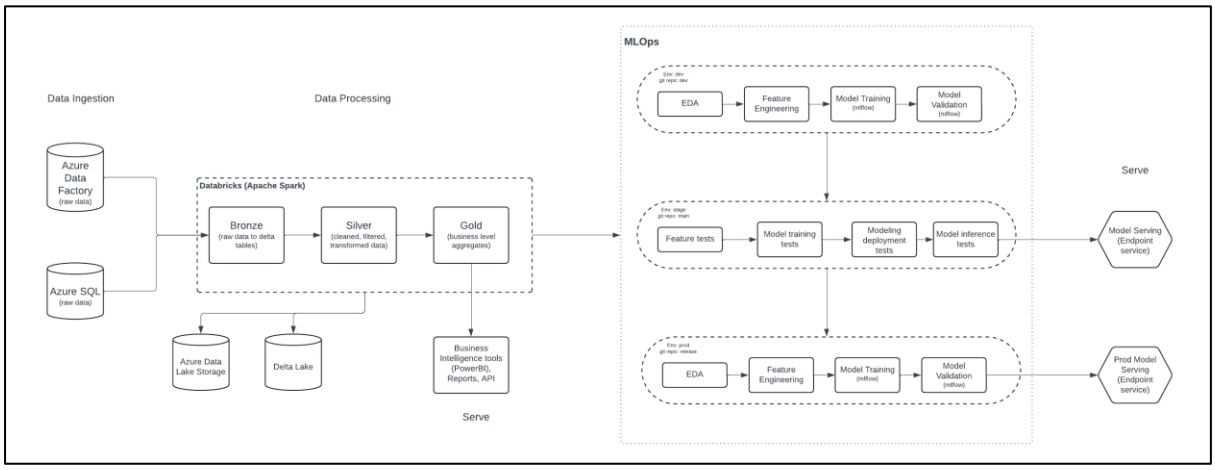
of testing various parameter combinations, the best model parameter is generated. The final best model statistics has an **accuracy of 76%** with the best **F1-score in class 1 of 0.81.** Following are its other plots.



Task 3

Data Processing and Analytics Pipeline

Building a pipeline involving data processing and analytics involves multiple steps from getting the raw data to serving the results. In the architecture proposed we cover a high level process to get data, process it, store it, build reports, build ML models, and finally deploy, and serve the model. I have built the architecture with a combination of Microsoft Azure and Databricks giving high value to scalability and MLOps.



Data Ingestion:

1. Data sources might be from various sources within Azure cloud services like Azure data factory, and Azure SQL.
2. We usually get raw and unprocessed data during the ingestion stage.

3. Added, we can also have multiple sources of data like files, databases, cloud services, etc.

Data Processing:

1. Data ingested will have to be processed. To process that Databricks uses medallion architecture.
2. It consists of three steps bronze, silver, and gold.
3. Bronze: In this layer the raw data is converted into delta tables.
4. Silver: In this layer, we do all transformations like filtering, aggregation, transformation, removal, etc. Usually, it is run in spark clusters and has the flexibility of using languages like Spark SQL, PySpark, Scala, etc.
5. Gold: This layer holds the Business-level aggregate data. These are called OLAP (Online Analytics Processing) tables which store ready made data for faster and simpler querying of data. It follows Snowflake and Star schema of data modeling.

MLOps

1. Development: In this environment, we pull data from data lakehouse and do EDA, feature engineering, train the model and finally evaluate the model performance. The git repo is called dev. At this stage data the Data Scientist will be playing around with model. Mlfow plays a huge role in in tracking the model, metrics, and parameters.
2. Staging: At this environment, we make sure the code, model developed is error free. It is a replica of the production environment and we should be able to run all test cases and make final corrections before deployment. We test the endpoint API also. The git repo is main.
3. Deployment: This is the final stage where the model is trained again on the production data and deployed after all successful tests. It is always advisable to move code from environment to environment rather than model itself. The git repo is called release. Finally based on how model results are consumed endpoints are shared. MFlow and MLRegistery are used to track, model, metrics, and deployment status.

Data Serving:

4. Usually data is consumed or served at two stages one is at the Gold layer where we use Business Intelligence tools like PowerBI, and Tableau are used to present the aggregated business data.
5. Usually models are consumed via an API that can give predictions service based on the input given.