## RMIT University

## ISYS3420 – Machine Learning for Decision Makers

## Assessment 2: Guided Notebook and Evaluation

## Assessment type: Evaluation Report

**Title:** Identifying a machine learning model that will help to predict share price above the threshold value.

**Full Name:** Rokshana Pervin

**Contact details:** Email: s3923324@student.rmit.edu.au

**Date of report:** 12/04/2023

Some data analysis tasks have been implemented to find the meaningful insights to inform investment decision accurately. These are:

- **Data wrangling:** Cleaned the data and joined the sources data appropriately where all key features regarding the stock price and share market values are included.
- **Exploratory analysis:** Here, correlation matrix function is used with absolute value function to find the correlated features precisely.
- **Feature engineering:** In this step, the best correlated features are selected for target variable using the heat plot and drop the correlated features for reducing the multicollinearity from the dataset.
- **Training and testing the models:** In this stage, hyperparameters are tuned using two techniques such as GridSearchCV and RandomisedSearchCV which reveal the best params or parameters to obtain the model's performance.
- **Evaluate the model:** Confusion matrix has been used where F1-score helps to provide the meaningful insight regarding the investment decision.

During engineering the features, some exploratory analysis have been completed using `corr()` and `data.abs()` functions to check the correlation between each features.

```
In [321]: corr_matrix = data.corr().abs() # The `corr()` method finds the correlation of each column in a DataFrame `data`. The
          # `data.abs()` function return a Series/DataFrame with absolute numeric value of each element.
          corr_matrix
```

Out[321]:

| | revenuePerShare | netIncomePerShare | operatingCashFlowPerShare | freeCashFlowPerShare | cashPerShare | bookValuePerShare | tangibl |
|---|---|---|---|---|---|---|---|
| revenuePerShare | 1.000000 | 0.998925 | 0.998609 | 0.998630 | 0.909948 | 0.995980 | |
| netIncomePerShare | 0.998925 | 1.000000 | 0.995472 | 0.995496 | 0.891063 | 0.999009 | |
| operatingCashFlowPerShare | 0.998609 | 0.995472 | 1.000000 | 0.999952 | 0.929898 | 0.990280 | |
| freeCashFlowPerShare | 0.998630 | 0.995496 | 0.999952 | 1.000000 | 0.929594 | 0.990339 | |
| cashPerShare | 0.909948 | 0.891063 | 0.929898 | 0.929594 | 1.000000 | 0.870181 | |
| ... | ... | ... | ... | ... | ... | ... | |
| priceEarningsToGrowthRatio | 0.002411 | 0.002084 | 0.002491 | 0.002536 | 0.003343 | 0.002013 | |
| priceSalesRatio | 0.002711 | 0.002344 | 0.002801 | 0.002851 | 0.003759 | 0.002263 | |
| enterpriseValueMultiple | 0.002552 | 0.002207 | 0.002638 | 0.002684 | 0.003539 | 0.002131 | |
| priceFairValue | 0.002492 | 0.002155 | 0.002575 | 0.002621 | 0.003456 | 0.002081 | |
| shareGrowth | 0.016659 | 0.015379 | 0.018655 | 0.018869 | 0.025918 | 0.014217 | |

161 rows × 161 columns

Since, maximum features are highly correlated with each other, ` upper triangle of correlation matrix` function is used to select the top correlated features.

```
In [61]: # Selected upper triangle of correlation matrix
upper = corr_matrix.where(np.triu(np.ones(corr_matrix.shape), k=1).astype(bool))
upper
```

Out[61]:

| | revenuePerShare | netIncomePerShare | operatingCashFlowPerShare | freeCashFlowPerShare | cashPerShare | bookValuePerShare | tangib |
|---|---|---|---|---|---|---|---|
| revenuePerShare | NaN | 0.998925 | 0.998609 | 0.998630 | 0.909948 | 0.995980 | |
| netIncomePerShare | NaN | NaN | 0.995472 | 0.995496 | 0.891063 | 0.999009 | |
| operatingCashFlowPerShare | NaN | NaN | NaN | 0.999952 | 0.929898 | 0.990280 | |
| freeCashFlowPerShare | NaN | NaN | NaN | NaN | 0.929594 | 0.990339 | |
| cashPerShare | NaN | NaN | NaN | NaN | NaN | 0.870181 | |
| ... | ... | ... | ... | ... | ... | ... | |
| priceEarningsToGrowthRatio | NaN | NaN | NaN | NaN | NaN | NaN | |
| priceSalesRatio | NaN | NaN | NaN | NaN | NaN | NaN | |
| enterpriseValueMultiple | NaN | NaN | NaN | NaN | NaN | NaN | |
| priceFairValue | NaN | NaN | NaN | NaN | NaN | NaN | |
| shareGrowth | NaN | NaN | NaN | NaN | NaN | NaN | |

161 rows × 161 columns

sort_values() function has been used to identify the most important features from the index of correlation matrix as those are the most correlated with the target variable `shareGrowth`.

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Markdown

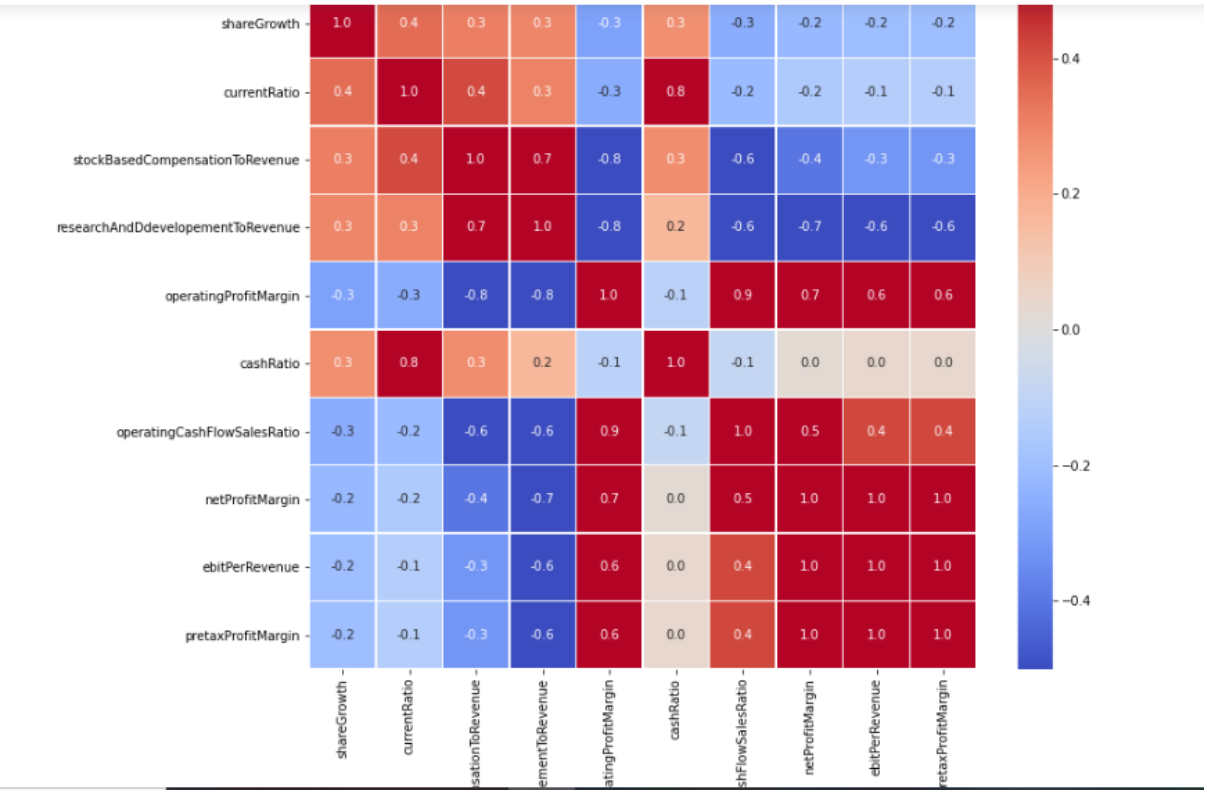
```
In [11]: df = corr_matrix.sort_values(['shareGrowth'], ascending=False).head(10)

In [12]: df
```

Out[12]:

| | revenuePerShare | cashPerShare | marketCap | peRatio | debtToEquity | debtToAssets | netDebtToEBITDA | currentRatio | intere |
|---|---|---|---|---|---|---|---|---|---|
| shareGrowth | 0.016659 | 0.025918 | 0.006505 | 0.014484 | 0.028551 | 0.147146 | 0.012355 | 0.350992 | |
| currentRatio | 0.024894 | 0.021772 | 0.034812 | 0.054984 | 0.086974 | 0.368482 | 0.019038 | 1.000000 | |
| stockBasedCompensationToRevenue | 0.016769 | 0.019808 | 0.005799 | 0.002803 | 0.000841 | 0.179464 | 0.006123 | 0.415309 | |
| researchAndDdevelopementToRevenue | 0.010997 | 0.013285 | 0.000857 | 0.009769 | 0.007147 | 0.151668 | 0.005591 | 0.303772 | |
| operatingProfitMargin | 0.023690 | 0.030561 | 0.005218 | 0.007196 | 0.034103 | 0.106736 | 0.010622 | 0.259044 | |
| cashRatio | 0.009411 | 0.027610 | 0.011576 | 0.033622 | 0.015276 | 0.271152 | 0.017611 | 0.801391 | |
| operatingCashFlowSalesRatio | 0.005456 | 0.000203 | 0.002326 | 0.006455 | 0.001496 | 0.042655 | 0.003532 | 0.211375 | |
| netProfitMargin | 0.014815 | 0.018024 | 0.004789 | 0.008118 | 0.027557 | 0.010751 | 0.040005 | 0.153976 | |
| ebitPerRevenue | 0.017441 | 0.021980 | 0.005157 | 0.008449 | 0.027790 | 0.000888 | 0.043099 | 0.136005 | |
| pretaxProfitMargin | 0.017441 | 0.021980 | 0.005157 | 0.008449 | 0.027790 | 0.000888 | 0.043099 | 0.136005 | |

10 rows × 86 columns

After that, the pearson correlation coefficient relationship chart or heat map is used to find the best correlated features with target variable and to reduce the multicollinearity from each independent features.




According to the heat map, it is observed that 'stockBasedCompensationToRevenue' variable is showing high multicollinearity with 'researchAndDdevelopementToRevene ' variable as they are having same information which may results the poor model perf ormance [1]. To reduce the risk and extend the performance of the model, it is consi dered to retain only one feature from both of those features using exploratory analysi s and finding correlation with target variable. From the given dataframe, it is noticed t hat 'researchAndDdevelopementToRevenue' is having high correlation with `shareGr owth` than 'stockBasedCompensationToRevenue' [3]. Therefore, 'researchAndDdev elopementToRevenue' feature will be more consistent than 'stockBasedCompensatio nToRevenue' feature to enhance the model performance adequately.

| | shareGrowth | currentRatio | stockBasedCompensationToRevenue | researchAndDdevelopementToRevenue | operatingProfitMargin | cashRatio | operatingCashFlow |
|---|---|---|---|---|---|---|---|
| 0 | 0.348744 | 1.533173 | 0.013945 | 0.078249 | 0.180515 | 0.664423 | |
| 1 | -0.082432 | 1.271979 | 0.003856 | 0.000000 | 0.069748 | 0.093500 | |
| 2 | 1.062603 | 1.363604 | 0.024877 | 0.068310 | 0.241473 | 0.360710 | |
| 3 | 0.196094 | 3.177350 | 0.012926 | 0.162538 | 0.390278 | 2.561694 | |
| 4 | 0.180224 | 0.976446 | 0.000392 | 0.000000 | -0.027043 | 0.135815 | |
| ... | ... | ... | ... | ... | ... | ... | |
| 538 | 0.265210 | 0.851140 | 0.010702 | 0.099666 | 0.161427 | 0.021368 | |
| 539 | 0.415486 | 1.447358 | 0.191973 | 0.254218 | -0.199658 | 0.410693 | |
| 540 | 5.201161 | 3.281676 | 0.117414 | 0.107730 | 0.020390 | 0.848138 | |
| 541 | 1.992131 | 3.731515 | 0.281483 | 0.226956 | -0.264234 | 0.334632 | |
| 542 | 0.321691 | 2.629014 | 0.010703 | 0.073003 | 0.287700 | 1.069767 | |

543 rows × 10 columns

Similarly, 'operatingCashFlowSalesRatio' and 'netProfitMargin' variables are showing multicollinearity with each other and 'netProfitMargin' would be closest feature to `shareGrowth` in accordance with exploratory analysis and following data frame.

| BasedCompensationToRevenue | researchAndDdevelopementToRevenue | operatingProfitMargin | cashRatio | operatingCashFlowSalesRatio | netProfitMargin | ebitPerRever |
|---|---|---|---|---|---|---|
| 0.013945 | 0.078249 | 0.180515 | 0.664423 | 0.197753 | 0.207438 | 0.177 |
| 0.003856 | 0.000000 | 0.069748 | 0.093500 | 0.089289 | 0.050149 | 0.065 |
| 0.024877 | 0.068310 | 0.241473 | 0.360710 | 0.293878 | 0.209136 | 0.244 |
| 0.012926 | 0.162538 | 0.390278 | 2.561694 | 0.400529 | 0.236939 | 0.253 |
| 0.000392 | 0.000000 | -0.027043 | 0.135815 | 0.024212 | -0.017902 | -0.027 |
| ... | ... | ... | ... | ... | ... | |
| 0.010702 | 0.099666 | 0.161427 | 0.021368 | 0.152731 | 0.121293 | 0.133 |
| 0.191973 | 0.254218 | -0.199658 | 0.410693 | 0.109333 | -0.207802 | -0.206 |
| 0.117414 | 0.107730 | 0.020390 | 0.848138 | 0.243941 | 0.040640 | 0.042 |
| 0.281483 | 0.226956 | -0.264234 | 0.334632 | 0.183915 | -0.266924 | -0.261 |
| 0.010703 | 0.073003 | 0.287700 | 1.069767 | 0.286741 | 0.239617 | 0.287 |

Target variable is created and removed collinear features to target variable (features which use target variable to calculate its own value).

In [337]:
```
target = df_final['shareGrowthAboveTwenty']
features = df_final.drop([
    'shareGrowth', 'shareGrowthAboveTwenty',
    'stockBasedCompensationToRevenue',
    'operatingCashFlowSalesRatio',
], axis=1)
features.shape
```

After that, collinear features are removed using `drop()` function while creating target variable.

In this way, features are selected and engineered for the particular machine learning model.

Here, feature engineering reveals the highly correlated independent features that were correlated with each other and made multicollinearity which could destroy the model performance as they were showing volatility.

In hypothetically, feature engineering is utmostly related to both machine learning engineers and decision-makers. Because, during this operation, if machine learning engineers do any mistake in selecting the right features or picking collinear features without knowing the causality than it will result the poor model performance or poor ethical outcomes which would be vulnerable for decision-maker to take the valuable decision.

## Model selection:

`GradientBoostingClassifier` model is selected for final solution. It is precisely indicated from the given table that GradientBoostingClassifier's F1 -score and accuracy is higher than ` RandomForestClassifier` and ` Votingclassifier` F1 -score and accuracy.

| Model Type | Precision | Recall | F1-Score | Accuracy |
|---|---|---|---|---|
| RandomForestClassifier | 0.64 | 0.49 | 0.56 | 0.71 |
| GradientBoostingClassifier | 0.64 | 0.56 | 0.60 | 0.72 |

| | | | | |
|---|---|---|---|---|
| Votingclassifier | 0.62 | 0.39 | 0.48 | 0.68 |

**Classification report for RandomForest Classifier:**

```
In [96]: print(classification_report(y_test, random_clf.predict(X_test)))
```

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.73 | 0.83 | 0.78 | 102 |
| 1 | 0.64 | 0.49 | 0.56 | 61 |
| accuracy | | | 0.71 | 163 |
| macro avg | 0.69 | 0.66 | 0.67 | 163 |
| weighted avg | 0.70 | 0.71 | 0.70 | 163 |

**Classification report for GradientBoosting Classifier:**

```
In [111]: print(classification_report(y_test, GradBoost_clf.predict(X_test)))
```

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.75 | 0.81 | 0.78 | 102 |
| 1 | 0.64 | 0.56 | 0.60 | 61 |
| accuracy | | | 0.72 | 163 |
| macro avg | 0.70 | 0.69 | 0.69 | 163 |
| weighted avg | 0.71 | 0.72 | 0.71 | 163 |

**Classification report for Voting Classifier:**

```
In [118]: print(classification_report(y_test, votingClassifier.predict(X_test)))
```

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.70 | 0.85 | 0.77 | 102 |
| 1 | 0.62 | 0.39 | 0.48 | 61 |
| accuracy | | | 0.68 | 163 |
| macro avg | 0.66 | 0.62 | 0.62 | 163 |
| weighted avg | 0.67 | 0.68 | 0.66 | 163 |

Here, metric selection can help to fulfill the business requirements as well as to find the right model by reducing the risk of investment. That means, the purpose is:

- Predicting True Positive (TP) more
- Reducing False Positive (FP) more.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F\text{-}measure = \frac{2*Recall*Precision}{Recall + Precision}$$

As RandomForest and Voting Classifier is showing low recall which means the hedge fund is missing out on good investments. Therefore, F1-score will be the most important metric for evaluating the performance of the GradientBoosting Classifier. Because F1-score or F-measure is in combination of precision (predicted values) and recall (actual values) where both deals with true positive compared to other metric.

## Hyperparameter selection:

- Criterion calculates the mean squared errors with improvement score by Friedman,
- max_depth = 1 refers the maximum depth limit and best performance of the tree,
- max_features = 'log2' refers the number of features that splitting best,
- loss = 'exponential' specifically for Gradient Boosting Algorithm that provides the 'probabilistic outputs', and
- learning_rate = 1.0 implies the compression of each tree [2].

```
GradBoost_clf = GradientBoostingClassifier(criterion='friedman_mse',
                                   max_depth=1,
                                   max_features='log2',
                                   loss = 'exponential',
                                   learning_rate= 1.0,random_state=0
                                   )
```

At this moment, this GradientBoosting Classifier algorithm would be used to suggest a hedge-fund manager makes specific investment. To validate this algorithm, some operational stages have been performed and these stages are:

- Demonstrate exploratory data analysis using heat map visualization to find the strong correlated features for target variable.
- Engineered features to remove the multicollinearity and extend the relationship between target variable and independent features.
- Tuned hyperparameter by following two techniques such GridsearchCV and RandomisedSearchCV

After completing all stages, it is considered that this model is reliable to predict the share price above the threshold value which would be more than 20% in a year.

Apart from this, I would use this model to select a portfolio of investment rather than a single investment.

In statistically, the final performance of my model was finding the absolute value using the lambda function.

I have achieved this performance by applying the `sorted()` function including reverse or iterate the values.

When I used the `reverse = True` parameter, the performance improved over multiple iterations and obtained top five features relevant with target variable.

```
In [374]: labels = features.columns.values
          weights = random_clf.feature_importances_
          top_features = sorted(list(zip(labels,weights)), reverse=True, key = lambda x: abs(x[1]))[0:5] # # Top five features have taken u
          top_labels = [x[0] for x in top_features] # for loop to plot the top n features
          top_weights = [x[1] for x in top_features]
          pp.pprint(top_labels)
          pp.pprint(top_weights) # `pprint()` explicitly notifies the recursion and also adds the ID of the dictionary.

          [   'researchAndDdevelopementToRevenue',
              'cashRatio',
              'currentRatio',
              'operatingProfitMargin',
              'pretaxProfitMargin']
          [   0.2956835031048366,
              0.29306527646598907,
              0.20780896832345835,
              0.0784902824956015,
              0.04579133188782326]
```

I would explain the performance of my model to a decision-maker by translating the technical language to non-technical language which is given in the jupyter notebook. Such as:

F1 -score would implies as a proportional ratio between accuracy and recall that seeks to balance both,

Accuracy would refer the proportion of all predictions that were correct.

Precision would refer the proportion of predicted positives that were correct.

Recall implies the proportion of actual positives that were correct.

RandomForest would refer randomly distributed trees,

GradientBoosting would implies the best possible next model, when combined with previous models, minimizes the overall prediction error.

A voting classifier needs a vote from another estimator or model.

The potential impacts of this performance are obtaining the F1–score of this model and getting the best features that would help to predict the share price growth accurately for a company’s good investment.

In the real world, a decision-maker might to know some additional information before making decisions. These are:

- Deficiencies and limitations of the machine learning model.
- Benefits and losses before using the machine learning model.
- What would be the risk in terms of using the model, data, or decision-making process?
- What is the likelihood of risk occurring?
- How we will mitigate the risk to improve the model performance?

I could improve my model by adding more external or historical data into the model.

I might improve my model performance in the real world by adding or subtracting the data or features from the model for increasing its accuracy and reliability to the decision-makers. Apart from this, I would use some statistical analysis and validation techniques to increase my model performance adequately.

In considering data, data leakage and data privacy issues would impact the performance or usability of the models.

Regarding ethics, privacy and long-term sustainability of the solution, I would consider to,

- manage the data privacy by following the European Union General Data Protection Regulation (GDPR) to avoid the massive fine that occurs in breaching the regulations.
- Increase the robustness which refers the machine learning model's ability for continuing to provide the future guidance.
- Addressing the accountability that implies the understanding, documenting of the machine learning and decision-making process.

Initially, I formed my anticipation regarding the model's performance that will meet the requirements of the hedge funds by eliminating the unneeded costs and predicting the future performance of U.S. stocks.

In my analysis, it is proved that the future performance of U.S. stocks will be good as I have used most relevant predictable features with the target feature `shareGrowth` that will help to predict the share price for a company over 20% in a year.

I believe this knowledge will be beneficial for my decision-maker as they were looking for a machine learning model that will help them to predict the share price above the threshold value 0.2.

## References:

[1] Lannge E J (2021) ‘Does removal of correlated variables affect the classification accuracy of machine learning algorithms?’, *Bachelor’s thesis in Statistics*, 1-18.

[2] Scikit-learn (2023) *sklearn.model_selection.RandomizedSearchCV,* Scikit-learn website, accessed 10 April 2023. sklearn.model_selection.RandomizedSearchCV — scikit-learn 1.2.2 documentation

[3] Zheng H and Wu Y (2019) ‘A XGBoost Model with Weather Similarity Analysis and Feature Engineering for Short-Term Wind Power Forecasting’, *applied sciences*, 9(3019): 2-12, doi:10.3390/app9153019