# Gruppenarbeit: Vorhersage einer «buy, hold, sell» - Strategie

**Gruppe 1**

Tim Ehrensperger, Marco Heiniger, Pascal Huser, Marc Tschudi

# Inhaltsverzeichnis

# Data Preprocessing I

## NaN and Zero Values

- Remove features with a percentage of NaN and Zero Values over thresholds of 50% and 40% respectively

## Outliers

- Handling per sector per year
- Replace data points above or below a certain quantile with the respective quantile value

## Imputing Missing Values

- Last preprocessing step before splitting the data set into training and test sets
- Per sector per year
- Impute NaN values with median

# Data Preprocessing II

## Train/Test split

- Played with different thresholds
- Decided to go with a 70/30-split for our final run

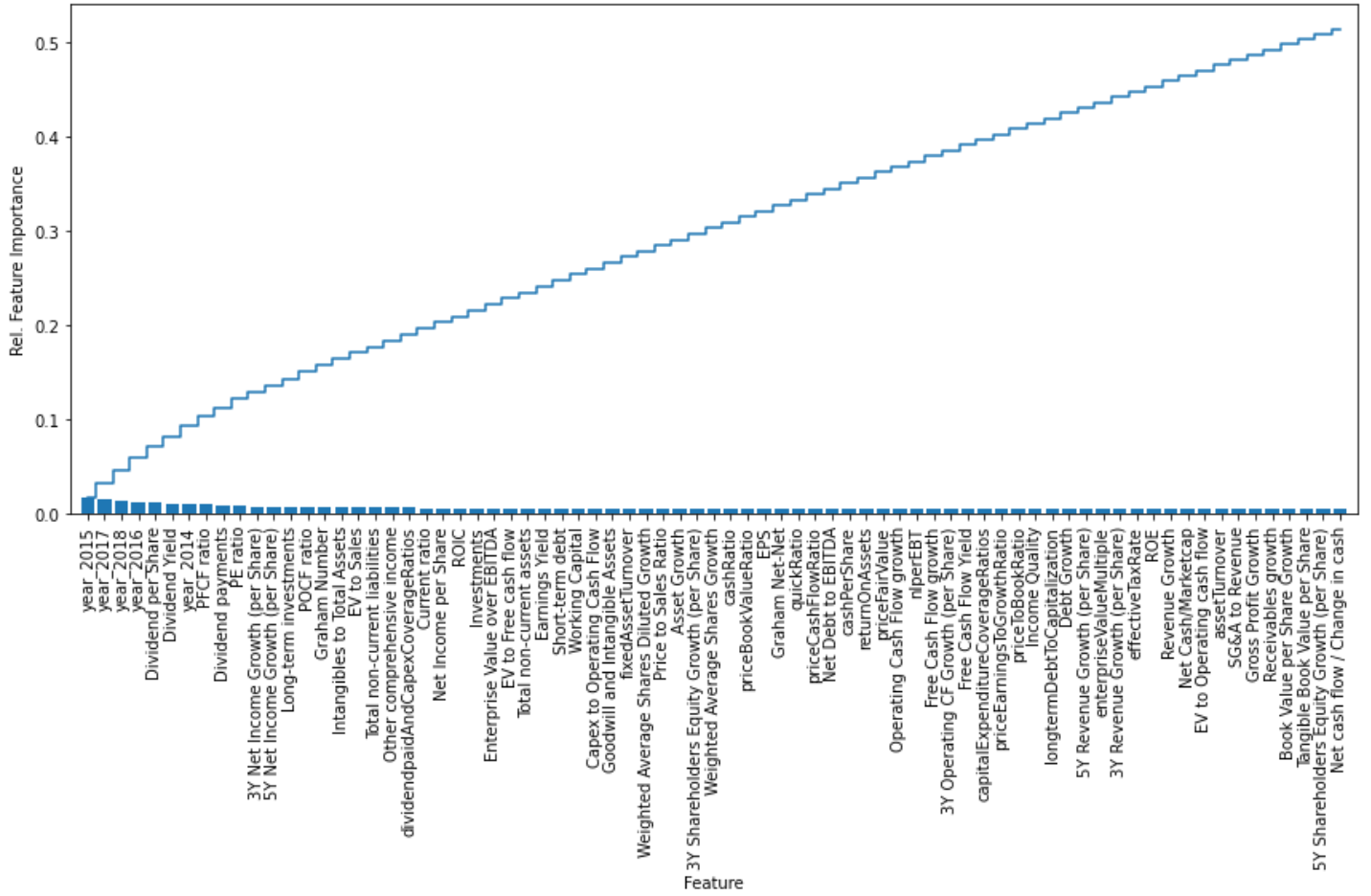
## Class Imbalance

- Balance after the train/test split on the train set → test set does not contain artificial data which happens while balancing
- K-nearest neighbors = 10

## Feature Selection

- Select all features with at least 0.55% relative feature importance → 81 features left
- Comes out that years and sectors are very important features

# Feature Selection



# Algorithms

| Random Forest / Decision Tree | SVM | LDA / QDA | Keras Sequential Model |
|---|---|---|---|
| - Intuitive classification process<br>- Flexible<br>- Two of the most used algorithms for classification problems | - Rather disappointing score<br>- Runtimes went into hours → not 100 percent sure if we have found the best parameters<br>- Unlikely that the score would improve onto the level of Random Forest | - Dimensioality reduction technique<br>- Reduces high-dimensional data sets onto a lower-dimensional space<br>- Less computational costs because of lower dimension<br>- LDA assumption of a common covariance matrix | - Deep learning framework<br>- Most used framework among top-5 winning teams on Kaggle<br>- Layer-by-Layer model creation<br>- Three layers resulted in the best score |

# Algorithms – Results

| Method | Best Score |
|---|---|
| Random Forest | 61.44% |
| Decision Tree | 60.39% |
| LDA and QDA | 57.58% |
| Keras Neural Network | 54.60% |
| Support Vector Machine | 41.09% |

# Conclusion

- At first, we asked ourselve what accuracy score we would be aiming for → 65-75%
- Just predicting «sell» would have resulted in a score of 53% → we managed to push the score up by prediting «buy» and «hold» in an acceptable manner
- We are convinced that if we woud get real world data we would achieve similar scores → we didn't artificially improve our test score by removing too many outliers
- More computational power would be useful for applying the GridSearch algorithm and trying out different threshold levels for the algorithms → thershold sensitivity as room for improvement
- Data preprocessing is a very important step and with a looser time restriction different ways of handling a task can be deeper investigated
- Feature selection as an important component