MGT-432 Data Science For Business

Initial Public Offering

Professor

Kenneth Younge

Group 7

Arthur Gassner
Zeineb Sahnoun
Mathieu Shiva

TAs

Maximilian Hofer
George Abi Younes

Introduction

1. **Problem statement in business:**

- The offering price of an IPO is the price at which the company sells shares to initial investors.
- The price at which shares eventually trade at the end of the first day determines if the offering price was underpriced.
- An accurate offering price is important for all stakeholders involved in an IPO.

2. **Stakeholders:**

- Issuing companies: an accurate prediction maximizes the amount of capital raised from the IPO
- Investment banks: an accurate prediction keeps the bank's reputation by better serving the clients
- Investors: an accurate prediction helps evaluate the investment opportunity and the the risk associated

3. Problem statement in analysis:

Given offering price and other business, management and financial data related to an IPO, we want to develop machine learning classifiers and regressors to predict trading price at the end of the first day.

4. Important metrics:

Depends on the prediction at hand:
As an example, P1: *“Will closing price at the end of the first day go up?”:*

- From the point of view of investors a FP is very costly since they would lose their investment. A FN is a lost opportunity. Precision is therefore more important than recall, and we use a F-beta score for this case.
- From the point of view of the investment bank, a FP has the same weight as a FN, but typically underpricing is most common (the first-day appreciation in the stock price is viewed as a "risk premium")
- From the point of view of the issuing company, a FN is very costly because they will lose funds they could have collected. On the other hand a FP is profitable to them. In this case Recall is more important than Precision.

Processing outline

Goal: Develop models for different predictions on the first day closing price of an IPO.

Main steps:

- **Data cleaning**
  - Correlated features
  - NaN values
- **Data preprocessing**
  - Feature engineering
- **ML models**

Text Preprocessing and Feature Engineering

Managers:

- **Separate** the managers (The managers are separated by \n)
- Compute the **average of the success rates** of each manager and store it as “managerSuccessAvg”

Risk factors:

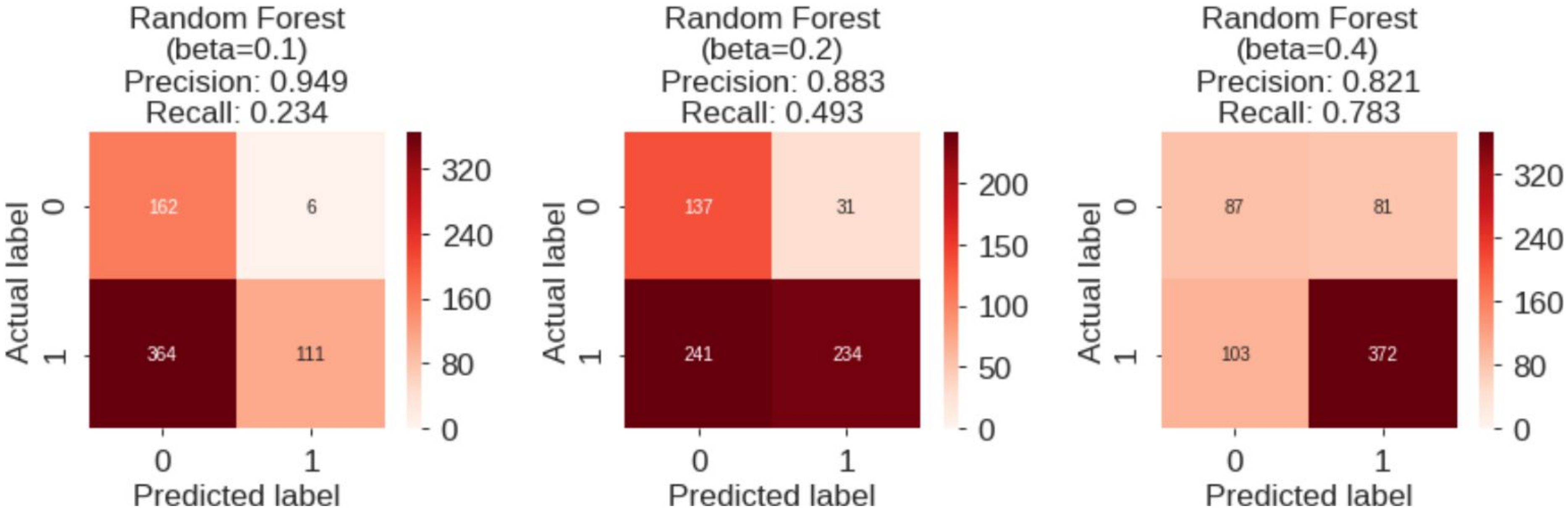
- Clean punctuation, remove stopwords, lemmatize, lowercase
- Create a corpus where each document is a risk factor entry
- Compute the TF-IDF representation of each of those documents
- Compute the **cosine-similarity** between **each** document's TF-IDF representation and **all** the TF-IDF representations of the **successful** IPO
- Compute and store the **average success rate** of the 100, 10 and 1 **most similar risk factor**

Predictions 1, …, 5 Methodology

1. Choose some **ML models** to try : KNN, Regularized Logit, Random Forest
2. Cross-validate on the training set to **tune the hyper-parameters**
3. Choose the hyper-parameter which gives the **greatest F1-score**
4. **Compare** the trained models performance on the test set
5. Choose the model which has the **highest AUC score**
6. Cross-validate on the training set to **pick the threshold** which maximizes the Fbeta-score

P1 Best Model: Random Forest, 700 trees

Confusion matrix comparisions between our best model with different betas



Random Forest (beta=0.1) Precision: 0.949 Recall: 0.234

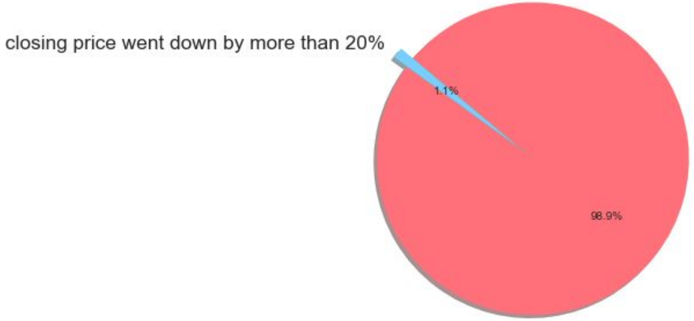
Random Forest (beta=0.2) Precision: 0.883 Recall: 0.493

Random Forest (beta=0.4) Precision: 0.821 Recall: 0.783

Results P1 -> P5

Prediction	Best Model	Best Hyper-parameter	AUC	Threshold
P1	RF	700 trees	0.73	0.76
P2	KNN	51 neighbors	0.62	0.74
P3	RF	700 trees	0.74	0.8
P4	RF	700 trees	0.81	0.61
P5	Balanced RF	31 trees	0.73	0.5

P5 : Unbalanced data



Problem

- Only 1% of labels classified as 1 (36 data points out of 3214)

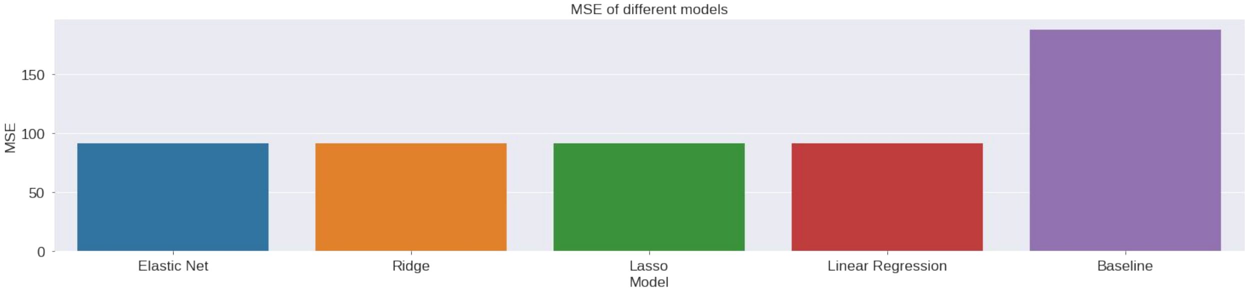
Trials

- Undersampling: 72 data points are not enough for training
- Oversampling: Add synthetic data points from under-represented class with built-in noise
- **Balanced Random Forest** from imbalanced-learn library worked best

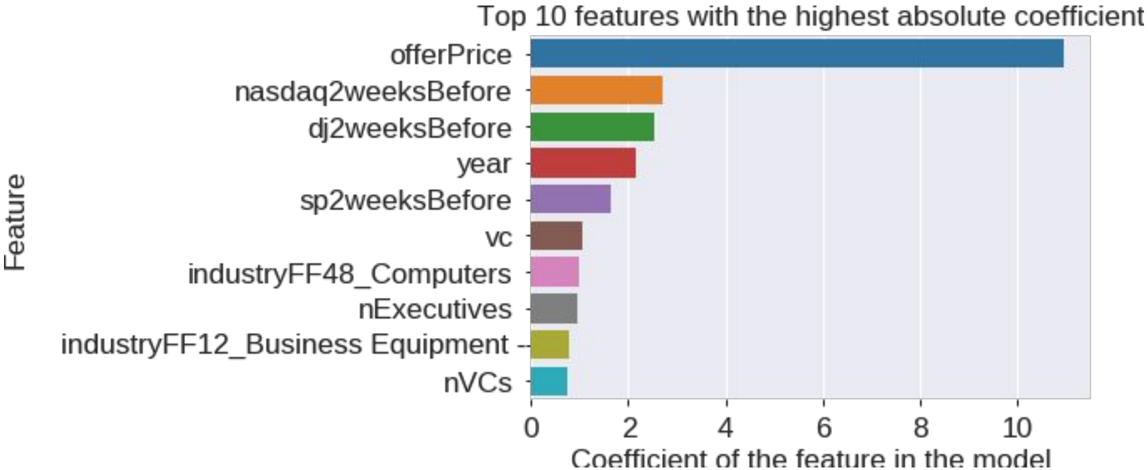
Methodology Prediction 6

1. Choose some **ML models** to try : **linear**, **lasso**, **ridge regression** and **Elastic Net**
2. Cross-validate on the training set to **tune the hyper-parameters**
3. Choose the hyper-parameter which gives the **lowest Mean Squared Error**
4. **Compare** the trained models performance on the test set
5. Choose the model which has the **lowest MSE**

P6 Best Model: Elastic Net, alpha of 0.01



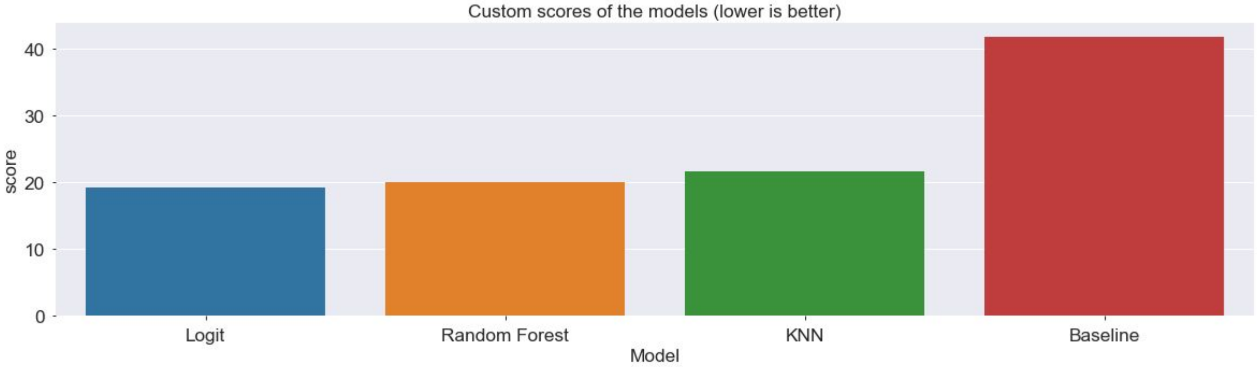
Baseline: predict the mean



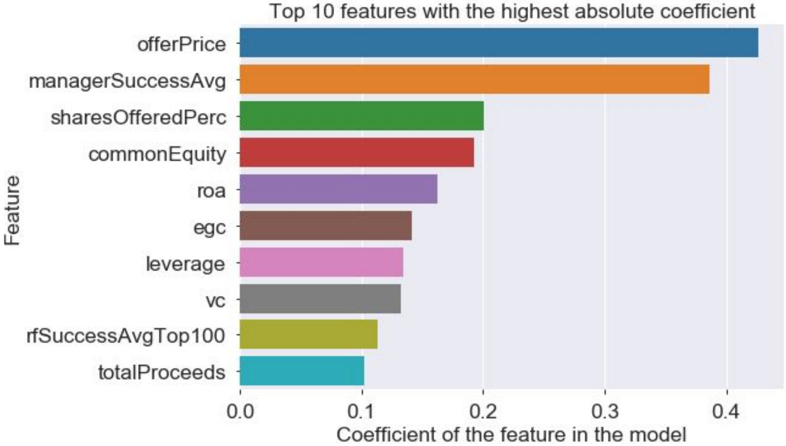
Methodology Prediction 7, 8, 9

1. Choose some **ML models** to try : **KNN**, **regularized logit**, **random forest**
2. Cross-validate on the training set to **tune the hyper-parameters**
3. Choose the hyper-parameter which gives the **lowest custom score**
4. **Compare** the trained models performance on the test set
5. Choose the model which has the **lowest custom score**

P7 Best Model: Regularized Logit, C of 0.01



Baseline: always predict the majority class label



Results P7->P9

Prediction	Best model	Hyper parameter	Custom score	Baseline custom score
P7	Regularized Logit	C = 0.01	19.3	41.7
P8	Random Forest	40 trees	8.2	13.7
P9	KNN	10 neighbors	12.1	9

Thank you for you attention !

Any Questions ?

Appendix: Trial and Error performed

1. Perform polynomial augmentation on the given features and add them as new features.
2. Drop features that have correlation larger to 0.9 to some other feature.
3. Drop features that have high rate of missing values.
4. Use xgboost classifier with unbalanced datasets.
5. Undersampling/Oversampling of unbalanced datasets.