University of California San Diego
Course: GPIM452 Big Data Analytics

Predicting Stock Profitability

with **Machine Learning** in **Python**

Author: Amy Huang

Date: June 2020

Problem Description

- Currently over 3,671 publicly traded stocks
- Average rate of return for Nasdaq stock: 6%
- Perfect Markets Theory
  → no way to accurately predict future value of stocks
- Goal: Find out what stocks or industries generate positive ROI

Financial Dataset

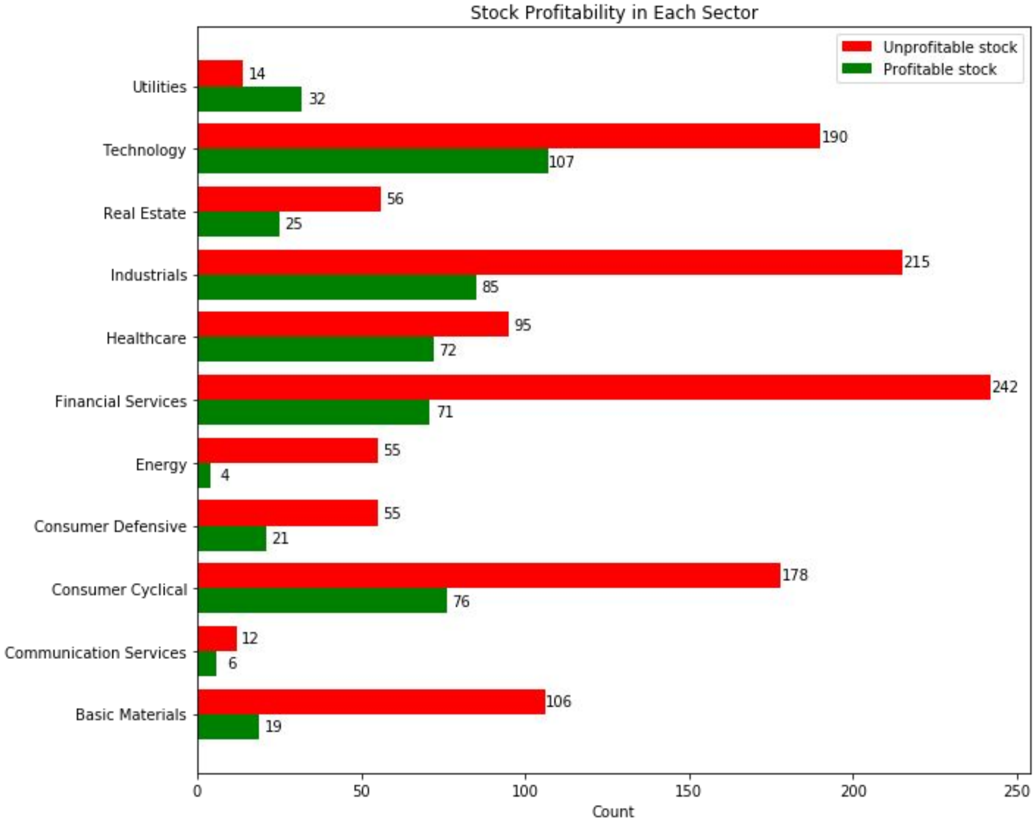
- 5 datasets that contain **224** financial indicators of 4,000+ stocks from **2014** to **2018**.
- Assuming **Price variance (VAR)** determines profitability
  - Positive Price VAR → Profitable
  - Negative Price VAR → Unprofitable
- Binary outcome variable
  - **Class = 1** if Price VAR > 0
  - **Class = 0** if Price VAR ≤ 0

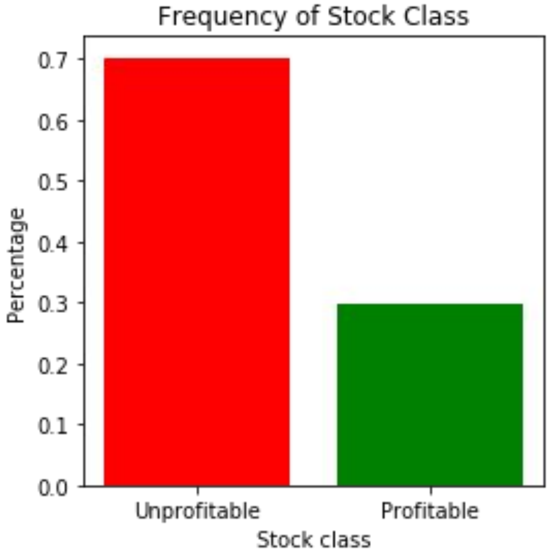
Year	# of stocks/rows
2014	3808
2015	4120
2016	4797
2017	4960
2018	4392

Analysis task

- Use 2017 data set to train and test the ML model and ran it on 2018 data
- Supervised learning using **scikit-learn package:**
  Use **decision tree classifier** to solve binary classification problem
- Explanatory variable: financial indicators
- Outcome variable: stock class

2017 Financial Dataset Overview





Pre-analysis: Challenges & Solutions

Challenges	Solutions
Duplicate columns with same values and/or different numbers of nulls	Omit columns with same values and keep columns with fewer nulls
Extreme values/outliers	Omit entries with any values below 1st percentile or above 99th percentile
Large amount of missing values (each entry has at least 1 nulls)	Remove columns with above average number of nulls and then remove entries with any nulls
Numerical columns are on different scales	Normalization based on Z-transform
Categorical information (Sector)	One-hot encoding
Large amount of features	Dimensionality reduction using feature importance score by scikit-learn
Effect of sectors	Construct two models and compare results

Analysis Roadmap

Data cleaning

1. Drop duplicate columns
   (columns with same values & columns with same names)
2. Drop columns with 1,000+ missing values
   (unrepresentative indicators)
3. Omit rows with any missing values
4. Omit rows with any values below 1st percentile and above 99th percentile

Preprocessing

1. Dimensionality reduction based on feature importance
   (# of features: 141 → 10)
2. Scale numerical columns with z-transformation
3. One-hot encode categorical columns (Sector)
4. Merge normalized numerical columns, one-hot columns, and outcome variable into one dataframe

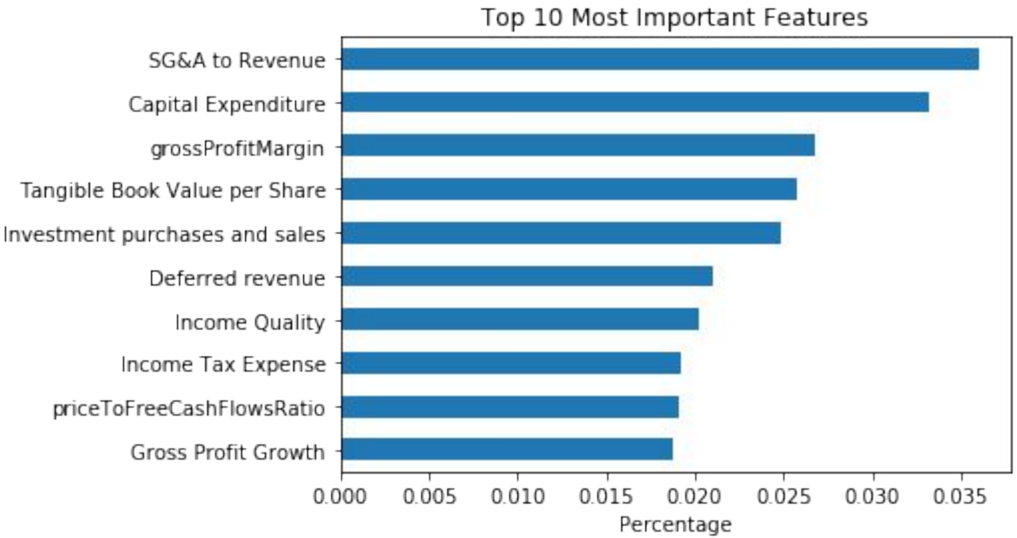
Analysis

1. Split data into train, test, and validation sets
2. Construct decision tree classifier
3. Address overfitting issue by validating the maximum depth of the decision tree
4. Hyperparameter tuning
5. Accuracy report and model improvement

Preprocessing

Determining Top 10 Important Features

- Used Decision Tree Classifier from scikit-learn package
    - Label: Class
    - Features: all financial indicators except Price VAR and Sector
    - Random state = 120
    - Class weight = balanced



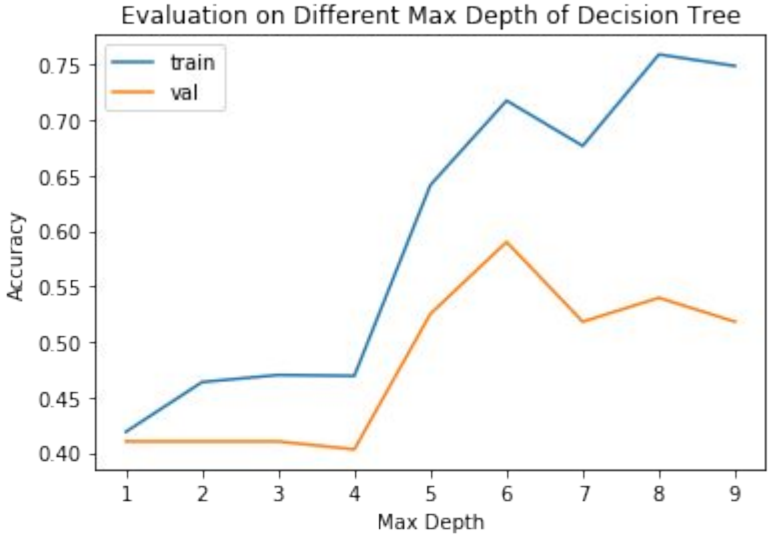
Analysis I - Splitting datasets

- Splitting data into train, test, and validation datasets
  - Test size: 20%
  - Validation size: 10% of the remaining data
- Shuffle the result to prevent the order of data from biasing the analysis outcome

Training samples: 1249
Validation samples: 139
Test samples: 348

Analysis II - Deciding Maximum Depth

Accuracy of validation dataset starts falling below its highest number (59%) once tree depth goes over **6**. This means the ability of generalization decreases when the tree splits more than 6 times.



Analysis III - Hyperparameter tuning

- Used GridSearchCV to perform 5-fold cross validation

```
GridSearchCV(cv=5, error_score='raise-deprecating',
          estimator=DecisionTreeClassifier(class_weight=None,
                                           criterion='gini', max_depth=None,
                                           max_features=None,
                                           max_leaf_nodes=None,
                                           min_impurity_decrease=0.0,
                                           min_impurity_split=None,
                                           min_samples_leaf=1,
                                           min_samples_split=2,
                                           min_weight_fraction_leaf=0.0,
                                           presort=False, random_state=120,
                                           splitter='best'),
          iid='warn', n_jobs=-1,
          param_grid={'class_weight': ['balanced'],
                      'criterion': ['gini', 'entropy'],
                      'max_depth': [1, 2, 3, 4, 5, 6],
                      'min_samples_split': [2, 3, 4, 5, 6, 7, 8, 9]},
          pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
          scoring=None, verbose=0)
```

Results - Best Estimators

- Model I: Important features & Sector

```
DecisionTreeClassifier(class_weight='balanced', criterion='entropy',
                       max_depth=3, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=9,
                       min_weight_fraction_leaf=0.0, presort=False,
                       random_state=120, splitter='best')
```

- Model II: Without Sector

```
DecisionTreeClassifier(class_weight='balanced', criterion='gini', max_depth=2,
                       max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort=False,
                       random_state=120, splitter='best')
```

Results - Accuracy Report

- Model I: Important features & Sector
  - Train accuracy: 0.69
  - Test accuracy: 0.65

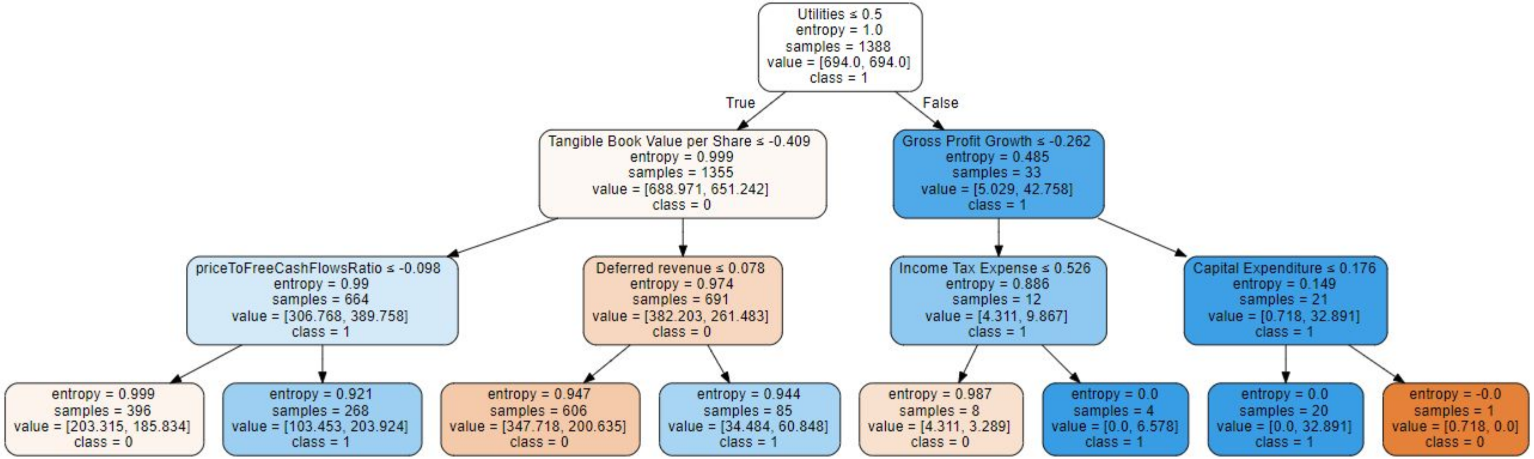
	precision	recall	f1-score	support
0	0.75	0.78	0.76	252
1	0.36	0.32	0.34	96
accuracy			0.65	348
macro avg	0.55	0.55	0.55	348
weighted avg	0.64	0.65	0.65	348

- Model II: Without Sector
  - Train accuracy: 0.64
  - Test accuracy: 0.63

	precision	recall	f1-score	support
0	0.76	0.72	0.74	252
1	0.35	0.39	0.36	96
accuracy			0.63	348
macro avg	0.55	0.55	0.55	348
weighted avg	0.64	0.63	0.64	348

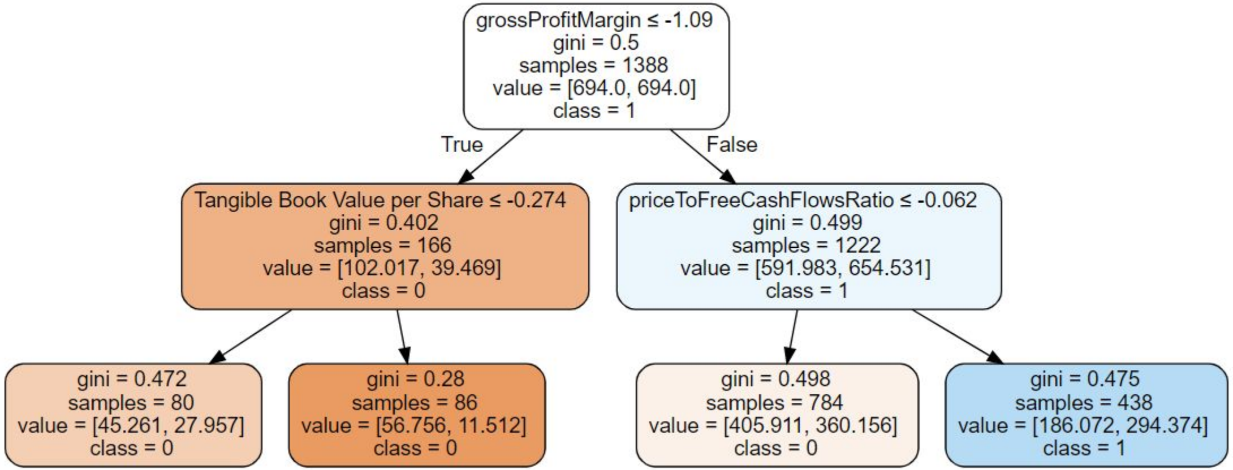
Results - Decision Trees

- Model I: Important features & Sector
  - The Utilities sector is determining factor, followed by tangible book value per share, and gross profit growth



Results - Decision Trees

- Model II: Without Sector
    - Gross profit margin becomes the determining factor, followed by tangible book value per share and price to free cash flows ratio



Application on 2018 data - Accuracy Report

- Model I: Important features & Sector

	precision	recall	f1-score	support
0	0.25	0.74	0.37	439
1	0.76	0.27	0.40	1330
accuracy			0.39	1769
macro avg	0.50	0.50	0.39	1769
weighted avg	0.63	0.39	0.39	1769

- Model II: Without Sector

	precision	recall	f1-score	support
0	0.26	0.73	0.39	439
1	0.78	0.32	0.45	1330
accuracy			0.42	1769
macro avg	0.52	0.53	0.42	1769
weighted avg	0.65	0.42	0.44	1769

Conclusions

- Model II has stronger ability to generalize on other datasets
- Top features to determine the success of a stock:
  - Model II
    - Gross profit margin
    - Tangible book value per share
    - Price to free cash flows ratio
- Stock market unpredictability explains the difficulty to accurately predict stock profitability

Thank you.