edX ‘Choose your own’ - Predicting dividend policy

Estanislao Maria Ferrer

08/20/2020

I. Introduction

Over the past few years financial literature has become quite popular, and disciplines like financial engineering and algorithmic trading has gained the attention of many traders and researchers: the idea of predicting future market fluctuations has become the number one priority among them. Furthermore, a vast amount of techniques has been discussed in both the private sector and the academic field, like predicting stock market fluctuations via **Dynamic Mode Decomposition** (Mann et al, 2016), **Pairs trading strategies** (Rad et al, 2016), etc. In fact Kakushadze et al (2018) has summarized the most popular trading strategies in a paper called **151 trading strategies**.

Even though there is few literature regarding **dividend policy** -one of the most important concepts in investments decisions-, Dewasiri et al (2019) has provided a starting point in their latest research **Determinants of dividend policy: evidence from an emerging and developing market** by focusing on finding which fundamental variables (such as free cash flow) are the most important ones for predicting what they call “Propensity to pay dividends”.

Unlike Dewasiri et al (2019), who performed a Logistic Regression on a 191 observations dataset to understand which variables matter the most, the following project will aim towards predicting when will a company meet the adequate conditions to start paying dividends, by applying more advanced techniques on a 1485 observations dataset extracted from _https://finviz.com/screener.ashx_, in order to provide a useful tool for investors when deciding their portfolios composition.

A. Overview

Once the main dataset has been loaded, we will first study how does each variable behaves independently. In the second place we will try to predict whether a company pays dividends or not, starting with a simple model and concluding with more advanced algorithms. In the third place, the final model will be used to predict dividend payout on a validation dataset and results will be discussed.

I will now describe some important financial variables for those who are not familiarized with such concepts.

1. Price to earnings ratio (PER): Used for determining whether a company is undervalued or overvalued by dividing the actual price of the company’s share by earnings per share.
2. Price to earnings to growth ratio (PEG): Used for determining whether a company is undervalued or overvalued by dividing the PER ratio by earnings estimated growth.
3. Price to sales and price to free cash flow (P__Sales and P__CF): Just like PER and PEG this ratios are used for determining whether a company is undervalued or overvalued by dividing the actual price of the company’s share by sales revenue per share and company’s free cash flow per share respectively.
4. ROA, ROE and ROI: Return on assets, equity, and investments of the company.
5. Long term debt to equity and Total debt to equity(LTD__Eq and TD__Eq): Indicates how leveraged a company is.

6. Age: Difference in years since their initial public offering (IPO) and actual date.
7. Gross_M, Oper_M, and Profit_M: Gross margin, operating margin, and net profit margin.

Reader can find more detailed explanations at __https://www.investopedia.com/__.

B. Loading and splitting datasets

Even though data was extracted from finviz (https://finviz.com/screener.ashx), an "xlsx" file has been provided in a **github** public repository and in order to extract it, the following code must be executed:

```r
url <- "https://github.com/estFerrer/edx_choose_your_own/blob/master/finviz.xlsx?raw=true"
temp_file <- tempfile(fileext = ".xlsx")
req <- GET(url,
          write_disk(path = temp_file))

fin_info <- readxl::read_excel(temp_file)
```

Before any attempt to describe the newly imported dataset, a simple glimpse will reveal that some variables are not in the correct format:

```
## Rows: 1,757
## Columns: 19
## $ Ticker   <chr> "GE", "F", "HTZ", "AMD", "NCLH", "DAL", "M", "AAPL", "T", ...
## $ Sector   <chr> "Industrials", "Consumer Cyclical", "Industrials", "Techno...
## $ Industry <chr> "Specialty Industrial Machinery", "Auto Manufacturers", "R...
## $ PER      <chr> "17.32", "-", "-", "162.69999999999999", "-", "-", "-", "3...
## $ PEG      <chr> "-", "-", "-", "4.5199999999999996", "-", "-", "-", "2.81"...
## $ P_Sales  <dbl> 0.65, 0.21, 0.02, 12.84, 0.63, 0.54, 0.09, 7.36, 1.21, 11....
## $ P_Book   <dbl> 1.71, 0.91, 0.16, 29.02, 0.75, 2.10, 0.80, 27.45, 1.21, 13...
## $ P_CF     <dbl> 27.50, 3.72, 0.09, 160.71, 21.59, 17.64, 10.45, 34.97, 19....
## $ Dividend <dbl> 0.01, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.01, 0.07, 0.01...
## $ EPS      <dbl> 0.38, -0.52, -1.75, 0.50, -5.04, -5.83, -10.16, 13.16, 1.6...
## $ ROA      <dbl> -0.02, -0.01, -0.01, 0.10, 0.06, -0.05, -0.15, 0.18, 0.02,...
## $ ROE      <dbl> -0.16, -0.07, -0.18, 0.22, 0.16, -0.27, -0.59, 0.71, 0.07,...
## $ ROI      <dbl> 0.02, 0.01, 0.04, 0.12, 0.09, 0.20, 0.07, 0.27, 0.07, 0.23...
## $ LTD_Eq   <dbl> 2.16, 3.89, 13.67, 0.15, 1.93, 2.23, 1.82, 1.31, 0.87, 0.5...
## $ TD_Eq    <dbl> 2.43, 5.68, 13.67, 0.21, 1.97, 2.84, 2.10, 1.57, 0.96, 0.6...
## $ Gross_M  <dbl> 0.23, 0.04, 0.43, 0.44, 0.39, 0.48, 0.37, 0.38, 0.54, 0.68...
## $ Oper_M   <dbl> 0.14, -0.04, 0.06, 0.09, 0.18, -0.05, -0.15, 0.25, 0.14, 0...
## $ Profit_M <dbl> -0.06, -0.02, -0.03, 0.08, 0.15, -0.11, -0.14, 0.21, 0.07,...
## $ Age      <dbl> 128.17, 48.17, 4.17, 40.83, 7.58, 13.33, 28.50, 39.67, 36....
```

PER and PEG ratios, which should be doubles, are being computed as strings! This happens for a simple reason: whenever the website can't find the information needed to calculate a specific financial ratio, or when the outcome might result as a negative value, it replaces it with a "-". The following code should help us clean the data and keep those observations which present valuable information:

```r
# When both PER and PEG ratios are presented with a "-",
# it means that no ratio can be calculated due to negative earnings per share,
# so we must keep those observations because we might lose valuable information by removing them.
# On the other hand, when one of the values is properly calculated but the other not,
# the entire observation must be removed because it could mean
# that the website could not find proper information for that company.
fin_info$cleaning <- ifelse(fin_info$PER=="-"&fin_info$PEG=="-","Keep",
                     ifelse((fin_info$PER=="-"&fin_info$PEG!="-")|(fin_info$PER!="-"&fin_info$PEG=="-")

# Now we filter all the observations we want to keep:
fin_info <- fin_info %>% filter(cleaning=="Keep")
fin_info$PER[fin_info$PER=="-"] <- 0
fin_info$PEG[fin_info$PEG=="-"] <- 0

# And then we parse from character to numeric:
fin_info$PER <- as.numeric(fin_info$PER)
fin_info$PEG <- as.numeric(fin_info$PEG)
```

Now that all variables has been stored with their correct format, we will encode our dependent variable and add some new predictors. This is a very important step since the main goal is to predict whether a company will pay dividends or not in the future:

```r
main_ds <- fin_info %>%
  select(-c(Ticker,Industry,cleaning)) %>% # 1
  mutate(Sector = str_replace(Sector,"\\s","_"),
         E_growth = ifelse(is.na(PER/PEG)=="TRUE",0,(PER/PEG)/100), # 2
         payer = ifelse(Dividend==0,"NO","YES")) %>% # 3
  select(-c(Dividend,PER,PEG))

# 1: First we get rid of useless information,
# 2: We calculate earning's estimated growth by dividing PER ratio by PEG.
# 3: We encode our dependent variable into two categories: YES (company pays dividends) and NO.
```

Finally, we split the main dataset in two: **train_set** for optimizing models and **validation** for evaluating each model's precision. It is important to mention that due to lack of data the validation set will be used to evaluate each individual model, meaning that we wont be using a final hold-out test set.

According to **The Data Detective** in *The 80/20 Split Intuition and an Alternative Split Method*, a practical way to split datasets is to calculate the "validation to train ratio": dividing 1 by the square root of the number of predictors (link provided at references).

```r
ratio <- 1/sqrt(ncol(main_ds)-1)
set.seed(10,sample.kind = "Rounding")
validation_index <- createDataPartition(main_ds$payer, times=1, p=ratio, list=FALSE)

train_set <- main_ds %>% slice(-validation_index)
validation <- main_ds %>% slice(validation_index)
```

C. Description

The glimpse() function will provide an effective insight of our dataset:

```
## Rows: 1,101
## Columns: 16
## $ Sector   <chr> "Consumer_Cyclical", "Industrials", "Technology", "Consume...
## $ P_Sales  <dbl> 0.21, 0.02, 12.84, 0.63, 0.54, 0.09, 7.36, 1.21, 11.24, 1....
## $ P_Book   <dbl> 0.91, 0.16, 29.02, 0.75, 2.10, 0.80, 27.45, 1.21, 13.38, 1...
## $ P_CF     <dbl> 3.72, 0.09, 160.71, 21.59, 17.64, 10.45, 34.97, 19.14, 53....
## $ EPS      <dbl> -0.52, -1.75, 0.50, -5.04, -5.83, -10.16, 13.16, 1.64, 5.7...
## $ ROA      <dbl> -0.01, -0.01, 0.10, 0.06, -0.05, -0.15, 0.18, 0.02, 0.15, ...
## $ ROE      <dbl> -0.07, -0.18, 0.22, 0.16, -0.27, -0.59, 0.71, 0.07, 0.40, ...
## $ ROI      <dbl> 0.01, 0.04, 0.12, 0.09, 0.20, 0.07, 0.27, 0.07, 0.23, 0.02...
## $ LTD_Eq   <dbl> 3.89, 13.67, 0.15, 1.93, 2.23, 1.82, 1.31, 0.87, 0.58, 0.3...
## $ TD_Eq    <dbl> 5.68, 13.67, 0.21, 1.97, 2.84, 2.10, 1.57, 0.96, 0.62, 0.4...
## $ Gross_M  <dbl> 0.04, 0.43, 0.44, 0.39, 0.48, 0.37, 0.38, 0.54, 0.68, 0.38...
## $ Oper_M   <dbl> -0.04, 0.06, 0.09, 0.18, -0.05, -0.15, 0.25, 0.14, 0.37, 0...
## $ Profit_M <dbl> -0.02, -0.03, 0.08, 0.15, -0.11, -0.14, 0.21, 0.07, 0.31, ...
## $ Age      <dbl> 48.17, 4.17, 40.83, 7.58, 13.33, 28.50, 39.67, 36.75, 34.4...
## $ E_growth <dbl> 0.000000000, 0.000000000, 0.359955752, 0.000000000, 0.0000...
## $ payer    <chr> "NO", "NO", "NO", "NO", "NO", "NO", "YES", "YES", "YES", "...
```

We can see that train_set contains financial information for over 1000 companies, only two variables are factors ("payer" and "sector"), while the rest of them are doubles, which indicates that either numerical data should be factorized, or categorical variables will have to be encoded.

It is also important to mention that not all variables are stored in the same unit of measure. For example "G_Margin" is a percentage oscillating between 0 and 1, and "Age" is a positive number that ranges between 0.92 and 115.33 meaning that numerical predictors will have to be scaled in order to achieve robust models.

D. Normalizing datasets

Since scaling the "main_ds" dataset entirely and then proceed to splitting might result into including information from the "validation" dataset (which should remain unknown) to the "train_set", we must first calculate the scaling parameters on the "train_set" and then scale "validation" with those parameters.

```r
# Calculating scaling parameters with train_set
normParam <- preProcess(train_set)

# Scaling train_set:
norm_train <- predict(normParam, train_set)

# Scaling validation with train_set parameters:
norm_validation <- predict(normParam, validation)
```

II. Exploratory analysis

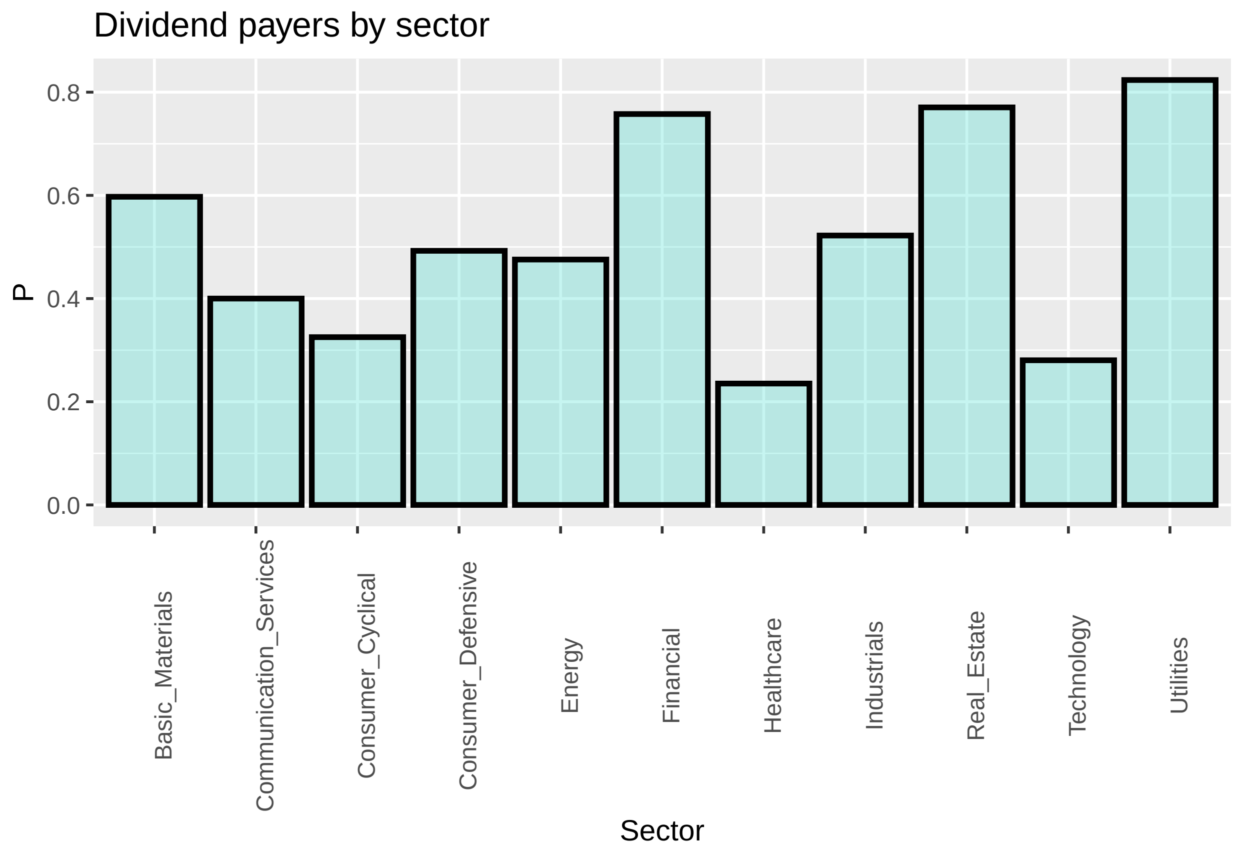
The following section will consist on studying the relationship between different predictors and our dependent variable.

Sector

Are there significant differences among sectors? Does the Utility sector presents a better and more stable environment for dividend policy than Technology sector?

```r
sector <- norm_train %>%
  group_by(Sector) %>% # 1
  summarize(P = mean(payer=="YES")) # 2
```

```r
# 1: First, we group all observations by each sector,
# 2: Then we compute for each sector the proportion of companies which pays dividends.

sector %>% ggplot(aes(Sector,P)) +
  geom_col(size=1,col="black",fill="turquoise",alpha=0.3) +
  theme(axis.text.x = element_text(angle = 90)) +
  ggtitle("Dividend payers by sector")
```

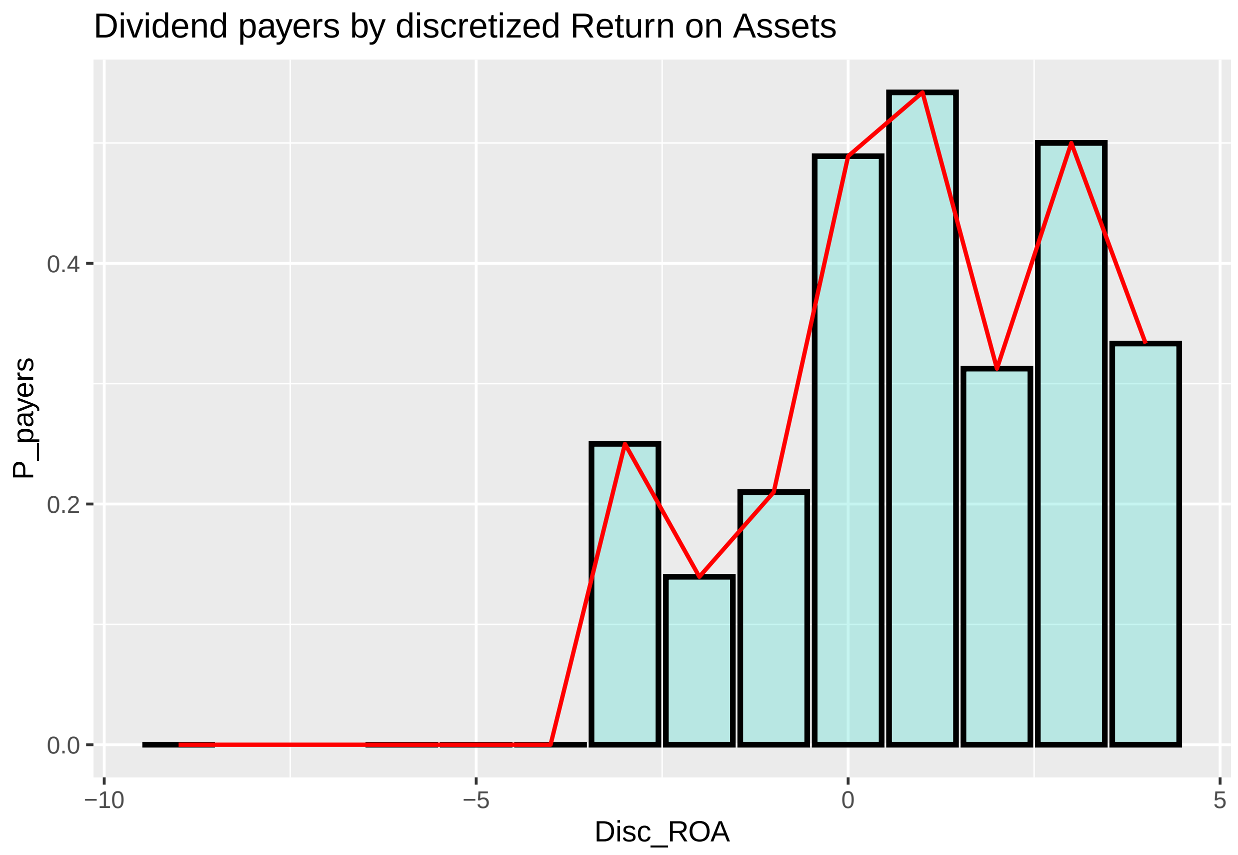


It seems that there is a significant variability among sectors being Financial, Real-estate, and Utility sectors the ones with higher proportion of companies which pays dividends to their share holders.

Return ratios

The same approach could be employed to understand how does return ratios impact dividend policy. We can take, for instance, return on assets (ROA), which can be defined as the return generated by all company's assets without taking into account it's financing sources (liabilities and equity):

```r
norm_train %>% mutate(Disc_ROA = round(ROA)) %>% # 1
  group_by(Disc_ROA)  %>% # 2
  summarize(P_payers = mean(payer=="YES")) %>%  # 3
  ggplot(aes(Disc_ROA, P_payers)) +
  geom_col(size=1,col="black",fill="turquoise",alpha=0.3) +
  geom_line(aes(Disc_ROA,P_payers),size=0.75,col="red", group=1) +
  ggtitle("Dividend payers by discretized Return on Assets") # 4
```

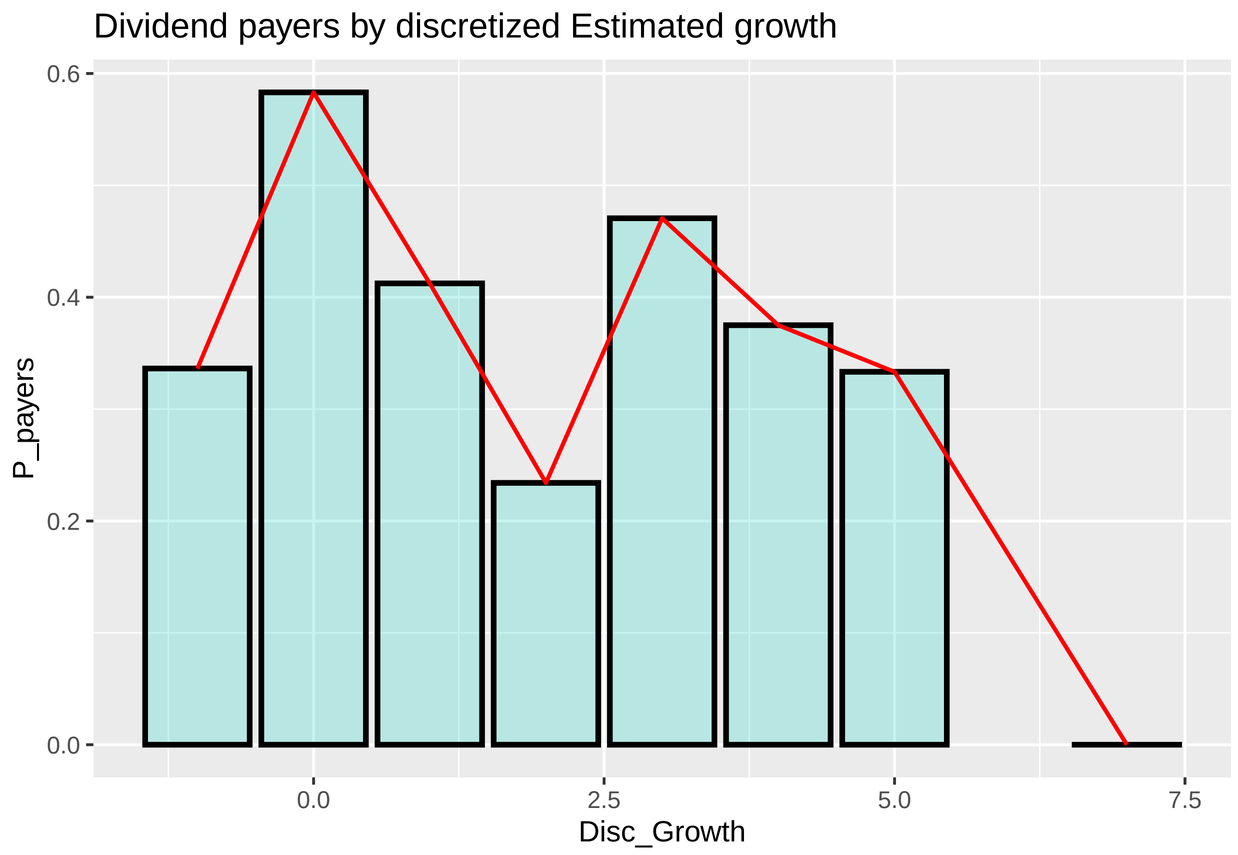


```
# 1: First we discretize the predictor. Because we have scaled all numeric variables,
#    we know that we can round each value to the next integer,
# 2: Then, we group by each discretized value,
# 3: After that, we compute the proportion of companies which pays dividends,
# 4: And we finally plot results
```

We can see that companies with higher return on assets are more likely to pay dividends.

Estimated growth

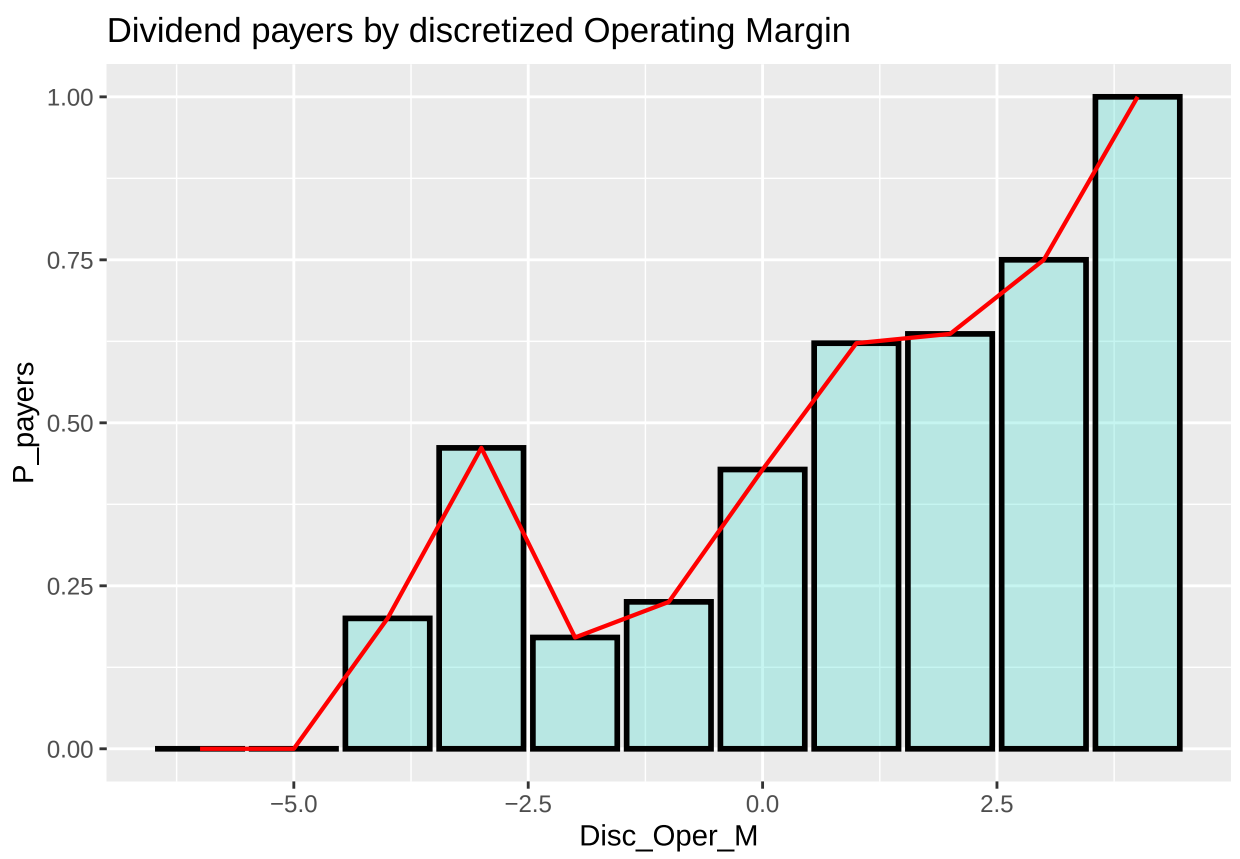
In **Section I - B** we used PER and PEG ratios to calculate Earnings estimated growth, and this is a very tricky indicator, since small businesses may have better perspectives than large ones because of the fact that a small **nominal** increase in sales may generate a big **proportional** movement in revenue in the same direction. Nevertheless, no company starts a dividend policy if their perspectives are not promising!



Data suggests that we were in fact, correct: enterprises with stable values of their estimated growth are more likely to pay dividends, and those companies with abnormally high values of estimated growth are not likely to pay dividends at all, since we might be in presence of new businesses.

Gross and net Profit margin, and Operating margin:

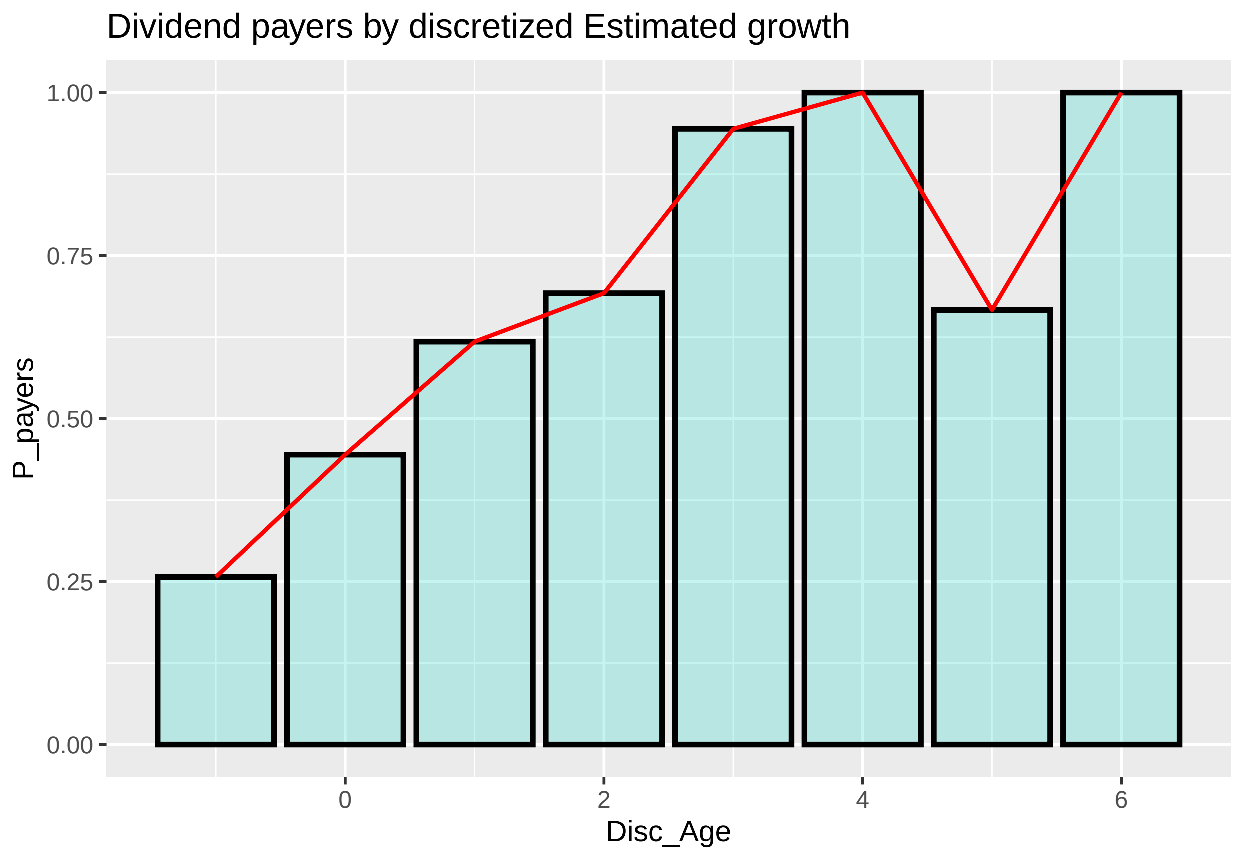
Operating margin indicates the percentage of revenue that represents profits after deducting variable costs of production:



Of course, companies with higher profit margins (specially operating and net profit margin) will be more likely to start a dividend policy.

Age:

One of the most important predictors is "Age", because the investor will not just care for "proper conditions" but he/she will also want to know "when" will a company start paying dividends.



Data suggests that there is indeed a clear relationship between a company's age and the probability it might start paying dividends!

III. Methodology: Building our predictive model

Even though most variables are pretty straight forward to understand, we have seen that there are some numerical predictors like earnings estimated growth (E_growth) that while being important, they are not linearly related to our dependent variable. Furthermore, we are also in presence of a categorical variable, "Sector".

Being said that, we will avoid conventional regressions that depends on clear relationships between predictors and the dependent variable (and other assumptions such as normal distribution of data) and will focus on those algorithms that can handle non-linear relationship between variables.

A. Encoding datasets by sector:

We have seen that one of our predictors is a categorical variable, and if we want to use any kind of technique rather than decision trees, we must encode this variable on both training and validation datasets:

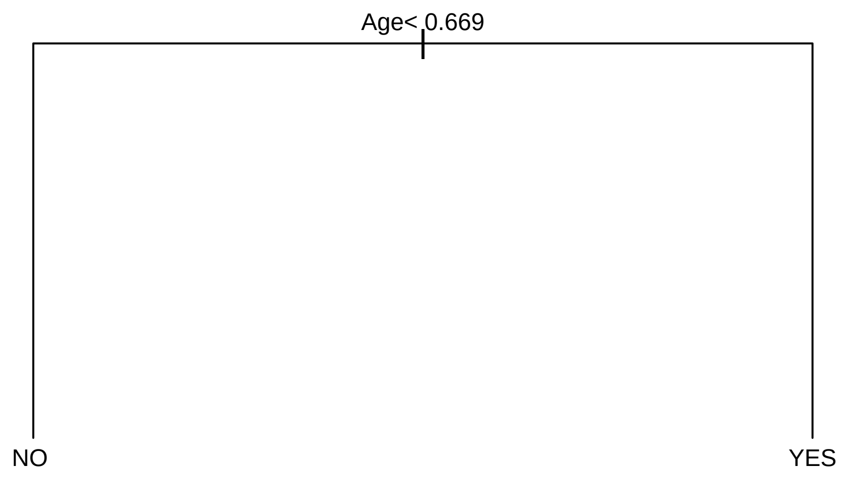
```r
norm_train <- norm_train %>%
  mutate(Sector = paste("Sec",Sector,sep="_"), Sec=1) %>%
  spread(key = Sector, value = Sec, fill = 0)

norm_validation <- norm_validation %>%
  mutate(Sector = paste("Sec",Sector,sep="_"), Sec=1) %>%
  spread(key = Sector, value = Sec, fill = 0)
```

Basically, we have created as many columns as different sectors, and each company will be assigned a 1 for its corresponding sector, and a 0 for the rest of them. The following table should be a clear example of an encoded dataset (not all sectors are shown):

```
## # A tibble: 5 x 5
##   Sec_Healthcare Sec_Industrials Sec_Real_Estate Sec_Technology Sec_Utilities
##            <dbl>           <dbl>           <dbl>          <dbl>         <dbl>
## 1              0               1               0              0             0
## 2              0               0               0              1             0
## 3              1               0               0              0             0
## 4              0               0               0              0             0
## 5              0               0               0              0             1
```

B. First stage: Naive approximation

We will first try to predict dividend policy with a simple model. Since we have both numerical and categorical predictors we can make use of a simple decision tree:



We can see that our first approach is not satisfactory at all, since the only variable used was Age, ignoring key financial ratios! In fact, if we evaluate the model with the confusionMatrix() function from the caret package we will see that results should be improved:

```
##                         Accuracy
## Sensitivity          0.8761468
## Specificity          0.4457831
## Pos Pred Value       0.6749117
## Neg Pred Value       0.7326733
## Precision            0.6749117
## Recall               0.8761468
## F1                   0.7624750
## Prevalence           0.5677083
## Detection Rate       0.4973958
## Detection Prevalence 0.7369792
## Balanced Accuracy    0.6609650
```

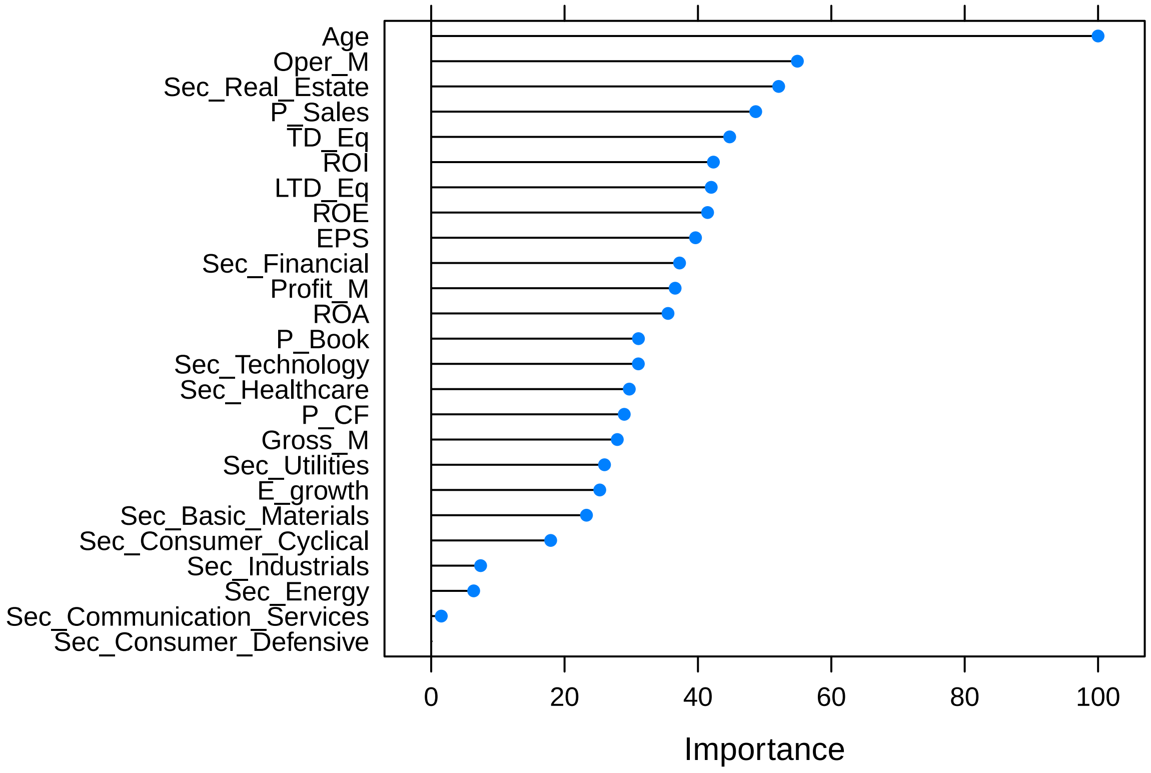
C. Second stage: Advanced models

Random forests:

Even though our naive model could be an interesting starting point, there's still much to be done in order to improve our predictive power. Now, instead of a single tree, we will try predicting whether a company pays dividend or not with an entire forest.

```
##                      Single_tree RF_Accuracy
## Sensitivity            0.8761468   0.8715596
## Specificity            0.4457831   0.6265060
## Pos Pred Value         0.6749117   0.7539683
## Neg Pred Value         0.7326733   0.7878788
## Precision              0.6749117   0.7539683
## Recall                 0.8761468   0.8715596
## F1                     0.7624750   0.8085106
## Prevalence             0.5677083   0.5677083
## Detection Rate         0.4973958   0.4947917
## Detection Prevalence   0.7369792   0.6562500
## Balanced Accuracy      0.6609650   0.7490328
```

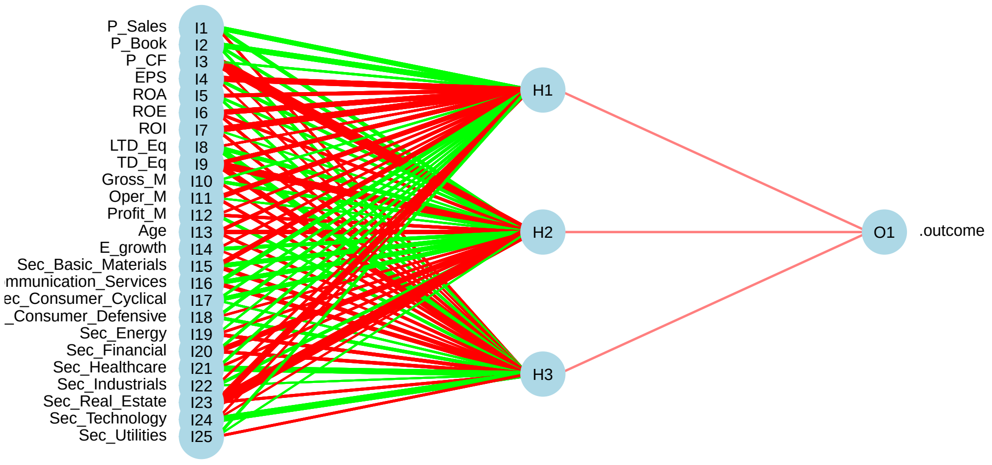
Accuracy measures reported by the confusionMatrix() function has dramatically improved. And more importantly, we have gained a new piece of valuable information: including more variables into the model can improve our accuracy!



Artificial Neural Networks:

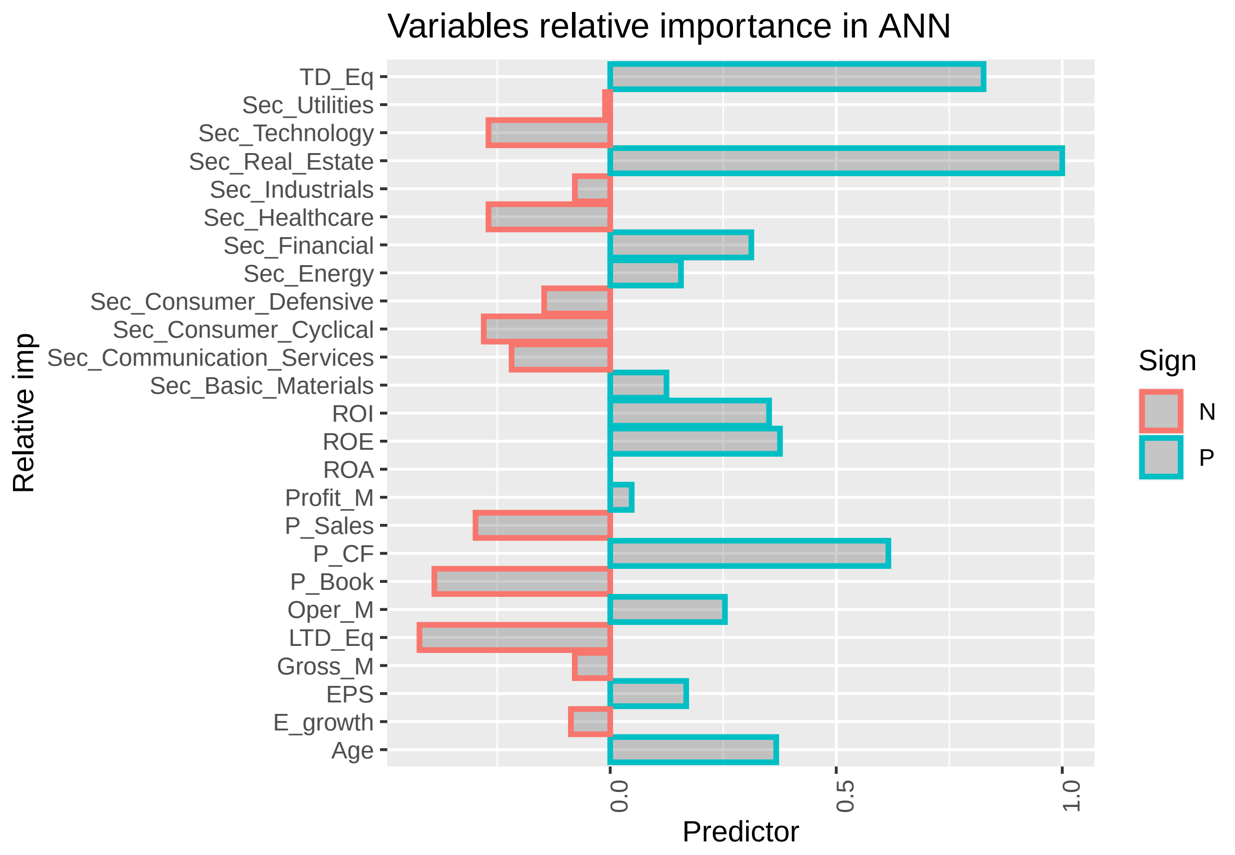
We will now use an algorithm not learned in previous courses, an artificial neural network. Basically, it woks by emulating human's learning process by simulating a network of neurons! Reader will find a much detailed explanation in **Neural Networks in R** by Barış Can Tayiz and **ANN Classification with 'nnet' Package in R** by Rizka Yolanda (link provided in references).

What ANN does is predicting outcomes by processing predictors through "nodes" which are organized within "layers": we can have multiple nodes in multiple layers, or a single node and a single layer. For illustrative purposes we will train a 3 node-single layered ANN on our training dataset:



We can see that each input variable (I1 to I25) is connected with three different nodes inside the hidden layer (H1, H2 and H3) and produce a specific output (O1).

Furthermore, we can dive into the model to check the relative importance of each predictor with the output!



ANN can be a powerful tool for understanding underlying patterns conventional approaches may overlook, but did it really help us improving our predictive power?

```
##                      Single_tree RF_Accuracy       ANN
## Sensitivity            0.8761468   0.8715596 0.7064220
## Specificity            0.4457831   0.6265060 0.7349398
## Pos Pred Value         0.6749117   0.7539683 0.7777778
## Neg Pred Value         0.7326733   0.7878788 0.6559140
## Precision              0.6749117   0.7539683 0.7777778
## Recall                 0.8761468   0.8715596 0.7064220
## F1                     0.7624750   0.8085106 0.7403846
## Prevalence             0.5677083   0.5677083 0.5677083
## Detection Rate         0.4973958   0.4947917 0.4010417
## Detection Prevalence   0.7369792   0.6562500 0.5156250
## Balanced Accuracy      0.6609650   0.7490328 0.7206809
```

Even though some accuracy measures are lower (comparing with other models), we can see that others like specificity and precision were highly improved.

So, what's left to do? We can optimize all models and ensemble them!

D. Third stage: Optimizing models

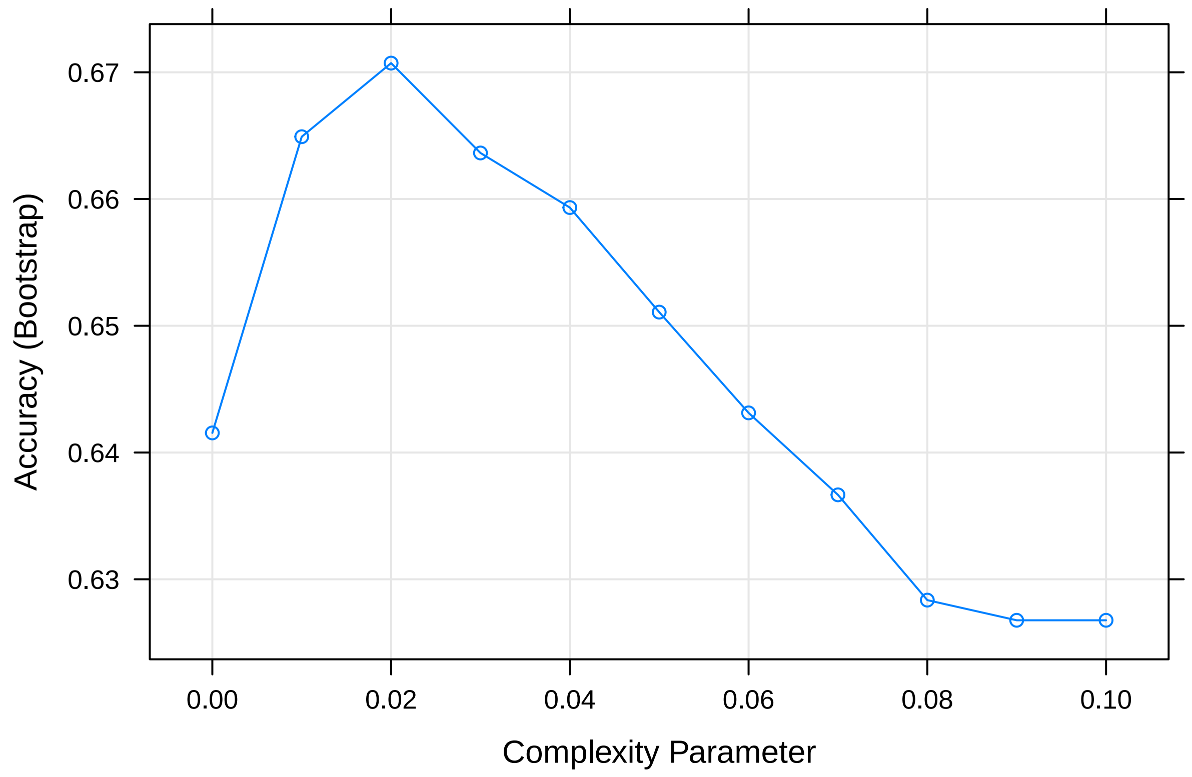
We have seen three different models that have reported, with no tuning at all, some satisfactory results. Now it is time to optimize and them, and check if teamwork aid us achieving more predictive power.

Decision trees: We will tune the *"complexity parameter"* which can be defined as "the minimum improvement in the model needed at each node".

Random forests: We will tune the *"mtry parameter"*, "Number of variables randomly sampled as candidates at each split"

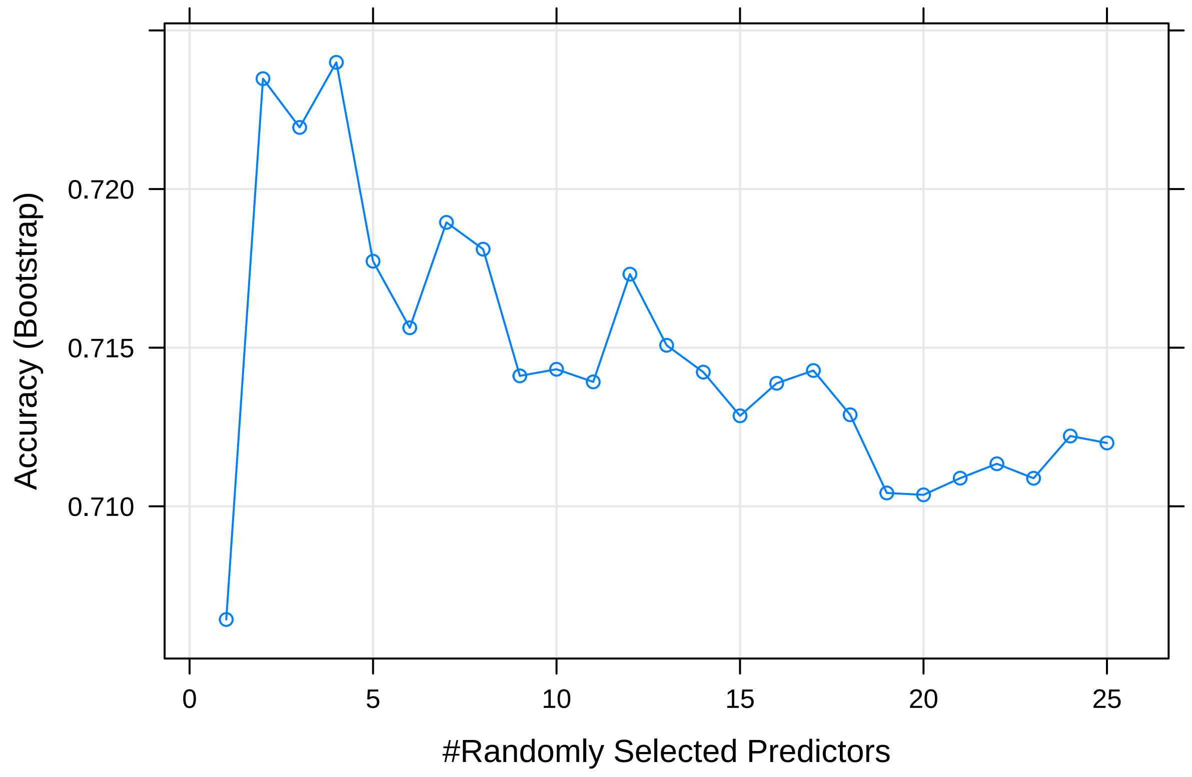
Artificial neural network (ANN): We will tune the *"size parameter"* (number of nodes in the hidden layer) and *"decay parameter"* (regularization parameter to avoid over training)

Single tree parameters:

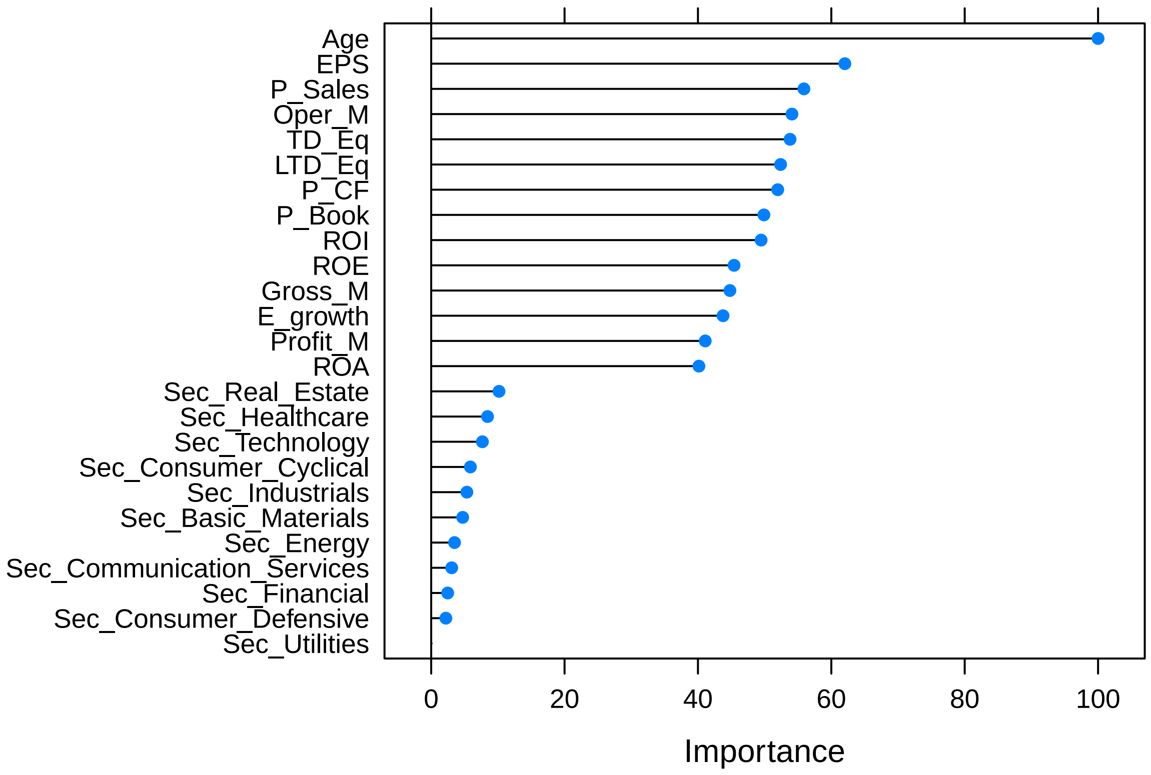


We can see that a complexity parameter of 0.02 can maximize our model's performance.

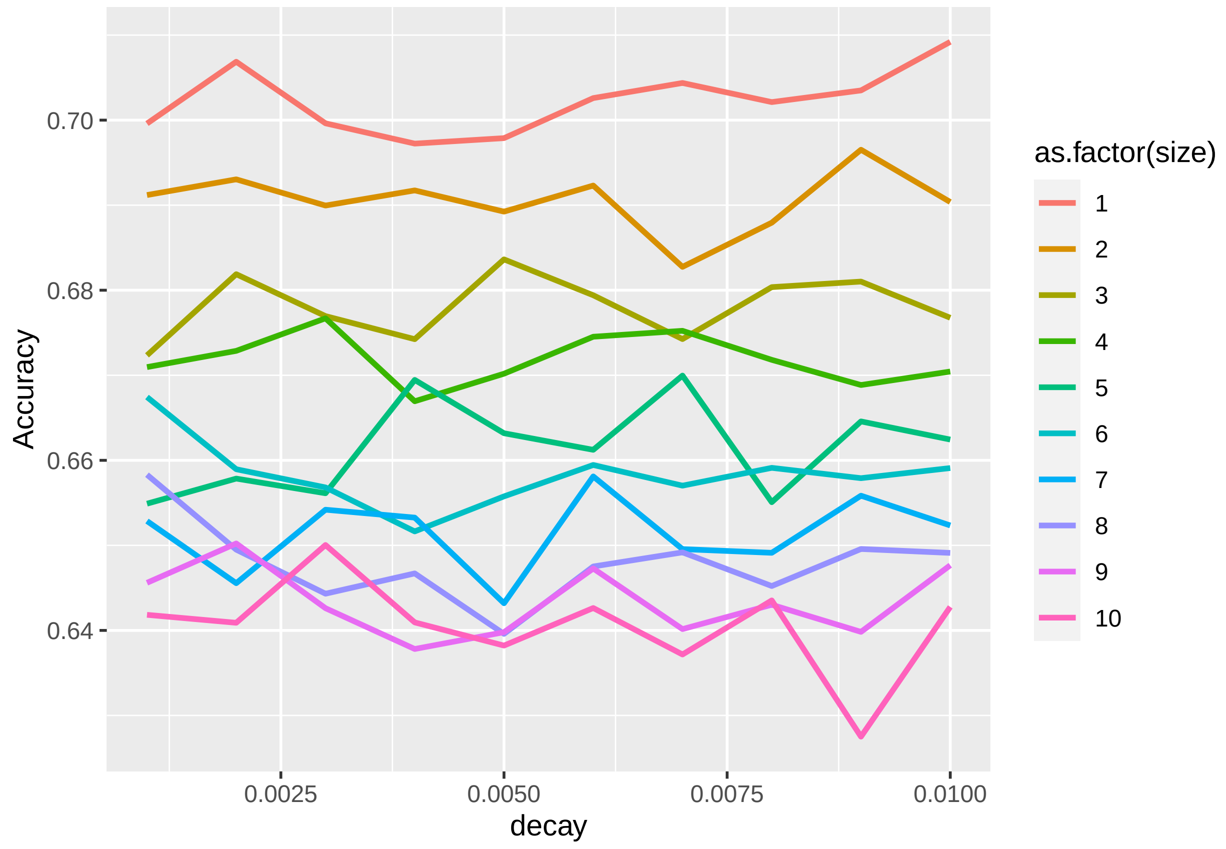
Random forest parameters:



By including only 4 variables in each split per tree, our random forest can maximize its accuracy. More importantly, we can also see that besides "Age", other variables like ROA, ROE, ROI and Operating Margin were relatively important:


Age
EPS
P_Sales
Oper_M
TD_Eq
LTD_Eq
P_CF
P_Book
ROI
ROE
Gross_M
E_growth
Profit_M
ROA
Sec_Real_Estate
Sec_Healthcare
Sec_Technology
Sec_Consumer_Cyclical
Sec_Industrials
Sec_Basic_Materials
Sec_Energy
Sec_Communication_Services
Sec_Financial
Sec_Consumer_Defensive
Sec_Utilities
0
20
40
60
80
100
Importance

ANN parameters:



Our model will maximize it's precision with a 1 node hidden layer and a 0.01 decay for regularization.

E. Fourth stage: Ensembling

We will ensemble all trained models. This means that for each company a model will "vote" on whether it pays dividends or not, and the final prediction will be the most voted one.

Since there are no perfect algorithms, we want one that can perform well on every dimension.

```
##                      Single_tree RF_Accuracy       ANN Final_model
## Sensitivity            0.8761468   0.8715596 0.7064220   0.8394495
## Specificity            0.4457831   0.6265060 0.7349398   0.6265060
## Pos Pred Value         0.6749117   0.7539683 0.7777778   0.7469388
## Neg Pred Value         0.7326733   0.7878788 0.6559140   0.7482014
## Precision              0.6749117   0.7539683 0.7777778   0.7469388
## Recall                 0.8761468   0.8715596 0.7064220   0.8394495
## F1                     0.7624750   0.8085106 0.7403846   0.7904968
## Prevalence             0.5677083   0.5677083 0.5677083   0.5677083
## Detection Rate         0.4973958   0.4947917 0.4010417   0.4765625
## Detection Prevalence   0.7369792   0.6562500 0.5156250   0.6380208
## Balanced Accuracy      0.6609650   0.7490328 0.7206809   0.7329778
```

IV. Results: Team work vs individuallity

I will first congratulate the reader for making it to the very end! Machine learning can be exhausting on its own, and adding financial concepts might have made it even more difficult, but you have made it, and I highly appreciate it!

We can see that adding more complex algorithms than a simple decision tree can help us achieve pretty satisfactory results. The NPV suggests that approximately 9 out of 12 companies we include in our portfolio will end up paying dividends, and on the other hand the PPV value suggests that approximately only 3 out of 12 companies which were not included will present an opportunity cost.

Besides, the F1 score and Balanced Accuracy also suggests that dividend policy can be predicted by fundamental analysis.

Each algorithm presented its strengths and weaknesses, and each one provided it's strengths to the final model, For example, ANN provided better specificity and PPV, and random forests allowed the model to improve it's NPV!

Overall, it appears that random forests have more predictive power on its own.

One very important thing to understand is that the key accuracy indicator in this specific field is the proportion of companies which were predicted to pay dividends (hence they were included in investor's portfolio) and actually did. Being the category "NO" the positive one in confusionMatrix() function, this is measured by the NPV value.

We can improve it by forcing a higher cut rate in the model. The predict() function allows to present output as the probability assigned by the model instead of the category. By default, most algorithms use a 0.50 rate cut, meaning that if an observation has a 51% probability to pay dividends, it will be assigned as "YES". If we force this rate to 0.75, NPV should improve:

```
##                       X075_cut
## Sensitivity          0.9770642
## Specificity          0.3012048
## Pos Pred Value       0.6474164
## Neg Pred Value       0.9090909
## Precision            0.6474164
## Recall               0.9770642
## F1                   0.7787934
## Prevalence           0.5677083
## Detection Rate       0.5546875
## Detection Prevalence 0.8567708
## Balanced Accuracy    0.6391345
```

One more result that must be discussed is the correlation between each numerical predictor and the probability for a company to pay dividends assigned by the final model. Even though correlation coefficient assumes linear relationship among variables, it should work as a fast and intuitive tool for analysis.

To do it, we will make use of the 'type' argument within predict() function, which allows to generate the probability of belonging into one category or another:

```
##           Correlation_with_probability
## P_Sales                   -0.105180335
## P_Book                    -0.094519365
## P_CF                      -0.026036986
## EPS                        0.159937163
## ROA                        0.380571043
## ROE                        0.328753480
## ROI                        0.349111059
## LTD_Eq                    -0.039435180
## TD_Eq                     -0.012235999
## Gross_M                   -0.008747675
## Oper_M                     0.444232047
## Profit_M                   0.372337954
## Age                        0.586298545
## E_growth                   0.088529027
```

Results were quite coherent as expected in the exploratory analysis: higher return ratios and margin will lead to a more stable situation for a company to start paying dividends, and the same might be said about earnings. On the other hand, overvalued and over-leveraged companies are less likely to pay dividends. Earnings expected growth has a small positive correlation, which matches our initial hypothesis.

A very important predictor is "Age" and it makes sense that it has a strong positive correlation.

V. Final thoughts

A. Conclusions

We first performed an exploratory analysis by discretizing each predictor and computing for each discretized value the proportion of dividend paying companies. This first insight allowed for us to start making some hypothesis on the relationship between each predictor and our dependent variable.

After doing so we performed a naive approach by using a single decision tree. Results were not satisfactory enough and the algorithm didn't use all predictors. In fact, the only variable used was "Age".

Fortunately, the caret package includes a vast number of techniques, from which we applied two very popular and complex ones: random forests ('rf') and a single layered artificial neural network ('nnet'). The same package allowed model tuning with a very simple line of code and by performing an ensemble of all optimized algorithms we were able to achieve satisfactory results.

B. Limitations

The first limitation one should mention is the fact that because of low computer power only three techniques were used, given that caret package includes tons of predictive models which could have provided additional accuracy.

The second limitation I must mention is the fact that the main dataset only included information from 1485 companies.

In the third place, because I wanted to go beyond course material, I used a technique not learned in previous courses, which may have not been the best available option.

Finally, not all variables were included in the exploratory analysis. I chose to do so because I wanted it to be as "friendly" as possible with readers who are not familiarized with financial literature.

C. Future work

One might say that future work could be adding more variables and avoiding categorical predictors (like sector) and, Of course, that more predictive algorithms could be applied in order to achieve higher prediction power.

I encourage the reader to do so, but I also think that a bigger dataset could be better than including more algorithms, because it will allow for multiple splits on the main dataset leading to higher predictive power, since I was not able to perform cross-validation or bootstrapping.

Being said that, I believe that future work should be focused on keeping the dependent variable as a continuous one instead of a categorical one, and applying proper techniques on predicting "how much" will a company pay, given it has been predicted to pay dividends in the foreseeable future.

Thank you, Estanislao Maria Ferrer.

VI. References

Research:

Dewasiri, N. J., Koralalage, W. B. Y., Azeez, A. A., Jayarathne, P. G. S. A., Kuruppuarachchi, D., & Weerasinghe, V. A. (2019). *Determinants of dividend policy: Evidence from an Emerging and Developing market.* Managerial Finance.

Kakushadze, Z., & Serur, J. A. (2018). *151 Trading Strategies.* Z. Kakushadze and JA Serur, 151.

Liu, B., Chang, L. B., & Geman, H. (2017). *Intraday pairs trading strategies on high frequency data: The case of oil companies.* Quantitative Finance, 17(1), 87-100.

Mann, J., & Kutz, J. N. (2016). *Dynamic mode decomposition for financial trading strategies.* Quantitative Finance, 16(11), 1643-1655.

Links:

https://towardsdatascience.com/finally-why-we-use-an-80-20-split-for-training-and-test-data-plus-an-alternative-method-oh-yes-edc77e96295d#:~:text=The%20Pareto%20Principle%20is%20also,come%20from%2020%25%20of

https://finviz.com/screener.ashx?v=152&f=ind_stocksonly&ft=4&o=-averagevolume&r=21&c=1,3,4,6,7,9,10,13,14,16,32,33,34,37,38,63,70

https://medium.com/@brscntyz/neural-network-in-r-e275302b6e44

https://medium.com/@yolandawiyono98/ann-classification-with-nnet-package-in-r-3c4dc14d1f14