עזריאלי
מכללה אקדמית להנדסה
ירושלים

דו"ח פרויקט

Machine Learning Project

מנחה הקורס: ד"ר רוני הורביץ

שמות חברי הצוות:

מתן חובר: 203314752

אלירן סרור:305404337

אמיר מסללאתי:203875166

חן שליו: 203205984

קישור ל-bitbucket : https://bitbucket.org/Matanch/r-studio-project/src/master/

מבוא

בפרויקט זה, במסגרת הקורס "כריית ידע ולמידת מכונה", נדרשנו לחקור DataSet המכיל נתונים גולמיים, המציגים שלל מידע פיננסי של 5910 חברות פולניות. המידע שלנו מכיל ציונים פיננסיים מהשנה הראשונה של החיזוי ולאחר 5 שנים.

עמודת המטרה- מה קרה בפועל לחברה, התבקשנו לבצע ניתוח לנתונים וכריית מידע על ה-DATA בכדי לחזות מי מהחברות עלולה לפשוט רגל על סמך הנתונים שבידינו.

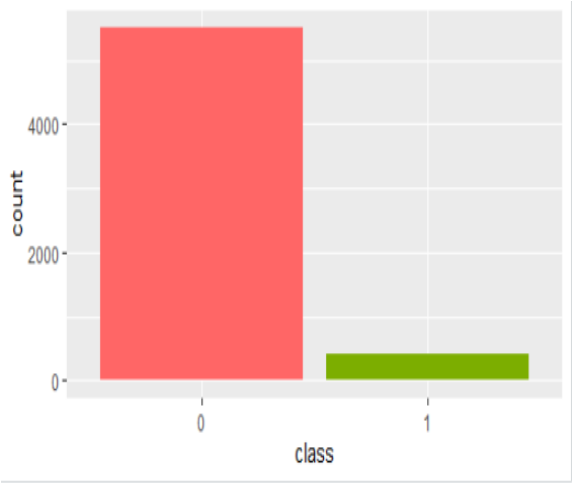
מטרות הפרויקט

- הרחבת הידע בחומר הנלמד, תוך שימוש בלמידת מודלים ונושאים חדשים.
- הפקת מודלים אשר באמצעותם יתקבלו תוצאות חיזוי אופטימאליות שבאמצעותן נצליח לחזות האם חברה פולנית תפשוט רגל או להיפך.
- זיהוי עמודות (תכונות) אשר משפיעות בצורה מובהקת על עמודת המטרה.
- מתן סיכום אשר אמור לייצר לבעלי העניין יכולת השפעה גבוהה בקבלת החלטות.

תהליך העבודה

תחילה בחנו את הנתונים שקיבלנו בקובץ והבנו שמדובר במאגר מידע גדול הכולל בתוכו המון מספרים הקשורים לניתוחים פיננסים, על מנת לנתח אותם נצטרך לבצע מספר פעולות הכוללות, ניקיון וסידור של הנתונים במטרה להגיע למידע הכי נכון ומדויק.

ביצענו בדיקת ראשונית לראות את היחס בעמודת המטרה – כבר בשלב זה זיהינו כי ישנה בעיה משמעותית של נתונים לא מאוזנים , אשר עלולים לפגוע באמינות המודלים.



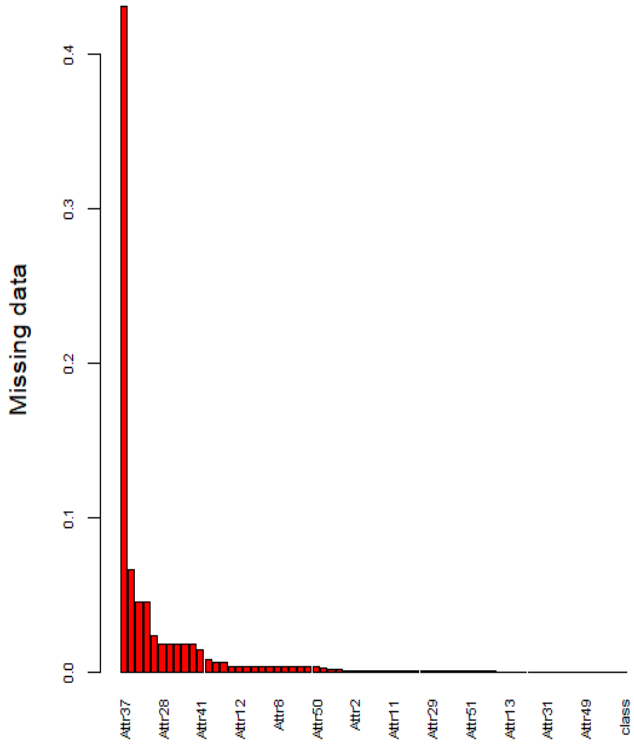
בכדי לסדר את הנתונים, השתמשנו בכלים שניתנו לנו במהלך הקורס ואף ביצענו מחקר מעמיק באתרי אינטרנט אשר הסבירו לנו כיצד עלינו להתמודד בכל שלב.

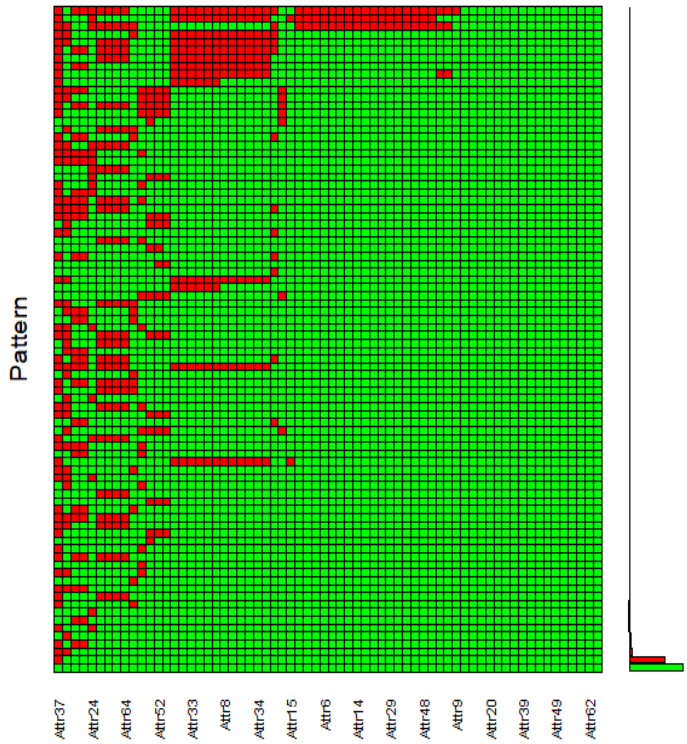
ניקיון הנתונים בוצע בצורה הבאה:

תחילה, טענו את כל הספריות שבהם נעזרנו לבנות את הגרפים והמודלים בהם בחרנו להשתמש.

שנית , בדקנו את כמות ה-NA בנתונים שלנו, וגילנו כי ישנו מספר רב של נתונים חסרים.

יצרנו ווקטור אשר נותן לנו אינדיקציה באילו עמודות קיימים ערכי NA. בנוסף, בחרנו לבצע פונקציה בשם aggr(df) על מנת לראות באופן גרפי את אחוז הנתונים החסרים בכל עמודה.





בעקבות בדיקה זו, החלטנו למחוק את תכונה-Attr37 מכיוון שקיימים המון ערכים חסרים (כמעט 45%) ועל כן, ניתן להסיר את העמודה מכיוון שלא נוכל להסיק באמצעותה דבר בהקשר לתכונת המטרה בנוסף, החלטנו לטפל גם בתוכנה Attr21 אשר היא מתארת את היחס בין המכירות לשנה הנוכחית לעומת השנה הקודמת. ישנם מקרים בהם חברה נפתחה במהלך השנה ולכן אין נתונים משנים קודמות, דבר זה מוביל לערכים חסרים בעמודה זו ולכן, את כל הערכים החסרים בעמודה זו נשנה ל-0.

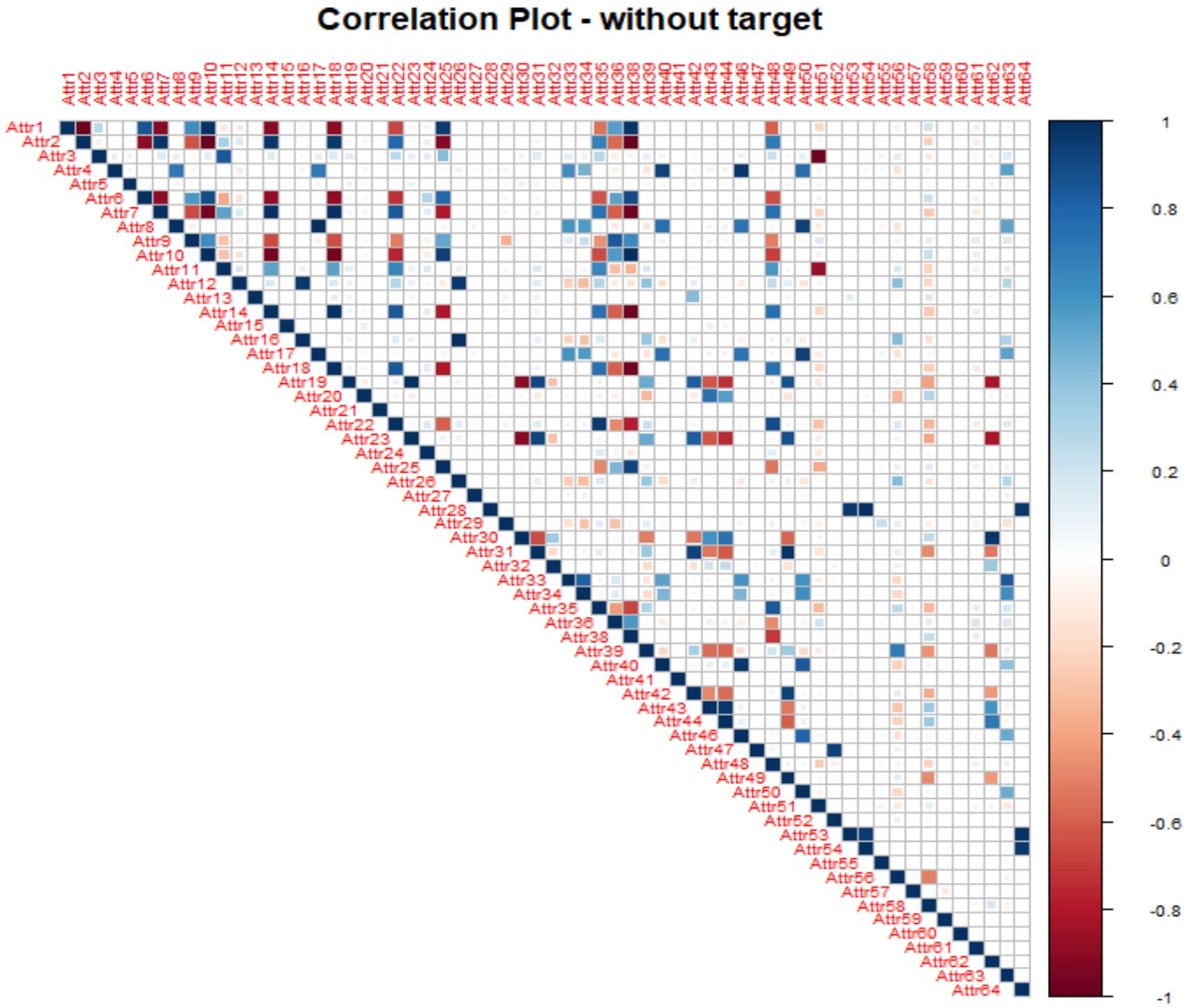
בנוסף, החלטנו לטפל גם בתוכנה Attr27 אשר היא המתארת את היחס בין הרווחים השוטפים לבין ההוצאות הפיננסיות של החברה (קרי הלוואות), לכן במצב בו קיים ערך חסר סביר להניח שזה משום שלחברה אין הלוואות או שזו טעות בהזנה של הנתון. לכן נשנה את הערכים החסרים ל-0.

את העמודות "Attr45" , "id" נסיר.

עמודת "id" הוסרה וזאת מכיוון שלא תורמת לחיזוי משתנה המטרה ואף עלולה לייצר מצב של verfitting. בנוסף, החלטנו להסיר גם את Attr45 מכיוון שהיא דומה מאוד ל-Attr60 כך שנוצרת כאן כפילות מיותרת של נתון.

לאחר מכן ביצענו בדיקת קורלציה(התאמה של מעל 80%), בדיקה שמראה את הקשר בין העמודות השונות במספר צורות, כלומר אם השתנות של תכונה מסוימת גוררת השתנות של תכונה אחרת. על מנת שנוכל לחזות נכון את התנהגות משתנה המטרה על סמך התכונות השונות עליהן להיות בלתי תלויות זו בזו. לשם בדיקה זו השתמשנו בגרף הcorrplot, אשר בודק התאמה בין עמודות, לצורך בדיקה זו הורדנו את עמודת ה-Target אשר לא תורמת לבדיקה זו.

<u>להלן תרשים ה-corrplot :</u>



לאחר מכן, יצרנו וקטור אשר מכיל את המספרים של העמודות אשר נמצאה בהם התאמה גבוהה ובסוף הורדנו אותם מהמודל שלנו.

Due to high correlation, these <<columns>> are recommended to be **removed:before** Treatment of NA values:

1 2 7 8 9 10 11 12 14 18 19 22 23 26 28 30 31 33 35 37 39 43 44 45 47 48 49 51 62

```
df_reduced_after_corr <- bankruptcy.raw[, -highlycorrelated]
colnames(df_reduced_after_corr)
[1] "Attr3"  "Attr4"  "Attr5"  "Attr6"  "Attr13" "Attr15" "Attr16" "Attr17" "Attr20" "Attr21"
11] "Attr24" "Attr25" "Attr27" "Attr29" "Attr32" "Attr34" "Attr36" "Attr39" "Attr41" "Attr42"
21] "Attr43" "Attr48" "Attr52" "Attr54" "Attr55" "Attr56" "Attr57" "Attr58" "Attr59" "Attr60"
31] "Attr61" "Attr62" "Attr63" "class"
```

טיפול בעמודות הנוספות עם ערכי NA:

לאחר מחקר שביצענו באינטרנט בנושא טיפול בערכי -NA נמצאו מספר שיטות אשר נותנות פתרון לבעיית הערכים החסרים(שיטת אמצע הקטע, מספר ייחודי או הזנת נתונים במודל MICE.

אנו הגענו להחלטה שהדבר הטוב ביותר לעשות (וזה גם מה שנלמד בכיתה),להכניס אליהם את הממוצע של אותה עמודה. לבסוף זה המודל שגם יישמנו מכיוון, שהוא שיקף את המידע בצורה הכי מדויקת.

```
> #AVERAGE FUNCTION
> # change NA  to the average of his col
> getmode <- function(v)
+ {
+   uniqv <- unique(v)
+   uniqv[which.max(tabulate(match(v, uniqv)))]
+ }
> #Function for removing outliers
> out.rem<-function(x)
+ {
+   x[which(x==outlier(x))]=NA
+   x
+ }
>
> imputed_dataset=df_reduced_after_corr %>% mutate_if(is.numeric, funs(replace(.,is.na(.), mean(., na.rm = TRUE)))) %>%
+   mutate_if(is.factor, funs(replace(.,is.na(.), getmode(na.omit(.)))))
>  any(is.na(imputed_dataset))
[1] FALSE
```

בדיקות EDA

בשלב הראשון בדקנו בעזרת פונקציית היסטוגרם אם יש לשדות חריגים (זנבות). במצבים רבים המידע התקבץ באזור מסוים והיו מדגמים עם ערכים גבוהים או נמוכים בפער משמעותי מאוד משאר המידע, ולכן החלטנו להוריד את החריגים באמצעות הפונקציות אשר בנינו במהלך ההרצאה.

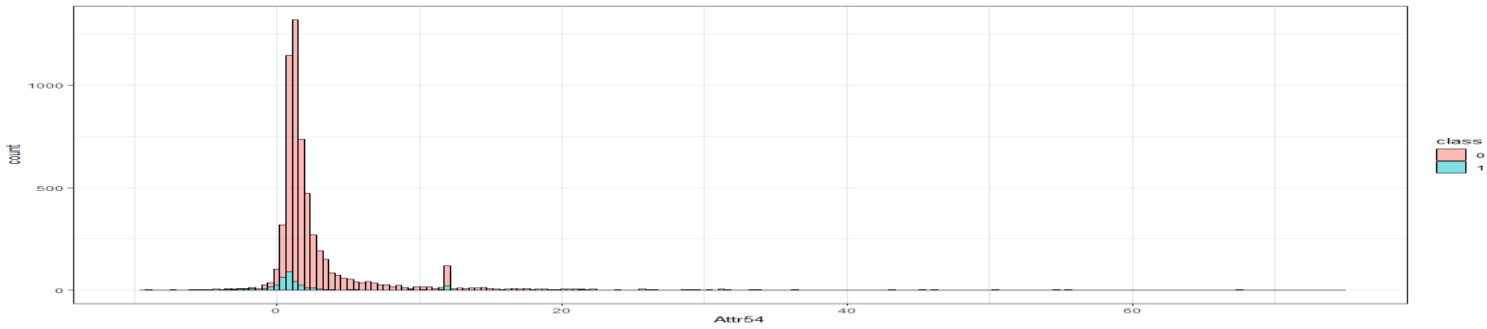
```
#--coercex
coercex.maxValue <- function(x,by){
  if(x<=by) return(x)
  return(by)
}

coercex.minValue <- function(x,by){
  if(x>=by) return(x)
  return(by)
}
```

כל תכונה אשר החלטנו להשאיר במודל (לאחר תרשים הקורלציה) ביצענו קיבוץ ערכים חריגים(במידה וקיימים), לבסוף בדיקה באמצעות **Bining** בכדי לבדוק האם עמודה זו רלוונטית למודל.

להלן דוגמא לשימוש בתהליך ה-EDA

בחרנו להתבונן בתכונה Attr54-

```
> #Attr54
> quantile(bankruptcy.prepared$Attr54, probs = c(0,0.2,0.5,1,2,3,4,5,10,25,50,75,90,92,93,94,95,97,98,99,99.2,99.5,99.75,99.80,99.82,9
9.83,99.85,99.87,99.90,100)/100)
          0%        0.2%        0.5%          1%          2%          3%          4%          5%         10%
-1088.7000000  -44.2193860  -10.5260850  -3.6240380  -0.9268510  -0.2154754   0.0607950   0.2523610   0.6353500
         25%         50%         75%         90%         92%         93%         94%         95%         97%
    1.0001250   1.4513000   2.5744500   7.2739300   9.9110320  11.7798800  12.1101893  12.2920500  19.4609300
         98%         99%       99.2%       99.5%      99.75%       99.8%      99.82%      99.83%      99.85%
   30.2333200  76.7289800 110.3348000 191.0879000 388.1549000 549.2325000 590.5352940 626.0661110 879.7178700
      99.87%       99.9%        100%
  883.9328880 1185.4668000 21702.0000000
> ggplot(bankruptcy.prepared,aes(Attr54)) + geom_histogram(aes(fill=class),color='black' ,bins=200 ,alpha=0.5) +theme_bw()+xlim(-10,75)
```

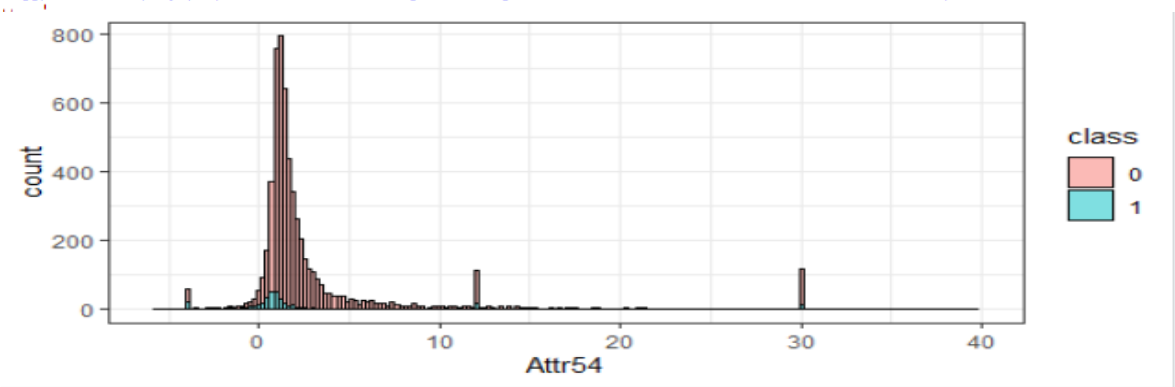


הוצאת חריגים

הוצאנו חריגים באמצעות הפונקציות שיצרנו (מופיעות בתרשים מלעיל).

גרף לאחר ניקוי החריגים

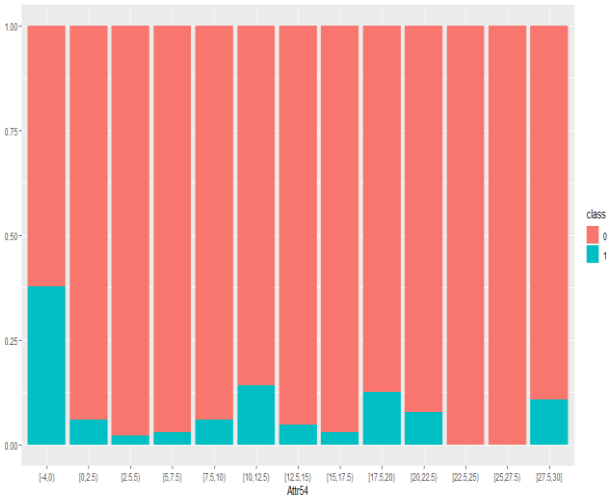
```
> bankruptcy.prepared$Attr54 <- sapply(bankruptcy.prepared$Attr54, coercex.maxValue, by = 30)
> bankruptcy.prepared$Attr54 <- sapply(bankruptcy.prepared$Attr54, coercex.minValue, by = -4)
> ggplot(bankruptcy.prepared,aes(Attr54)) + geom_histogram(aes(fill=class),color='black' ,bins=200 ,alpha=0.5) +theme_bw()+xlim(-6,40)
```

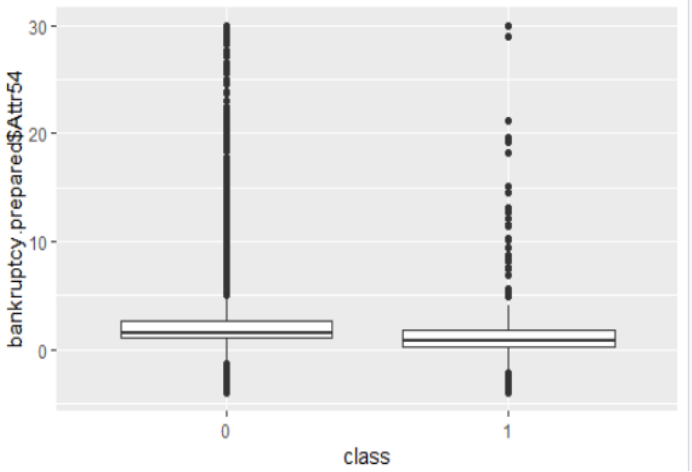


ביצוע Bins

```
> #Bins
> breaks.54<- c(-4,0,2.5,5,7.5,10,12.5,15,17.5,20,22.5,25,27.5,30)
> bins.A54<- cut(bankruptcy.prepared$Attr54, breaks=breaks.54 , include.lowest = TRUE,right = FALSE)
> summary(bins.A54)
   [-4,0)    [0,2.5)    [2.5,5)    [5,7.5)   [7.5,10) [10,12.5) [12.5,15) [15,17.5) [17.5,20) [20,22.5) [22.5,25) [25,27.5)
 [27.5,30]
      214       4179        718        228        101        177         62         33         24         26          7         11
      130
>
```

ניתוח העמודה





זיהינו חוסר מגמה בעמודה זו ובנוסף מספר גבוהה של חריגים , לכן מחקנו אותה מהמודל.

```
> #we can delete this Attribute
> bankruptcy.prepared$Attr54<-NULL
```

מודלי חיזוי

בשלב האחרון הרצנו שבעה מודלים ולצורך כך חילקנו את מסד הנתונים Train ו-Test, ביחס של 70:30. להלן המודלים:

- Naive Bayes
- Decision Tree
- Random Forest
- KNN
- Logistic Regression
- AdaBoost
- Tree Bag
- Naive Bayes after balance
- Neural nets

בבדיקות הנ"ל החלטנו שההסתברות שחברה תפשוט רגל היא כאשר תוצאות ה-prediction הן מעל 0.5.
בנוסף החלטנו לייחס חשיבות גדולה יותר לrecall כיוון שמדובר בנתוני פשיטות רגל.

Naive Bayes model

confusion matrix:

```
            actual.NB
predicted.NB    0    1
           0 1489   60
           1  161   63
```

precsion and recall:

```
> precsion
[1] 0.28125
> recall
[1] 0.5121951
>
```

accuracy

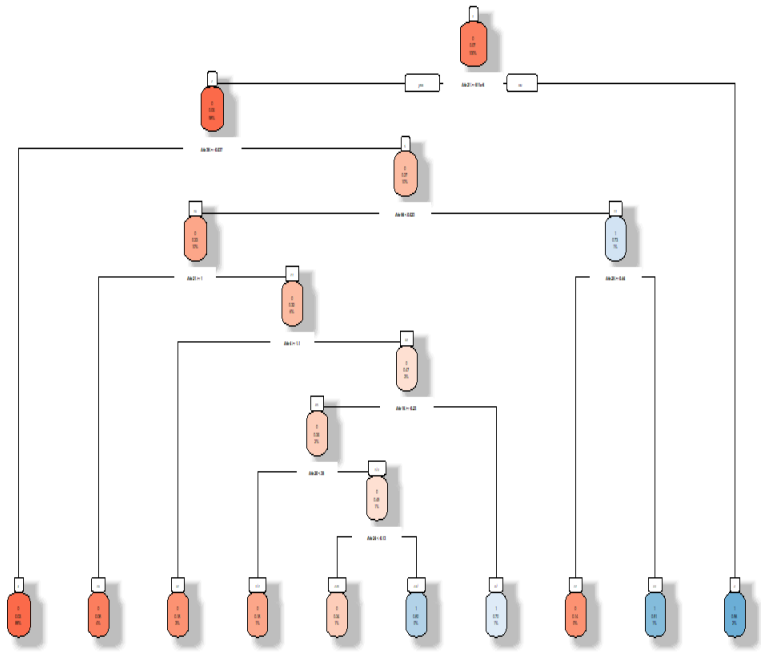
```
accu.NB
threshold       AUC omission.rate sensitivity specificity prop.correct     Kappa
      0.5 0.6212577       0.71875     0.28125   0.9612653    0.8753525 0.3004569
```

מסקנות המודל: ב28.1% מהמקרים שהמודל חוזה שאכן העסק יפשוט רגל הוא אכן צודק.

בנוסף, המודל מצליח לחזות רק ב-51.2% הצלחה עסקים שאכן יפשטו את הרגל.

Decision Tree

```
> conf_matrix.DT
            actual.DT
predicted.DT    0    1
           0 1646   85
           1    4   38
```



שגיאת התרשים:

```
#error of the rpart
error.rpart <- 1-(sum(diag(conf_matrix.DT))/sum(conf_matrix.DT))
error.rpart #0.0676819
.] 0.05019741
```

precsion and recall

```
> precsion
[1] 0.9047619
> recall
[1] 0.3089431
```

מסקנות מהמודל: ב90% מהמקרים שהמודל חוזה שאכן העסק יפשוט רגל הוא אכן צודק,

בנוסף המודל מצליח לחזות רק ב-30% הצלחה עסקים שאכן יפשטו את הרגל אמנם אחוז הדיוק

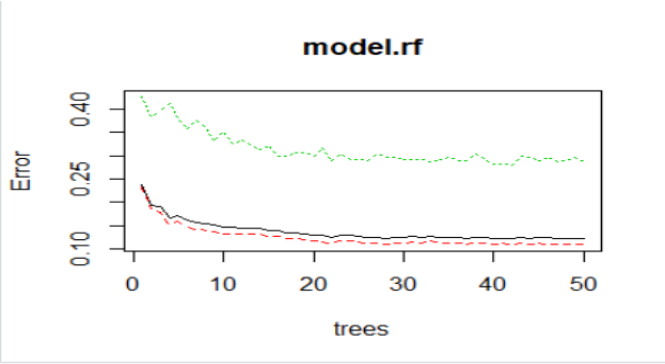
בפונקציית החיזוי של העץ היא גבוהה, אך עדין אנו מצליחים למצוא רק 30%

מהמקרים זה עדיין לא מספיק.

Random forest

תחילה, ניתן לראות כי כאשר בחרנו ב150 עצים אחוז הטעות היה רק 6.16%.

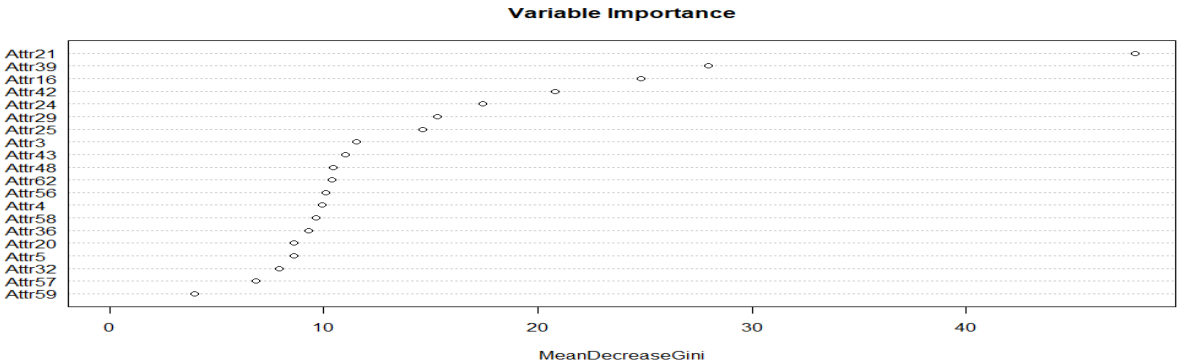
בנוסף, ניתן לראות מהגרף הבא כי כאשר אנו עוברים את ה50 עצים, אחוז הטעות נשאר קבוע באיזור ה5%.



לכן נחתוך את היער ומראש נגדיר לאלגוריתם רק 50 עצים.

לאחר מכן בדקנו את חשיבות התכונות לפי סדר במודל הנ"ל:

```
# Variable Importance Plot
varImpPlot(model.rf, sort = T, main="Variable Importance")
```



ניתן לראות כי התכונה החשובה במודל לפי תרשים זה הינה Attr21 וכן הלאה .

להלן תוצאות המודל:

```
Confusion Matrix and Statistics

            actual.RF
predicted.RF    0    1
           0 1471   43
           1  179   80
```

Precsion and Recall

```
> precsion
[1] 0.3088803
> recall
[1] 0.6504065
```

Accuracy

```
accu
threshold       AUC omission.rate sensitivity specificity prop.correct
      0.5 0.7709608     0.3495935   0.6504065   0.8915152    0.8747885
    Kappa
0.3585007
```

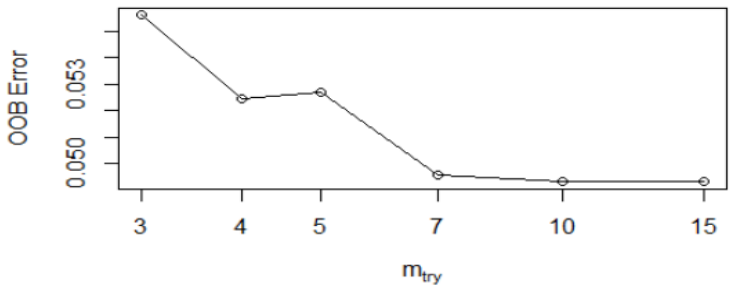
לאחר בחינה של תוצאות המודל ואף ביצוע מחקר מעמיק באינטרנט בכדי לנסות לשפר את תוצאות המודל של ה-Random Forest מצאנו דרך אשר הובילה לשינוי בתוצאות.

ביצענו בדיקה בכדי לגלות מהם מספר הענפים והפיצולים האופטימאלי בכל עץ,משמעות של מספר הענפים האופטימלי הינו – מספר המשתנים שנדגמו באופן אקראי.

בבדיקה זו נמצא כי מספר פיצולים האופטימאלי הינו 10.

```
bestmtry<-tuneRF(x=df_train[,-ncol(df_train)], y=df_train$class, stepFactor=1.5, improve=1e-5, ntree=50 ,doBest = F)
print(bestmtry)   # chosen mtry=9
```

```
> print(bestmtry)
       mtry   OOBError
3.OOB     3 0.05777133
4.OOB     4 0.05777133
6.OOB     6 0.05462896
9.OOB     9 0.05245347
13.OOB   13 0.05632101
```



להלן תוצאות המודל:

```
> confusion.matrix_RF
            actual.RF
predicted.RF    0    1
           0 1472   19
           1  178  104
```

precsion and recall

```
> precsion
[1] 0.3687943
> recall
[1] 0.8455285
>
```

accuracy

```
accu
threshold       AUC omission.rate sensitivity specificity prop.correct     Kappa
      0.5 0.6372962     0.7023411   0.2976589   0.9769335    0.8623801 0.3587627
```

ניתן לראות כי בעקבות התהליך שבוצע על מנת להקטין את מספר הענפים והפיצולים תרם לשיפור תוצאות המודל באופן מובהק.

מסקנות מהמודל: ב-36% מהמקרים שהמודל חוזה שאכן העסק יפשוט רגל הוא אכן צודק

בנוסף המודל מצליח לחזות רק ב-84.5% הצלחה שעסקים שאכן יפשטו את הרגל.

KNN model (אלגוריתם שכן קרוב)

אלגוריתם השכן הקרוב או ***k*-Nearest Neighbors algorithm** (או בקיצור **k-NN**) הוא אלגוריתם חסר פרמטרים לסיווג ולרגרסיה מקומית .בשני המקרים הקלט תלוי ב k-התצפיות הקרובות במרחב התכונות k-NN . יכול לשמש לסיווג או לרגרסיה:

- **k-NNלסיווג** –בהינתן קלט של דוגמה חדשה, האלגוריתם משייכה לקבוצה. הדוגמה משויכת למחלקה הנפוצה ביותר בקרב k השכנים הקרובים) כאשר k מוגדר כמספר חיובי שלם, בדרך כלל מספר קטן .(אם k=1 האובייקט משויך למחלקה של השכן הבודד הקרוב ביותר.
- **k-NNלרגרסיה** –בהינתן דוגמה חדשה, האלגוריתם מחזיר ערך מאפיין לדוגמה. ערך זה הוא ממוצע ערכים של ערכי k השכנים הקרובים ביותר.

k-NNהוא אלגוריתם לימוד מבוסס מופעים, או למידה עצלה ,שבו הפונקציה מקורבת באופן מקומי בלבד וכל החישובים נדחים עד סיווגה. אלגוריתם k-NN הוא מבין האלגוריתמים הפשוטים ביותר בתחום למידת המכונה.

אופן שימוש האלגוריתם :

השכנים נלקחים מתוך סדרת אובייקטים של מחלקה (עבור k-NN לסיווג) או אפיון הערך (עבור k-NN לרגרסיה)ידועים.

חסרון האלגוריתם: חיסרון בולט של האלגוריתם הוא רגישותו למבנה המקומי של הנתונים.

להלן תוצאות המודל: במודל שלנו בחרנו לחקור 3 סוגים של ערכים (שכנים) – 1, 5, ו-20.

Knn = 1

```
> confusion.matrix_kNN.1
      actual.kNN
kNN.1    0    1
    0 1644  116
    1    6    7
```

Precision and Recall

```
> precision.kNN.1   #0.4
[1] 0.5384615
> recall.kNN.1   #0.01626016
[1] 0.05691057
>
```

Knn = 5

```
> confusion.matrix_kNN.5
      actual.kNN
kNN.5    0    1
    0 1646  117
    1    4    6
```

Precision and Recall

```
> precision.kNN.5   #0.4117647
[1] 0.6
> recall.kNN.5   #0.06140351
[1] 0.04878049
>
```

Knn = 20

```
> confusion.matrix_kNN.20
       actual.kNN
kNN.20    0    1
     0 1643  119
     1    7    4
>
```

Precision and Recall

```
precision.kNN.20   #0.4444444
] 0.3636364
recall.kNN.20   #0.07017544
] 0.03252033
```

מסקנות ממודל שכן קרוב: ניתן לראות כי כאשר ערך ה-Knn שווה ל-5 , המודל נותן תוצאות טובות יותר.

עזריאלי
מכללה אקדמית להנדסה
ירושלים

Logistic Regression

תחילה, ביצענו מבחן אנובה בכדי לראות האם קיימת התאמה בין העמודות.

```
> anova(model.LR, test="Chisq")
Analysis of Deviance Table

Model: binomial, link: logit

Response: class

Terms added sequentially (first to last)


       Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
NULL                    4136     2085.2
Attr3   1  191.760      4135     1893.4 < 2.2e-16 ***
Attr4   1    7.090      4134     1886.3 0.0077519 **
Attr5   1   11.086      4133     1875.2 0.0008698 ***
Attr16  1   79.959      4132     1795.3 < 2.2e-16 ***
Attr17  1    1.325      4131     1794.0 0.2497236
Attr20  1    0.938      4130     1793.0 0.3328118
Attr21  1  296.052      4129     1497.0 < 2.2e-16 ***
Attr24  1    1.216      4128     1495.8 0.2700973
Attr25  1   26.382      4127     1469.4 2.801e-07 ***
Attr27  1    0.010      4126     1469.4 0.9216976
Attr29  1    7.761      4125     1461.6 0.0053393 **
Attr32  1    0.243      4124     1461.4 0.6221706
Attr36  1    3.702      4123     1457.7 0.0543417 .
Attr39  1   14.729      4122     1442.9 0.0001241 ***
Attr42  1    0.934      4121     1442.0 0.3337583
Attr43  1   13.849      4120     1428.2 0.0001981 ***
Attr48  1    4.566      4119     1423.6 0.0326107 *
Attr52  1    3.961      4118     1419.6 0.0465766 *
Attr54  1    0.717      4117     1418.9 0.3970876
Attr55  1    0.858      4116     1418.0 0.3543692
Attr56  1    0.201      4115     1417.8 0.6535541
Attr57  1    0.285      4114     1417.6 0.5936047
Attr58  1    2.427      4113     1415.1 0.1192264
Attr59  1    0.027      4112     1415.1 0.8695287
Attr61  1   10.472      4111     1404.6 0.0012123 **
Attr62  1    0.404      4110     1404.2 0.5249747
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
There were 19 warnings (use warnings() to see them)
```

ניתן לראות כי קיימות עמודות אשר תוצאותן זהות לעמודות אחרות.

להלן תוצאות המודל :

```
> confusion_matrix.LR
   actual.LR
       0    1
  0 1638   77
  1   12   46
```

Precision and Recall

```
> precsion
[1] 0.7931034
> recall
[1] 0.3739837
```

מסקנות מהמודל: ב79% מהמקרים שהמודל חוזה שאכן העסק יפשוט רגל הוא אכן צודק.

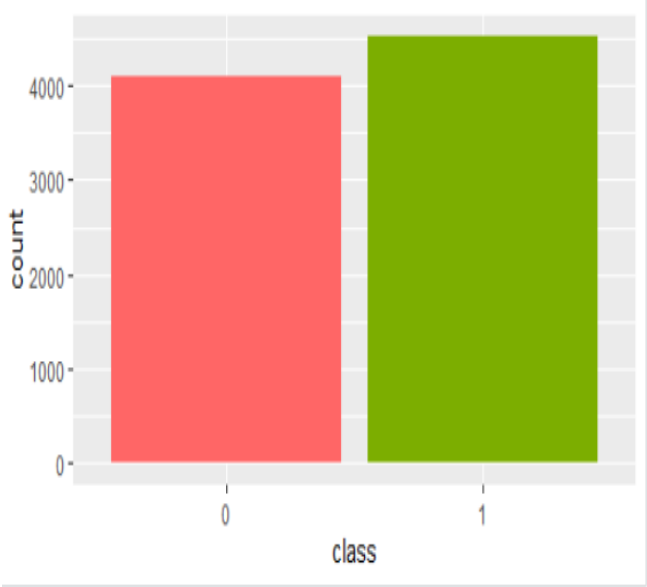
בנוסף המודל מצליח לחזות רק ב-37.5% הצלחה עסקים שאכן יפשטו את הרגל.

מודלים של איזון המידע

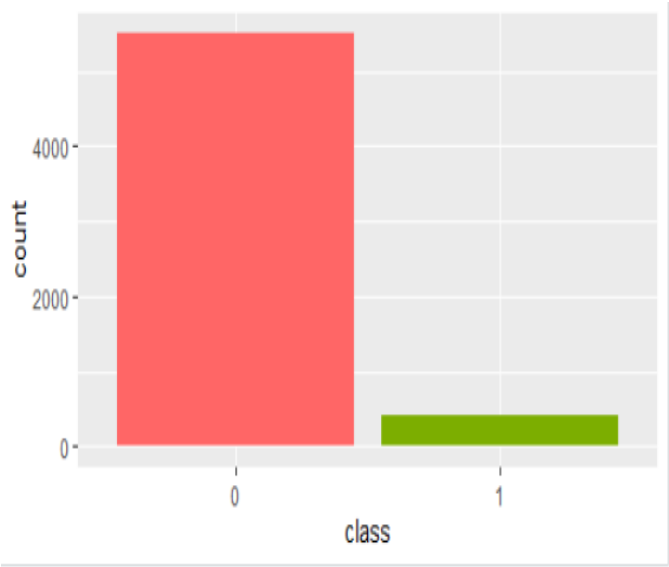
בעקבות מחקר מעמיק שביצענו על המודלים שבוצעו עד כה , הגענו למסקנה כי אנו לא כל כך מרוצים מתוצאות המודלים. בכדי לנסות ולשפר את תוצאות המודלים, החלטנו לנסות ולאזן את נתוני המודל . כמו שראינו בתחילה , קיים הבדל ניכר בין ערכי עמודת ה- Target שלנו, מה שגורם מצב של חוסר איזון בנתונים.

להלן תרשים המשקף את איזון הנתונים לפני ואחרי:

After Boosting



Before Boosting



בכדי טפל במצב זה, ביצענו מחקר באינטרנט על מנת למצוא אילו מודלים מתאימים במצב של חוסר באיזון נתונים. בחרנו לבחון שלושה מודלים אשר לטעמינו נראו הכי מעניינים.

ההמודלים שבחרנו היינם: AdaBoost , Tree Bag ו-Naive Bayes(שאותו אנו מכירים מההרצאות ומתרגילי הבית).

AdaBoost Model

AdaBoost, קיצור של Boosting Adaptive , הוא מטא-אלגוריתם למידת מכונה . ניתן להשתמש בו בשילוב עם סוגים רבים אחרים של אלגוריתמי למידה כדי לשפר את הביצועים .הפלט של אלגוריתמי הלמידה האחרים ('הלומדים החלשים') משולב לסכום משוקלל המייצג את התפוקה הסופית של המסווג המוגבר AdaBoost .הוא אדפטיבי במובן זה שלומדים חלשים הבאים עוקבים לטובת המקרים המסווגים על ידי סיווגים קודמים AdaBoost .רגיש לנתונים ולמחשבים רועשים .בחלק מהבעיות זה יכול להיות פחות רגיש לבעיית ההישג יתר מאשר אלגוריתמי למידה אחרים .הלומדים האינדיבידואליים יכולים להיות חלשים, אך כל עוד הביצועים של כל אחד מהם מעט טובים יותר מאשר ניחוש אקראי, ניתן להוכיח כי המודל הסופי יתכנס למלומד חזק.

להלן תוצאות המודל:

Precision and Recall

```
> confusion.matrix.ADA
                Observed Class
Predicted Class    0    1
              0 1635   55
              1   15   68
```

```
precsion # 0.8192771
1] 0.8192771
recall #0.5528455
1] 0.5528455
```

שגיאת האלגוריתם , ואחוז התאמת האלגוריתם:

```
#AFTER BOOSTING THE ALGORITHEM HAVE ERROR OF 3.9%
print(pred$error)
] 0.03948111
#AFTER BOOSTING THE ALGORITHEM HAVE correction of 96.05%
auc.ADA<-(1-pred$error)
auc.ADA ##0.9605
] 0.9605189
```

מסקנות מהמודל: ניתן לראות כי כאשר מבצעים מניפולציות על מנת לאזן את הנתונים במודל , התוצאות גדלות. לכן כנראה קיימת בעיה של **חוסר איזון בנתונים** ב DataSet שלנו.

Tree Bag Model

```
> confusion_matrix.TB
    actual.TB
pred    0    1
   0 1181   52
   1   85 1301
```

Precision and Recall

```
> precsion
[1] 0.9386724
> recall
[1] 0.9615669
```

מסקנות מהמודל: ב93% מהמקרים שהמודל חוזה שאכן העסק יפשוט רגל הוא אכן צודק.

בנוסף המודל מצליח לחזות רק ב-96% הצלחה עסקים שאכן יפשטו את הרגל.

ניתן להבחין כי איזון הנתונים משפיע משמעותית על אמינות המודל.

Naive Bayes (After Balacne)

```
> conti_matrix.NB.balance
                  actual.NB.balance
predicted.NB.balance    0    1
                   0 1174   63
                   1   92 1290
```

Precision and Recall

```
> precsion.balance
[1] 0.9334298
> recall.balance
[1] 0.9534368
```

accuracy

```
accu.NB.balance
threshold       AUC omission.rate sensitivity specificity prop.correct
      0.5 0.9412501    0.06657019   0.9334298   0.9490703    0.9408171
    Kappa
0.8814161
```

מסקנות מהמודל: ב93% מהמקרים שהמודל חוזה שאכן העסק יפשוט רגל הוא אכן צודק.

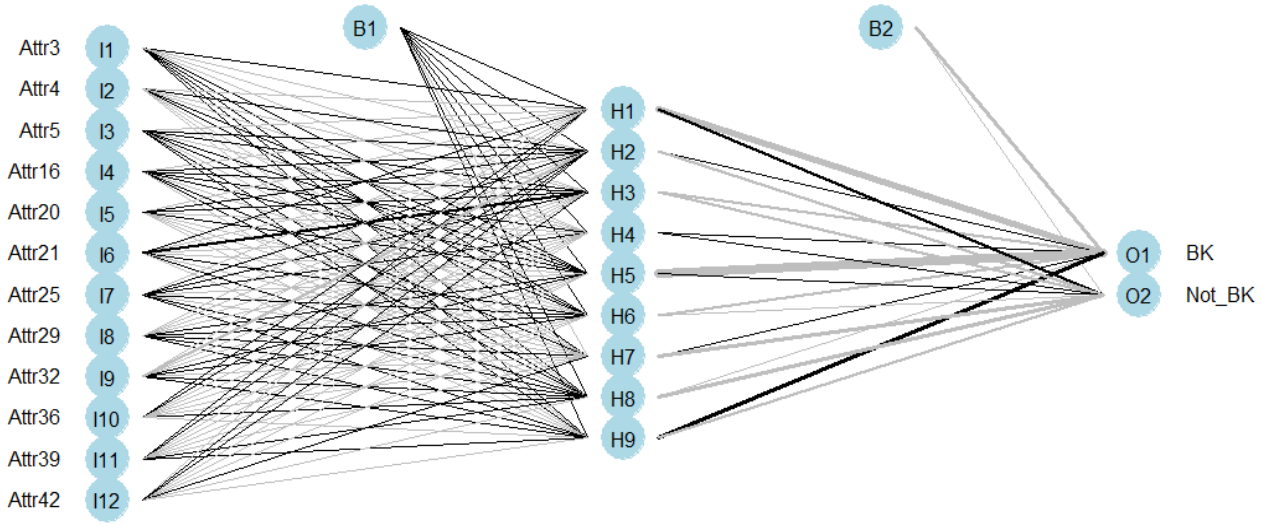
בנוסף המודל מצליח לחזות רק ב-95% הצלחה עסקים שאכן יפשטו את הרגל.

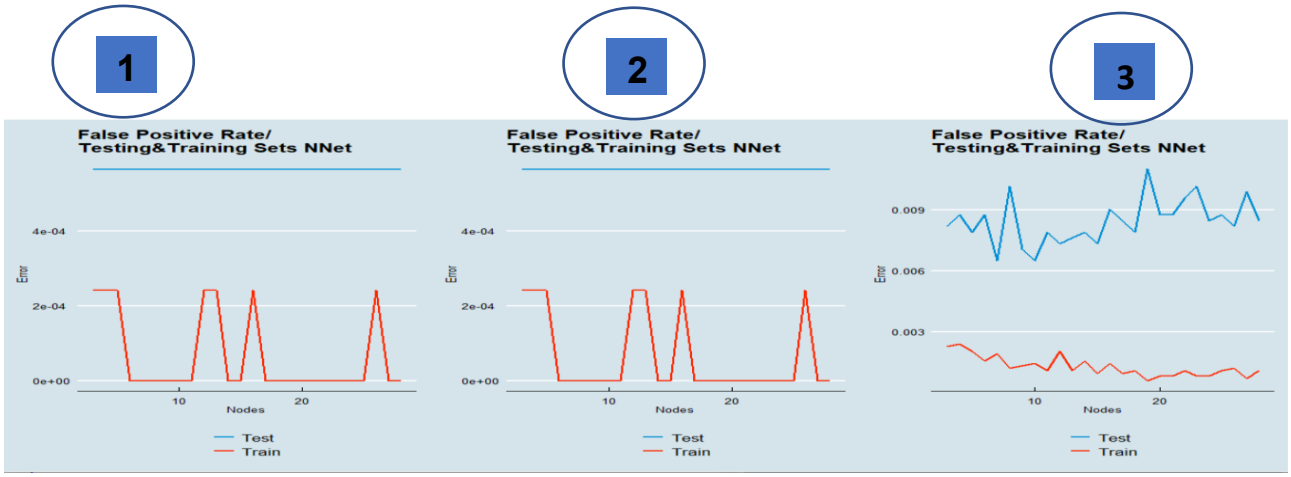
ניתן להבחין כי איזון הנתונים משפיע משמעותית על אמינות המודל.

Neural nets

מודל נוסף שהחלטנו לחקור וללמוד עליו הוא רשת נוירונים, מודל זה הינו המודל האחרון והקשה מכולם, במודל זה ביצענו מחקר מעמיק אודות רשת הנוירונים אשר זכינו לראות בהרצאה הראשונה והסתקרנו מתכונותיה. במודל יצרנו 25 תרשימים להשוואה.

להלן דוגמא מהתרשיתרשים מספר 7



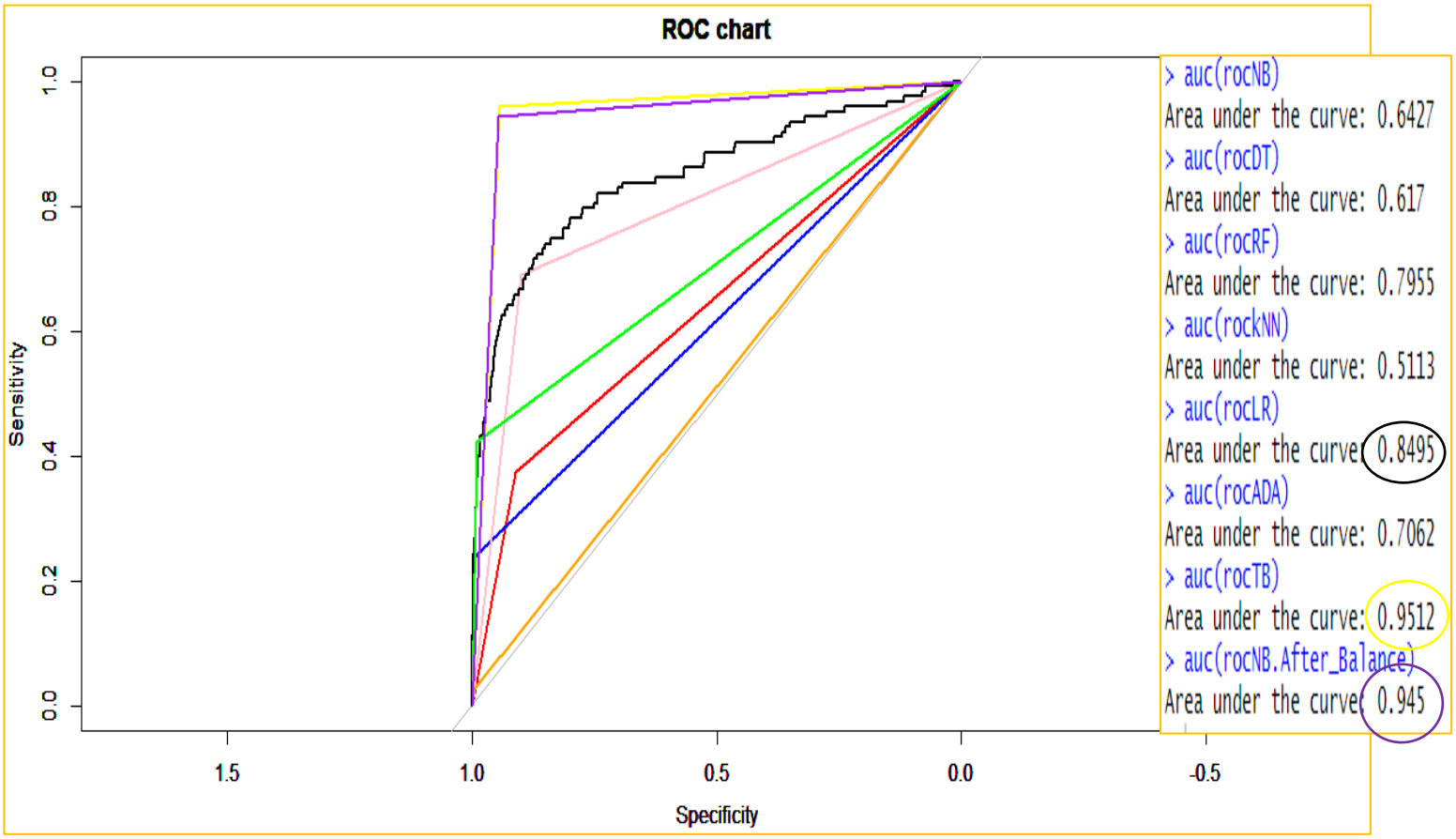


ראינו כי ב-Train set קיימת שגיאה לא יציבה, ביצענו מספר אלגוריתמים בכדי לייצב את השגיאה ואכן השגנו מטרה זו (**להלן התרשים מלעיל** . רשת הנוירונים מזהה באופן מובהק את בעיית הפרויקט, האם יפשטו רגל או לא.

מסקנות המודלים

בשלב הבא ביצענו בדיקת **rock chart** לצורך השוואה בין המודלים.

התוצאות שהתקבלו הן:



אדום – Naive Bayes | **כחול-** Decision Tree | **שחור –** logistic regression | **ורוד-** RF

כתום – Knn | **ירוק -** ADAboost | **צהוב –** TreeBag | **סגול -** Naive Bayes after balance

מתוצאות ההשוואה ע"פ התרשים הנ"ל ניתן לראות שהמודל הטוב ביותר לצורך בדיקה זו הוא:

TreeBag , שבו השטח שמתחת לגרף הוא **95.12%.**

מודל נוסף אשר נותן תוצאה גבוהה הינו : בייס נאיבי המאוזן (לאחר איזון נתוני עמודת המטרה)

שבו השטח שמתחת לגרף הוא **94.5%**, בנוסף גם המודל של רגרסיה לוגיסטית שבו השטח מתחת

לגרף היינו : **84.95%.**

סיכום

מעבודה זו הפקנו רבות על כריית ידע ולמידת מכונה, התמודדנו עם כמות רבה של נתונים פיננסיים, הגדרנו זמנים ויעדים מראש, ביצענו את העבודה בעזרת שיתוף פעולה בין חברי הצוות, ולמדנו רבות אודות ניתוח וניקיון נתונים, וכל זאת במטרה לבצע ניתוח מדויק של מידע השמור בדאטה פריים ואכן להגיע לחיזוי ותוצאות ברמת דיוק גבוהה לעמודת המטרה.

בנינו מודלים אשר לא נלמדו במהלך הקורס על מנת לחקור את הנושא לעומק וזאת בכדי להשיג תוצאות אופטימליות. הגענו למסקנה כי איזון הנתונים משפיע משמעותית על אמינות המודל.

לבסוף קיבלנו תוצאות המסבות לנו סיפוק רב באמצעות מחקר מעמיק ויצירת מודלים מעניינים.