Stockholm School of Economics
7313 – Data Science Analytics

Business Use Case Report

For iZettle

Haihui, Li	41506
Bing, Fu	41507

Stockholm School of Economics
7313 – Data Science Analytics
Haihui Li, Bing Fu

Introduction

Business use case

iZettle is a FinTech company focusing on small companies. It mainly covers two business areas: payment facilitation and loan lending. The payment facilitation area is the bulk part of iZettle's business model for the moment. In this line of business, iZettle persuades small enterprises into installing iZettle credit card readers and charge 1.85% of the total amount in every transaction via the card readers as service fees. Compare to traditional card reader solutions offered by banks, iZettle doesn't charge a monthly subscription fee, which is 5000 to 6000 SEK, according to an iZettle employee. For this reason, iZettle can approach small companies and facilitate transactions for them with iZettle card readers.

Currently, iZettle is considering expanding their business in the toy store area, where some online toy stores are planning to go offline and set up physical stores. Five of the twelves online stores have approached iZettle for their card reader solutions. But the problem is: "Whom should iZettle focus on?"

Hereto, we initiate a business use case to address this line of questions, but we don't confine ourselves only to the toy store industry. Can we develop a model to distinguish potential customers with financial data? The target is whether a small enterprise is worth to approach. The features are financial data such as turnover.

We will first start with identifying potential toy stores out of the 12 with unsupervised learning. By identifying the target customers, we can learn features needed for our model design. From there, we can develop a more general model to identify potential customer.

The business value of this model is that it can save costs of searching potential customers and help make better business decisions for iZettle.

Data Description

We make use of two datasets: IZettleAR and IZettleOrder. IZettleAR is dataset contains 2015-2018's financial data of the 12 companies. IZettleOrder is dataset with 2014-2015's online transaction data of customers. There are connected with merchant_id. In this business use case, we focus more on IZettleAR and use IZettleOrder as a reference to check and estimate data. For example, we tried to estimate 2014 and 2015 turnover values in IZettleAR for one of the stores by grouping and aggregating IZettleOrder.

IZettleAR has 60 observations and 8 variables. It is 2014-2018's financial data of the 12 toy store companies. The 8 variables are:

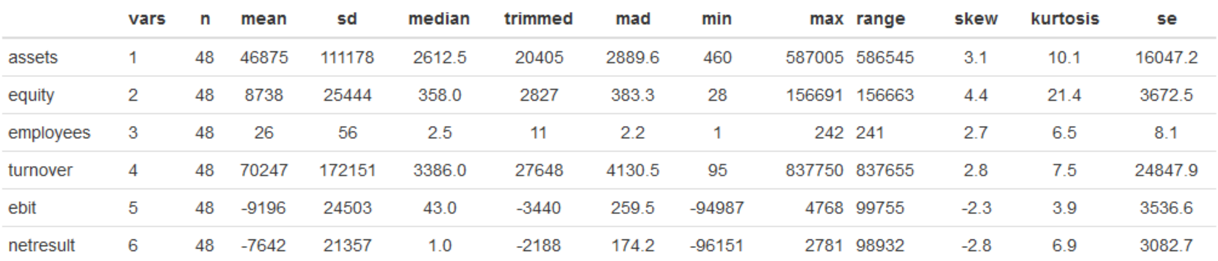
`merchant_id`: Company id
`year`: Year of annual report
`assets`: Total assets in kSEK
`equity`: Total equity in kSEK
`employees`: Number of employees
`turnover`: Turnover in kSEK

`ebit`: Earnings before interest and taxes in kSEK
`netresult`: Net results in kSEK

However, the dataset doesn't come in perfect form. First, it use 100K SEK for all 2014 and 2015 `turnover` values. Second, it has so many missing values. After fixing the input problem of `turnover`, we fill in the missing values with three general strategies:

1. Assume a certain ratio to back out value. For example, in the case of `merchant_id = 2026296`, we fill in the 2014 turnover by the 2014 `turnover/ebit` is the same with 2015, and back out 2014's `turnover` with the ratio and 2014's `ebit`
2. Assume values are the same over the years. For example, we fill in employee values with the one in later years
3. Refer to IZettleOrder for 2014 and 2015's turnover values
4. Omit observations missing all the values, except `year`

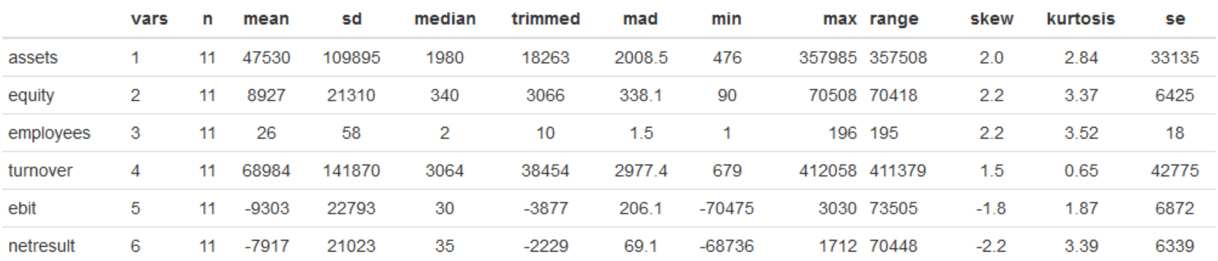
In this manner, we drop 12 of the 60 observations (a 20% loss) and completely disregard `merchant_id =` 8244704, because it has no values at all. As a results, we only study 11 of the 12 firms. The left data is like:



	vars	n	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
assets	1	48	46875	111178	2612.5	20405	2889.6	460	587005	586545	3.1	10.1	16047.2
equity	2	48	8738	25444	358.0	2827	383.3	28	156691	156663	4.4	21.4	3672.5
employees	3	48	26	56	2.5	11	2.2	1	242	241	2.7	6.5	8.1
turnover	4	48	70247	172151	3386.0	27648	4130.5	95	837750	837655	2.8	7.5	24847.9
ebit	5	48	-9196	24503	43.0	-3440	259.5	-94987	4768	99755	-2.3	3.9	3536.6
netresult	6	48	-7642	21357	1.0	-2188	174.2	-96151	2781	98932	-2.8	6.9	3082.7

Table1. Data Description

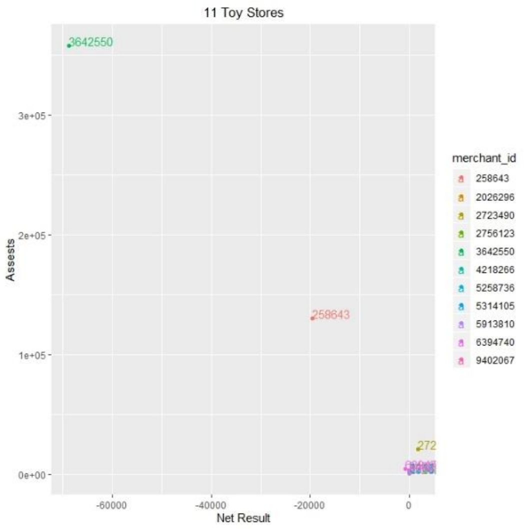
To better illustrate the data of different merchants, we group them by `merchant_id` by taking averages. This data has 11 observations and 8 variables.



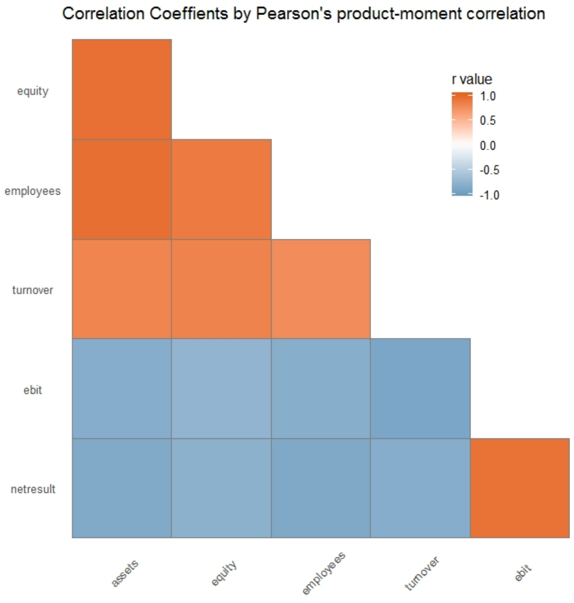
	vars	n	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
assets	1	11	47530	109895	1980	18263	2008.5	476	357985	357508	2.0	2.84	33135
equity	2	11	8927	21310	340	3066	338.1	90	70508	70418	2.2	3.37	6425
employees	3	11	26	58	2	10	1.5	1	196	195	2.2	3.52	18
turnover	4	11	68984	141870	3064	38454	2977.4	679	412058	411379	1.5	0.65	42775
ebit	5	11	-9303	22793	30	-3877	206.1	-70475	3030	73505	-1.8	1.87	6872
netresult	6	11	-7917	21023	35	-2229	69.1	-68736	1712	70448	-2.2	3.39	6339

Table2. Data Description by Merchant

We see that assets, equity, employee number, turnover, EBIT, and net income vary a lot among different companies. From the skewness and kurtosis on Table 2, we see that big companies have high assets, turnover, and equity and a lot of employees. Meanwhile, some companies make huge losses. If we depict the 11 firms with two dimension- `assets` and `netresult` we see that big companies have big losses while small ones have positive net income. It seems to be possible to group companies by their scale and net income.



Graph1. Toy Stores


Correlation Coeffients by Pearson's product-moment correlation

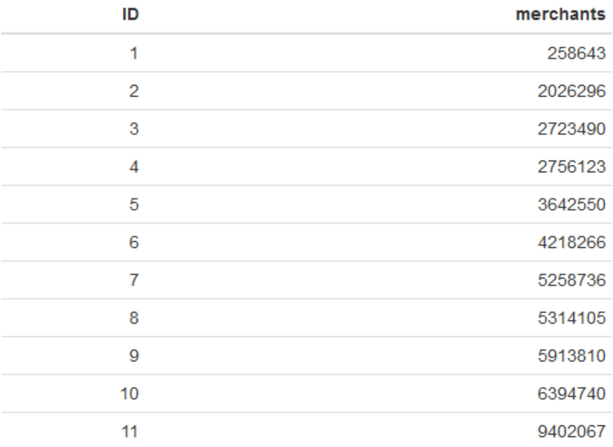
Graph2. Correlation

We also notice that `assets`, `equity`, `turnover`, and `employees` are highly correlated, and , `ebit` and `netresult` are highly correlated. This is expceted as a larger company tends to have more turnover and employees, and net income is calculated from EBIT. The take away is that we should aviod colinearitying a problem by choosing one feature of the related features, but not all of them.

Method

Unsupervised Learning - Whom Should iZettle Focus on

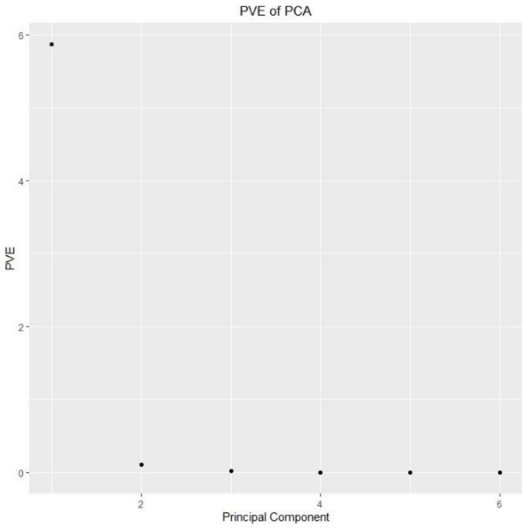
To identify the potential customers for iZettle, we conduct Principal Component Analysis and K-Mean Clustering to find out groups on the grouped data. PCA is necessary for K-Mean Clustering, as we have more than two variables to plot. In addition, we have mentioned that some variables are highly correlated. For K-Mean Clustering, we specify it to look for two groups with 50 different initial configurations. To better plot in the merchants, we use IDs as follow to represent them:



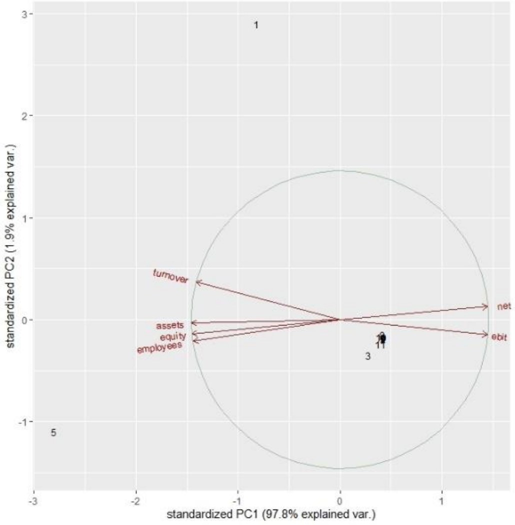
ID	merchants
1	258643
2	2026296
3	2723490
4	2756123
5	3642550
6	4218266
7	5258736
8	5314105
9	5913810
10	6394740
11	9402067

Table3. Merchant ID

In the PCA method, we find that the first principal component captures almost everything of the features variation, by standard deviation of 97.8%. Two groups are discerned, one with positive PC1 score and negative PC2 score, the other with different scores. With respect to the factor loadings, we see that PC1 is mainly drive by two forces: a negative force – company scale (`assets`, `equity`, `turnover`, and `employees`) and a positive force – net income (`ebit` and `netresult`). The two merchant outside of the circle in Graph 4 are merchant 258643 and 3642550.

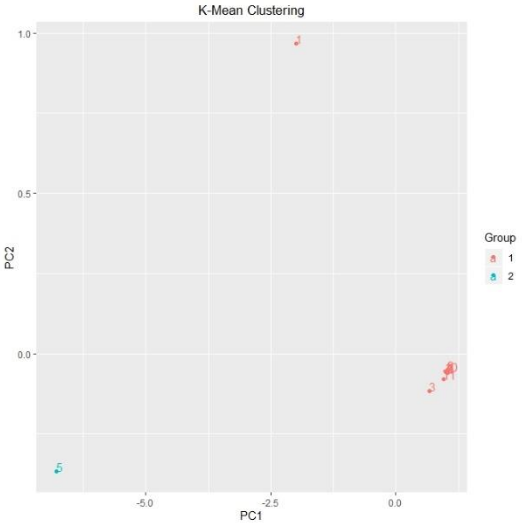


Graph3. PVE



Graph4. Merchant PC Score

In the K-Mean Clustering, we catch merchant 3642550 as an outsider, while other merchants are in the same group.

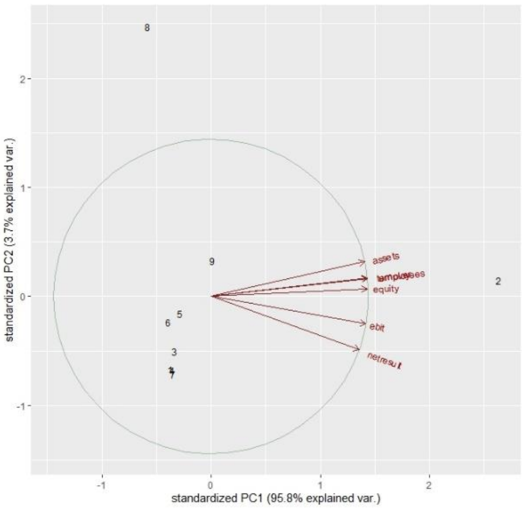


Graph5. K-Mean Clustering with Two Groups

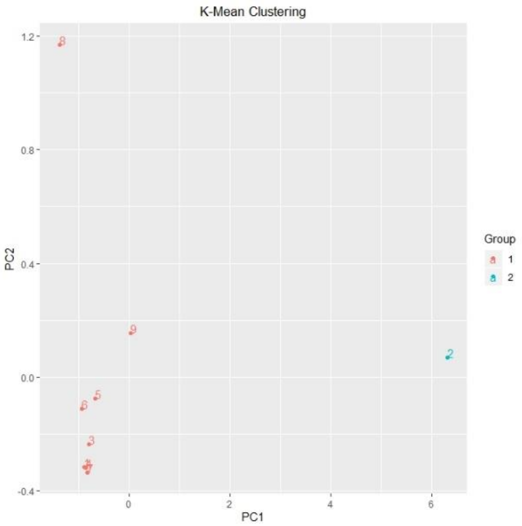
But who are the customers iZettle want to deal with? Or, which group is which? Through analyzing merchant 258643 and 3642550, we notice that they have very large turnover and negative net income. It means that they can't be the potential customers, because their turnover is so high that installing iZettle card readers will have a negative impact on the P/L, compared to subscribing traditional card readers, as calculated with the following formula:

$$\text{Monthly IZettle Fees} = \text{Turover(kSEK)} \times 1.85\% \times 1000 \div 12$$

Therefore, the potential customers are in the red group. Through the PCA method, we see that the larger a company is, the less likely it being a potential customer. Also, the more scaled profit it has, the more likely it being a potential customer.



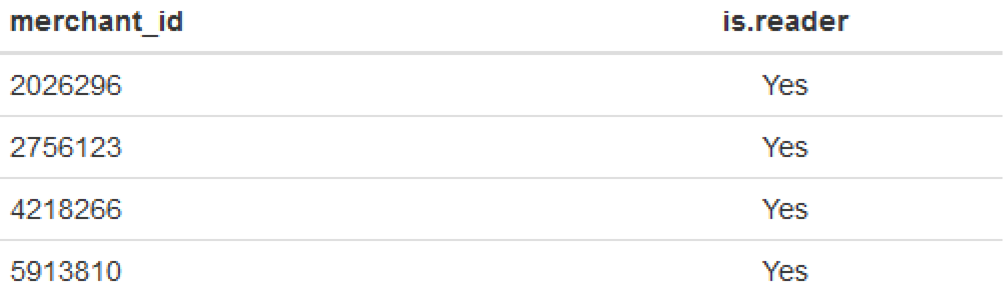
Graph6. PCA Second Time



Graph7. K-Mean Clustering Second Time

We carry on our grouping methods with the remaining companies. This time we single out merchant 2026296, which has a relatively small turnover. But there isn't any distinct pattern in this further subgrouping. We can't rule out any of those companies.

However, the first attempt tells us that company scale and company net profit play important roles in separating the companies. With the formula we established, we scrutinize the annual data year by year, narrow the number of potential companies down to four after scrutinizing the original data:

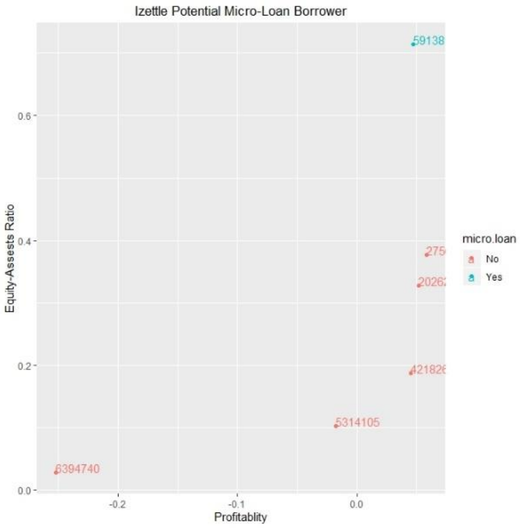


merchant_id	is.reader
2026296	Yes
2756123	Yes
4218266	Yes
5913810	Yes

Table4. Merchants to Approach

Though merchant 5258736 and 6394740 have a Monthly IZettle Fees below 5000 SEK, but if we look into yearly data, they have a Monthly IZettle Fees more than 5000 SEK in recent years. With this train of thoughts, we get the values of our target `is.reader`. On top of that, we can build more general models with supervised learning.

In addition, merchant 5314105 can be indifferent, as its Monthly IZettle Fees would be just above 6000 SEK, equal to the highest traditional subscription fee. But if it shows interest in iZettle card readers, iZettle should avoid it, because it may give up using the card reader in the midway.



Graph8. Merchants 5913810 is a reliable borrower

To identify reliable micro-loans borrower, we require that the company should have an equity-to-assets ratio higher than 50% and net profit ratio (`netresult/turnover`) larger than 0, so we add these two features to the grouped data. In this way, we find that a potential loan taker would be merchant 5913810.

Supervised Learning - A General Model

In the last unsupervised learning session, we find a very general rule to rule out companies. If a company's Monthly IZettle Fees is larger than 5000 SEK, it won't be interested in iZettle card reader, because the company would be better off to take the traditional subscription offered by banks. We can back out that the turnover is 3243 kSEK, from the following formula:

$$\text{Monthly IZettle Fees} = \text{Turover(kSEK)} \times 1.85\% \times 1000 \div 12$$

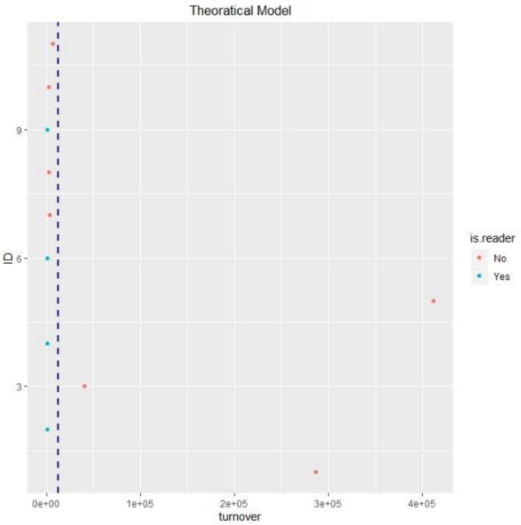
Based on this theory, we can develop a decision tree model by hand. But the problem with this approach is that this tree's decision on a company changes from year to year. For example, if a company has Monthly IZettle Fees 200 SEK and 6000 SEK in year 1 and year 2 respectively. The classifier will grant it in year 1, but overturn its decision in year 2.

Therefore, apart from a theoretical tree classifier, we could train an appropriate model to see if time effect (`year`) plays a role.

1. Theoretical: One Internal Node Tree Model

We can design a one-internal node Tree Classifier, by setting a turnover threshold of 3243kSEK on the average turnover of merchants.

$$Potential\ Customer\ = \begin{cases} Yes, & if\ tunover\ average <\ 3243kSEK \\ No, & if\ turnover\ average\ >= 3243kSEK \end{cases}$$



Graph9. Theoretical Tree Classifier

However, it doesn't perform well. As shown in Graph9, it can't not completely separate potential customers from other stores. The Test Accuracy is 63.3%, if we include merchant 5314105, the Test Accuracy will be 72.7%. The problem is with the average turnover, because it can be affected by extreme values and it doesn't put higher weights on turnover of recent years.

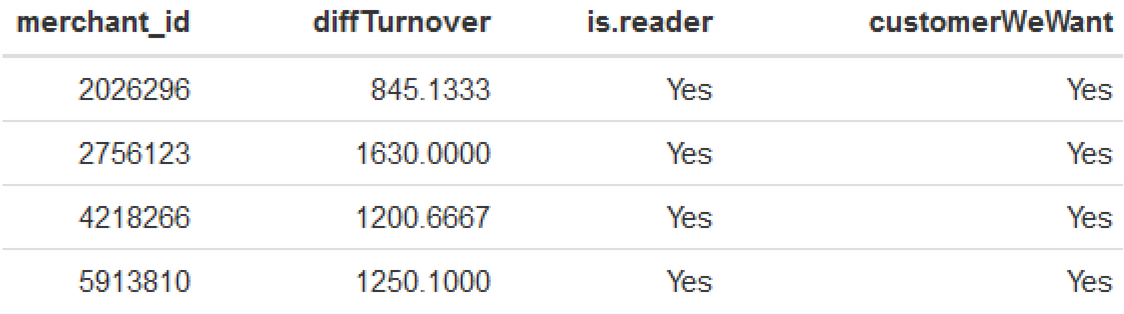
A natural solution is to put different weights on different year's turnover while calculating the average turnover. The more recent the data is, the more weight it gets.

$$Potential\ Customer\ = \begin{cases} Yes, & if\ weighted\ tunover\ average\ <\ 3243kSEK \\ No, & if\ weighted\ turnover\ average\ >=3243kSEK \end{cases}$$

For example, we have three years' turnover of a merchant. We assign a weight 1/ (1+2+3) to year 1, 2/(1+2+3) to year 2, and, 3/(1+2+3) to year three. In this way, we can bag all the potential customers with our data.

We develop a Classification Tree by hand. The tree model is wrapped up in the function "izreader". It requires two argument: the data and the turnover threshold we want to use, see Appendix for details.

Results of Theoretical Model



merchant_id	diffTurnover	is.reader	customerWeWant
2026296	845.1333	Yes	Yes
2756123	1630.0000	Yes	Yes
4218266	1200.6667	Yes	Yes
5913810	1250.1000	Yes	Yes

Table5. Improved Theoretical Model

What we are doing here is translating our subjective decision-making process into an automatic classification model. If we have the list of the true potential customers, we will be able to improve our model.

2. Statistical Learning Models

We use 50% of the original data set to fit five different types of models: Logit, LDA, QDA, Tree, and SVM with linear kernel. The specifications for training are five-time repeated ten-fold cross-validation and optimal specificity, as the positive case is "No" while we want to spot potential customers. In addition, we tune our SVM model with grid search of Cost.

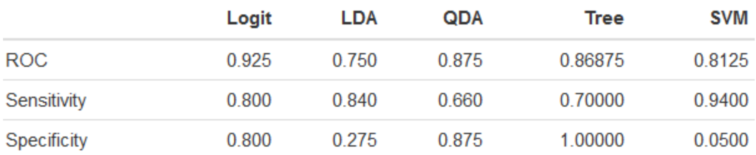
We use two feature sets. The first one {`year, turnover`}, and the second one is {`year, profitablility, equity.to.assets`}. Here are the definitions:

```
profitablility: netresult/turnover
```

```
equity.to.assets: equity/assets
```

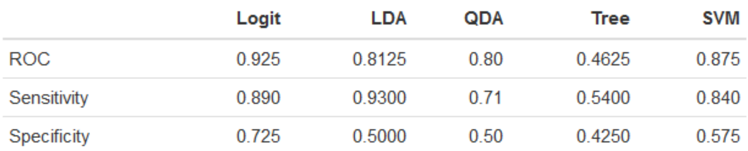
Results of Statistical Learning Models

The first feature set yields better results.



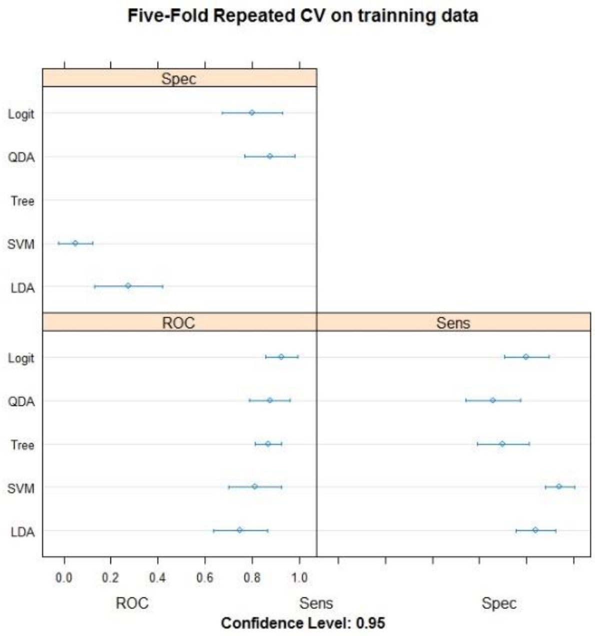
	Logit	LDA	QDA	Tree	SVM
ROC	0.925	0.750	0.875	0.86875	0.8125
Sensitivity	0.800	0.840	0.660	0.70000	0.9400
Specificity	0.800	0.275	0.875	1.00000	0.0500

Table6. First Feature Set



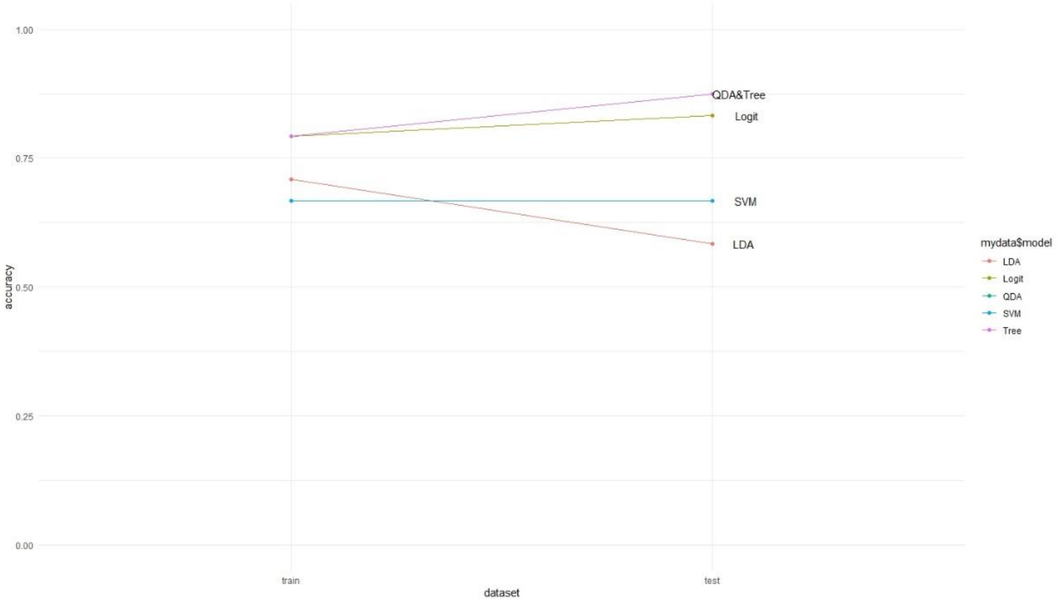
	Logit	LDA	QDA	Tree	SVM
ROC	0.925	0.8125	0.80	0.4625	0.875
Sensitivity	0.890	0.9300	0.71	0.5400	0.840
Specificity	0.725	0.5000	0.50	0.4250	0.575

Table7. Second Feature Set

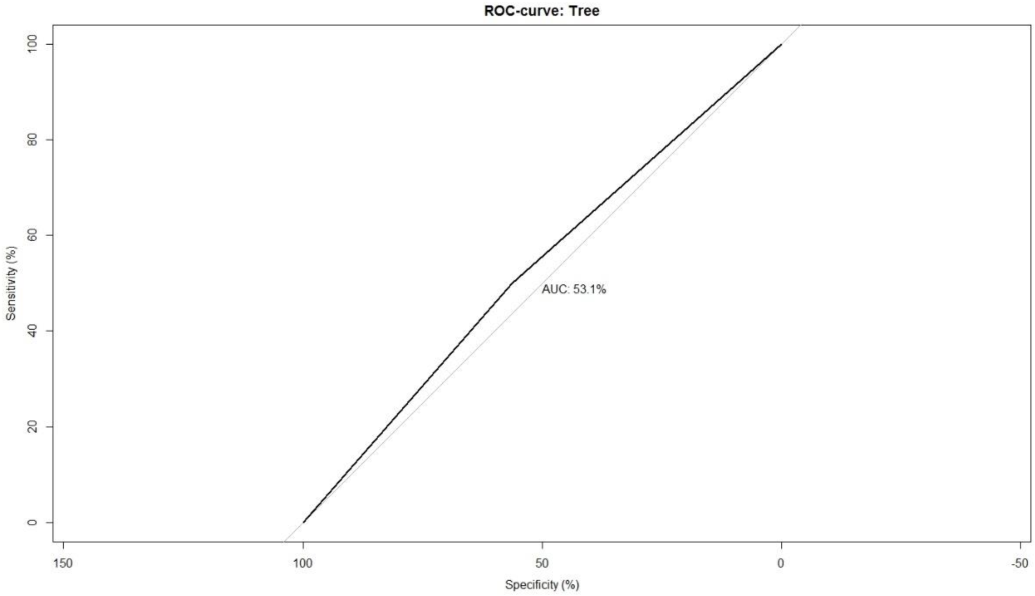


Graph10. First Feature Set Model Comparison

Our best model is Tree with the second feature set. Compared to other models, except QDA, it is extremely sensitive to potential customers, with a specificity of 1. It is better than QDA, because QDA needs to estimate a lot of parameters with so few observations. The most important feature is turnover, while year doesn't explain anything.



Graph11. First Feature Set Model Accuracy



Graph12. First Feature Set Model Accuracy

In conclusion, in the statistical learning method, the effect of time doesn't play a role. But the tree method is may use a turnover cutoff similar to our theoretical model. Apart from that, it is trained on quite a few observations. Therefore, we suggest using our theoretical model "izreader", for it captures the reality of the line of the business.

Discussion

iZettle is looking to expand its business by attracting new users. Our business use case creates business value in that it helps iZettle spot potential customers and saves the costs of customers searching. We identify four potential customers, who will bring in annual reader services fees of 77,922 SEK. If we include, the indifferent merchant, the tally would be 12,4860 SEK per year. If we discount the tally with 1% interest rate as a perpetuity, it would be worth of 12,486,000 SEK. This is only the case of toy stores, our model will also apply to other businesses, for it only need annual turnover data.

Assume that the indifferent monthly turnover is 3243kSEK, the annual reader services fees would be 60,000 SEK. If we can identify 100 such customers a year, there would be an extra 6,000,000 SEK revenue to iZettle. We can achieve that by digging in data base, finding potential customers and talking them into installing iZettle card readers, for who might not know that having credit card readers is also a possible option.

iZettle card readers may cost more than their selling price 199 SEK. That is why we should avoid indifferent merchants, as they may give up using iZettle card readers midway, and that would be a loss to iZettle.

Our recommendations are:

1. Merchant 2026296, 2756123, 4218266, and 5913810 are potential customers for the card reader business. Merchant 5314105 may be indifferent, so if it approaches iZettle, turn the offer down.
2. Merchant 5913810 is a reliable loan borrower, if we require that the company should have an equity-to-assets ratio higher than 50% and net profit ratio larger than 0.
3. iZettle should make use of our minimalistic theoretical model “izreader” to screen merchants in the database and spot potential customers. The cost is basically nothing compared to the potential rewards.
4. iZettle should consider establishing an offline sales force team to sell iZettle readers, which focus on physical stores with very few employees. For instance, no more than 3 employees.
5. iZettle should consider the following problems: What should they do to retain old customers when they grow too large and realize that the traditional credit card payment solutions are cheaper? How iZettle can grow with these customers?
6. Online payment solutions such as Alipay and Swish can be substitutes for SMEs. iZettle should consider how they can fend off these new players.

Appendix

Theoretical Model

```
izreader =
function(data, turnoverThred){
  library(dplyr)
  library(ggplot2)
  library(formattable)
  dfsth = data
  mid = unique(dfsth$merchant_id)
  for (i in 1: length(mid)){
    mdf = dfsth[dfsth$merchant_id == mid[i],]
    nyears = nrow(mdf)
    wghs = 1:nyears
    wghs = wghs/sum(wghs)
    # Matrix Multiplication
    avg.turnover = wghs %*% mdf$turnover
    dfsth$diffTurnover[dfsth$merchant_id == mid[i]] = avg.turnover
  }
  result_tabble =
    unique(dfsth %>% select(merchant_id,diffTurnover)) %>%
    mutate(is.reader = if_else(diffTurnover < turnoverThred, "Yes","No"))
  # Show the table
  print(formattable(result_tabble))
  result_tabble$is.reader = as.factor(result_tabble$is.reader)
  return(result_tabble)
}
```

Calling Packages

```
library(dplyr)
library(RMySQL)
library(ggplot2)
library(ggbiplot)
library(caret)
library(doSNOW)
library(psych)
library(lubridate)
library(formattable)
library(tidyr)
library(pROC)
library(AUC)
setwd("C:/Users/41506/7313/izttle/business use case")
```

Getting and Processing iZettle Order Data

```
rm(list = ls())
#setwd("D:/7313 prediction model")
#library(RMySQL)
#con = dbConnect(MySQL(), dbname = "ToyStorey",
```

```r
#host = "db.cfda.se", port = 3306, user = "toystorey",
#password = "toys@sse")
#df = fetch(dbSendQuery(con,"SELECT * FROM IZettle.IZettleOrder"), n=-1)
#write.csv(df,"izt_order.csv")
iz = read.csv('izt_order.csv')
iz = iz[,-1]

#================================ Preparing IZettle Order Transaction Data ===
==================================
# Hold out the test data(rows whose target is NA)
iz = iz[!is.na(iz$target),]

# Drop 'device': it is the target itself
dt_iz = subset(iz,select = -device)

# Turn datastamp into month (a factor with 12 levels)
dt_iz$datestamp = as.Date(dt_iz$datestamp)
dt_iz$purchase_month = months(dt_iz$datestamp)
dt_iz$purchase_month = as.factor(dt_iz$purchase_month)



# Factorizing(target,currency,gender,country, merchant_id)
dt_iz$currency = as.factor(dt_iz$currency)
dt_iz$gender = as.factor(dt_iz$gender)
dt_iz$country = as.factor(dt_iz$country)
# Target needs other level names!!!!!!!
dt_iz$target = ifelse(dt_iz$target == 1, "Desktop","Others")
dt_iz$target = as.factor(dt_iz$target)
dt_iz$merchant_id = as.factor(dt_iz$merchant_id)

dt_iz$cid = as.factor(dt_iz$cid)


# Fixing Purchase Amount
# Drop Currency, because it provides the same information as country

#1. Keep Purchase Amount > 0(It can't be negative)
dt_iz = dt_iz[dt_iz$purchase_amount > 0,]

#2. Divide Purchase Amount by 100.(It hasn't properly indicated "cent")
dt_iz$purchase_amount = dt_iz$purchase_amount/100

#3. Currency exchange: changing other currencies into SEK
eur_index = which(dt_iz$currency == "EUR")
dt_iz[eur_index,"purchase_amount"] = dt_iz[eur_index,"purchase_amount"]*10
dt_iz[eur_index,"currency"] = "SEK"

nok_index = which(dt_iz$currency == "NOK")
```

```
dt_iz[nok_index,"purchase_amount"] = dt_iz[nok_index,"purchase_amount"]*1.04
dt_iz[nok_index,"currency"] = "SEK"



#==== Birth Year Modifying
# 1. Add a "MissingYear" Value
dt_iz$MissingYear = ifelse(is.na(dt_iz$birthyear),"Y","N")
dt_iz$MissingYear = as.factor(dt_iz$MissingYear)


#2. Add a "fake_year" feature (Yes: reporting fake birth year; No: reporting
real birth year)
# Identify fake birthyear (birthyear < 1965 and birthyear = 2011)

dt_iz$FakeYear = rep("N",length(dt_iz$id))
dt_iz$FakeYear = ifelse(dt_iz$birthyear <= 1965|dt_iz$birthyear == 2011, 'Y',
'N')
dt_iz$FakeYear[is.na(dt_iz$FakeYear)] = 'Y'
dt_iz$FakeYear = as.factor(dt_iz$FakeYear)


#3. Filling birthyear's none values with average
dt_iz$birthyear[is.na(dt_iz$birthyear)] = round(mean(dt_iz$birthyear,na.rm =
T))
```

Getting and Processing IZettle Annual Report Data

```
#============================== Getting IZettle Merchant Data
con = dbConnect(MySQL(), dbname = "ToyStorey",
                host = "db1.cfda.se", port = 3306,
                user = "toystorey", password = "toys@sse")
df = fetch(dbSendQuery(con, "Select *
                       FROM IZettle.IZettleAR"), n=-1)
df.backup = df
#============================== Preparing Merchant Data ====================
#  Fixing 2014 AND 2015 Turnover
for (i in 1:nrow(df)){
  if (is.na(df[i,"turnover"])){
    next
  }
  else {
    if (df[i,"year"] == 2014) {
      df[i,"turnover"] = df[i,"turnover"]*100
    }
    else if (df[i,"year"] == 2015){
      df[i,"turnover"] = df[i,"turnover"]*100
    }
    else
    {next}
  }
}

# =========================== Data Description ===========================

# assests, equity, employees, turnoverm ebit, netresults have NAs

#=========================== More Data Preparasion ========================
df$merchant_id = as.factor(df$merchant_id)
str(df)
# Drop merchant_id = 8244704, becasue it has basically no values

df= df %>%
      filter(merchant_id != "8244704")

# merchant_id = 2026296, missing 2014 turnover, so fill in 2015's turnover/ #
 ebit ratio * 2014 EBIT
df[df$merchant_id == "2026296" & df$year == 2014,"turnover"] =
  df[df$merchant_id == "2026296" & df$year == 2015,"turnover"] *
  df[df$merchant_id == "2026296" & df$year == 2014,"ebit"]/df[df$merchant_id
== "2026296" & df$year == 2015,"ebit"]


# merchant_id = 2756123, misisng employess values in 2014, 2015, 2016
# Use 2017's employee number 1
df[df$merchant_id == "2756123"&(df$year %in% c(2014,2015,2016)),"employees"]
= df[df$merchant_id == "2756123"& df$year == 2017,"employees"]

# merchant_id = 3642550 missing all values in 2018, drop it
df = df %>%
        filter(!(merchant_id == "3642550" & year == 2018))

# merchant_id = 4218266 missing employees value in 2016
# Use 2017's employees value
df[df$merchant_id == "4218266"&df$year == 2016,"employees"] = df[df$merchant_
id == "4218266"&df$year == 2017,"employees"]

# merchant_id = 4218266 missing turnover value in 2014
# use the transaction data to fill in
df[df$merchant_id == "4218266"&df$year == 2014,"turnover"] =
                                                        dt_iz %>%
                                                          filter(merchant_id
== 4218266, year == 2014) %>%
                                                          summarise(sales = s
um(purchase_amount)/1000)
# merchant_id = 4218266 missing turnover value in 2015
# use the transaction data to fill in
df[df$merchant_id == "4218266"&df$year == 2015,"turnover"] =
  dt_iz %>%
  filter(merchant_id == 4218266, year == 2015) %>%
  summarise(sales = sum(purchase_amount)/1000)

# merchant_id = 4218266 missing employees value in 2014, 2015
# Use 2017's employees value
df[df$merchant_id == "4218266"&df$year == 2014,"employees"] = df[df$merchant_
id == "4218266"&df$year == 2017,"employees"]
df[df$merchant_id == "4218266"&df$year == 2015,"employees"] = df[df$merchant_
id == "4218266"&df$year == 2017,"employees"]


# merchant_id = 5913810 missing assets, equity, employees value in 2014
# Use 2015's value
df[df$merchant_id == "5913810"&df$year == 2014,"assets"] = df[df$merchant_id
== "5913810"&df$year == 2015,"assets"]
df[df$merchant_id == "5913810"&df$year == 2014,"equity"] = df[df$merchant_id
== "5913810"&df$year == 2015,"equity"]
df[df$merchant_id == "5913810"&df$year == 2014,"employees"] = df[df$merchant_
id == "5913810"&df$year == 2015,"employees"]


# merchant_id = 6394740 miss netresult in 2017 and 2018
# Use 2017's EBIT and 2018's EBIT to approximate
df[df$merchant_id == "6394740"&df$year == 2017,"netresult"] = df[df$merchant_
id == "6394740"&df$year == 2017,"ebit"]
df[df$merchant_id == "6394740"&df$year == 2018,"netresult"] = df[df$merchant_
id == "6394740"&df$year == 2018,"ebit"]


# Drop the rows with no values at all. The reason is that those companies may
# not even exist back then

df = na.omit(df)
# In total, we lost 12 observations out of 60 (20%)
```

IZettle AR Data Description

```
# Descriptive
formattable(psych::describe(df[,-c(1,2)]),
            align =c("l","c","c","c","c", "c", "c", "c", "r"))

# Grouping data by merchant_id
df.group = df %>%
            group_by(merchant_id) %>%
            summarise_each(funs(mean))
df.group$year = as.integer(df.group$year)
df.group$employees = as.integer(df.group$employees)
df.group$id = 1:11
```

```r
# Descriptive of Group Data
formattable(psych::describe(df.group[,-c(1,2,9)]),
            align =c("l","c","c","c","c", "c", "c", "c", "r"))

# Ggplot it!
ggplot(df.group,
       aes(x = netresult, y = assets,color = merchant_id),label=merchant_id )
 +
  geom_point() +
  geom_text(aes(label=merchant_id),hjust=0, vjust=0) +
  ggtitle("11 Toy Stores") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylab("Assests") +
  xlab("Net Result")

# GGPLOT corelation graph
ggiraphExtra::ggCor(df[,-c(1,2)])
```

PCA

```r
# ========================== PCA ================================

merchants = as.character(unique(df.group$merchant_id))
mernames = as.character(1:11)
merchants = data.frame(ID = mernames, merchants = merchants)

formattable(merchants)

df.pca = as.data.frame(subset(df.group, select =  - c(merchant_id,year,id)))
pca.data = prcomp(df.pca, scale. = T)

# Plotting: Biplot and PVE plot
ggbiplot::ggbiplot(pca.data,circle = T, labels = mernames)
pve = pca.data$sdev^2
pve.df = data.frame(pve = pve, id = 1:length(pve))

ggplot(pve.df,
       aes(x = id, y = pve) ) +
  geom_point() +
  ggtitle("PVE of PCA") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylab("PVE") +
  xlab("Principal Component")
```

K-Mean Clustering

```r
# ====================== K-Mean Clustering =======================
set.seed(7313)
pca.mat = pca.data$x
km.out = kmeans(pca.mat,2,nstart = 50)
km.df = data.frame(ID = mernames, pca.mat[,c(1,2)], Group = km.out$cluster )
km.df$Group = as.factor(km.df$Group)

ggplot(km.df,
       aes(x = PC1, y = PC2,color = Group),label=ID ) +
  geom_point() +
  geom_text(aes(label=ID),hjust=0, vjust=0) +
  ggtitle("K-Mean Clustering") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylab("PC2") +
  xlab("PC1")



merchant_class = data.frame(mechant = merchants, class = km.out$cluster)
```

Second Run of PCA&K-Mean

```
# ===================== Decision Making: Rule out Merchant 1 and 5
# 1    258643
# 5   3642550
# They have following in commom:
#  1. High Assets and High Equity, high turnover
#  2. Poor profitability: negative netresult and ebit
# Are they the merchants we want to approach ?
# 286558*0.0185*1000/12
# Unlikely, monthly turnover*fees > 6000, puls they have negative results

# ===================== Subgroup ========================
df.group.sub =
  df.group %>%
      filter(merchant_id != "258643", merchant_id != "3642550")



merchants = as.character(unique(df.group.sub$merchant_id))
mernames = as.character(1:length(df.group.sub$merchant_id))
merchants = data.frame(ID = mernames, merchants = merchants)


df.pca = as.data.frame(subset(df.group.sub, select =  - c(merchant_id,year,i
d)))
pca.data = prcomp(df.pca, scale. = T)

# Plotting: Biplot and PVE plot

ggbiplot::ggbiplot(pca.data,circle = T, labels = mernames)
pve = pca.data$sdev^2
pve.df = data.frame(pve = pve, id = 1:length(pve))

ggplot(pve.df,
       aes(x = id, y = pve) ) +
  geom_point() +
  ggtitle("PVE of PCA") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylab("PVE") +
  xlab("Principal Component")



set.seed(7313)
pca.mat = pca.data$x
km.out = kmeans(pca.mat,2,nstart = 50)
km.df = data.frame(ID = mernames, pca.mat[,c(1,2)], Group = km.out$cluster )
km.df$Group = as.factor(km.df$Group)

ggplot(km.df,
       aes(x = PC1, y = PC2,color = Group),label=ID ) +
  geom_point() +
  geom_text(aes(label=ID),hjust=0, vjust=0) +
  ggtitle("K-Mean Clustering") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylab("PC2") +
  xlab("PC1")




df.group =
  df.group %>%
  mutate(profitablility = netresult/turnover, equity.to.assets = equity/asset
s)

df.group =
  df.group %>%
  mutate(IZettle.fees =  turnover*1000*0.0185/12)

df.group.sub =
  df.group %>%
  filter(merchant_id != "258643", merchant_id != "3642550")

df.iZettle.reader =
df.group.sub %>%
 filter(IZettle.fees <= 5000)
```

Indentifying Customers and Borrowers

```
# ======================== Find Customers ======================================
=========
reader.merchant.id = df.iZettle.reader$merchant_id
df$is.reader = ifelse(df$merchant_id %in% reader.merchant.id,"Yes","No")
df$IZettle.fees = df$turnover*1000*0.0185/12
```

```
# But merchant 5258736, 5314105, 6394740 are nor potential customers if we lo
ok into
# yearly data instead of average data
df$is.reader[df$merchant_id == "5258736"] = "No"
df$is.reader[df$merchant_id == "5314105"] = "No"
df$is.reader[df$merchant_id == "6394740"] = "No"
df$is.reader = as.factor(df$is.reader)

reader.table = unique(df %>%
  group_by(merchant_id) %>%
  select(merchant_id, is.reader))

reader.table = reader.table[reader.table$is.reader == "Yes",]
formattable(reader.table, align =c("l","c","c","c","c", "c", "c", "c", "r"))

# But merchant 5314105 can be potential customers as its turnover is very clo
se to
# the threshold of IZettle fees < traditional card readers fixed fees

# ================ Identify Potential Customers with IZettle Reader==========
=

# We keep the only four potential customers
df.iZettle.reader = df.iZettle.reader[df.iZettle.reader$merchant_id %in%uniqu
e(df$merchant_id[df$is.reader == "Yes"]),]

df.group$is.reader = rep("No",nrow(df.group))
df.group$is.reader[df.group$merchant_id %in%unique(df$merchant_id[df$is.reade
r == "Yes"])] = "Yes"


plot.new()

ggplot(df.group,
       aes(x = profitablility, y = equity.to.assets,color = is.reader),label=
merchant_id ) +
  geom_point() +
  geom_text(aes(label=merchant_id),hjust=0, vjust=0) +
  ggtitle("IZettle Potential Customers") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylab("Equity-Assests Ratio") +
  xlab("Profitablity")




# ================ Identify Promising Micro-Loan borrower ===========

df.iZettle.reader =
  df.iZettle.reader %>%
  mutate(micro.loan = ifelse(equity.to.assets >0.5 & profitablility > 0, "Yes
","No"))

df.iZettle.reader$micro.loan = as.factor(df.iZettle.reader$micro.loan)

plot.new()

ggplot(df.iZettle.reader,
       aes(x = profitablility, y = equity.to.assets,color = micro.loan),label
=merchant_id ) +
  geom_point() +
  geom_text(aes(label=merchant_id),hjust=0, vjust=0) +
  ggtitle("IZettle Potential Micro-Loan Borrower") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylab("Equity-Assests Ratio") +
  xlab("Profitablity")

# We can loan to merchant 5913810, with criteria: 1. more than 50% E/A ratio,
# 2. positive profitability
```

Theratical Model

```
# Theoratical

df.group$is.reader = ifelse(df.group$merchant_id %in% reader.table$merchant_i
d, "Yes","No")
df.group$is.reader = as.factor(df.group$is.reader)
df.group$binary = ifelse(df.group$is.reader == "Yes", 1,0)

ggplot(df.group,
       aes(x = turnover, fill= is.reader),binwidth = 10) +
  geom_bar(position = "dodge") +
  ggtitle("Theoratical Model") +
  theme(plot.title = element_text(hjust = 0.5))+
  # To make it look more clear, we add the threshold by 10000, because the bi
ns are too wide
  geom_vline(aes(xintercept= 5000*12/(1000*0.0185)+10000),color="darkblue", l
inetype="dashed", size=1)

ggplot(df.group,
       aes(y = 1:11, x = turnover, color = is.reader)) +
  geom_point() +
  ggtitle("Theoratical Model") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylab("ID") +
  # To make it look more clear, we add the threshold by 10000
  geom_vline(aes(xintercept= 5000*12/(1000*0.0185)+10000),color="darkblue", l
inetype="dashed", size=1)
```

Theratical Model- IZREADER

```
# =========================== Improved Theoratical Model =============
# Assigning Different Weights
# Writting a funciton

izreader =
function(data, turnoverThred){
  library(dplyr)
  library(ggplot2)
  library(formattable)
  dfsth = data
  mid = unique(dfsth$merchant_id)
  for (i in 1: length(mid)){
    mdf = dfsth[dfsth$merchant_id == mid[i],]
    nyears = nrow(mdf)
    wghs = 1:nyears
    wghs = wghs/sum(wghs)
    # Matrix Multiplication
    avg.turnover = wghs %*% mdf$turnover
    dfsth$diffTurnover[dfsth$merchant_id == mid[i]] = avg.turnover
  }
  result_tabble =
    unique(dfsth %>% select(merchant_id,diffTurnover)) %>%
    mutate(is.reader = if_else(diffTurnover < turnoverThred, "Yes","No"))
  # Show the table
  print(formattable(result_tabble))
  result_tabble$is.reader = as.factor(result_tabble$is.reader)
  return(result_tabble)
}

sth = izreader(df,3243)

formattable(sth %>%
              filter(is.reader == "Yes") %>%
              mutate(customerWeWant = ifelse(merchant_id %in% c("2026296","27
56123","4218266","5913810"),
                                             "Yes","No")))



# we have too few observations, so we should use few features to avoid overfi
tting
```

Statistical Learning

```
# we have too few observations, so we should use few features to avoid overfi
tting

# ======================== Statistical Learning =======================

# ========================= First Feature Set ========================
df =
  df %>%
  mutate(profitablility = netresult/turnover, equity.to.assets = equity/asset
s)


features = c("year","turnover"
             ,"is.reader")
train.data = df[,features]




# spliting 50%/50% training and test sets
indexes = createDataPartition(train.data$is.reader,
                              times = 1,
                              p = 0.5,
                              list = F)

train = train.data[indexes,]
test = train.data[-indexes,]

# trainning models
ctrl = trainControl(method = "repeatedcv",
                    number = 10,
                    repeats = 5,
                    classProbs = T,
                    summaryFunction = twoClassSummary)

# Logit
fit.logit = caret::train(is.reader~., data = train, method = "glm",
                         family = binomial(), trControl = ctrl,
                         metric = "Spec")

logit.pred = predict(fit.logit, test)
confusionMatrix(logit.pred, test$is.reader)
logit.pred = predict(fit.logit, train)
confusionMatrix(logit.pred, train$is.reader)

# SVM with Linear Kernel
grid <- expand.grid(C = c(0,0.01, 0.05, 0.1, 0.25, 0.5, 0.75, 1, 1.25, 1.5,
1.75, 2,5))
fit.svm = caret::train(is.reader~., data = train,
                       method = "svmLinear",preProcess = c("center", "scale
"),
                       tuneLength = 10,
                       trControl = ctrl,
                       tuneGrid = grid,
                       metric = "Spec")
svm.pred = predict(fit.svm, test)
```

```
confusionMatrix(svm.pred, test$is.reader)
svm.pred = predict(fit.svm, train)
confusionMatrix(svm.pred, train$is.reader)

# LDA
fit.lda = caret::train(is.reader~., data = train, method = "lda",
                         trControl = ctrl,
                         metric = "Spec")

lda.pred = predict(fit.lda, test)
confusionMatrix(lda.pred, test$is.reader)
lda.pred = predict(fit.lda, train)
confusionMatrix(lda.pred, train$is.reader)


# QDA
fit.qda = caret::train(is.reader~., data = train, method = "qda",
                       trControl = ctrl,
                       metric = "Spec")

qda.pred = predict(fit.qda, test)
confusionMatrix(qda.pred, test$is.reader)
qda.pred = predict(fit.qda, train)
confusionMatrix(qda.pred, train$is.reader)


# Tree Methods
fit.tree = caret::train(is.reader~., data = train, method = "rpart",
                       trControl = ctrl,
                      metric = "Spec")

tree.pred = predict(fit.tree, test)
confusionMatrix(tree.pred, test$is.reader)
tree.pred = predict(fit.tree, train)
confusionMatrix(tree.pred, train$is.reader)





set.seed(7313)
results = resamples(list(
  Logit = fit.logit, LDA = fit.lda, QDA = fit.qda, Tree = fit.tree, SVM = fi
t.svm ))
dotplot(results,main = "Five-Fold Repeated CV on trainning data")

compare_vec = apply(results$values[,-1],2,mean,na.rm = T)
compare_mat = matrix(compare_vec,ncol = 5,nrow = 3, byrow = F)
compare_mat = data.frame(compare_mat,row.names = c("ROC","Sensitivity","Speci
ficity"))
colnames(compare_mat) = c("Logit","LDA","QDA","Tree","SVM")


formattable(compare_mat)


pred = predict(fit.tree, test, type = "prob")
plot.roc(train$is.reader, pred$No, percent = T, main = "ROC-curve: Tree", pri
nt.auc = T)


# =================== Second Feature Set ======================
features = c("year","profitablility","equity.to.assets"
             ,"is.reader")
train.data = df[,features]

glimpse(train.data)


# spliting 50%/50% training and test sets
indexes = createDataPartition(train.data$is.reader,
                              times = 1,
                              p = 0.5,
                              list = F)

train = train.data[indexes,]
test = train.data[-indexes,]

# trainning models
ctrl = trainControl(method = "repeatedcv",
                    number = 10,
                    repeats = 5,
                    classProbs = T,
                    summaryFunction = twoClassSummary)

# Logit
fit.logit = caret::train(is.reader~., data = train, method = "glm",
                         family = binomial(), trControl = ctrl,
                         metric = "Spec")

logit.pred = predict(fit.logit, test)
confusionMatrix(logit.pred, test$is.reader)


# SVM with Linear Kernel
grid <- expand.grid(C = c(0,0.01, 0.05, 0.1, 0.25, 0.5, 0.75, 1, 1.25, 1.5,
1.75, 2,5))
```

```
fit.svm = caret::train(is.reader~., data = train,
                       method = "svmLinear",preProcess = c("center", "scale
"),
                       tuneLength = 10,
                       trControl = ctrl,
                       tuneGrid = grid,
                       metric = "Spec")
svm.pred = predict(fit.svm, test)
confusionMatrix(svm.pred, test$is.reader)

# LDA
fit.lda = caret::train(is.reader~., data = train, method = "lda",
                       trControl = ctrl,
                       metric = "Spec")

lda.pred = predict(fit.lda, test)
confusionMatrix(lda.pred, test$is.reader)

# QDA
fit.qda = caret::train(is.reader~., data = train, method = "qda",
                       trControl = ctrl,
                       metric = "Spec")

qda.pred = predict(fit.qda, test)
confusionMatrix(qda.pred, test$is.reader)



# Tree Methods
fit.tree = caret::train(is.reader~., data = train, method = "rpart",
                        trControl = ctrl,
                        metric = "Spec")

tree.pred = predict(fit.tree, test)
confusionMatrix(tree.pred, test$is.reader)



set.seed(7313)
results = resamples(list(
  Logit = fit.logit, LDA = fit.lda, QDA = fit.qda, Tree = fit.tree, SVM = fi
t.svm ))
dotplot(results,main = "Five-Fold Repeated CV on trainning data")

compare_vec = apply(results$values[,-1],2,mean,na.rm = T)
compare_mat = matrix(compare_vec,ncol = 5,nrow = 3, byrow = F)
compare_mat = data.frame(compare_mat,row.names = c("ROC","Sensitivity","Speci
ficity"))
colnames(compare_mat) = c("Logit","LDA","QDA","Tree","SVM")
```

```
formattable(compare_mat)


logit.pred = predict(fit.logit, test)
confusionMatrix(logit.pred, test$is.reader)
logit.pred = predict(fit.logit, train)
confusionMatrix(logit.pred, train$is.reader)

lda.pred = predict(fit.lda, test)
confusionMatrix(lda.pred, test$is.reader)
lda.pred = predict(fit.lda, train)
confusionMatrix(lda.pred, train$is.reader)

qda.pred = predict(fit.qda, test)
confusionMatrix(qda.pred, test$is.reader)
qda.pred = predict(fit.qda, train)
confusionMatrix(qda.pred, train$is.reader)

tree.pred = predict(fit.tree, test)
confusionMatrix(tree.pred, test$is.reader)
tree.pred = predict(fit.tree, train)
confusionMatrix(tree.pred, train$is.reader)


svm.pred = predict(fit.svm, test)
confusionMatrix(svm.pred, test$is.reader)
svm.pred = predict(fit.svm, train)
confusionMatrix(svm.pred, train$is.reader)
```

Economic Value Calculation

```
df.iZettle.reader %>%
  select(merchant_id,IZettle.fees) %>%
  mutate(annualFees = IZettle.fees*12) %>%
  tally(annualFees)

df.group %>%
  select(merchant_id,IZettle.fees)%>%
  filter(merchant_id == "5314105") %>%
  mutate(annualFees = IZettle.fees*12) %>%
  tally(annualFees)

46888 + 7792
```