Biz2Credit / Biz2X

Data Science Communications Intern Assignment

Tung-Ting(Nelson) Wang

Feb 3, 2022

Table of Contents

Introduction:

(The completed code is on Github.)
To find out how devices usage interacts with the market and what kind of companies might be the potential valuable customers for Biz2Credit, I utilized Python to perform data wrangling, Excel to generate descriptive statistics and correlation, and RapidMiner to gain insights from the decision tree. Questions below are from Biz2Credit and me are answered in the Business Analytics sections:

Device Usage:

1. Which device contributed to the maximum number of applications in 12 months period?
2. How did the application distribution change over time?
3. Did "application channel" have any impact on the trend of device usage?
4. Which Industry contributed most in 2019?

Potential most valuable Customers:

1. What's the relationship between attributes?
2. How much amount of annual revenue can make the company a valuable customer?
3. What kind of customers can be potential valuable customers to Biz2Credit?

Data Wrangling:

(I'll use "-" to represent the code, including library and syntax. For example, df.info() will be "-info", df.drop() will be "-drop", etc. Besides, I'll use " " to represent the column name. For instance, the industry will be "Industry", the device will be "Device" etc.)

Null Value:

On Jupyter, We can easily discover that there are null values inside the dataset. In "-info", "Industry" and "Personal CreditScore" contain null-values. Besides, we can find out that "Personal CreditScore" and "Annual Revenue" also have null values represented by 0 in the Data Dictionary.

First, "Industry" only contains 21 null values, and it is a categorical attribute. In this case, replacing the null value with a categorical variable "Unkown" would be the best way to handle the null value rather than drop the entire column.

Second, "Personal CreditScore" contains 5170 null values, which is 5165(value 0)+5 (NaN in Python). The total percentage of null values is 12% (5170 / 41572). It's acceptable, thus, I use mean to replace all the null values.

Third, "Annual Revenue" contains 5732 null values, which is also a small amount of the whole dataset. Thus, I also use the mean to replace the null value.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41572 entries, 0 to 41571
Data columns (total 15 columns):
Business ID                    41572 non-null int64
Application ID                 41572 non-null int64
Application Date               41572 non-null datetime64[ns]
Customer Type                  41572 non-null object
Application Channel            41572 non-null object
Amount Requested               41572 non-null int64
Industry                       41551 non-null object
Census Region                  41572 non-null object
Business Legal Structure       41572 non-null object
Age of Business(Months)        41572 non-null int64
Personal CreditScore           41567 non-null float64
Annual Revenue                 41572 non-null int64
Device                         41572 non-null object
Self Directed                  41572 non-null object
Self Submit                    41572 non-null object
dtypes: datetime64[ns](1), float64(1), int64(5), object(8)
memory usage: 4.8+ MB
```

11	Personal CreditScore	Credit Score of the application. "0" means data is missing
12	Annual Revenue	Annual Revenue . "0" means data is missing

```
df["Industry"] = df["Industry"].fillna("Unknown")
```

```
df["Personal CreditScore"].loc[df["Personal CreditScore"]==0].count()
```

```
5165
```

Treatment:

```
#Personal Credit has null value both "NaN" and "0"(on codebook)
pc_mean = df["Personal CreditScore"].mean()
df["Personal CreditScore"]= df["Personal CreditScore"].fillna(pc_mean)
df["Personal CreditScore"] = df["Personal CreditScore"].replace(0,pc_mean)
```

Result:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41572 entries, 0 to 41571
Data columns (total 15 columns):
Business ID                    41572 non-null int64
Application ID                 41572 non-null int64
Application Date               41572 non-null datetime64[ns]
Customer Type                  41572 non-null object
Application Channel            41572 non-null object
Amount Requested               41572 non-null int64
Industry                       41572 non-null object
Census Region                  41572 non-null object
Business Legal Structure       41572 non-null object
Age of Business(Months)        41572 non-null int64
Personal CreditScore           41572 non-null float64
Annual Revenue                 41572 non-null int64
Device                         41572 non-null object
Self Directed                  41572 non-null object
Self Submit                    41572 non-null object
dtypes: datetime64[ns](1), float64(1), int64(5), object(8)
memory usage: 4.8+ MB
```

]: `df["Personal CreditScore"].loc[df["Personal CreditScore"]==0].count()`

]: 0

```
df["Annual Revenue"].loc[df["Annual Revenue"]==0].count()
```

5732

Treatment:

```
#Annual Revenue
ar_mean = df["Annual Revenue"].mean()
df["Annual Revenue"]= df["Annual Revenue"].replace(0,ar_mean)
```

Result:

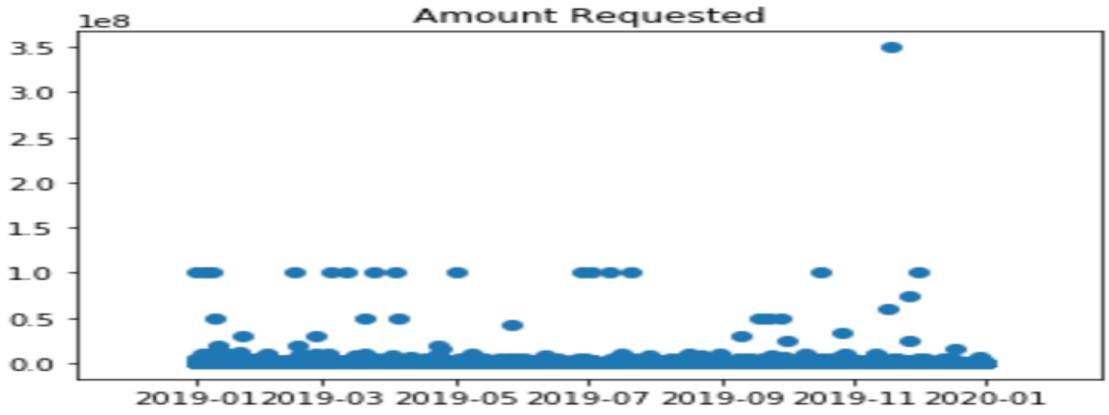
```
df["Annual Revenue"].loc[df["Annual Revenue"]==0].count()
```

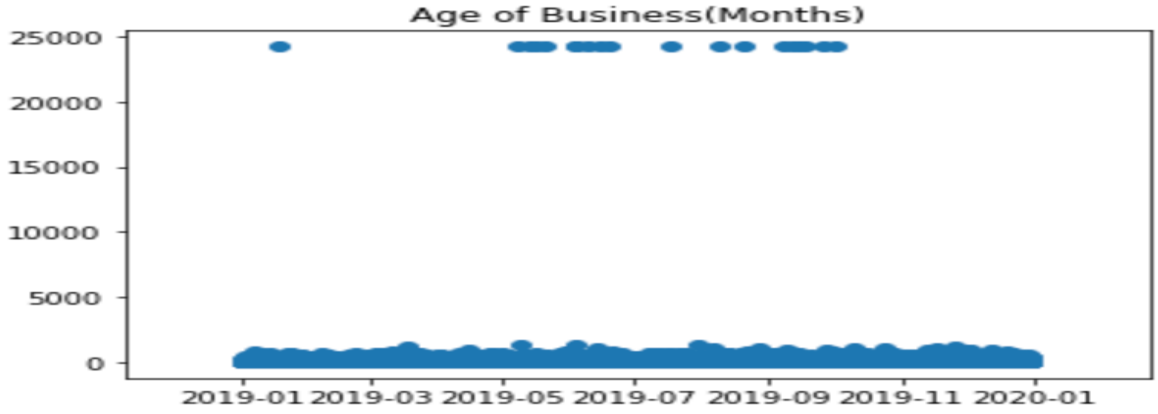
0

Outlier:

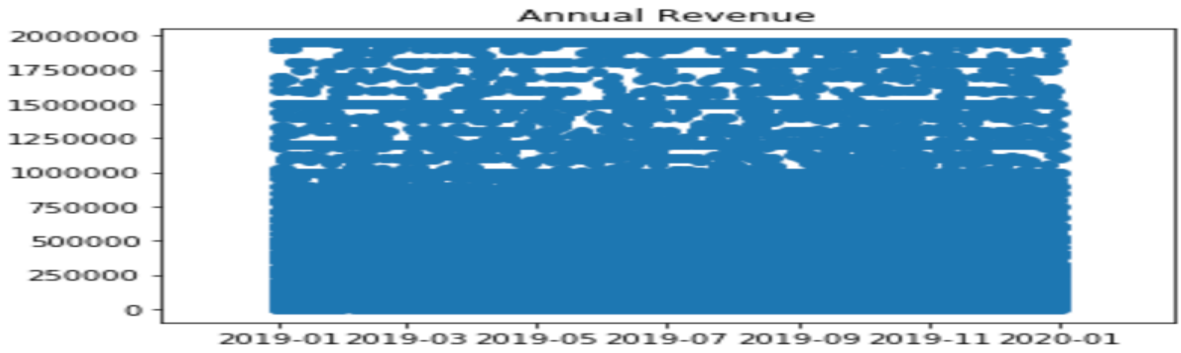
Utilizing scatter plots to discover outliers, I found out that only "Amount Requested" and "Age of Business(months)“ have outliers. “Personal CreditScore” and “Annual Revenue” don’t contain outliers but only null values, which were solved earlier. For “Amount Requested”, requiring 350,000,004 loans is nearly impossible. For "Age of Business(months)”, a company established 24,238 months, which is 2019 years, is also nearly impossible. Having a further examination, I found out that "Amount Requested" only has 1 outlier, and "Age of Business(months)“ has 19. Since both contain fewer outliers, dropping those rows is the best way for the analysis.

```
skip = 0 #Control Flow
#Utilize Scatter plot to
for i in df:
    if skip >2: #Skip BusinessID,App ID, and Date
        try:
            df[i]/1   # to see if it's numeric
            plt.title(i)
            plt.scatter(df["Application Date"], df[i],color = "green")
            plt.show()
            plt.close()
        except: #Skip non-numeric data
            continue
    skip = skip+1
```


Amount Requested
1e8
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
2019-01 2019-03 2019-05 2019-07 2019-09 2019-11 2020-01


Age of Business(Months)
25000
20000
15000
10000
5000
0
2019-01 2019-03 2019-05 2019-07 2019-09 2019-11 2020-01


Personal CreditScore
900
800
700
600
500
400
2019-01 2019-03 2019-05 2019-07 2019-09 2019-11 2020-01


Annual Revenue
2000000
1750000
1500000
1250000
1000000
750000
500000
250000
0
2019-01 2019-03 2019-05 2019-07 2019-09 2019-11 2020-01

```
df["Amount Requested"].loc[df["Amount Requested"]>200000000].count()
```

```
1
```

```
df["Age of Business(Months)"].loc[df["Age of Business(Months)"]>20000].count()
```

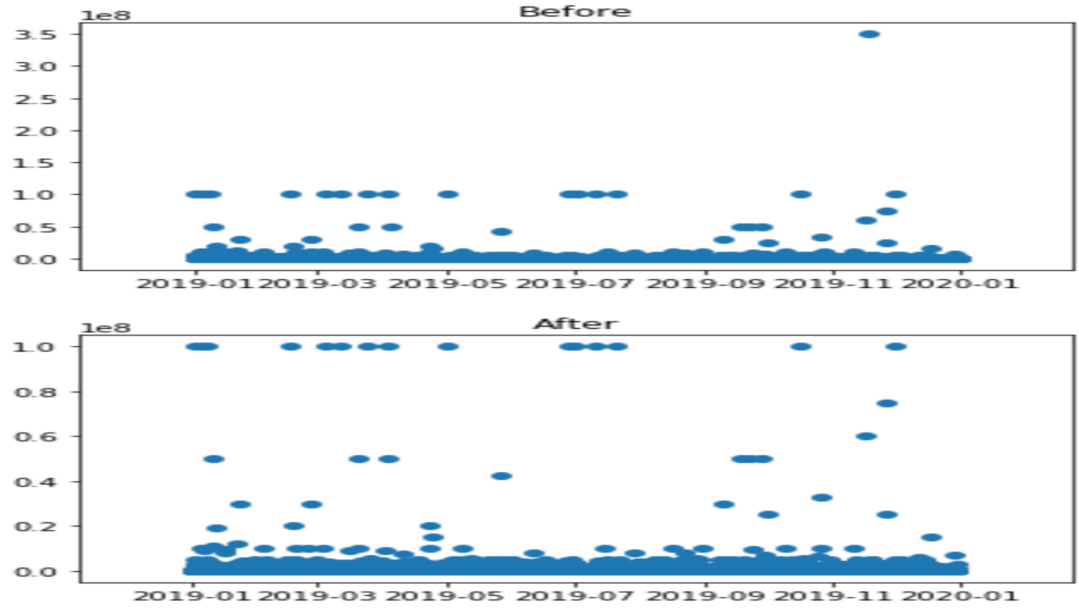
```
19
```

Treatment: (“Amount Requested”)

```
#1.Amount Requested:
plt.title("Before")
plt.scatter(df["Application Date"], df["Amount Requested"])#Scatter plot before the outlier is dropped
plt.show()
plt.close()
otlr = 0
for i in df["Amount Requested"]:# Find the value of the outlier
    if i > 150000000: # In the diagram, we can easily find out the outlier is larger than 1.5e8
        otlr = i
        df.drop(df[df["Amount Requested"]==otlr].index.values, axis= 0, inplace=True) # Drop the rows

plt.title("After")
plt.scatter(df["Application Date"], df["Amount Requested"])
plt.show()
plt.close()#Scatter plot after dropping the outlier
```

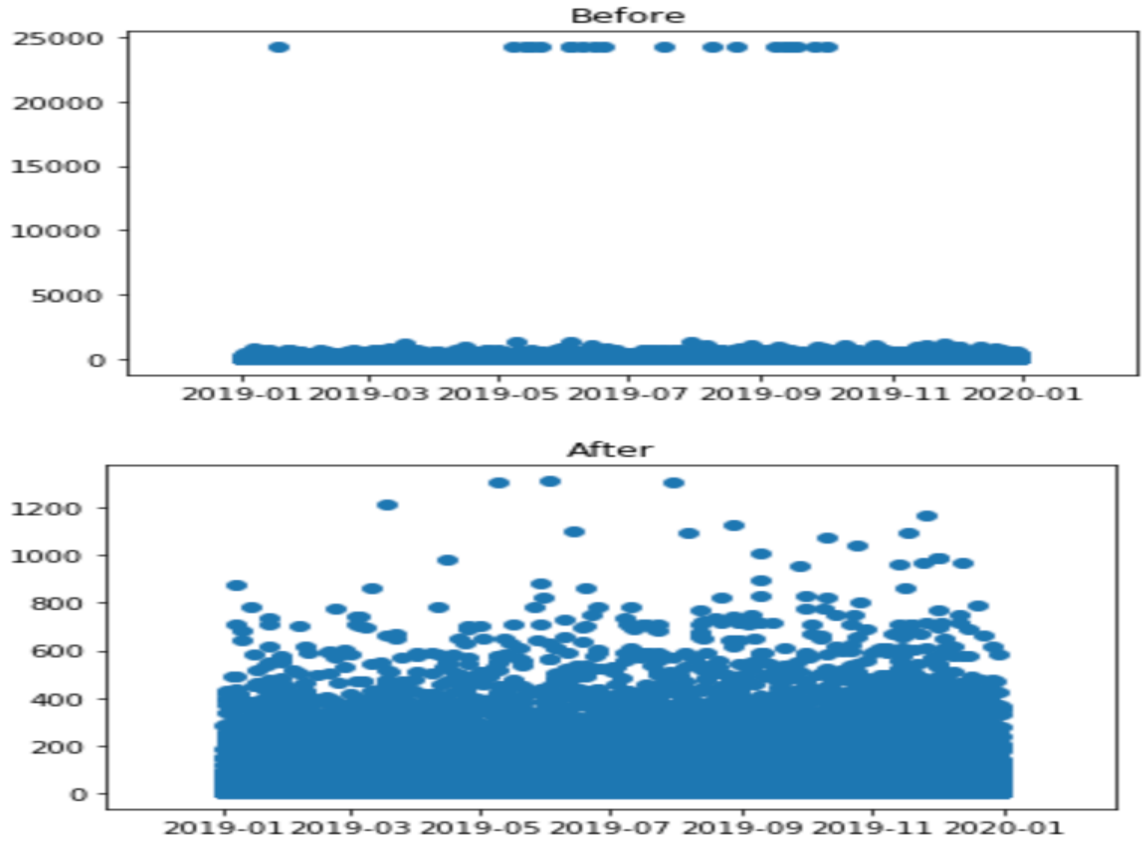
Result:



Treatment: (“Age of Business(Months)”)

```
#2. Age of Business(Months)
plt.title("Before")
plt.scatter(df["Application Date"], df["Age of Business(Months)"])#Scatter plot before the outlier is dropped
plt.show()
plt.close()
otlr = 0
for i in df["Age of Business(Months)"]:# Find the value of the outlier
    if i > 15000: # In the diagram, we can easily find out the outlier is larger than 15,000 months
        otlr = i
        df.drop(df[df["Age of Business(Months)"]==otlr].index.values, axis= 0, inplace=True) # Drop the rows
plt.title("After")
plt.scatter(df["Application Date"], df["Age of Business(Months)"])
plt.show()
plt.close()#Scatter plot after dropping the outlier
```

Result:



Data Analysis pt. 1: Device Usage

1. Which device contributed to the maximum number of applications in 12 months period?

I used “-groupby” to find out the number of applications from each device. As we can see in the data frame, “phone” contributed to the maximum number of applications, which is '19,235'.

Code / Results:

```
dev_grp = df.groupby(df["Device"], as_index=False)["Application ID"].count()
dev_grp.sort_values(["Application ID"], ascending= False)
```

	Device	Application ID
2	phone	19235
0	computer	16591
4	unspecified	4850
3	tablet	727
1	mobileapp	149

2. How did the application distribution change over time?

First, I set the values in date order. To count the number of applications in each device, I created a binary variable for each device. For instance, 0 means not on this device. 1 means on this

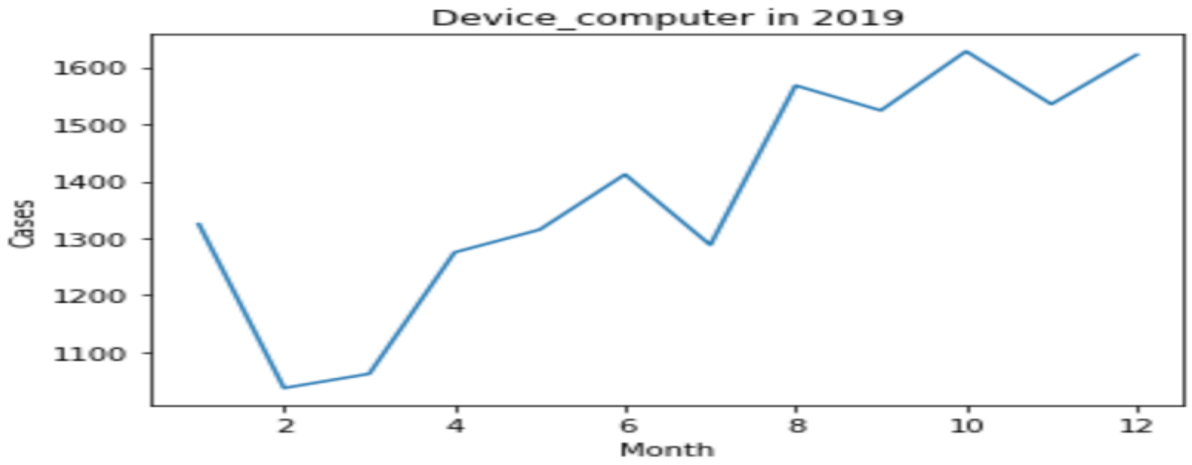
device. Second, I grouped up the “Application Date” and summed up all the dummy variables. Last, I created a line chart for each device to see the trend of the application distribution. In conclusion, almost all kinds of devices have continued to increase in applications. Only “tablet” has a dramatic increase and decrease in applications between August and October. In my point of view, it is probably caused by the launch of the iPad 7 on September 25 in 2019, which suddenly reminded the customers to try their iPad to see if it functions as usual. After that, they switched back to the device they are more comfortable with, making the usage drop dramatically.

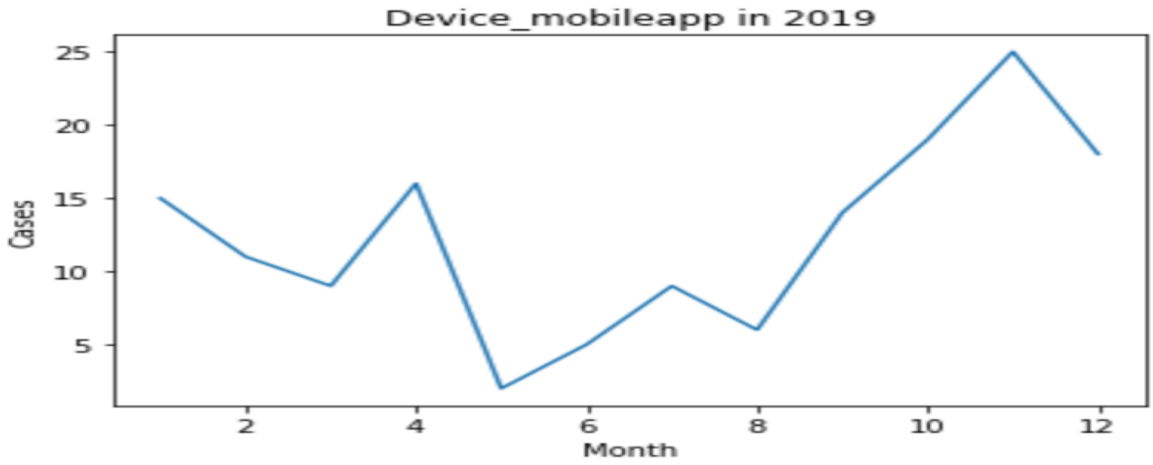
Code / Results:

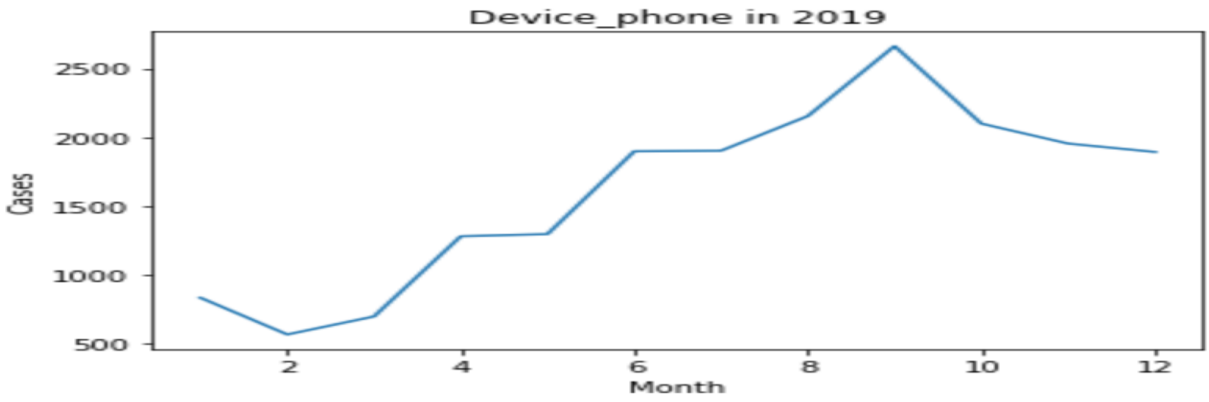
```
df= df.sort_values("Application Date", ascending = True)
dum = pd.get_dummies(df)
device = dum.iloc[:,45:50]
device["Application Date"] = df["Application Date"]
gp = device.groupby(["Application Date"], as_index=False).sum()
device_date = gp.groupby(gp["Application Date"].dt.month).sum().reset_index()
device_date = pd.DataFrame(device_date)
device_date
```

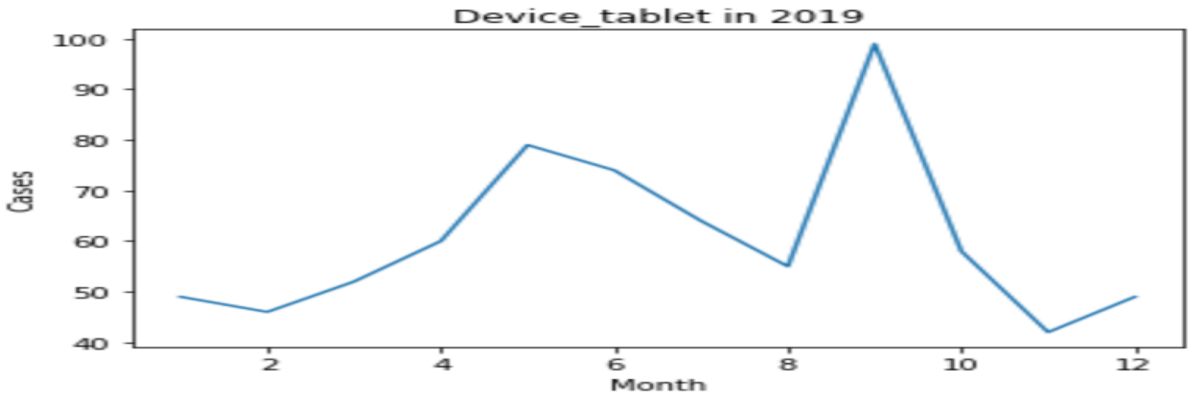
	Application Date	Device_computer	Device_mobileapp	Device_phone	Device_tablet	Device_unspecified
0	1	1325.0	15.0	833.0	49.0	352.0
1	2	1037.0	11.0	566.0	46.0	314.0
2	3	1062.0	9.0	697.0	52.0	354.0
3	4	1275.0	16.0	1280.0	60.0	303.0
4	5	1315.0	2.0	1296.0	79.0	381.0
5	6	1412.0	5.0	1898.0	74.0	368.0
6	7	1288.0	9.0	1902.0	64.0	431.0
7	8	1568.0	6.0	2154.0	55.0	462.0
8	9	1524.0	14.0	2663.0	99.0	451.0
9	10	1628.0	19.0	2099.0	58.0	490.0
10	11	1535.0	25.0	1954.0	42.0	472.0
11	12	1622.0	18.0	1893.0	49.0	472.0

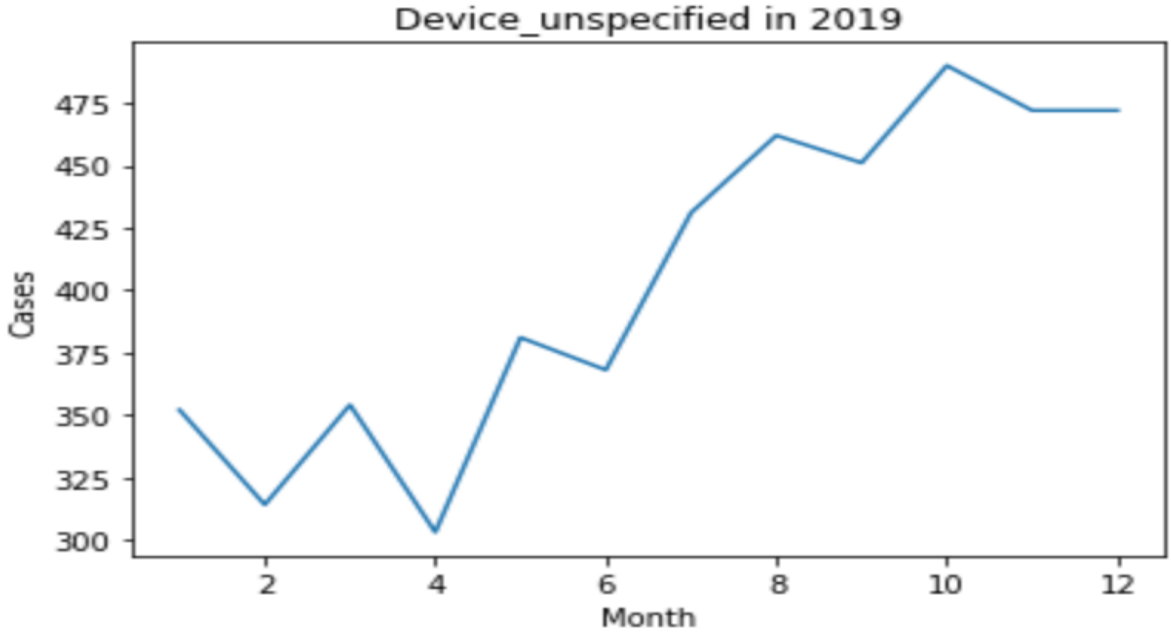
```
skip=0 #Control Flow
for i in device_date:
    if skip>0:
        plt.title(i+ " in 2019")
        plt.plot(device_date["Application Date"], device_date[i])
        plt.xlabel("Month")
        plt.ylabel("Cases")
        plt.show()
        plt.close
    skip +=1
```






Device_phone in 2019
Cases
Month
2500
2000
1500
1000
500
2
4
6
8
10
12


Device_tablet in 2019
Cases
Month
100
90
80
70
60
50
40
2
4
6
8
10
12


Device_unspecified in 2019
Cases
Month
475
450
425
400
375
350
325
300
2
4
6
8
10
12

3. Did "application channel" have any impact on the trend of device usage?

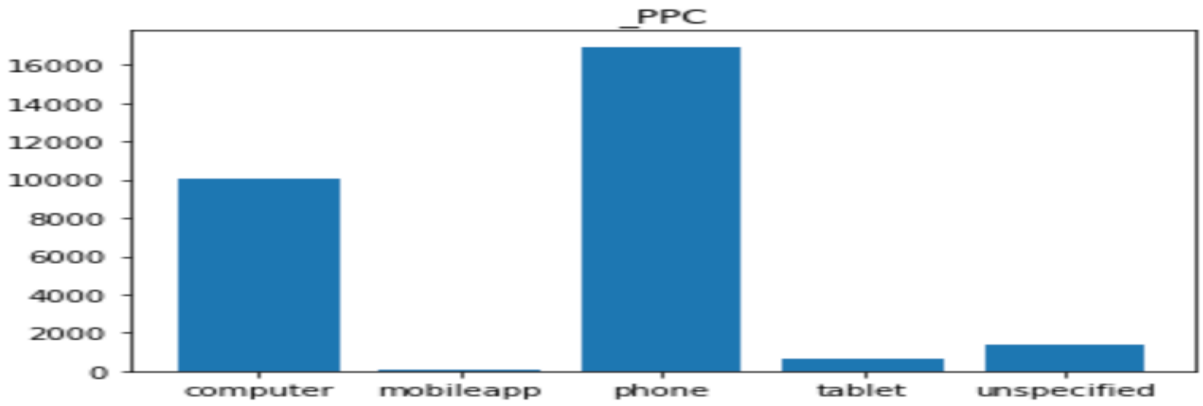
First, I created a data frame containing the device name and each application channel to see if the channel has an impact on the device usage. Second, I used a Bar chart to compare the device usage between each channel to see the difference. As we can see, “PPC(pay by click)” has the maximum number of applications on “phone”. “Referal” has the maximum number of applications on “computer”. “SEO(search engine optimization)” has a relatively even distribution on “computer”, ”phone”, and ”unspecified”. Last, I created line charts of each channel to compare those device usage line charts in which each channel has the most applications. In short, two similar track of trend line implies that “application channel” certainly has impacts on device usage.

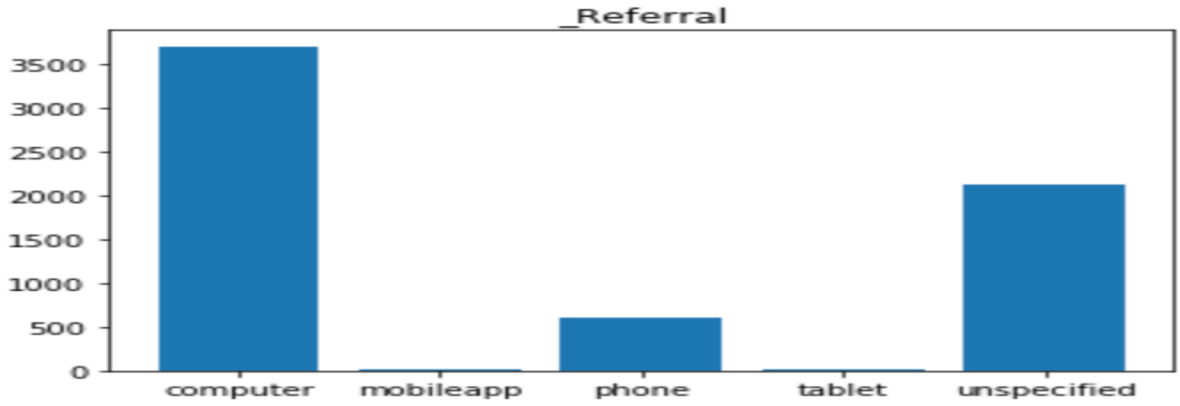
Code / Results:

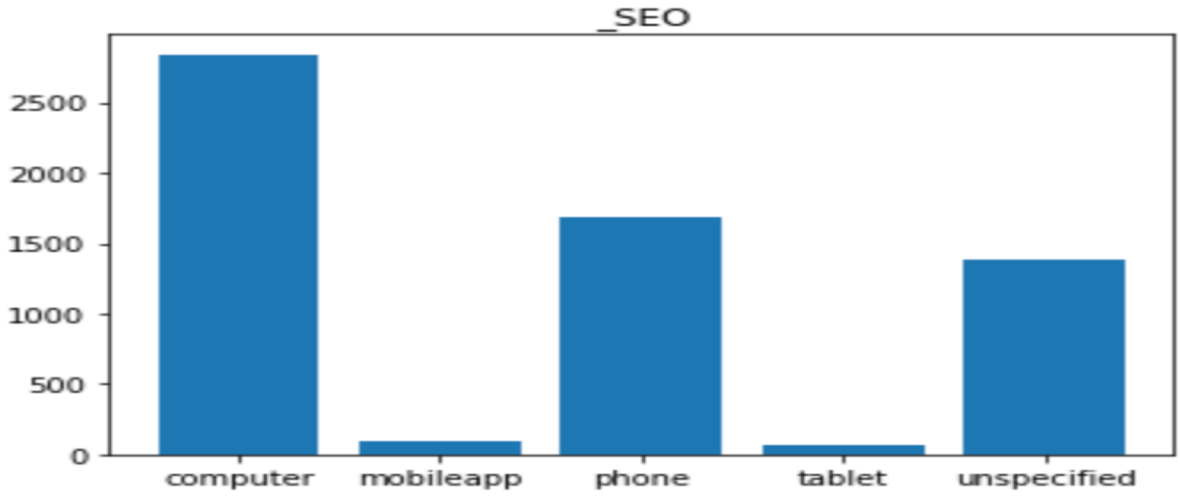
```
dev_channel = dum.iloc[:,9:12]
dev_channel["Device"] = df["Device"]
dev_channel = dev_channel.groupby(["Device"], as_index=False).sum()
dev_channel = pd.DataFrame(dev_channel)
dev_channel
```

	Device	Application Channel_PPC	Application Channel_Referral	Application Channel_SEO
0	computer	10049.0	3700.0	2842.0
1	mobileapp	54.0	8.0	87.0
2	phone	16953.0	596.0	1686.0
3	tablet	639.0	20.0	68.0
4	unspecified	1342.0	2121.0	1387.0

```
skip=0
for i in dev_channel:
        if skip>0:
            tt = i.replace("Application Channel_","")
            plt.title(tt)
            plt.bar(dev_channel["Device"], dev_channel[i])
            plt.show()
            plt.close
        skip+=1
```



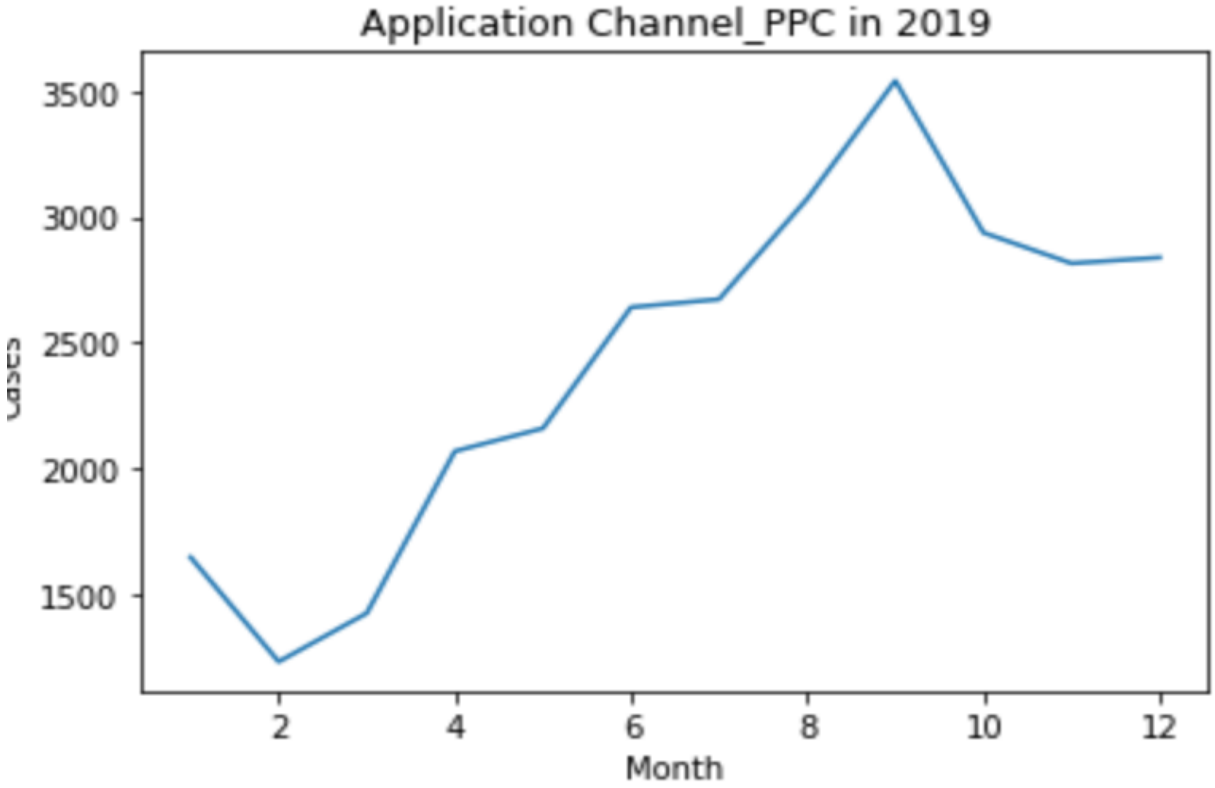


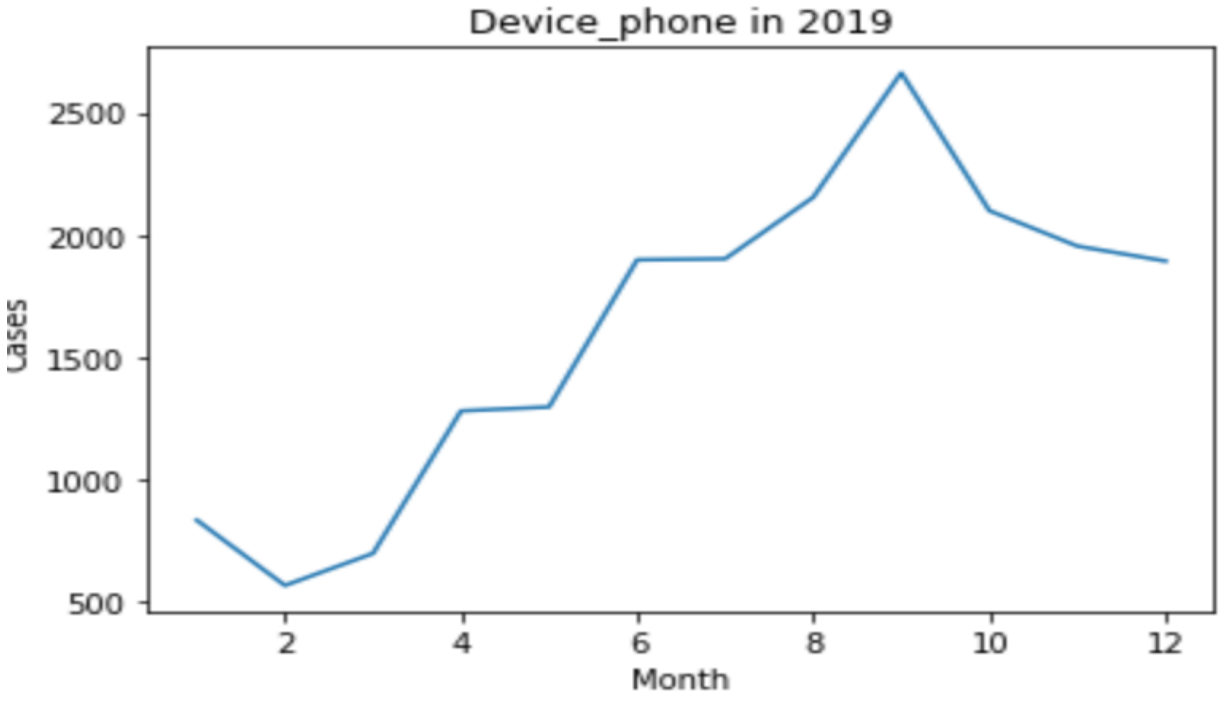


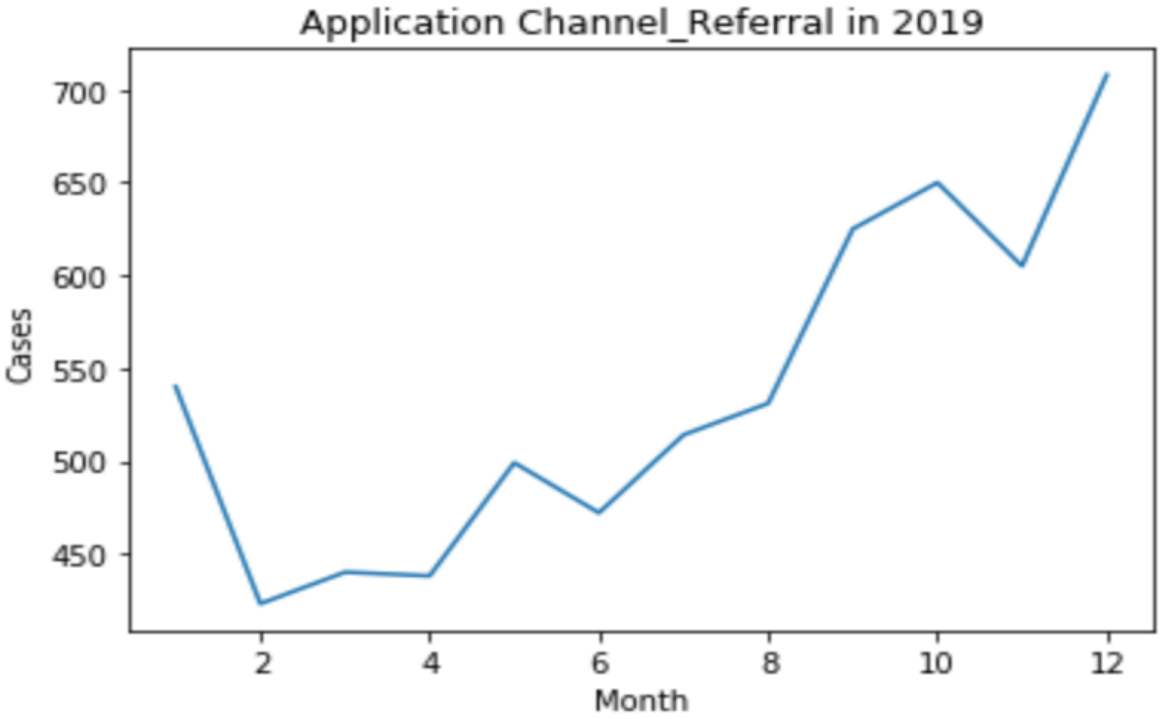
```
dev_channel = dum.iloc[:,9:12]
dev_channel["Application Date"] = df["Application Date"]
gp2 = dev_channel.groupby(["Application Date"], as_index=False).sum()
devc = gp2.groupby(gp2["Application Date"].dt.month).sum().reset_index()
devc = pd.DataFrame(devc)
devc
```

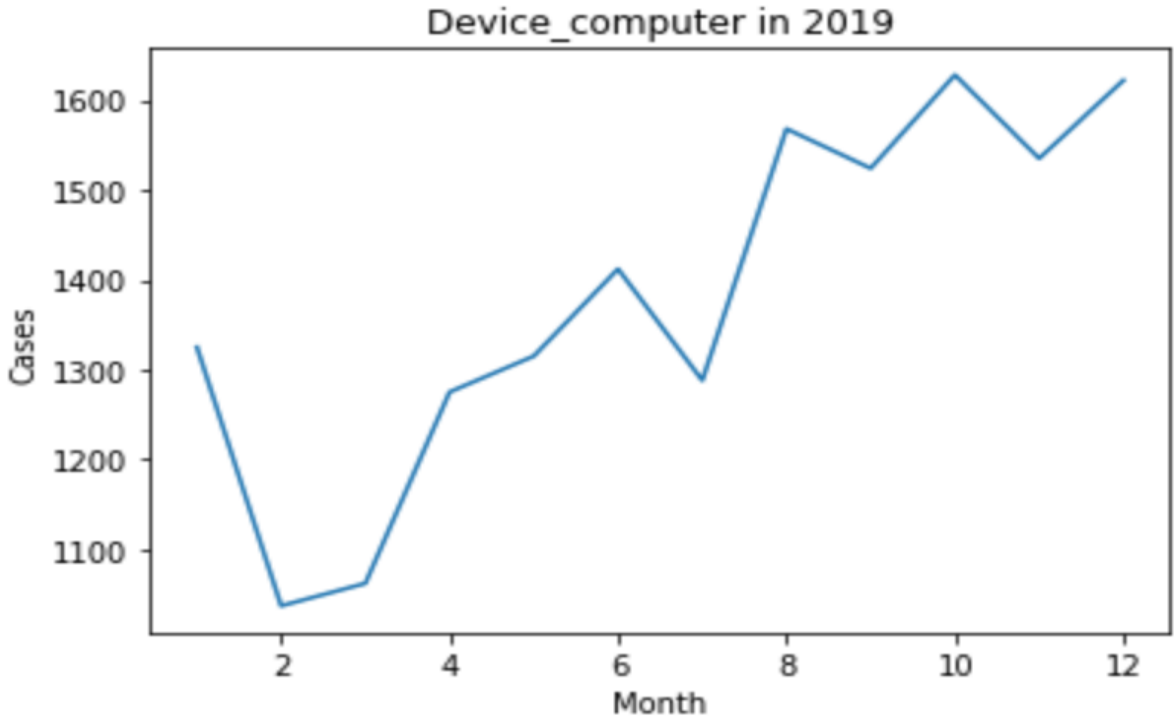
	Application Date	Application Channel_PPC	Application Channel_Referral	Application Channel_SEO
0	1	1645.0	540.0	389.0
1	2	1229.0	423.0	322.0
2	3	1422.0	440.0	312.0
3	4	2067.0	438.0	429.0
4	5	2158.0	499.0	416.0
5	6	2640.0	472.0	645.0
6	7	2672.0	514.0	508.0
7	8	3072.0	531.0	642.0
8	9	3542.0	625.0	584.0
9	10	2937.0	650.0	707.0
10	11	2815.0	605.0	608.0
11	12	2838.0	708.0	508.0

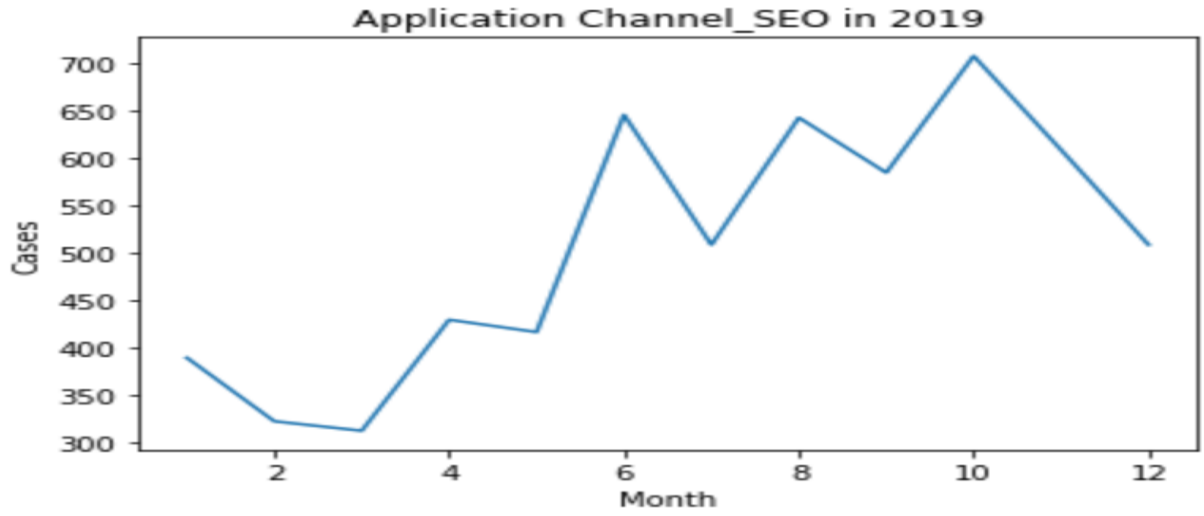
```
skip=0 #Control Flow
for i in devc:
    if skip>0:
        plt.title(i+ " in 2019")
        plt.plot(devc["Application Date"], devc[i])
        plt.xlabel("Month")
        plt.ylabel("Cases")
        plt.show()
        plt.close()
    skip +=1
```

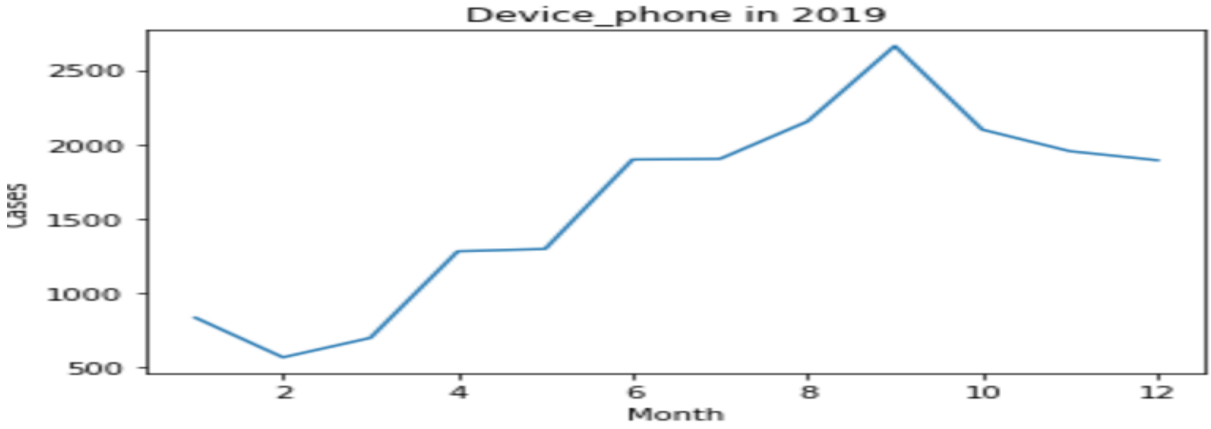

Application Channel_PPC in 2019
3500
3000
2500
2000
1500
2
4
6
8
10
12
Month

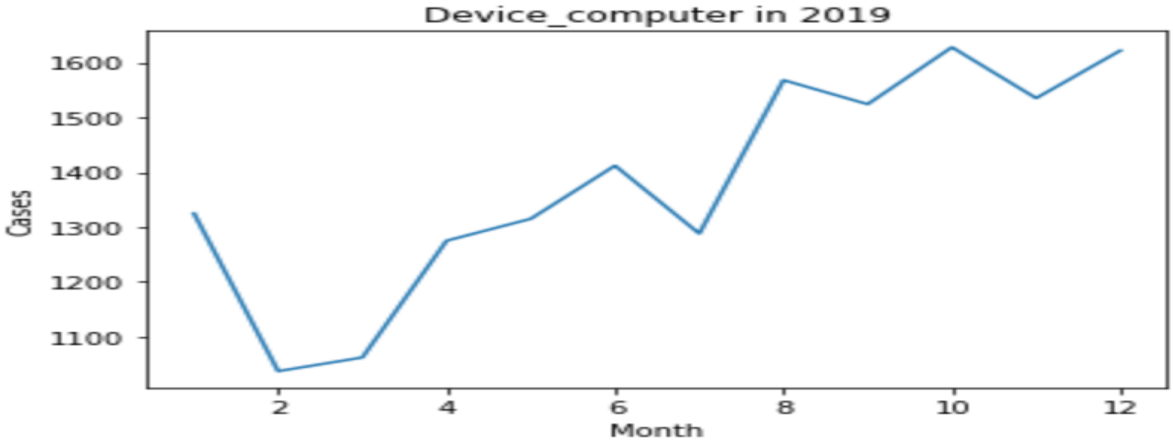

Device_phone in 2019
2500
2000
1500
1000
500
2
4
6
8
10
12
Month


Application Channel_Referral in 2019
700
650
600
550
500
450
Cases
2
4
6
8
10
12
Month


Device_computer in 2019
1600
1500
1400
1300
1200
1100
Cases
2
4
6
8
10
12
Month


Application Channel_SEO in 2019
Cases
Month
700
650
600
550
500
450
400
350
300
2
4
6
8
10
12


Device_phone in 2019
Cases
Month
2500
2000
1500
1000
500
2
4
6
8
10
12


Device_computer in 2019
Cases
Month
1600
1500
1400
1300
1200
1100
2
4
6
8
10
12

4. Which Industry contributed most in 2019?

In my point of view, the industry that has the maximum number of applications contributed the most in 2019. Thus, I used “-groupby” “Industry” to find out which industry has the maximum number of applications in 2019. In the diagram, “Other industry” has the maximum number of applications and “Retail” is the second-highest. However, “Other Industry” is a composition of many other industries that cannot be categorized. Since “Other Industries” might contain a hundred industries, it is not fair to compare it with a single industry. Thus, the “Retail” industry contributed the most in 2019.

Code / Results:

```
#Count of applications:
cnt=[]
cnt=pd.DataFrame(cnt)
ind = df.groupby(df["Industry"],as_index=False).count()
cnt["Industry"]=ind["Industry"]
cnt["num_Application"] = ind["Application ID"]
cnt.sort_values(["num_Application"], ascending= False)
```

	Industry	num_Application
11	Other Services (except Public Administration)	7823
16	Retail Trade	6640
0	Accommodation and Food Services	5175
4	Construction	4505
17	Transportation and Warehousing	3248
3	Arts, Entertainment, and Recreation	2232
7	Health Care and Social Assistance	2064
12	Professional, Scientific, and Technical Services	1825
14	Real Estate and Rental and Leasing	1527
20	Wholesale Trade	1256
9	Manufacturing	1026
8	Information Technology	1006
5	Educational Services	849
6	Finance and Insurance	774
2	Agriculture, Forestry, Fishing and Hunting	631
1	Administrative and Support and Waste Managemen...	439
19	Utilities	182
15	Research and Development	139
10	Mining, Quarrying, and Oil and Gas Extraction	115
13	Public Administration	75
18	Unknown	21

Data Analysis pt. 2: Potential Valuable Customers

1. What's the relationship between attributes?

To find the relationship between each attribute, I will need the correlation table for the dataset. Thus, I utilized python to transform all the categorical data from literal data into numerical data, making all the data applicable to the statistical model. Next, I used "-unique" to find out the unique value of each categorical data, gave each category a number, and exported the numerical dataset to Excel. Lastly, used "correlation" in Excel to generate the table. In the diagram, we can easily find out that "Amount Requested" has the weakest relationship with any other attributes. On the other hand, "Self-Directed" and "Annual Revenue" have the strongest relationship with others, which is represented in dark green or dark red.

2. How much amount of annual revenue can make the company a valuable customer?

In my point of view, the customer that has more annual revenue than most of the companies is valuable to Biz2Credit, since the bigger the company is the larger the amount of loan it needs. To find out how much amount is profitable, I need the distribution of the "Annual Revenue". I deleted the literal data, utilized descriptive statistics to find the "mean" and "standard deviation", and found the range of one standard deviation. In this case, it is a significant right-skewed

distribution. Thus, I used “Median” to calculate, instead of “Mean”. Approximately 68% of companies’ annual revenue falls between $0 to $815,188. Any company with annual revenue of more than $815,188 will be defined as “Valuable”.

3. What kind of customers can be potential valuable customers to Biz2Credit?

(Method is in this section. The answer is in the next section)

To answer this question, I used the supervised learning model- Decision tree to find the criteria. To choose the right parameters, I used the metrics generated by “regression” to filter. First, I deleted all the attributes whose P-value is more than 0.05. With the deletion of those attributes, the R square didn’t change, which means the attributes do not have a significant impact on the model. Second, I selected five attributes with the biggest absolute t stats, which will avoid the deletion of the binary attribute. Last, deleted the attribute that has a too low coefficient to the regression model. By doing so, the parameters for the decision model were selected- “Personal CreditScore”, “Application Channel”, “Business Legal Structure”, and “Self-Directed”.

Question 1:

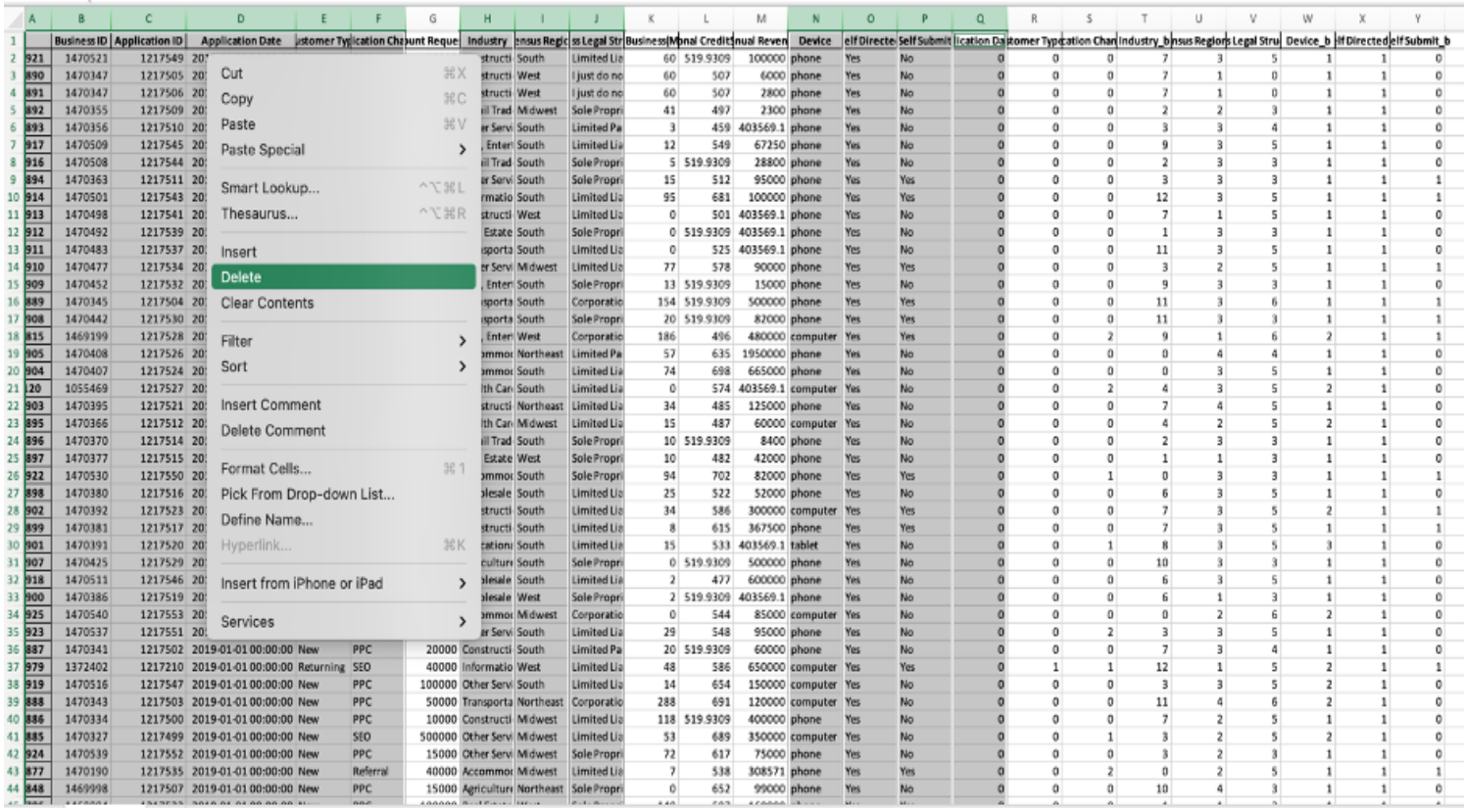
```
#Find out the unique value of each categorical data
skip = 0
for i in df:
    if skip>2:
        try:
            df[i]/1 #Skip numerical data
        except:
            df[i+"_b"]= 0
            print(f'{i}:\n\t ', df[i].unique())
    skip+=1
```

```
Customer Type:
          ['New' 'Returning']
Application Channel:
          ['PPC' 'Referral' 'SEO']
Industry:
          ['Construction' 'Retail Trade'
 'Other Services (except Public Administration)'
 'Arts, Entertainment, and Recreation' 'Information Technology'
 'Real Estate and Rental and Leasing' 'Transportation and Warehousing'
 'Accommodation and Food Services' 'Health Care and Social Assistance'
 'Wholesale Trade' 'Educational Services'
 'Agriculture, Forestry, Fishing and Hunting' 'Manufacturing'
 'Mining, Quarrying, and Oil and Gas Extraction'
 'Administrative and Support and Waste Management and Remediation Services'
 'Utilities ' 'Professional, Scientific, and Technical Services'
 'Finance and Insurance' 'Research and Development'
 'Public Administration' 'Unknown']
Census Region:
          ['South' 'West' 'Midwest' 'Northeast' 'Other']
Business Legal Structure:
          ['Limited Liability Company' 'I just do not know' 'Sole Proprietorship'
 'Limited Partnership' 'Corporation' 'Non Profit Corp' 'Partnership']
Device:
          ['phone' 'computer' 'tablet' 'unspecified' 'mobileapp']
Self Directed:
          ['Yes' 'No']
Self Submit:
          ['No' 'Yes']
```

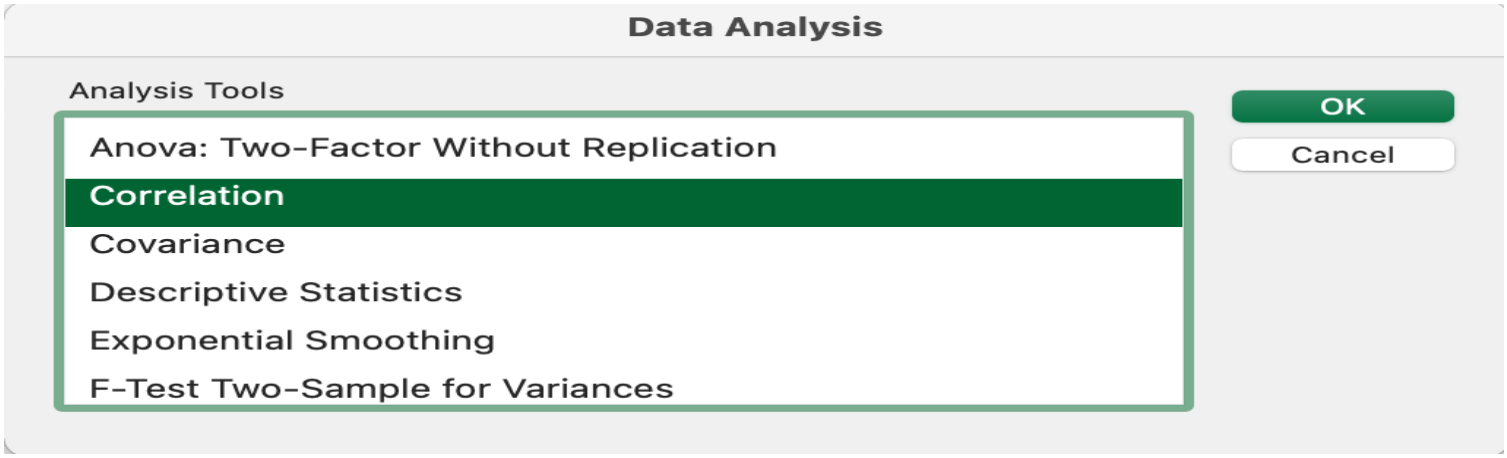
```
df.loc[df["Customer Type"] == "Returning","Customer Type_b"] = 1
df.loc[df["Application Channel"] == "SEO","Application Channel_b"] = 1
df.loc[df["Application Channel"] == "Referral","Application Channel_b"] = 2
df.loc[df["Census Region"] == "West","Census Region_b"] = 1
df.loc[df["Census Region"] == "Midwest","Census Region_b"] = 2
df.loc[df["Census Region"] == "South","Census Region_b"] = 3
df.loc[df["Census Region"] == "Northeast","Census Region_b"] = 4
df.loc[df["Business Legal Structure"] == "Non Profit Corp","Business Legal Structure_b"] = 1
df.loc[df["Business Legal Structure"] == "Partnership","Business Legal Structure_b"] = 2
df.loc[df["Business Legal Structure"] == "Sole Proprietorship","Business Legal Structure_b"] = 3
df.loc[df["Business Legal Structure"] == "Limited Partnership","Business Legal Structure_b"] = 4
df.loc[df["Business Legal Structure"] == "Limited Liability Company","Business Legal Structure_b"] = 5
df.loc[df["Business Legal Structure"] == "Corporation","Business Legal Structure_b"] = 6
df.loc[df["Device"] == "phone","Device_b"] = 1
df.loc[df["Device"] == "computer","Device_b"] = 2
df.loc[df["Device"] == "tablet","Device_b"] = 3
df.loc[df["Device"] == "mobileapp","Device_b"] = 4
df.loc[df["Self Directed"] == "Yes","Self Directed_b"] = 1
df.loc[df["Self Submit"] == "Yes","Self Submit_b"] = 1


df.loc[df["Industry"] == "Real Estate and Rental and Leasing","Industry_b"] = 1
df.loc[df["Industry"] == "Retail Trade","Industry_b"] = 2
df.loc[df["Industry"] == "Other Services (except Public Administration)","Industry_b"] = 3
df.loc[df["Industry"] == "Health Care and Social Assistance","Industry_b"] = 4
df.loc[df["Industry"] == "'Accommodation and Food Services","Industry_b"] = 5
df.loc[df["Industry"] == "Wholesale Trade","Industry_b"] = 6
df.loc[df["Industry"] == "Construction","Industry_b"] = 7
df.loc[df["Industry"] == "Educational Services","Industry_b"] = 8
df.loc[df["Industry"] == "Arts, Entertainment, and Recreation","Industry_b"] = 9
df.loc[df["Industry"] == "Agriculture, Forestry, Fishing and Hunting","Industry_b"] = 10
df.loc[df["Industry"] == "Transportation and Warehousing","Industry_b"] = 11
df.loc[df["Industry"] == "Information Technology","Industry_b"] = 12
df.loc[df["Industry"] == "Finance and Insurance","Industry_b"] = 13
df.loc[df["Industry"] == "Manufacturing","Industry_b"] = 14
```

```
df.to_excel("Dataset_binary.xlsx")
```



Amount Requested	Age of Business(Months)	Personal CreditScore	Customer Type_b	Application Channel_b	Industry_b	Census Region_b	Business Legal Structure_b	Device_b	Self Directed_b	Self Submit_b	Annual Revenue
10000	60	519.9308827	0	0	7	3	5	1	1	0	100000
10000	60	507	0	0	7	1	0	1	1	0	6000
8000	60	507	0	0	7	1	0	1	1	0	2800
60000	41	497	0	0	2	2	3	1	1	0	2300
10000	3	459	0	0	3	3	4	1	1	0	403569.1412
5500	12	549	0	0	9	3	5	1	1	0	67250
7500	5	519.9308827	0	0	2	3	3	1	1	0	28800
25000	15	512	0	0	3	3	3	1	1	1	95000
30000	95	681	0	0	12	3	5	1	1	1	100000
100000	0	501	0	0	7	1	5	1	1	0	403569.1412
80000	0	519.9308827	0	0	1	3	3	1	1	0	403569.1412
85000	0	525	0	0	11	3	5	1	1	0	403569.1412
15000	77	578	0	0	3	2	5	1	1	1	90000
500000	13	519.9308827	0	0	9	3	3	1	1	0	15000
10000	154	519.9308827	0	0	11	3	6	1	1	1	500000
25000	20	519.9308827	0	0	11	3	3	1	1	1	82000
50000	186	496	0	2	9	1	6	2	1	1	480000
100000	57	635	0	0	0	4	4	1	1	0	1950000
50000	74	698	0	0	0	3	5	1	1	0	665000
60000	0	574	0	2	4	3	5	2	1	0	403569.1412
10000	34	485	0	0	7	4	5	1	1	0	125000
10000	15	487	0	0	4	2	5	2	1	0	60000
20000	10	519.9308827	0	0	2	3	3	1	1	0	8400
10000	10	482	0	0	1	1	3	1	1	0	42000
25000	94	702	0	1	0	3	3	1	1	1	82000
100000	25	522	0	0	6	3	5	1	1	0	52000
125000	34	586	0	0	7	3	5	2	1	1	300000
25000	8	615	0	0	7	3	5	1	1	1	367500
25000	15	533	0	1	8	3	5	3	1	0	403569.1412
590000	0	519.9308827	0	0	10	3	3	1	1	0	500000
100000	2	477	0	0	6	3	5	1	1	0	600000
10000	2	519.9308827	0	0	6	1	3	1	1	0	403569.1412
150000	0	544	0	0	0	2	6	2	1	0	85000
110000	29	548	0	2	3	3	5	1	1	0	95000
20000	20	519.9308827	0	0	7	3	4	1	1	0	60000
40000	48	586	1	1	12	1	5	2	1	1	650000
100000	14	654	0	0	3	3	5	2	1	0	150000
50000	288	691	0	0	11	4	6	2	1	0	120000
10000	118	519.9308827	0	0	7	2	5	1	1	0	400000
500000	53	689	0	1	3	2	5	2	1	0	350000
15000	72	617	0	0	3	2	3	1	1	0	75000
40000	7	538	0	2	0	2	5	1	1	1	308571
15000	0	652	0	0	10	4	3	1	1	0	99000



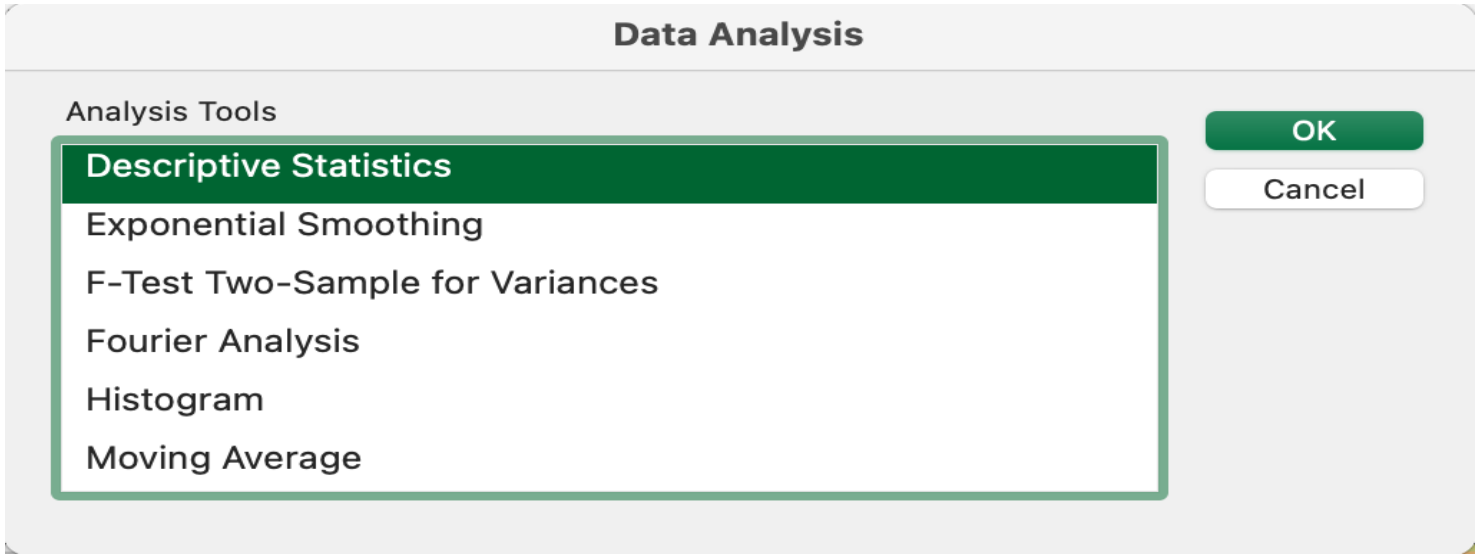
Weakest

	Amount Requested	Age of Business(Months)	Personal CreditScore	Customer Type_b	Application Channel_b	Industry_b	Census Region_b	Business Legal Structure_b	Device_b	Self Directed_b	Self Submit_b	Annual Revenue
Amount Requested	1											
Age of Business(Months)	0.028583526	1										
Personal CreditScore	0.025107064	0.208531453	1									
Customer Type_b	-0.006612222	0.185976447	0.180434868	1								
Application Channel_b	0.008453873	0.235407322	0.262127942	0.401354721	1							
Industry_b	0.010952414	0.020171914	-0.014215115	-0.138683927	-0.084509787	1						
Census Region_b	0.001997256	0.021111242	0.009303319	0.053672263	0.064105333	-0.030201835	1					
Business Legal Structure_b	0.007035584	0.13728605	0.292347716	0.246656108	0.25602935	0.011491738	0.02657836	1				
Device_b	0.017283957	0.019467168	0.105684034	-0.493184898	-0.048992612	0.080841111	-0.035162101	0.040356128	1			
Self Directed_b	-0.001993327	-0.260836071	-0.310038565	-0.689752771	-0.584398044	0.1150753	-0.055854	-0.332830221	0.188614413	1		
Self Submit_b	-0.007166536	0.023485271	0.103096631	-0.120871813	-0.038198551	0.0563634	-0.029044479	0.074221538	0.153896619	0.190381328	1	
Annual Revenue	0.076952913	0.232033462	0.3821663	0.297973358	0.39415659	-0.037868899	0.023870367	0.292089862	0.005908969	-0.459697514	-0.019804208	1

: Positively strong

: Negatively strong

Question 2:



Annual Revenue	
Mean	459279.7672
Standard Error	2690.085498
Median	266833.5
Mode	403569.1412
Standard Deviation	548355.3051
Sample Variance	3.00694E+11
Kurtosis	1.94960058
Skewness	1.727486475
Range	1949999
Minimum	1
Maximum	1950000
Sum	19083992888
Count	41552

	From:	To:
68% 1 standard deviation:	0	815188.8051

Question 3:

SUMMARY OUTPUT

Regression Statistics	
Multiple R	0.556473571
R Square	0.309662835
Adjusted R Square	0.309480031
Standard Error	455669.7209
Observations	41552

	Coefficients	Standard Error	t Stat	P-value
Intercept	-338679.8805	20968.52068	-16.15182518	1.66297E-58
Amount Requested	0.017116451	0.001048516	16.32444911	1.01541E-59
Age of Business(Months)	399.2540007	25.07674344	15.92128586	6.64925E-57
Customer Type_b	-2374.319004	9931.504556	-0.239069417	0.811052935
Personal CreditScore	1362.346262	28.63519612	47.57593615	0
Application Channel_b	105115.6885	3747.638526	28.04851314	1.6643E-171
Industry_b	215.4566523	466.613803	0.461745132	0.64426652
Census Region_b	-2728.260957	2229.86376	-1.223510156	0.221144011
Business Legal Structure_	34182.54628	1661.735991	20.5703833	1.48208E-93
Device_b	25620.53939	3824.655398	6.698783739	2.12836E-11
Self Directed_b	-336184.3328	8503.60608	-39.53432574	0
Self Submit_b	1901.694041	7572.82559	0.251120803	0.801721955

	Coefficients	Standard Error	t Stat	P-value	Lower 95%	Upper 95%	Lower 95.0%	Upper 95.0%
Intercept	-347667.015	18436.55052	-18.85748717	5.45879E-79	-383803.043	-311530.987	-383803.043	-311530.987
Amount Requested	0.01711542	0.001048329	16.32638539	9.83858E-60	0.015060674	0.019170167	0.015060674	0.019170167
Age of Business(Months)	399.6665942	25.00086053	15.98611351	2.36911E-57	350.6643804	448.6688081	350.6643804	448.6688081
Personal CreditScore	1363.427822	28.41556354	47.98172732	0	1307.732718	1419.122925	1307.732718	1419.122925
Application Channel_b	104902.8415	3734.337002	28.09142329	5.094E-172	97583.46218	112222.2207	97583.46218	112222.2207
Business Legal Structure_b	34207.61503	1645.303753	20.79106364	1.60108E-95	30982.78498	37432.44509	30982.78498	37432.44509
Device_b	26349.57369	3254.081295	8.097392567	5.76648E-16	19971.50573	32727.64165	19971.50573	32727.64165
Self Directed_b	-334255.211	6791.961413	-49.21335547	0	-347567.599	-320942.824	-347567.599	-320942.824

	Coefficients	Standard Error	t Stat	P-value	Lower 95%	Upper 95%	Lower 95.0%	Upper 95.0%
Intercept	-350606.829	18508.75088	-18.94276017	1.09945E-79	-386884.371	-314329.287	-386884.371	-314329.287
Amount Requested	0.017651812	0.001052078	16.77805089	5.69764E-63	0.015589718	0.019713907	0.015589718	0.019713907
Personal CreditScore	1452.853888	28.01521832	51.85945265	0	1397.943469	1507.764306	1397.943469	1507.764306
Application Channel_b	112008.383	3728.651462	30.03991768	3.7433E-196	104700.1475	119316.6185	104700.1475	119316.6185
Business Legal Structure_b	35757.23888	1647.047563	21.7099006	6.269E-104	32528.99092	38985.48683	32528.99092	38985.48683
Self Directed_b	-334223.3747	6567.450563	-50.89088551	0	-347095.7163	-321351.0331	-347095.7163	-321351.0331

	Coefficients	Standard Error	t Stat	P-value	Lower 95%	Upper 95%	Lower 95.0%	Upper 95.0%
Intercept	-355341.5253	18568.96769	-19.13631017	2.81994E-81	-391737.0935	-318945.9571	-391737.0935	-318945.9571
Personal CreditScore	1464.242533	28.10137902	52.10571809	0	1409.163238	1519.321828	1409.163238	1519.321828
Application Channel_b	112399.1942	3741.144211	30.04406884	3.3123E-196	105066.4727	119731.9157	105066.4727	119731.9157
Business Legal Structure_b	35784.22923	1652.597411	21.65332524	2.1138E-103	32545.10346	39023.355	32545.10346	39023.355
Self Directed_b	-333277.3604	6589.340356	-50.57825858	0	-346192.6064	-320362.1144	-346192.6064	-320362.1144

Data Analysis pt. 2 (contd.):

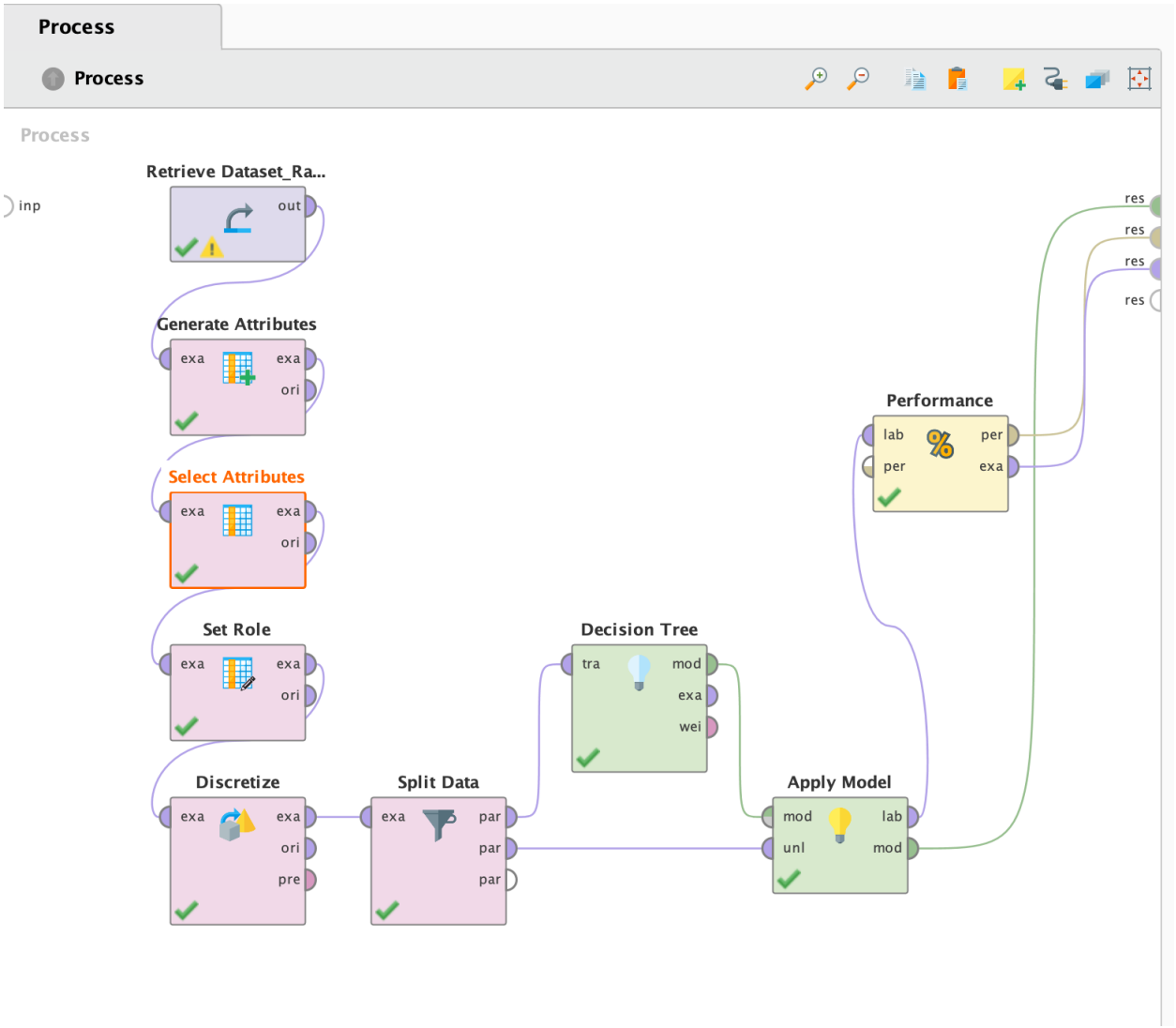
Supervised learning(Decision tree)

After all the metrics have met the demand of the supervised learning model, I can now apply the data to RapidMiner, a software that can easily execute the data mining process, to answer the third question. Since overfitting occurs when a predictive model is made overly complex to fit the quirks of a given sample data, I only chose four attributes for the decision tree.

On RapidMiner, I imported the organized data, generated the attribute that represents “valuable”, and selected the five attributes, including the attribute I just generated. Next, I set up the label to represent the dependent variable for the prediction, utilized “discretize” to avoid overfitting and help with generalization, separated the data in the ratio

of 7:3 for training and testing, and applied it to the decision tree model to see the performance.

Process / Results:

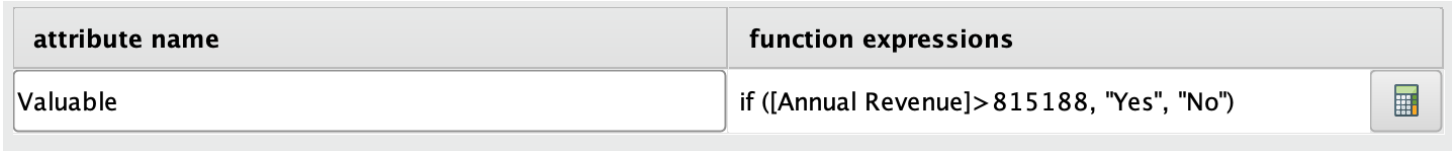


(Entire process)

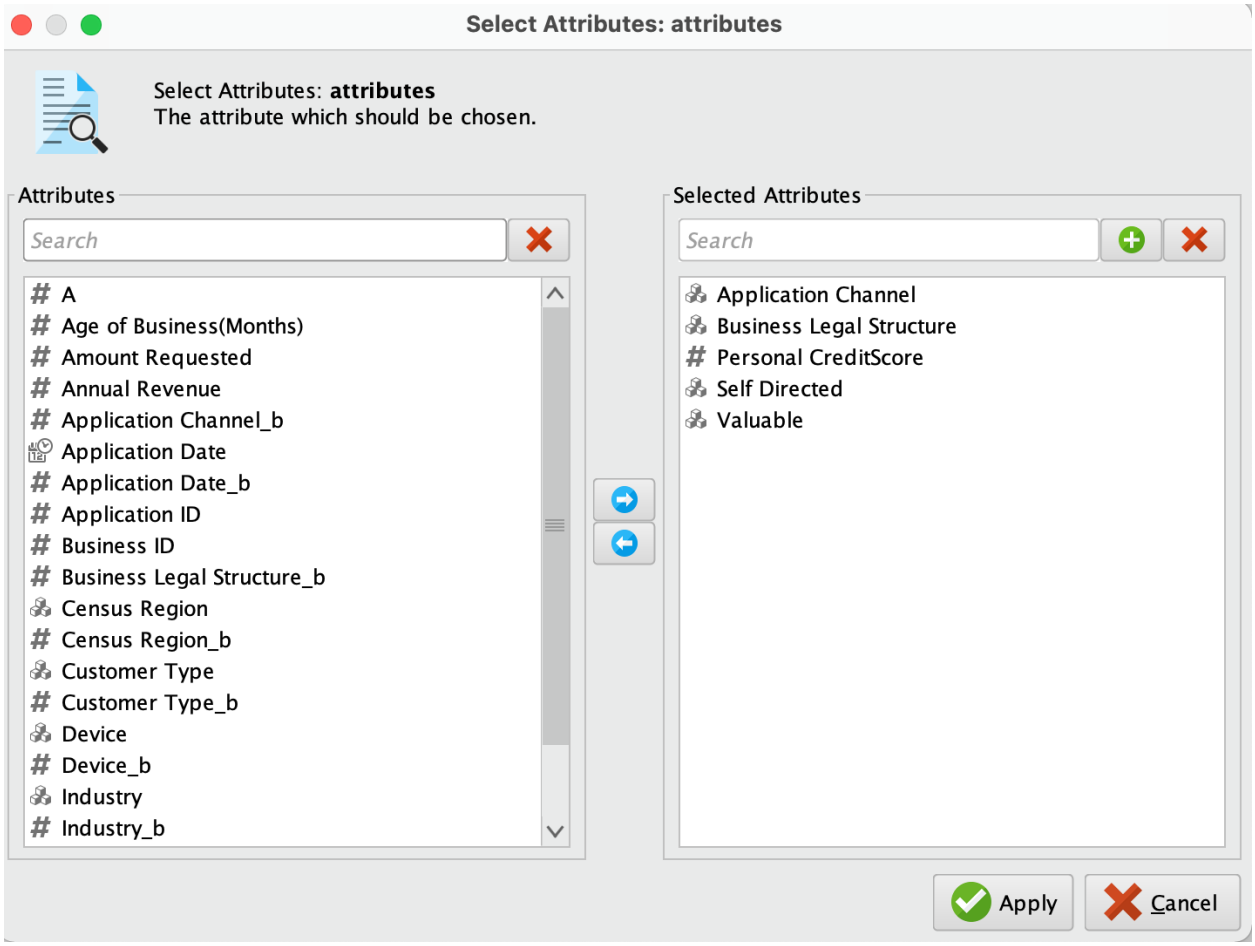
Customer ...	Application...	Amount Re...	Industry	Census Re...	Business L...	Age of Bus...	Personal Cr...	Annual Rev...	Device	Self Directed	Self Submit
New	Referral	50000	Retail Trade	West	Corporation	52	582	192600	computer	Yes	No
New	PPC	85000	Other Servic...	South	Sole Proprie...	3	572	500000	computer	Yes	No
New	PPC	75000	Retail Trade	West	Limited Liab...	11	800	327273	computer	Yes	No
New	PPC	50000	Other Servic...	South	I just do not...	0	571	403569.141	computer	Yes	No
New	SEO	10000	Retail Trade	South	Sole Proprie...	194	649	520000	computer	No	No
New	PPC	6000	Other Servic...	South	Limited Liab...	3	613	48000	phone	Yes	No
New	Referral	10000	Professional...	Midwest	Limited Liab...	29	673	180000	computer	No	No
New	Referral	50000	Transportati...	South	Corporation	142	600	600000	computer	No	No
New	PPC	10000	Construction	Midwest	Sole Proprie...	44	542	125000	phone	Yes	No
New	Referral	10000	Retail Trade	South	Limited Liab...	23	561	120000	computer	No	No
New	PPC	70000	Wholesale T...	South	Limited Liab...	0	583	403569.141	phone	Yes	No
New	PPC	80000	Agriculture, ...	South	Limited Liab...	0	563	403569.141	phone	Yes	No
New	PPC	8000	Other Servic...	South	Sole Proprie...	125	635	125000	computer	Yes	Yes
New	PPC	10000	Retail Trade	West	Sole Proprie...	26	519.931	100000	phone	Yes	No
New	PPC	6000	Arts, Enterta...	Midwest	Limited Part...	0	474	403569.141	phone	Yes	No
New	PPC	10000	Other Servic...	West	Sole Proprie...	43	519.931	85000	phone	Yes	No
New	PPC	8000	Wholesale T...	Midwest	Corporation	350	487	40000	phone	Yes	No
New	PPC	15000	Other Servic...	South	Sole Proprie...	4	570	300000	phone	Yes	Yes
New	PPC	15000	Other Servic...	South	I just do not...	0	470	403569.141	phone	Yes	No
New	PPC	7000	Construction	West	Sole Proprie...	95	510	65000	phone	Yes	No
New	PPC	35000	Transportati...	Midwest	Partnership	8	484	450000	phone	Yes	No

ExampleSet (41,552 examples, 0 special attributes, 25 regular attributes)

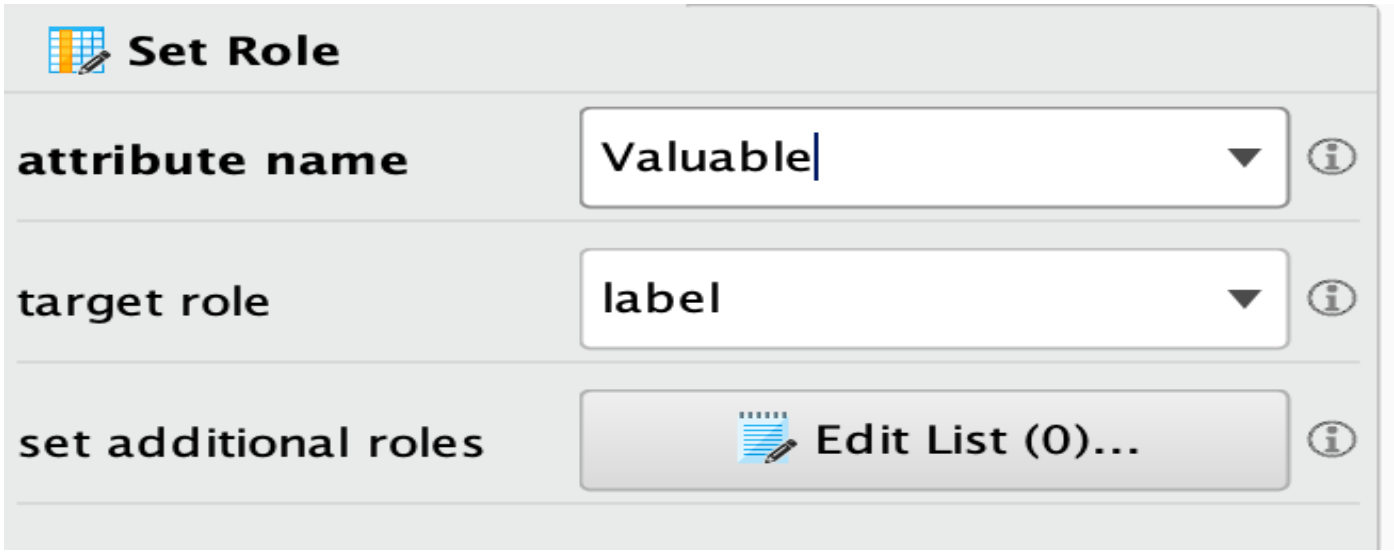
(Dataset)



(Generate Attribute)

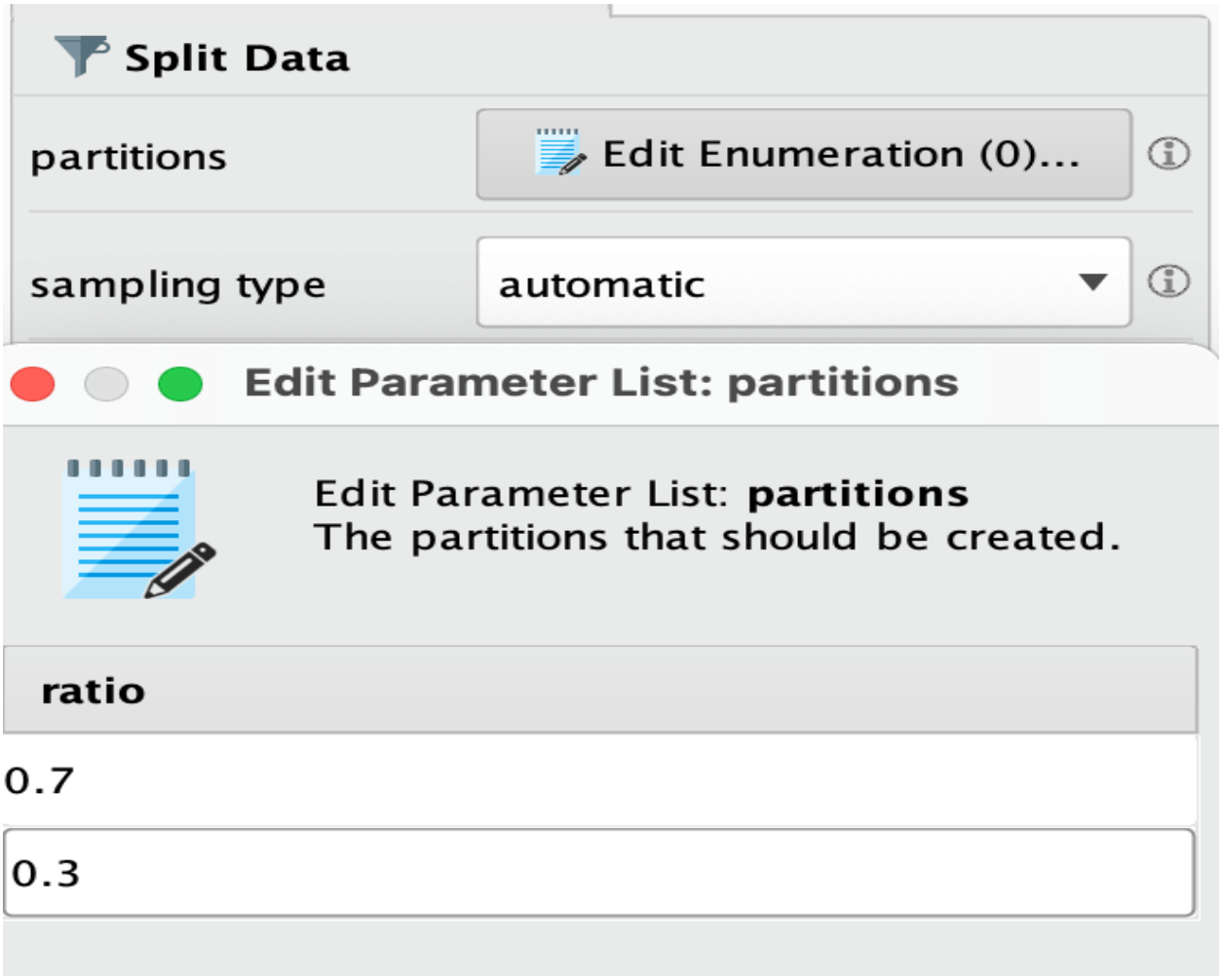


(Select Attribute)



(Set Role)

(Discretize)

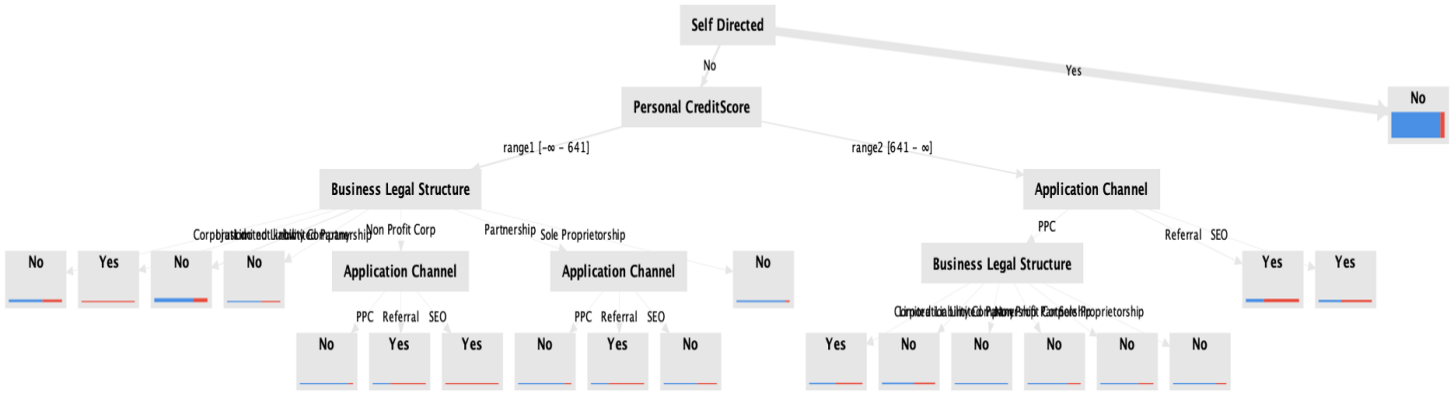


(Split Data)

accuracy: 85.42%

	true No	true Yes	class precision
pred. No	9950	1375	87.86%
pred. Yes	442	699	61.26%
class recall	95.75%	33.70%	

(Performance)



(The result of Decision Tree)

Data Analysis pt. 2 (contd.): Prediction

1. What kind of customers can be potential valuable customers to Biz2Credit?

In the Confusion Matrix, although the 33.7% accuracy of the true positive rate (recall) needs to be fixed, the overall 85.42% accuracy is

decent for the prediction. Based on this decision tree, here's my conclusion.

In many routes, only seven routes have "Yes", which means they have earned more annual revenue than $815,188. However, there are only two routes that have a significant number of "Yes", other "Yes" only contain less than 10 items. The only difference between the two routes is "Application Channel". One of our target audiences is through " Referral", which has 1186 out of 1798 "valuable" companies. The other one is through "SEO", which has 516 out of 894 "valuable" companies.

In my point of view, those applicants that have the potential to become a big company are too busy with the bigger deal. They are less likely to engage in the earlier tier of the application process. Plus, financially stable companies that deal with banks frequently will make the applicants' credit scores higher. At last, the decision tree shows that the applicants who met the criteria above will be our target audience if they reach out to Biz2Credit through "Referral" or "SEO". I believe those outstanding companies would not select their partner through only advertisement. Surely, they tend to have more connections to ensure their partner is trustworthy and profitable. Therefore, applying for the loan through "referral" is the best way. As to "SEO", companies might know Biz2Credit from other business places or from their employees. With the lack of connection to Biz2Credit, the only way in three of the "application channel" is through "SEO", which is searching Biz2credit online.

Again, the bigger the company is the larger amount of loan it needs, making Biz2Credit yield more profit. The company that has the criteria,

which are “not self-directed”, “higher credit score”, “through referral” or “SEO”, can be a potential valuable customer for Biz2Credit.