Managing Staffing Inefficiencies Using Analytics

BA638\_Data Driven Decision Making & Optimization

Professor: Carol J. Cagle

Team 2 : Hien Do

Khanh Dang

Ravi Bandi

# **Overview**

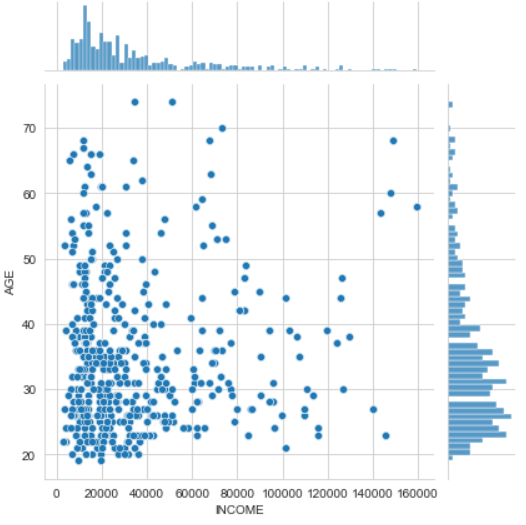
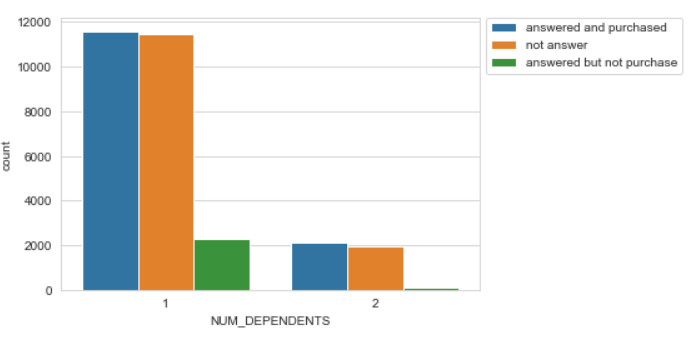
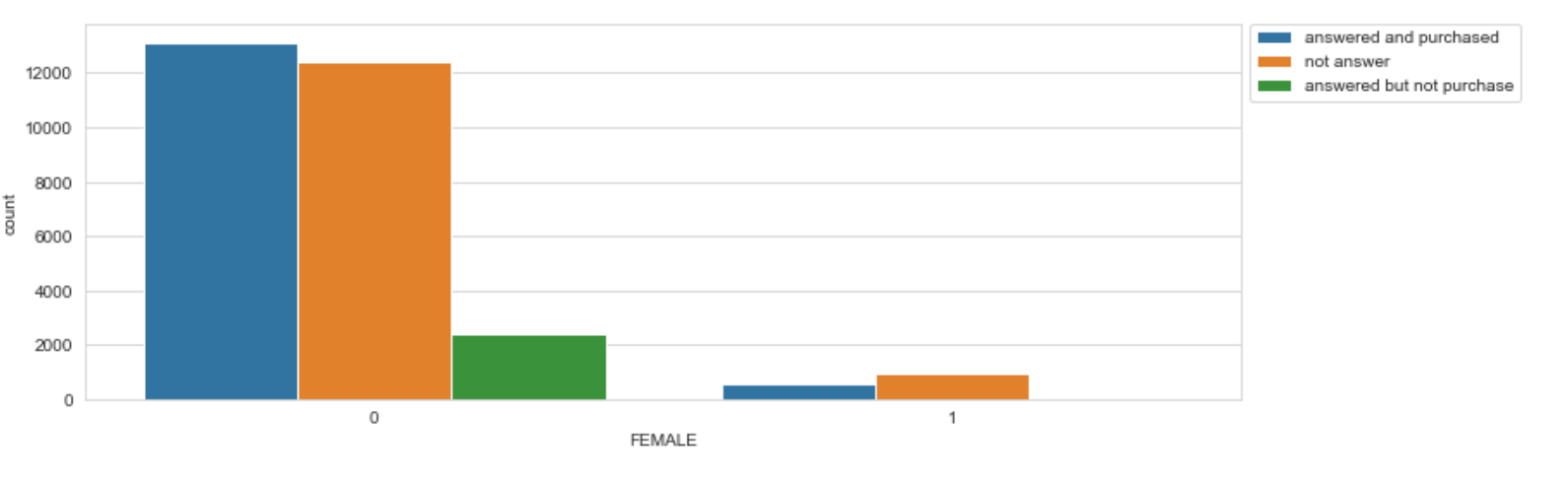
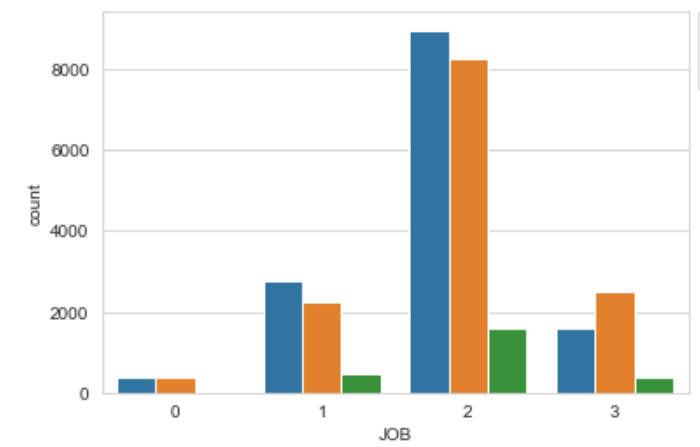
AdviseInvest is a venture-backup company and the company’s business is to provide a tool to ordinary customers for accessing financial services. While running their business, they have faced some challenges which negatively impact on their growth rate. In terms of staff’s capacity, sales representatives are underutilized since customer don’t pick up the phone call and company still need to pay salary for staff who were scheduled to provide consultancy. Another business problem is the operational process, due to inefficient process, it takes longer to proceed customer’s request. This results in customer do not want to answer the phone, particularly 50% of customers do not answer the phone after scheduling. Additionally, the company is unable to identify potential customers who are interested in financial services from the online form. Even though customers pick up the phone call, they might not determine to buy financial plans which leads to decreasing of sale revenue. AdviseInvest is a startup so their brand’s name is still new to customers, thus customers do not trust the company enough to use their financial services. Those problems resulting in the drop of customer subscription signup and increasing costs, and underutilization of employee. There are some stakeholders are involved in addressing those issues which include management level (general director, director of each department), sales representative and all staff who work under process, targeted customers, financial institutions as customers will give their account information for the company to manage, a data analyst who performs data analytics for AdviseInvest company, and other related departments like marketing, product development department. The objectives are improving customer onboarding process to avoid unanswered calls, optimizing employee schedule to resolve underutilization, figuring out potential customers who are interested in financial services. In this case, we will identify the business problems of AdviseInvest company and from them to figure out accurate analytical techniques.

# **Model**

Regarding to identified business problems of AvdviseInvest company, we will work on the historical data to understand root causes and current circumstances of business. There were only 55% customers answering the phone, while more than 8% of them answering but not purchasing the product. Thus, we will build a classification model to predict and classify customers into different groups who may not pick-up phone, may pick up phone. After we identify customers who may pick up phone, we are expecting 84% of them purchasing one or the other products offered based on historical data.

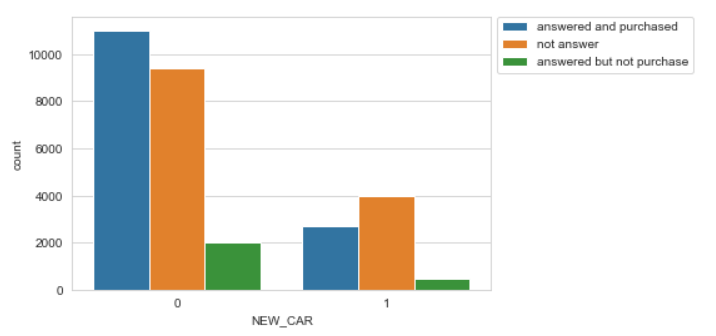
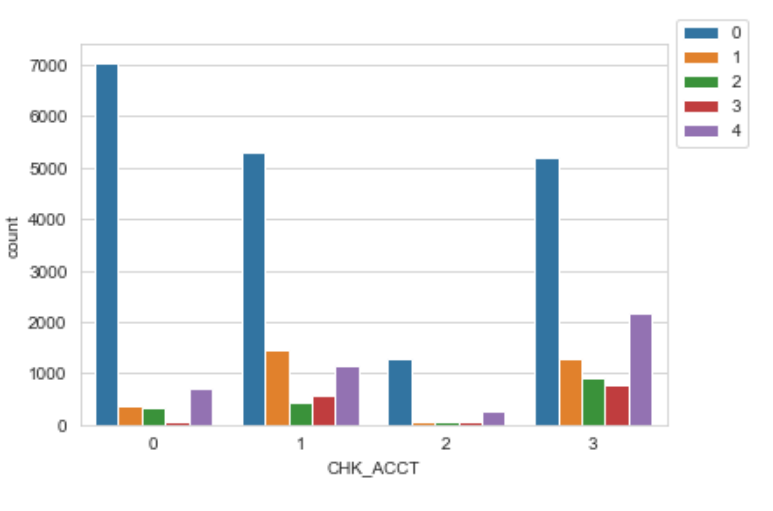
## ***Descriptive Analytics***

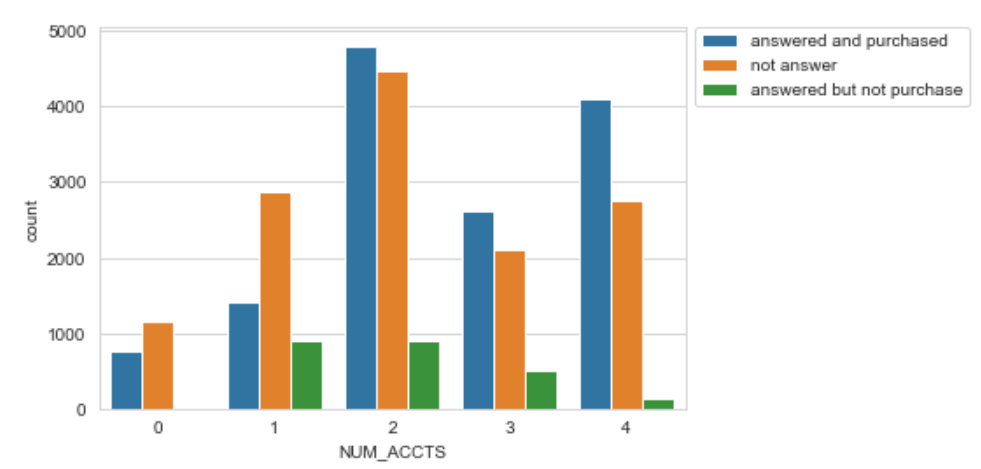
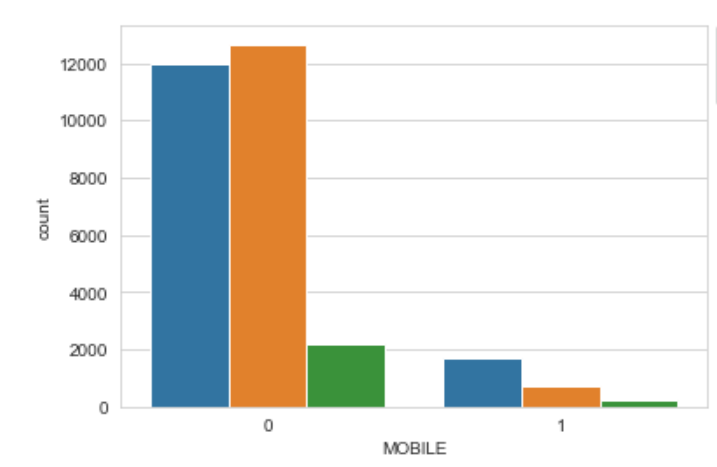
**Customers’ demography**:



The average income of our customers is around $33,000 per year, while most of our customers have annual income less than $25,000 per year. In this portfolio, there is high percentage of young generation customers as 75% of them are less than 40 years old with the average age of 35. Most of our customers are female (94.5% compare to 5.5%) which is explained for high proportion of responding to the call is female customers. 63% of our customers are mid-level job’s position who are the most potential with highest sale compare to other groups. In contrast, unemployed customers account for only 2% in the portfolio who may not interested on purchasing financial plans. 85% of customers have one dependent; however, the graph shows that having different number of dependents does not show much effect on customers answering the calls or purchasing the subscription. Nearly 70% of customers own their property which is a sign that they have stable income and high chance that they may have interest on other types of financial investments.

**Customers’ behavior:**



75% of our customer do not have a new car in the last three months. The bar chart shows that people who didn’t buy a new car in the last three months are more likely to answer company calls and opting for one of the financial products. In terms of checking accounts and saving accounts, in our customer portfolio, most of our customers do not have a saving account which makes up 63%, however, 33% of them have a Checking account value more than $2,000. Only 6.5% of our customers do not have open accounts. The bar chart also shows that the greater number of accounts customers have, higher the chances of customer answering the call and purchasing the product. About providing a phone number, 90% of our customers do not tend to provide their mobile phone number for following up. From given information about customer’s demography and buying behavior, we can figure out who are our targeted customers.

## ***Diagnostic Analytics****:*

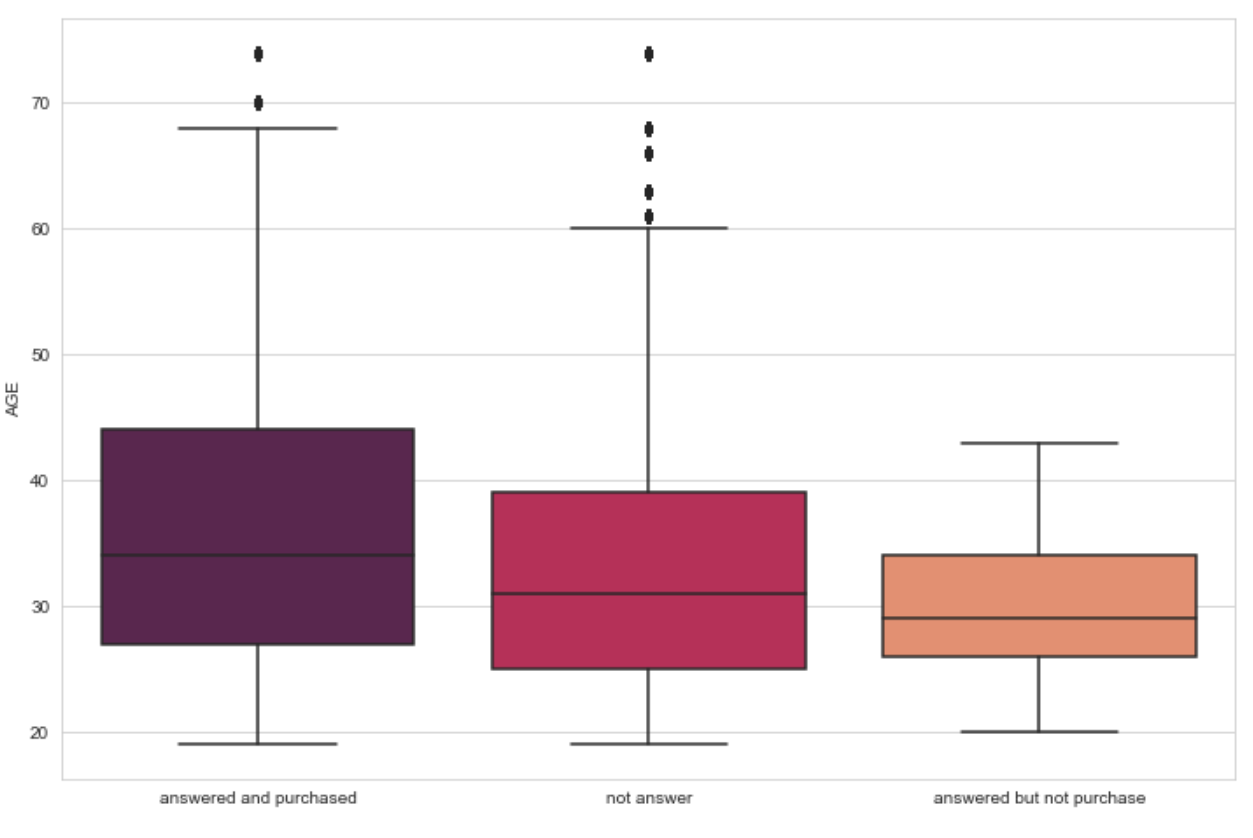
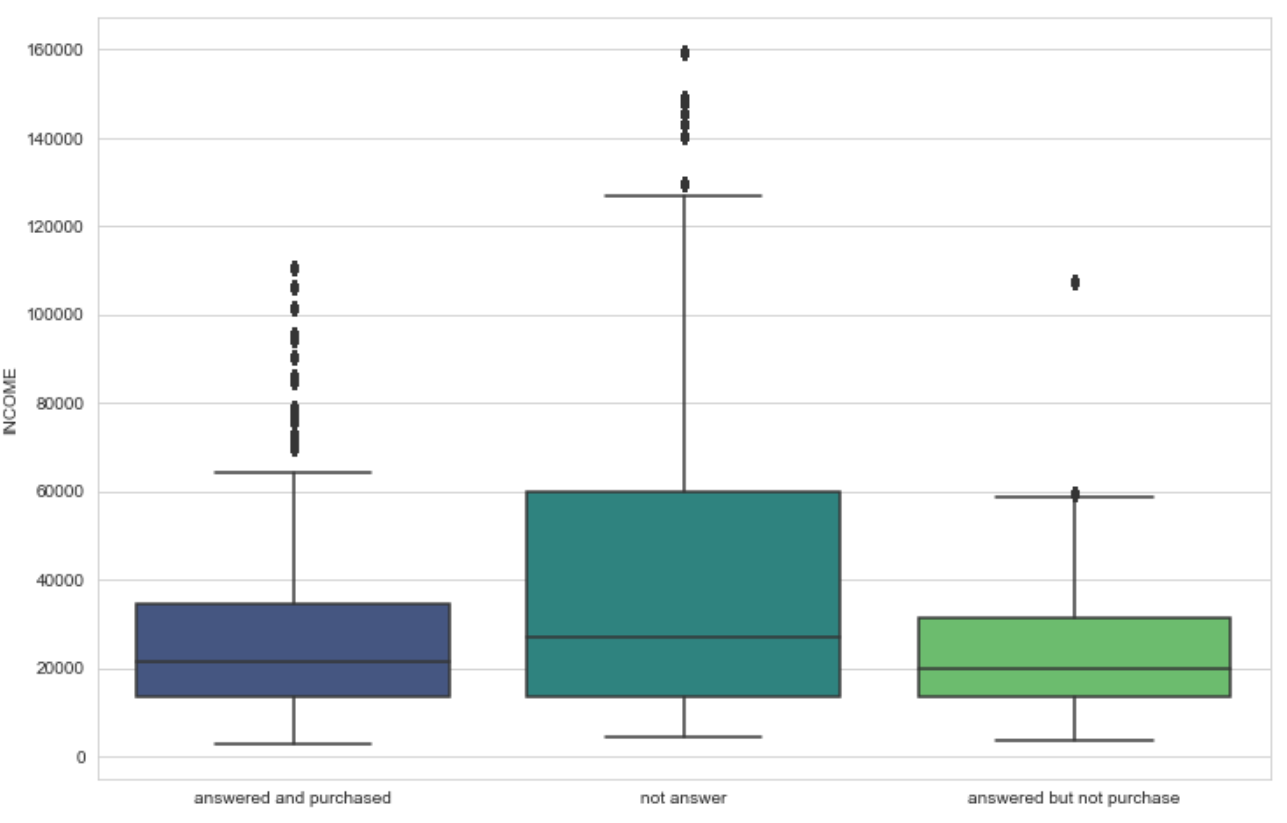


Figure. Boxplot of Income and Product Figure. Boxplot of Age and Product

According to the descriptive analytics, although we see that only 55% customers answered the phone, 83.6% of them purchased the product if they answered the phone. Therefore, we should focus on figuring out customers who will answer the phone more. In the Boxplot1, customers who usually opt for financial services are medium to low income or debt ridden because they would like to opt services to optimize their savings and expenditure.

Older customers are more likely to answer the phone and purchase the product than younger customers because at that age people start to have plan to save for their retirement, however in our portfolio, most of our customers are young customers.

The scatterplot of income and age separated by gender (appendix 2) shows that customers with higher age usually have higher income, besides, female customers tend to answer the call and complete the purchase, than the male customers (50% compare with 36%) it can be explained by that female customers are expected to have longer life expectancy, spend more for health care and household services than Male customers. However, female customers in our portfolio are young which is one of reasons for why our sales growth is low.

It is quite true that job type does not result in customer’s decision to have financial plans or not because unemployed or entry level customers usually have low or medium income, they opt for financial service to curb their overspending which help them to realize where need to cut back while middle or management customers with higher income opt for financial service because they may want to set their financial goals for long-term/future investments.

When it comes to rent/own residence, the bar chart shows that people who do not pay rent or own a residence (appendix 1) are more likely to answer the call than who pay the rent. It can be explained that when customers own a house, it is more likely that they are more in debt (mortgages or house payment) therefore they may opt for financial service to eliminate their debt, that helps them to avoid high-interest rate which will continue to shrink their wallet.

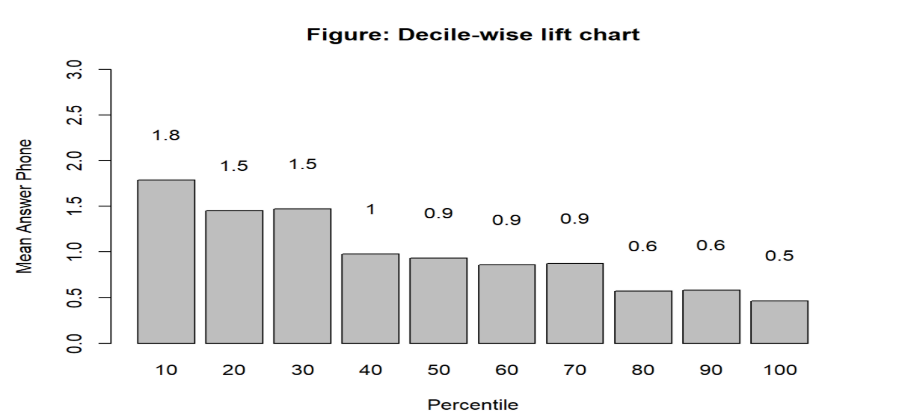
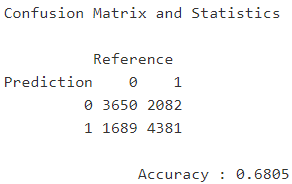
In terms of checking account/saving accounts the statistic tables (appendix 2) show that people with higher balance in checking accounts and saving accounts prefer to handle finances well for which they are answering our calls and opt for services. However, the percentage of customers who have higher balances in checking account/saving accounts is quite small in our customer portfolio. Customers with low balance or no account are more than half who signed up for a call. However, they are not answering the call or not completing the purchase. Like customers who have many numbers of accounts, they are more likely to optimize their spending.

The statistic table of mobile (appendix 2) shows that more than 71% of customers providing their mobile phone number answered the phone and more than 64% of customers out of the 71% purchased the products, because when customer provide their private contact number, means they are more likely to answer the call as it is accessible, however in our portfolio, the number of customers provided their mobile phone is small (10%).

## ***Predictive Model***

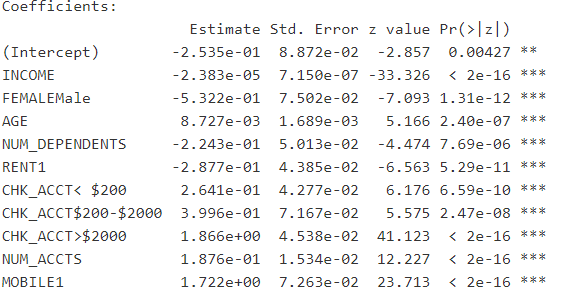
To identify or classify potential customers we are using a predictive model where we have tried to classify a customer who may answer the call to the customer who may not answer the call. To do this, we have considered some factors such as Income, Gender, Age, Number of dependents, Rent, Number of accounts, Checking account and Mobile. Some of the factors were not included in the model as they have similar behavior as some of the factors that are included in the model.

*Results*



According to the confusion matrix, although the accuracy rate is not high (68.05%), the percentage of customers who will answer the phone if we predict ‘they may answer the phone’ is higher than not using the predictive model (68% compared with 55%) and likewise for customers who are predicted to not answer the phone.

The decile lift chart indicates that the top 10% of clients from the predictive model will answer the call to sale representatives nearly twice higher compared to 10% random selection of customers.

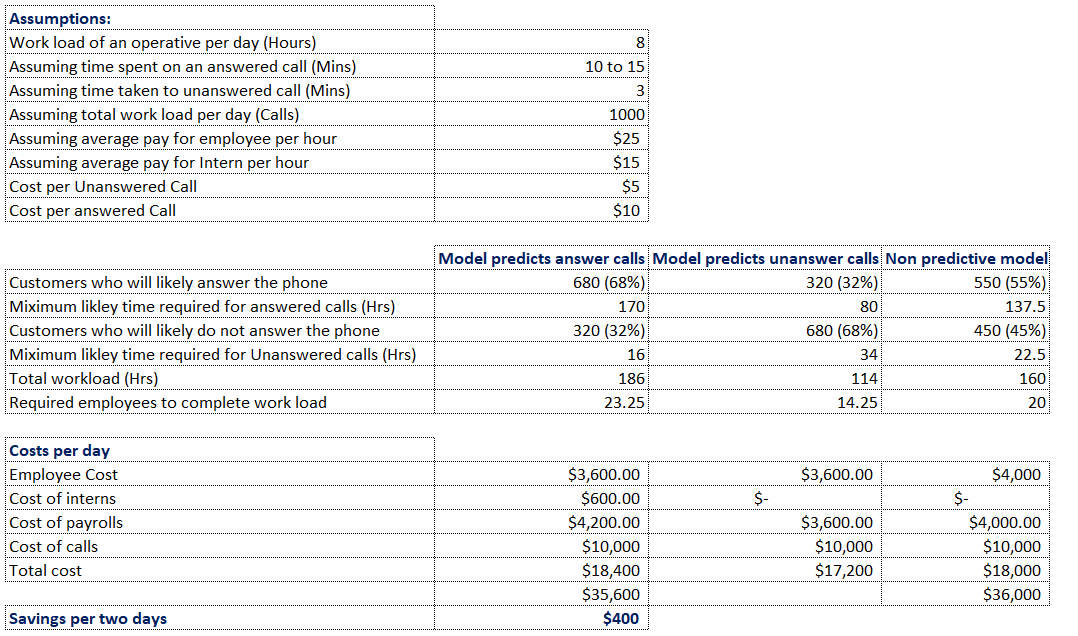


**Potential customers:**

Our most potential patrons are Middle-aged Females with low to medium income range who have a smaller number of dependents and does not pay rent or own a residence but hold the higher balance in checking or saving accounts and have a higher number of accounts and provide a mobile phone number.

# **Recommendations**

**Staff Scheduling:**



Due to inefficient onboarding process, many customers need to wait for a long time to communicate with sales, thus we will introduce chat customer services where a representative can handle similar requests. Specifically, at first when customers interest on the products they usually ask the general questions and initial requirements of financial plans, the chat will be more efficient to handle those requests. In case customers want to have more detailed information about the services, they can request a call from the company’s representative. Additionally, introduce chatbots when direct customer representative not required. In term of staff, each of the representatives must be subject matter experts of a certain area or one of the products. Sale representatives do not struggle to remember and find the answers for many types of questions. This will not only help saving time of process but also being more professional while communicating to customers. In the online form, the mobile number should be mandatory to fulfil; thus, representatives ease to follow up with customers when necessary. Since AdviseInvest is a start-up, not many clients know about this company, enhancing the brand’s reputation to customers plays a vital role to increase the growth rate of the business. We will raise customer’s awareness by cooperating with well-known financial institutions such as having advertising about AdviseInvest on their website. Therefore, we can get the customer’s attention about our company when they go to financial institutions website. Due to the potential groups of customers that have been identified from the predictive model, we will also have advertising on the grocery stores or grocery website to approach attention from customers who are female, medium-income and low number of dependents. A quick survey with three specific questions about call’s quality which includes “How easy was your onboarding process?” (1: not easy at all, 5: very easy), “How efficiently do you find our sale representatives base?” (1: not efficient at all, 5: very efficient), and “Do you have any feedback to improve our services?” (No/Yes, please specific if yes). From the customer’s feedback, we can improve the process and conduct sale training to enhance the customer’s experience. Additionally, we will pay more attention to cybersecurity such as get as many safety certifications as possible for the business which are the symbol for trust such as Better Business Bureau. At the same time, advertising more about the financial safety practices the company follows which will raise awareness among customer and reduces the scepticism.

**Further considerations:**

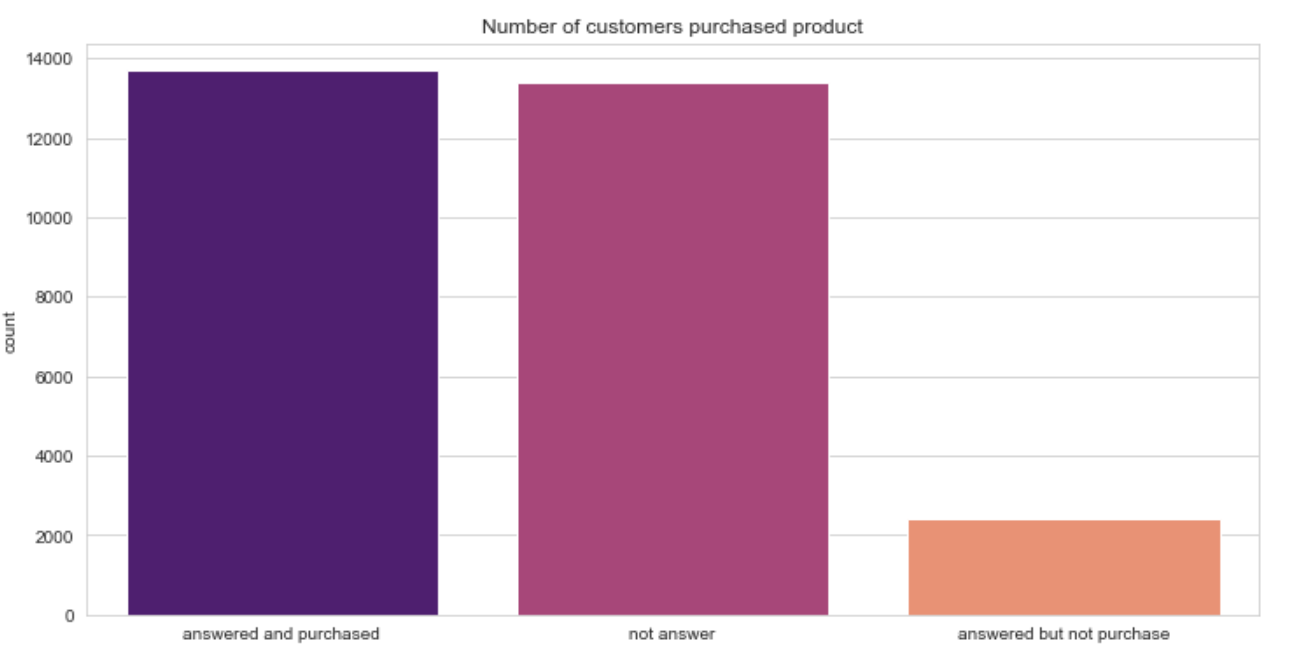
For further future model development, we can consider the call data information i.e., call duration and type of product-specific information communicated based on which we can serve tailored needs for the customer. and using the same information as above we can identify customers who consume less time on call and make a purchase, to prioritize them over a customer who consumes more call time who purchases the product. Assign the (customers) equal workload to representatives based on the total time consumption by customers on call. (*representative one has 2 customers, customer one uses 10 minutes and customer two uses 50 minutes. Similarly, representative 2 has 2 customers, customer 1 uses 30 minutes and customer two use 30 minutes hence the workload of both representatives is equal to 1 hour*). Proposing incentives for representatives based on performance.

# **References**

Dessislava A. Pachamanova 2015, “Case: Managing Staffing Inefficiencies Using Analytics”, Vol. 16, No. 1, September 2015, p. 23. https://pubsonline.informs.org/doi/pdf/10.1287/ited.2015.0146cs

# **Appendix**

1. **Descriptive Analytics**



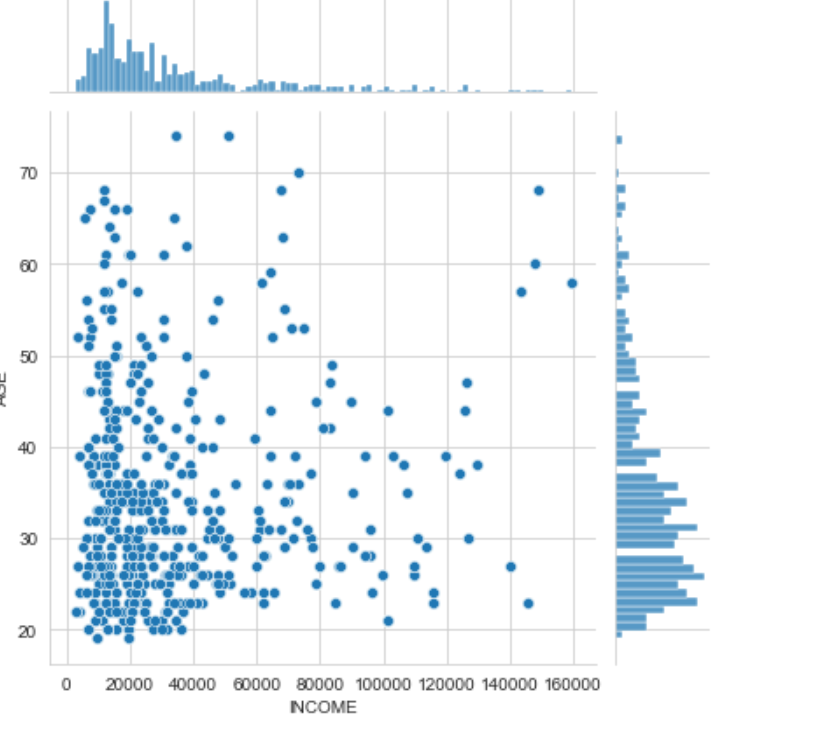
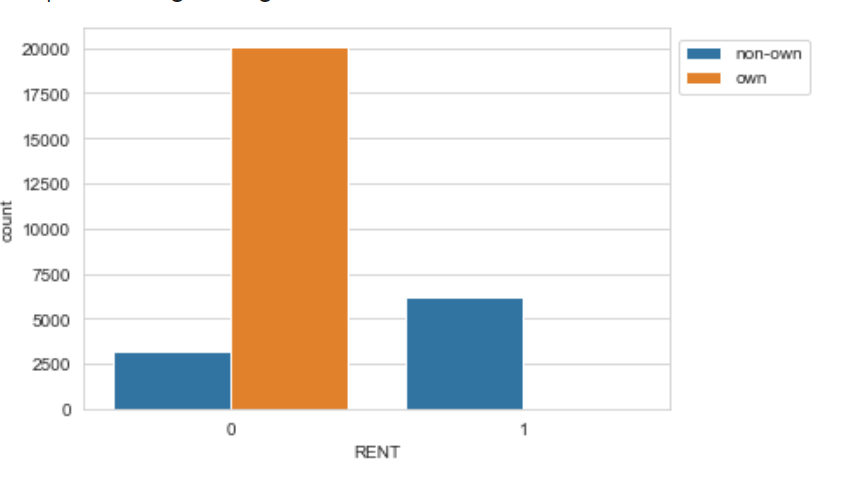


Figure. Rent and own-Residence

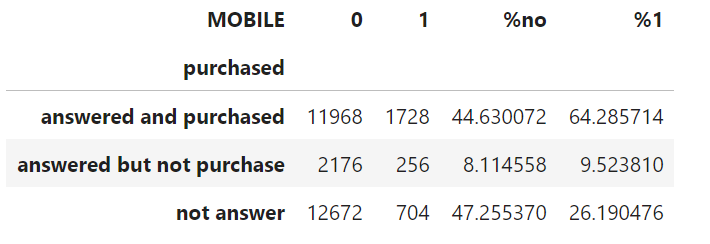


1. **Diagnostic Analytics**

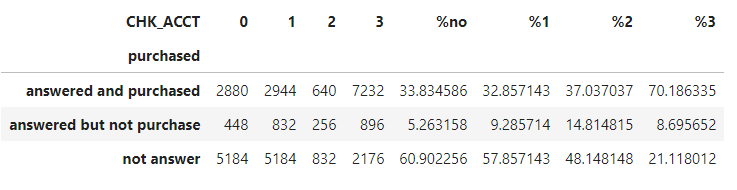
Figure. The Scatterplot of income and age separated by gender



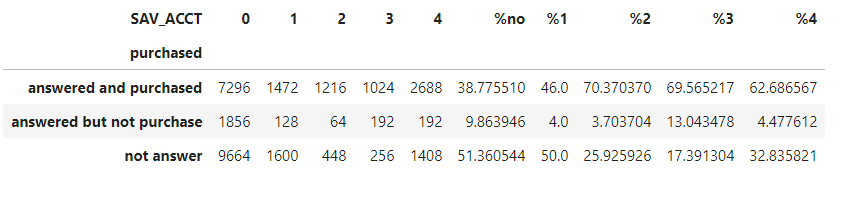
Table. Statistic Table of Mobile



Statistic Table of Checking Account



Statistic Table of Saving Account



1. **Exploring data using Python**

**import** numpy **as** np

**import** pandas **as** pd

In [2]:

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** sns

**from** plotnine **import** **\***

**%matplotlib** inline

In [3]:

df **=** pd**.**read\_csv('AdviseInvestData.csv')

In [4]:

df

Out[4]:

|  | **OBS.** | **ANSWERED** | **INCOME** | **FEMALE** | **AGE** | **JOB** | **NUM\_DEPENDENTS** | **RENT** | **OWN\_RES** | **NEW\_CAR** | **CHK\_ACCT** | **SAV\_ACCT** | **NUM\_ACCTS** | **MOBILE** | **PRODUCT** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | 133 | 1 | 13520 | 0 | 23 | 0 | 1 | 1 | 0 | 0 | 0 | 2 | 0 | 1 | 2 |
| **1** | 1509 | 0 | 14780 | 0 | 22 | 2 | 1 | 0 | 1 | 0 | 3 | 0 | 3 | 0 | 0 |
| **2** | 1530 | 1 | 37570 | 0 | 62 | 2 | 1 | 0 | 0 | 1 | 3 | 0 | 4 | 0 | 4 |
| **3** | 1886 | 0 | 12450 | 0 | 33 | 2 | 1 | 0 | 1 | 0 | 1 | 0 | 2 | 0 | 0 |
| **4** | 2463 | 0 | 12400 | 0 | 48 | 1 | 2 | 0 | 0 | 1 | 2 | 1 | 4 | 0 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **29499** | 9998377 | 1 | 11630 | 0 | 44 | 2 | 1 | 0 | 1 | 1 | 3 | 2 | 2 | 0 | 4 |
| **29500** | 9998970 | 0 | 28640 | 0 | 34 | 1 | 2 | 0 | 1 | 0 | 2 | 0 | 2 | 0 | 0 |
| **29501** | 9999108 | 1 | 34480 | 0 | 74 | 1 | 1 | 0 | 1 | 0 | 3 | 0 | 3 | 0 | 4 |
| **29502** | 9999701 | 1 | 10550 | 0 | 30 | 2 | 1 | 0 | 1 | 1 | 3 | 0 | 1 | 0 | 1 |
| **29503** | 9999738 | 0 | 125790 | 0 | 44 | 3 | 1 | 0 | 0 | 0 | 1 | 0 | 4 | 1 | 0 |

29504 rows × 15 columns

In [5]:

df**.**info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 29504 entries, 0 to 29503

Data columns (total 15 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 OBS. 29504 non-null int64

1 ANSWERED 29504 non-null int64

2 INCOME 29504 non-null int64

3 FEMALE 29504 non-null int64

4 AGE 29504 non-null int64

5 JOB 29504 non-null int64

6 NUM\_DEPENDENTS 29504 non-null int64

7 RENT 29504 non-null int64

8 OWN\_RES 29504 non-null int64

9 NEW\_CAR 29504 non-null int64

10 CHK\_ACCT 29504 non-null int64

11 SAV\_ACCT 29504 non-null int64

12 NUM\_ACCTS 29504 non-null int64

13 MOBILE 29504 non-null int64

14 PRODUCT 29504 non-null int64

dtypes: int64(15)

memory usage: 3.4 MB

## **Exploring data by visualization**

**1. Classification to predict customers picking up phone call or not**

* The number of customers who answerd the phone

In [6]:

df['ANSWERED']**.**value\_counts()

Out[6]:

1 16128

0 13376

Name: ANSWERED, dtype: int64

In [9]:

**import** warnings

warnings**.**filterwarnings("ignore")

sns**.**countplot(df['ANSWERED'])

plt**.**title('Bar chart the number of customers picking up the phone')

Out[9]:

Text(0.5, 1.0, 'Bar chart the number of customers picking up the phone')

****

* Exploring the impact of job type on picking up the phone

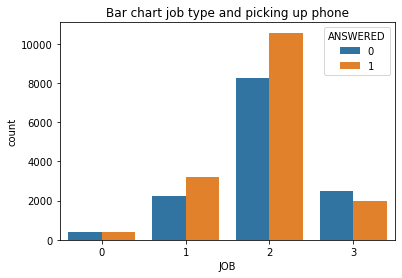
In [10]:

sns**.**countplot(df['JOB'], hue**=**'ANSWERED', data**=**df)

plt**.**title('Bar chart job type and picking up phone')

Out[10]:

Text(0.5, 1.0, 'Bar chart job type and picking up phone')

****

In [11]:

*# The percentage of each job type on picking up the phone*

job\_answer **=** pd**.**DataFrame(df**.**groupby(['JOB','ANSWERED'])**.**count()['OBS.'])**.**reset\_index()**.**pivot\_table(values**=**'OBS.', index**=**'JOB',

columns **=** 'ANSWERED')

job\_answer['percentage'] **=** job\_answer[1]**/**(job\_answer[0]**+**job\_answer[1])**\***100

job\_answer

Out[11]:

| **ANSWERED** | **0** | **1** | **percentage** |
| --- | --- | --- | --- |
| **JOB** |  |  |  |
| **0** | 384 | 384 | 50.000000 |
| **1** | 2240 | 3200 | 58.823529 |
| **2** | 8256 | 10560 | 56.122449 |
| **3** | 2496 | 1984 | 44.285714 |

* Exploring the impact of income on picking up phone

In [12]:

plt**.**figure(figsize**=**(16,4))

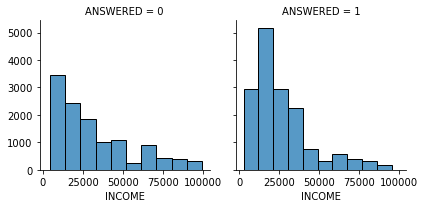
income\_map **=** sns**.**FacetGrid(data**=**df[df['INCOME'] **<** 100000], col**=**'ANSWERED')

income\_map**.**map(sns**.**histplot, 'INCOME', bins**=**10)

Out[12]:

<seaborn.axisgrid.FacetGrid at 0x225bcc5a348>

<Figure size 1152x288 with 0 Axes>

****

In [18]:

*# summary statistics of income basing on 'answer'*

df**.**groupby('ANSWERED')['INCOME']**.**mean(), df**.**groupby('ANSWERED')['INCOME']**.**median()

Out[18]:

(ANSWERED

0 40979.808612

1 27794.801587

Name: INCOME, dtype: float64,

ANSWERED

0 27120

1 21265

Name: INCOME, dtype: int64)

In [17]:

*# summary statistics of income basing on job*

df**.**groupby('JOB')['INCOME']**.**min(), df**.**groupby('JOB')['INCOME']**.**max(), df**.**groupby('JOB')['INCOME']**.**mean(), df**.**groupby('JOB')['INCOME']**.**median(), df**.**groupby('JOB')['INCOME']**.**std()

Out[17]:

(JOB

0 6090

1 2760

2 3380

3 12090

Name: INCOME, dtype: int64,

JOB

0 145550

1 115900

2 159450

3 148960

Name: INCOME, dtype: int64,

JOB

0 27012.500000

1 26902.705882

2 30148.843537

3 58491.857143

Name: INCOME, dtype: float64,

JOB

0 12710

1 18810

2 21735

3 47090

Name: INCOME, dtype: int64,

JOB

0 37185.812073

1 22591.282219

2 24773.798323

3 38946.255171

Name: INCOME, dtype: float64)

* Exploring the impact of gender on picking up the phone

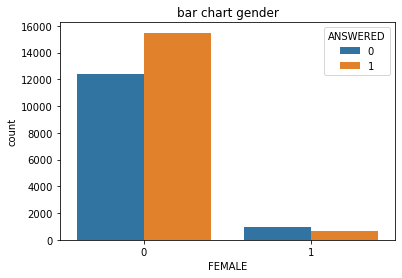
In [19]:

sns**.**countplot(data**=**df, x**=**'FEMALE', hue **=** 'ANSWERED')

plt**.**title('bar chart gender')

Out[19]:

Text(0.5, 1.0, 'bar chart gender')

****

In [21]:

*# The percentage of female on picking up the phone*

female\_answer **=** pd**.**DataFrame(df**.**groupby(['FEMALE','ANSWERED'])**.**count()['OBS.'])**.**reset\_index()**.**pivot\_table(values**=**'OBS.', index**=**'FEMALE',

columns **=** 'ANSWERED')

female\_answer['percentage'] **=** female\_answer[1]**/**(female\_answer[0]**+**female\_answer[1])**\***100

female\_answer

Out[21]:

| **ANSWERED** | **0** | **1** | **percentage** |
| --- | --- | --- | --- |
| **FEMALE** |  |  |  |
| **0** | 12416 | 15488 | 55.504587 |
| **1** | 960 | 640 | 40.000000 |

In [23]:

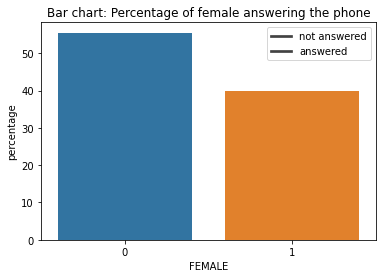
sns**.**barplot(data**=**female\_answer**.**reset\_index(),x**=**'FEMALE',y**=**'percentage')

plt**.**title('Bar chart: Percentage of female answering the phone')

plt**.**legend(['not answered', 'answered'])

Out[23]:

<matplotlib.legend.Legend at 0x225bd2e3bc8>

****

* Exploring the impact of age on answering the phone

In [24]:

*# The summary statistics*

df**.**groupby('ANSWERED')['AGE']**.**mean(), df**.**groupby('ANSWERED')['AGE']**.**median()

Out[24]:

(ANSWERED

0 33.837321

1 35.543651

Name: AGE, dtype: float64,

ANSWERED

0 31.0

1 33.5

Name: AGE, dtype: float64)

In [25]:

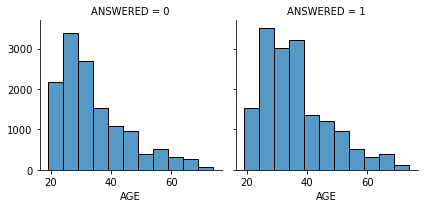
*# The distribution plot*

age\_map **=** sns**.**FacetGrid(data**=**df, col**=**'ANSWERED')

age\_map**.**map(sns**.**histplot, 'AGE', bins**=**11)

Out[25]:

<seaborn.axisgrid.FacetGrid at 0x225be36a248>

****

In [22]:

**import** warnings

warnings**.**filterwarnings("ignore")

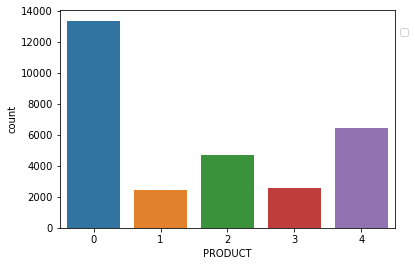
sns**.**countplot(df['PRODUCT'])

plt**.**legend(loc**=**'center left', bbox\_to\_anchor**=**(1, 0.9))

No handles with labels found to put in legend.

Out[22]:

<matplotlib.legend.Legend at 0x2848c937588>

****

* The impact of number of dependents on picking up the phone

In [27]:

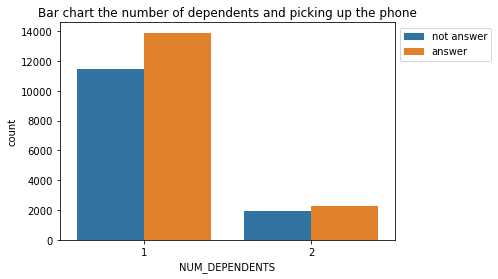
sns**.**countplot(df['NUM\_DEPENDENTS'], hue**=**'ANSWERED', data**=**df )

plt**.**legend(labels **=** ['not answer', 'answer'],loc**=**'center left', bbox\_to\_anchor**=**(1, 0.9))

plt**.**title('Bar chart the number of dependents and picking up the phone')

Out[27]:

Text(0.5, 1.0, 'Bar chart the number of dependents and picking up the phone')

****

In [29]:

*# The percentage*

dep\_answer **=** pd**.**DataFrame(df**.**groupby(['NUM\_DEPENDENTS','ANSWERED'])**.**count()['OBS.'])**.**reset\_index()**.**pivot\_table(values**=**'OBS.', index**=**'NUM\_DEPENDENTS',

columns **=** 'ANSWERED')

dep\_answer['percentage'] **=** dep\_answer[1]**/**(dep\_answer[0]**+**dep\_answer[1])**\***100

dep\_answer

Out[29]:

| **ANSWERED** | **0** | **1** | **percentage** |
| --- | --- | --- | --- |
| **NUM\_DEPENDENTS** |  |  |  |
| **1** | 11456 | 13888 | 54.797980 |
| **2** | 1920 | 2240 | 53.846154 |

* The impact of rent on picking up the phone

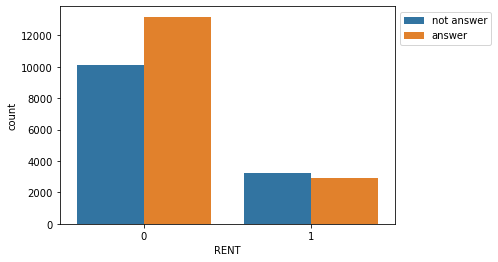
In [26]:

sns**.**countplot(df['RENT'], hue**=**'ANSWERED', data**=**df )

plt**.**legend(labels **=** ['not answer', 'answer'],loc**=**'center left', bbox\_to\_anchor**=**(1, 0.9))

Out[26]:

<matplotlib.legend.Legend at 0x2848c96dc48>

****

In [31]:

*# The percentage*

rent\_answer **=** pd**.**DataFrame(df**.**groupby(['RENT','ANSWERED'])**.**count()['OBS.'])**.**reset\_index()**.**pivot\_table(values**=**'OBS.', index**=**'RENT',

columns **=** 'ANSWERED')

rent\_answer['percentage'] **=** rent\_answer[1]**/**(rent\_answer[0]**+**rent\_answer[1])**\***100

rent\_answer

Out[31]:

| **ANSWERED** | **0** | **1** | **percentage** |
| --- | --- | --- | --- |
| **RENT** |  |  |  |
| **0** | 10112 | 13184 | 56.593407 |
| **1** | 3264 | 2944 | 47.422680 |

* The impact of own residence on picking up the phone

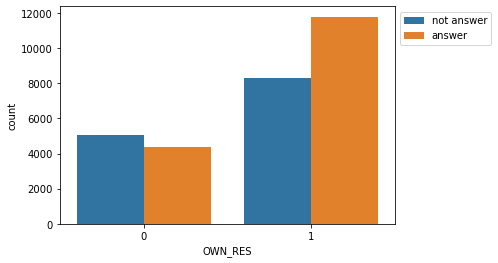
In [32]:

sns**.**countplot(df['OWN\_RES'], hue**=**'ANSWERED', data**=**df )

plt**.**legend(labels **=** ['not answer', 'answer'],loc**=**'center left', bbox\_to\_anchor**=**(1, 0.9))

Out[32]:

<matplotlib.legend.Legend at 0x225be364208>

****

In [33]:

*# The percentage*

own\_answer **=** pd**.**DataFrame(df**.**groupby(['OWN\_RES','ANSWERED'])**.**count()['OBS.'])**.**reset\_index()**.**pivot\_table(values**=**'OBS.', index**=**'OWN\_RES',

columns **=** 'ANSWERED')

own\_answer['percentage'] **=** own\_answer[1]**/**(own\_answer[0]**+**own\_answer[1])**\***100

own\_answer

Out[33]:

| **ANSWERED** | **0** | **1** | **percentage** |
| --- | --- | --- | --- |
| **OWN\_RES** |  |  |  |
| **0** | 5056 | 4352 | 46.258503 |
| **1** | 8320 | 11776 | 58.598726 |

* The impact of new car on picking up the phone

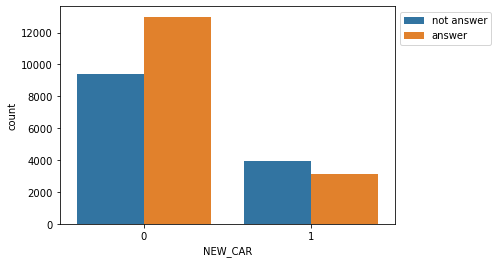
In [34]:

sns**.**countplot(df['NEW\_CAR'], hue**=**'ANSWERED', data**=**df )

plt**.**legend(labels **=** ['not answer', 'answer'],loc**=**'center left', bbox\_to\_anchor**=**(1, 0.9))

Out[34]:

<matplotlib.legend.Legend at 0x225be55f488>

****

In [35]:

*# The percentage*

car\_answer **=** pd**.**DataFrame(df**.**groupby(['NEW\_CAR','ANSWERED'])**.**count()['OBS.'])**.**reset\_index()**.**pivot\_table(values**=**'OBS.', index**=**'NEW\_CAR',

columns **=** 'ANSWERED')

car\_answer['percentage'] **=** car\_answer[1]**/**(car\_answer[0]**+**car\_answer[1])**\***100

car\_answer

Out[35]:

| **ANSWERED** | **0** | **1** | **percentage** |
| --- | --- | --- | --- |
| **NEW\_CAR** |  |  |  |
| **0** | 9408 | 12992 | 58.000000 |
| **1** | 3968 | 3136 | 44.144144 |

* The impact of checking account on picking up the phone

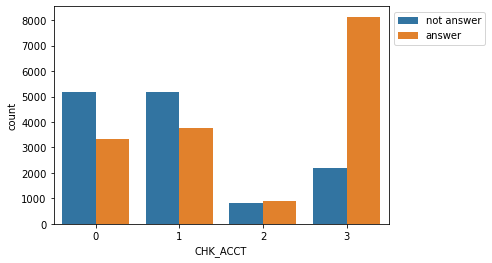
In [36]:

sns**.**countplot(df['CHK\_ACCT'], hue**=**'ANSWERED', data**=**df )

plt**.**legend(labels **=** ['not answer', 'answer'],loc**=**'center left', bbox\_to\_anchor**=**(1, 0.9))

Out[36]:

<matplotlib.legend.Legend at 0x225be5948c8>

****

In [37]:

*# The percentage*

check\_answer **=** pd**.**DataFrame(df**.**groupby(['CHK\_ACCT','ANSWERED'])**.**count()['OBS.'])**.**reset\_index()**.**pivot\_table(values**=**'OBS.', index**=**'CHK\_ACCT',

columns **=** 'ANSWERED')

check\_answer['percentage'] **=** check\_answer[1]**/**(check\_answer[0]**+**check\_answer[1])**\***100

check\_answer

Out[37]:

| **ANSWERED** | **0** | **1** | **percentage** |
| --- | --- | --- | --- |
| **CHK\_ACCT** |  |  |  |
| **0** | 5184 | 3328 | 39.097744 |
| **1** | 5184 | 3776 | 42.142857 |
| **2** | 832 | 896 | 51.851852 |
| **3** | 2176 | 8128 | 78.881988 |

* The impact of saving accounts on picking up phone

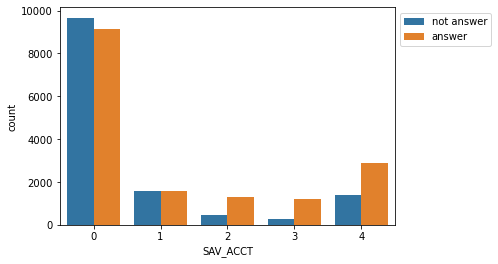
In [38]:

sns**.**countplot(df['SAV\_ACCT'], hue**=**'ANSWERED', data**=**df )

plt**.**legend(labels **=** ['not answer', 'answer'],loc**=**'center left', bbox\_to\_anchor**=**(1, 0.9))

Out[38]:

<matplotlib.legend.Legend at 0x225be6b26c8>

****

In [39]:

*# The percentage*

sav\_answer **=** pd**.**DataFrame(df**.**groupby(['SAV\_ACCT','ANSWERED'])**.**count()['OBS.'])**.**reset\_index()**.**pivot\_table(values**=**'OBS.', index**=**'SAV\_ACCT',

columns **=** 'ANSWERED')

sav\_answer['percentage'] **=** sav\_answer[1]**/**(sav\_answer[0]**+**sav\_answer[1])**\***100

sav\_answer

Out[39]:

| **ANSWERED** | **0** | **1** | **percentage** |
| --- | --- | --- | --- |
| **SAV\_ACCT** |  |  |  |
| **0** | 9664 | 9152 | 48.639456 |
| **1** | 1600 | 1600 | 50.000000 |
| **2** | 448 | 1280 | 74.074074 |
| **3** | 256 | 1216 | 82.608696 |
| **4** | 1408 | 2880 | 67.164179 |

* The impact of # accounts on picking up phone

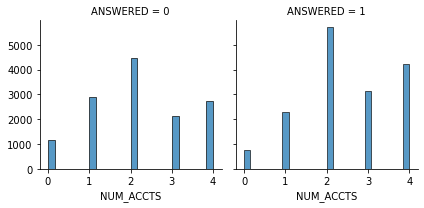
In [40]:

num\_map **=** sns**.**FacetGrid(data**=**df, col**=**'ANSWERED')

num\_map**.**map(sns**.**histplot, 'NUM\_ACCTS')

Out[40]:

<seaborn.axisgrid.FacetGrid at 0x225be54a2c8>

****

In [41]:

*# grouping into 2 types: account & non account*

**def** num\_acc(x):

**if** x **==** 0:

**return** 'non account'

**else**:

**return** 'account'

In [42]:

df['account'] **=** df['NUM\_ACCTS']**.**apply(**lambda** x: num\_acc(x))

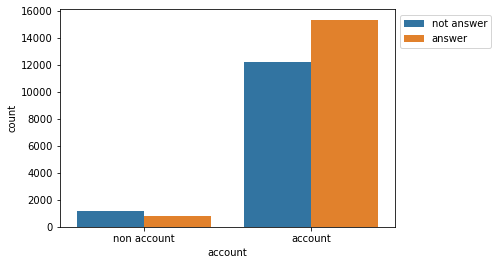
In [43]:

sns**.**countplot(df['account'], hue**=**'ANSWERED', data**=**df )

plt**.**legend(labels **=** ['not answer', 'answer'],loc**=**'center left', bbox\_to\_anchor**=**(1, 0.9))

Out[43]:

<matplotlib.legend.Legend at 0x225be4f83c8>

****

In [44]:

*# The percentage*

acc\_answer **=** pd**.**DataFrame(df**.**groupby(['account','ANSWERED'])**.**count()['OBS.'])**.**reset\_index()**.**pivot\_table(values**=**'OBS.', index**=**'account',

columns **=** 'ANSWERED')

acc\_answer['percentage'] **=** acc\_answer[1]**/**(acc\_answer[0]**+**acc\_answer[1])**\***100

acc\_answer

Out[44]:

| **ANSWERED** | **0** | **1** | **percentage** |
| --- | --- | --- | --- |
| **account** |  |  |  |
| **account** | 12224 | 15360 | 55.684455 |
| **non account** | 1152 | 768 | 40.000000 |

* The impact of mobile on picking up the phone

In [45]:

*# The percentage*

mobile\_answer **=** pd**.**DataFrame(df**.**groupby(['MOBILE','ANSWERED'])**.**count()['OBS.'])**.**reset\_index()**.**pivot\_table(values**=**'OBS.', index**=**'MOBILE',

columns **=** 'ANSWERED')

mobile\_answer['percentage'] **=** mobile\_answer[1]**/**(mobile\_answer[0]**+**mobile\_answer[1])**\***100

mobile\_answer

Out[45]:

| **ANSWERED** | **0** | **1** | **percentage** |
| --- | --- | --- | --- |
| **MOBILE** |  |  |  |
| **0** | 12672 | 14144 | 52.744630 |
| **1** | 704 | 1984 | 73.809524 |

In [46]:

sns**.**barplot(data**=**mobile\_answer**.**reset\_index(),x**=**'MOBILE',y**=**'percentage')

plt**.**title('Percentage of mobile answering the phone')

Out[46]:

Text(0.5, 1.0, 'Percentage of mobile answering the phone')

****

**2. Classification to predict who will buy the product if they pick up the phone**

In [47]:

*# Grouping 2 types: purchased product or not*

**def** product\_group(x):

**if** x **==** 1:

**return** 'not purchase'

**else**:

**return** 'purchase'

In [48]:

*# removing customers who did not pick up the phone*

df2 **=** df[df['PRODUCT']**!=**0]

In [49]:

df2

Out[49]:

|  | **OBS.** | **ANSWERED** | **INCOME** | **FEMALE** | **AGE** | **JOB** | **NUM\_DEPENDENTS** | **RENT** | **OWN\_RES** | **NEW\_CAR** | **CHK\_ACCT** | **SAV\_ACCT** | **NUM\_ACCTS** | **MOBILE** | **PRODUCT** | **account** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | 133 | 1 | 13520 | 0 | 23 | 0 | 1 | 1 | 0 | 0 | 0 | 2 | 0 | 1 | 2 | non account |
| **2** | 1530 | 1 | 37570 | 0 | 62 | 2 | 1 | 0 | 0 | 1 | 3 | 0 | 4 | 0 | 4 | account |
| **5** | 2667 | 1 | 14030 | 0 | 28 | 2 | 1 | 1 | 0 | 1 | 0 | 0 | 2 | 0 | 4 | account |
| **6** | 2812 | 1 | 9320 | 0 | 24 | 2 | 1 | 0 | 1 | 0 | 3 | 0 | 2 | 0 | 2 | account |
| **7** | 3555 | 1 | 11750 | 0 | 68 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 3 | non account |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **29495** | 9994393 | 1 | 33680 | 0 | 23 | 2 | 1 | 1 | 0 | 0 | 3 | 3 | 4 | 1 | 4 | account |
| **29497** | 9995719 | 1 | 5180 | 0 | 29 | 2 | 1 | 0 | 1 | 0 | 3 | 0 | 2 | 0 | 1 | account |
| **29499** | 9998377 | 1 | 11630 | 0 | 44 | 2 | 1 | 0 | 1 | 1 | 3 | 2 | 2 | 0 | 4 | account |
| **29501** | 9999108 | 1 | 34480 | 0 | 74 | 1 | 1 | 0 | 1 | 0 | 3 | 0 | 3 | 0 | 4 | account |
| **29502** | 9999701 | 1 | 10550 | 0 | 30 | 2 | 1 | 0 | 1 | 1 | 3 | 0 | 1 | 0 | 1 | account |

16128 rows × 16 columns

In [50]:

df2['purchased'] **=** df2['PRODUCT']**.**apply(**lambda** x: product\_group(x))

In [51]:

df2

Out[51]:

|  | **OBS.** | **ANSWERED** | **INCOME** | **FEMALE** | **AGE** | **JOB** | **NUM\_DEPENDENTS** | **RENT** | **OWN\_RES** | **NEW\_CAR** | **CHK\_ACCT** | **SAV\_ACCT** | **NUM\_ACCTS** | **MOBILE** | **PRODUCT** | **account** | **purchased** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | 133 | 1 | 13520 | 0 | 23 | 0 | 1 | 1 | 0 | 0 | 0 | 2 | 0 | 1 | 2 | non account | purchase |
| **2** | 1530 | 1 | 37570 | 0 | 62 | 2 | 1 | 0 | 0 | 1 | 3 | 0 | 4 | 0 | 4 | account | purchase |
| **5** | 2667 | 1 | 14030 | 0 | 28 | 2 | 1 | 1 | 0 | 1 | 0 | 0 | 2 | 0 | 4 | account | purchase |
| **6** | 2812 | 1 | 9320 | 0 | 24 | 2 | 1 | 0 | 1 | 0 | 3 | 0 | 2 | 0 | 2 | account | purchase |
| **7** | 3555 | 1 | 11750 | 0 | 68 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 3 | non account | purchase |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **29495** | 9994393 | 1 | 33680 | 0 | 23 | 2 | 1 | 1 | 0 | 0 | 3 | 3 | 4 | 1 | 4 | account | purchase |
| **29497** | 9995719 | 1 | 5180 | 0 | 29 | 2 | 1 | 0 | 1 | 0 | 3 | 0 | 2 | 0 | 1 | account | not purchase |
| **29499** | 9998377 | 1 | 11630 | 0 | 44 | 2 | 1 | 0 | 1 | 1 | 3 | 2 | 2 | 0 | 4 | account | purchase |
| **29501** | 9999108 | 1 | 34480 | 0 | 74 | 1 | 1 | 0 | 1 | 0 | 3 | 0 | 3 | 0 | 4 | account | purchase |
| **29502** | 9999701 | 1 | 10550 | 0 | 30 | 2 | 1 | 0 | 1 | 1 | 3 | 0 | 1 | 0 | 1 | account | not purchase |

16128 rows × 17 columns

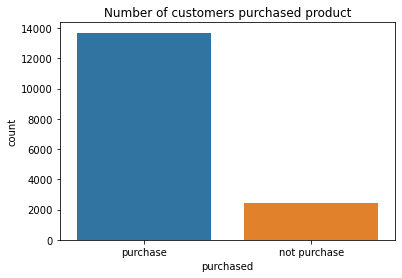
In [52]:

sns**.**countplot(df2['purchased'])

plt**.**title('Number of customers purchased product')

Out[52]:

Text(0.5, 1.0, 'Number of customers purchased product')

****

In [53]:

plt**.**figure(figsize**=**(16,4))

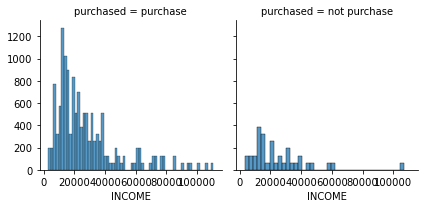
income\_purchased **=** sns**.**FacetGrid(data**=**df2, col**=**'purchased')

income\_purchased**.**map(sns**.**histplot, 'INCOME')

Out[53]:

<seaborn.axisgrid.FacetGrid at 0x225be8c7e88>

<Figure size 1152x288 with 0 Axes>

****

1. **Predictive Model using R Program**

*# Converting some variables into factors*

df$FEMALE<-factor(df$FEMALE, levels = c(0,1), labels=c("Female", "Male"))

df$JOB<-factor(df$JOB, levels=c(0,1,2,3), labels = c("unemployed", "entry", "middle", "managment"))

df$RENT<-as.factor(df$RENT)

df$OWN\_RES<-as.factor(df$OWN\_RES)

df$NEW\_CAR<-as.factor(df$NEW\_CAR)

df$CHK\_ACCT<-factor(df$CHK\_ACCT, levels=c(0,1,2,3), labels = c("No", "< $200", "$200-$2000", ">$2000"))

df$SAV\_ACCT<-as.factor(df$SAV\_ACCT)

df$MOBILE<-as.factor(df$MOBILE)

*# Partioning data into training and validation data*

set.seed(101)

train.index<-sample(c(1:dim(df)[1]), dim(df)[1]\*0.6)

valid.index<-setdiff(c(1:dim(df)[1]), train.index)

df.train<-df[train.index,]

df.valid<-df[valid.index,]

*# variables: variables: income, female, age, num\_dependents, rent, chk\_accts, num\_accts, mobile*

mod1<-glm(ANSWERED ~ INCOME+FEMALE+AGE+NUM\_DEPENDENTS+RENT+CHK\_ACCT+NUM\_ACCTS+MOBILE, data=df.train, family = "binomial")

summary(mod1)

##

## Call:

## glm(formula = ANSWERED ~ INCOME + FEMALE + AGE + NUM\_DEPENDENTS +

## RENT + CHK\_ACCT + NUM\_ACCTS + MOBILE, family = "binomial",

## data = df.train)

##

## Deviance Residuals:

## Min 1Q Median 3Q Max

## -2.0079 -0.9759 0.4531 0.9181 2.2872

##

## Coefficients:

## Estimate Std. Error z value Pr(>|z|)

## (Intercept) -2.535e-01 8.872e-02 -2.857 0.00427 \*\*

## INCOME -2.383e-05 7.150e-07 -33.326 < 2e-16 \*\*\*

## FEMALEMale -5.322e-01 7.502e-02 -7.093 1.31e-12 \*\*\*

## AGE 8.727e-03 1.689e-03 5.166 2.40e-07 \*\*\*

## NUM\_DEPENDENTS -2.243e-01 5.013e-02 -4.474 7.69e-06 \*\*\*

## RENT1 -2.877e-01 4.385e-02 -6.563 5.29e-11 \*\*\*

## CHK\_ACCT< $200 2.641e-01 4.277e-02 6.176 6.59e-10 \*\*\*

## CHK\_ACCT$200-$2000 3.996e-01 7.167e-02 5.575 2.47e-08 \*\*\*

## CHK\_ACCT>$2000 1.866e+00 4.538e-02 41.123 < 2e-16 \*\*\*

## NUM\_ACCTS 1.876e-01 1.534e-02 12.227 < 2e-16 \*\*\*

## MOBILE1 1.722e+00 7.263e-02 23.713 < 2e-16 \*\*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## (Dispersion parameter for binomial family taken to be 1)

##

## Null deviance: 24390 on 17701 degrees of freedom

## Residual deviance: 19939 on 17691 degrees of freedom

## AIC: 19961

##

## Number of Fisher Scoring iterations: 4

mod1.pred<-predict(mod1, df.valid, type="response")

confusionMatrix(factor(ifelse(mod1.pred>0.5,1,0)), factor(df.valid$ANSWERED))

## Confusion Matrix and Statistics

##

## Reference

## Prediction 0 1

## 0 3650 2082

## 1 1689 4381

##

## Accuracy : 0.6805

## 95% CI : (0.672, 0.6889)

## No Information Rate : 0.5476

## P-Value [Acc > NIR] : < 2.2e-16

##

## Kappa : 0.3592

##

## Mcnemar's Test P-Value : 1.731e-10

##

## Sensitivity : 0.6836

## Specificity : 0.6779

## Pos Pred Value : 0.6368

## Neg Pred Value : 0.7217

## Prevalence : 0.4524

## Detection Rate : 0.3093

## Detection Prevalence : 0.4857

## Balanced Accuracy : 0.6808

##

## 'Positive' Class : 0

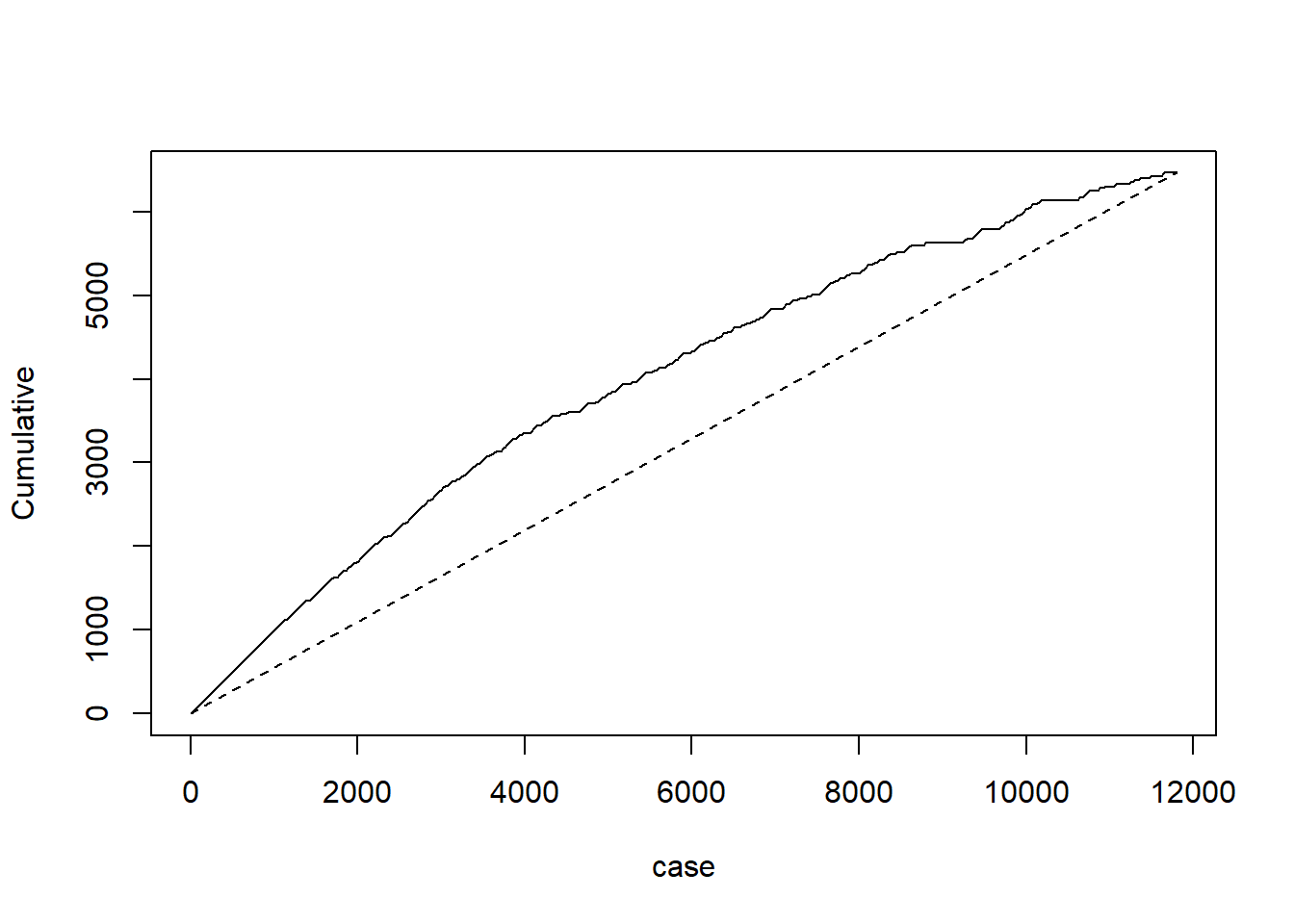
##

# Lift Chart

gain<-gains(df.valid$ANSWERED, mod1.pred, groups=length(mod1.pred))

plot(c(0, gain$cume.pct.of.total\*sum(df.valid$ANSWERED))~c(0, gain$cume.obs), xlab="case", ylab="Cumulative", main=" ", type="l")

lines(c(0, sum(df.valid$ANSWERED))~c(0,dim(df.valid)[1]), lty=2)

****

#Decile Chart

decile.data<-data.frame("actual" = df.valid$ANSWERED, "probability" = mod1.pred)

gain.mod1<-gains(decile.data$actual, decile.data$probability)

midpoints.mod1 <-barplot(gain.mod1$mean.resp/mean(decile.data$actual), names.arg= gain.mod1$depth, ylim = c(0,3),

xlab = "Percentile", ylab = "Mean Answer Phone", main = "Figure: Decile-wise lift chart")

text(midpoints.mod1, gain.mod1$mean.resp/mean(decile.data$actual)+0.5, labels=round(gain.

mod1$mean.resp/mean(decile.data$actual), 1), cex = 1)