2023

CA1

Statistical Techniques for DA

EDUARDO J. MATOS ROMERO

Index

Introduction

The report presented here examines data collected from past marketing campaigns, specifically focusing on whether or not customers subscribed to a term deposit. The primary goal of this data collection by the bank was to gain insights on customer behaviour and evaluate the success of their marketing strategies.

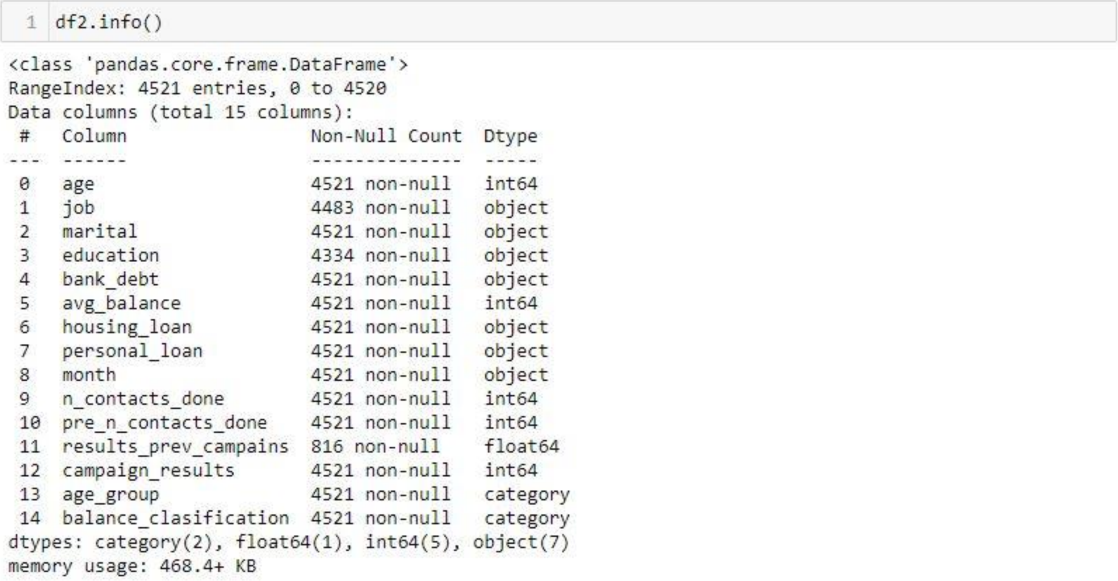
The findings from the conducted analysis have been summarised, and the resulting conclusions drawn.

- **The purpose of this report is to profile the key characteristics of the customer who is more inclined to subscribe to a term deposit based on our analysis.**
    - Assuming we seek customers who are willing to invest long-term without taking unnecessary risks.
- It have been conducted following the next steps:
    - Data overview.
    - Comprehensive analysis of Features
    - The probability of a target customer successfully subscribing to a term deposit.

The data collected to perform this report is the result of questions that have been done to 4521 customers of a bank in order to profile them. It was as of interest their:

- age
- job
- marital: Marital status
- education: Education level achieved.
    - Primary
    - Secondary
    - Tertiary
- bank_debt: Whether or not they have a debt in their bank account.
- avg_balance: Average yearly balance in their bank account.
- housing_loan: Whether or not they have a housing loan.
- personal_loan: Whether or not they have a personal loan.
- month: Month they were contacted last time.
- duration: Last contact duration.
- n_contacts_done: Number of contacts performed during this campaign and for this client.
- pre_n_contacts_done: Number of contacts performed before this campaign and for this client.
- results_prev_campains: outcome of the previous marketing campaign.
    - failure
    - nonexistent
    - success
- campaign_results:: Whether or not the client has subscribed to a term deposit.
- age_group: Groups of customers based on their age.
    - Young: Population from 19 to 34.
    - Adul: Population from 35 to 59.
    - Elderly: Population from 60 to 87.
- balance_clasification: Groups of customers based on their balance.
    - low: Population from -3.333 to 1.421€.

- standard: Population from 1.422 to 4.000€.
- high: Population from 4.001 to 71.188€



```
df2.info()
```

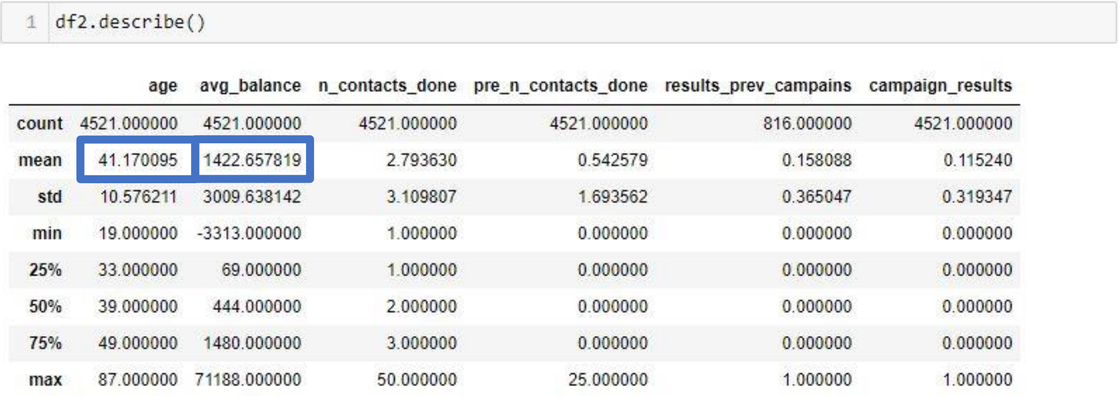
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4521 entries, 0 to 4520
Data columns (total 15 columns):
 #   Column                Non-Null Count  Dtype   
---  ------                --------------  -----   
 0   age                   4521 non-null   int64   
 1   job                   4483 non-null   object  
 2   marital               4521 non-null   object  
 3   education             4334 non-null   object  
 4   bank_debt             4521 non-null   object  
 5   avg_balance           4521 non-null   int64   
 6   housing_loan          4521 non-null   object  
 7   personal_loan         4521 non-null   object  
 8   month                 4521 non-null   object  
 9   n_contacts_done       4521 non-null   int64   
 10  pre_n_contacts_done   4521 non-null   int64   
 11  results_prev_campains 816 non-null    float64 
 12  campaign_results      4521 non-null   int64   
 13  age_group             4521 non-null   category
 14  balance_clasification 4521 non-null   category
dtypes: category(2), float64(1), int64(5), object(7)
memory usage: 468.4+ KB
```

Illustration 1. Columns name and datatypes.

Data Overview

At the beginning of the analysis, an overview of the data was carried out, where the following conclusions were drawn:

1. Data description of numeric features gathered from customers.



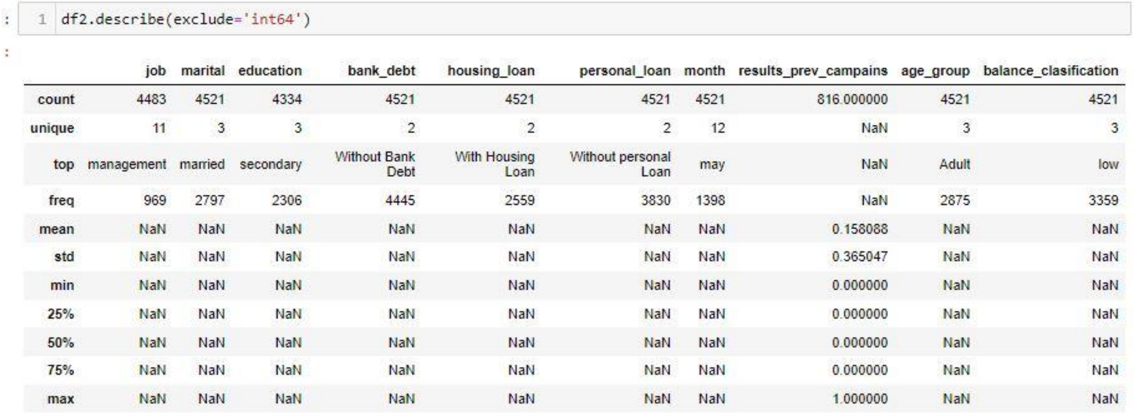
```
df2.describe()
```

	age	avg_balance	n_contacts_done	pre_n_contacts_done	results_prev_campains	campaign_results
count	4521.000000	4521.000000	4521.000000	4521.000000	816.000000	4521.000000
mean	41.170095	1422.657819	2.793630	0.542579	0.158088	0.115240
std	10.576211	3009.638142	3.109807	1.693562	0.365047	0.319347
min	19.000000	-3313.000000	1.000000	0.000000	0.000000	0.000000
25%	33.000000	69.000000	1.000000	0.000000	0.000000	0.000000
50%	39.000000	444.000000	2.000000	0.000000	0.000000	0.000000
75%	49.000000	1480.000000	3.000000	0.000000	0.000000	0.000000
max	87.000000	71188.000000	50.000000	25.000000	1.000000	1.000000

Illustration 2. Descriptive statistics values of numeric features.

- The mean age is 41.
- Mean balance in their bank account 1423€.

2. Data description of categorical features gathered from customers.



```
df2.describe(exclude='int64')
```

	job	marital	education	bank_debt	housing_loan	personal_loan	month	results_prev_campains	age_group	balance_clasification
count	4483	4521	4334	4521	4521	4521	4521	816.000000	4521	4521
unique	11	3	3	2	2	2	12	NaN	3	3
top	management	married	secondary	Without Bank Debt	With Housing Loan	Without personal Loan	may	NaN	Adult	low
freq	969	2797	2306	4445	2559	3830	1398	NaN	2875	3359
mean	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.158088	NaN	NaN
std	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.365047	NaN	NaN
min	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.000000	NaN	NaN
25%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.000000	NaN	NaN
50%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.000000	NaN	NaN
75%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.000000	NaN	NaN
max	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1.000000	NaN	NaN

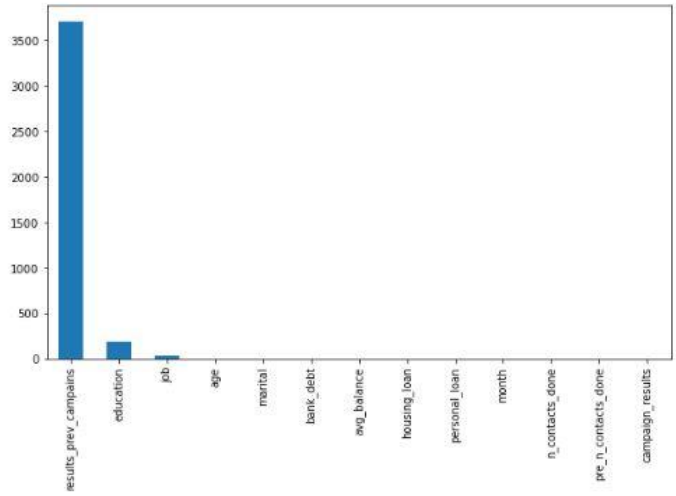
Illustration 3. Descriptive statistics values of categorical features.

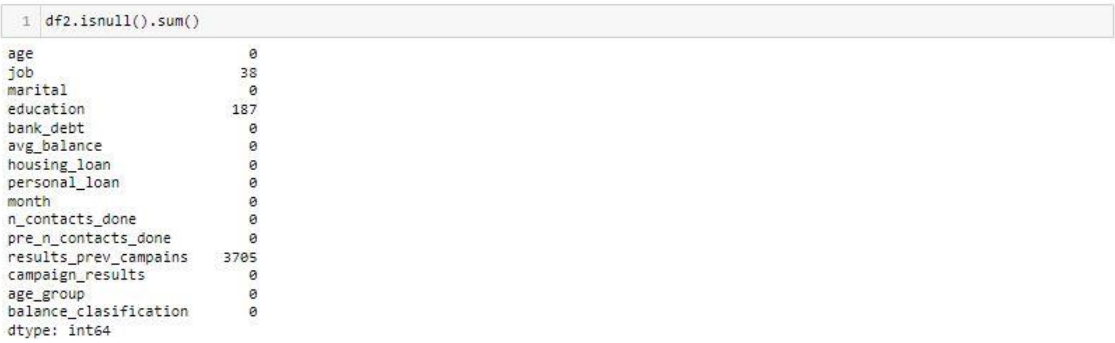
- ○ Working in management
- ○ Married
- ○ With a secondary level of education.
- ○ Without a bank debt.
- ○ With a housing loan
- ○ Without a personal loan
- ○ Adult
- ○ With a low average balance in their bank account.

3. Data collection

```
df2.isnull().sum().nlargest(20).plot(kind='bar', figsize=(10, 6))
```

```
<AxesSubplot:>
```





```
df2.isnull().sum()
```

```
age                        0
job                       38
marital                    0
education                187
bank_debt                  0
avg_balance                0
housing_loan               0
personal_loan              0
month                      0
n_contacts_done            0
pre_n_contacts_done        0
results_prev_campains   3705
campaign_results           0
age_group                  0
balance_clasification      0
dtype: int64
```

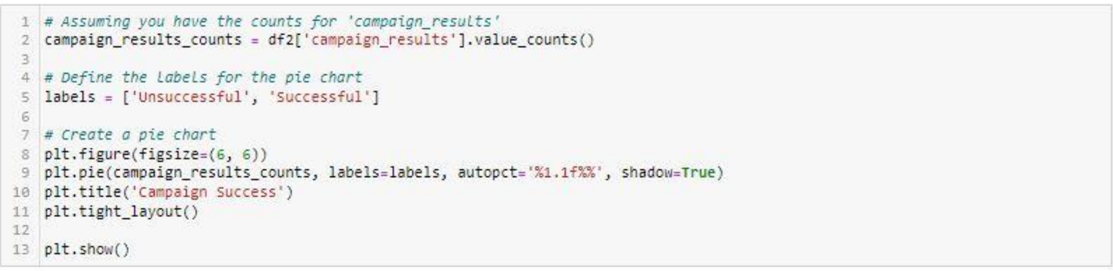
Illustration 4. Missing values.

- We may decide not to include information about the results of previous campaigns in further analysis because the number of customers from whom we have collected information is too small.
    - From the 4521 customers contacted, we were unable to collect information from 3705 of them.

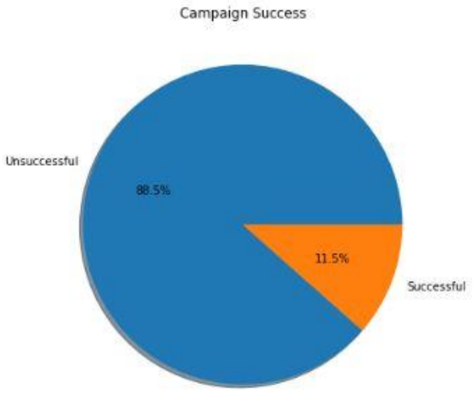
Comprehensive Analysis of Features

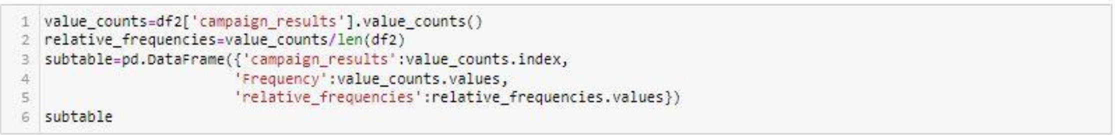
Once it was discovered what the data is that we are working on, it was decided to carry out an analysis of the characteristics that we could consider relevant to draw an approach of what could be an interesting profile of customers that we could consider developing further strategies to reach out.

Campaign Success

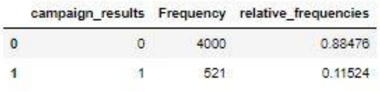


```
# Assuming you have the counts for 'campaign_results'
campaign_results_counts = df2['campaign_results'].value_counts()

# Define the labels for the pie chart
labels = ['Unsuccessful', 'Successful']

# Create a pie chart
plt.figure(figsize=(6, 6))
plt.pie(campaign_results_counts, labels=labels, autopct='%1.1f%%', shadow=True)
plt.title('Campaign Success')
plt.tight_layout()

plt.show()
```





```
value_counts=df2['campaign_results'].value_counts()
relative_frequencies=value_counts/len(df2)
subtable=pd.DataFrame({'campaign_results':value_counts.index,
                       'Frequency':value_counts.values,
                       'relative_frequencies':relative_frequencies.values})
subtable
```



	campaign_results	Frequency	relative_frequencies
0	0	4000	0.88476
1	1	521	0.11524

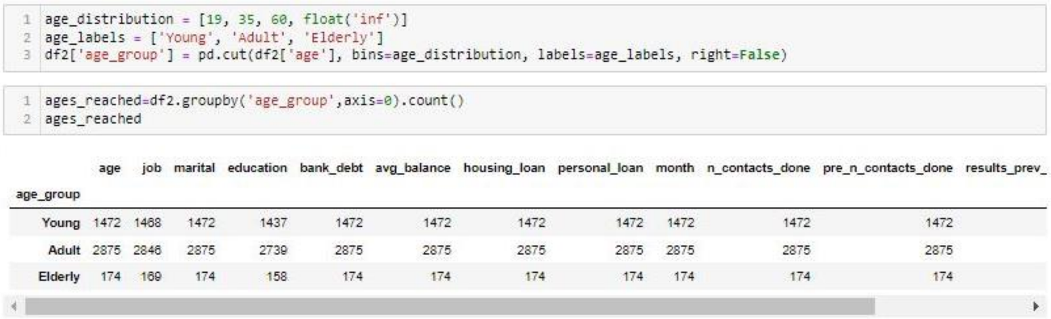
Illustration 5. Campaign results.

The previous campaign results we are working with are:

- 11.5% of customers who were contacted were successfully subscribed to a term deposit.
- 89.5% of customers who were contacted were unsuccessfully subscribed to a term deposit.

Campaign Results by Group of Ages

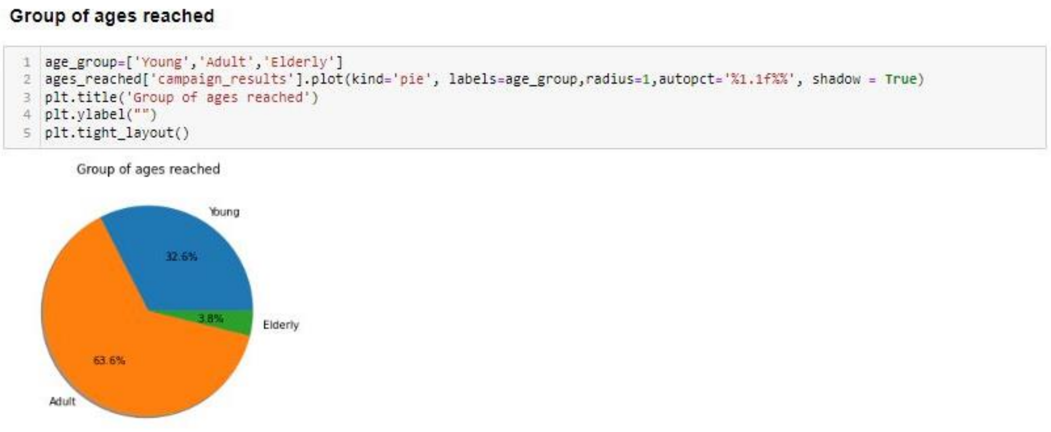
To conduct a study of the success of previous campaigns by age group, we decided to group customers aged 19-34 as young, 35-59 as adults and 60-87 as older.



```
age_distribution = [19, 35, 60, float('inf')]
age_labels = ['Young', 'Adult', 'Elderly']
df2['age_group'] = pd.cut(df2['age'], bins=age_distribution, labels=age_labels, right=False)
```

```
ages_reached=df2.groupby('age_group',axis=0).count()
ages_reached
```

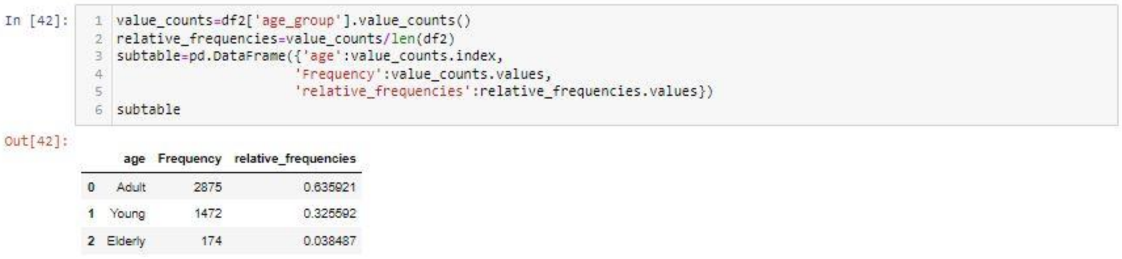
	age	job	marital	education	bank_debt	avg_balance	housing_loan	personal_loan	month	n_contacts_done	pre_n_contacts_done	results_prev_
age_group												
Young	1472	1468	1472	1437	1472	1472	1472	1472	1472	1472	1472	
Adult	2875	2846	2875	2739	2875	2875	2875	2875	2875	2875	2875	
Elderly	174	169	174	158	174	174	174	174	174	174	174	



Group of ages reached

```
age_group=['Young','Adult','Elderly']
ages_reached['campaign_results'].plot(kind='pie', labels=age_group,radius=1,autopct='%1.1f%%', shadow = True)
plt.title('Group of ages reached')
plt.ylabel("")
plt.tight_layout()
```

Illustration 6. Customers contacted by group of ages.



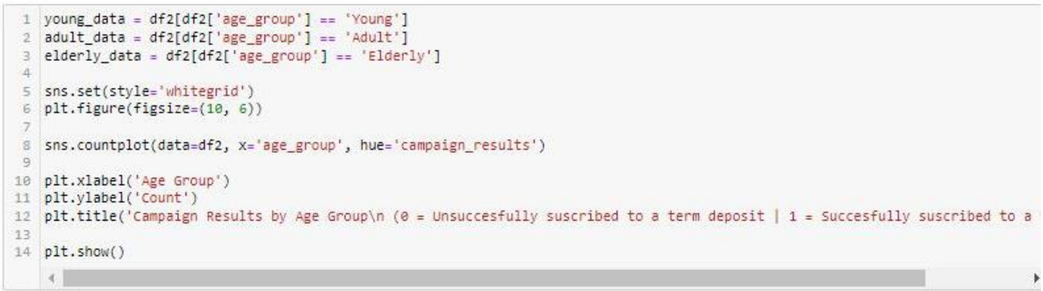
```
In [42]: value_counts=df2['age_group'].value_counts()
         relative_frequencies=value_counts/len(df2)
         subtable=pd.DataFrame({'age':value_counts.index,
                                'Frequency':value_counts.values,
                                'relative_frequencies':relative_frequencies.values})
         subtable
Out[42]:
```

	age	Frequency	relative_frequencies
0	Adult	2875	0.635921
1	Young	1472	0.325592
2	Elderly	174	0.038487

Illustration 7. Relative frequencies of customers contacted by group of ages.

In previous campaigns:

- 63.5% of customers contacted were adults.
- 32.5% of customers contacted were young.
- 4% of customers contacted were elderly.



```
young_data = df2[df2['age_group'] == 'Young']
adult_data = df2[df2['age_group'] == 'Adult']
elderly_data = df2[df2['age_group'] == 'Elderly']

sns.set(style='whitegrid')
plt.figure(figsize=(10, 6))

sns.countplot(data=df2, x='age_group', hue='campaign_results')

plt.xlabel('Age Group')
plt.ylabel('Count')
plt.title('Campaign Results by Age Group\n (0 = Unsuccesfully suscribed to a term deposit | 1 = Succesfully suscribed to a

plt.show()
```

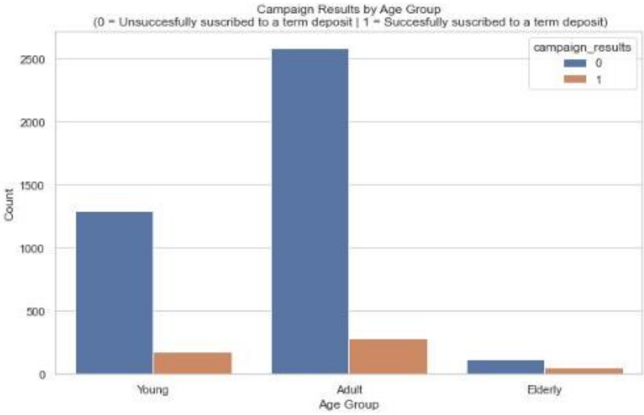
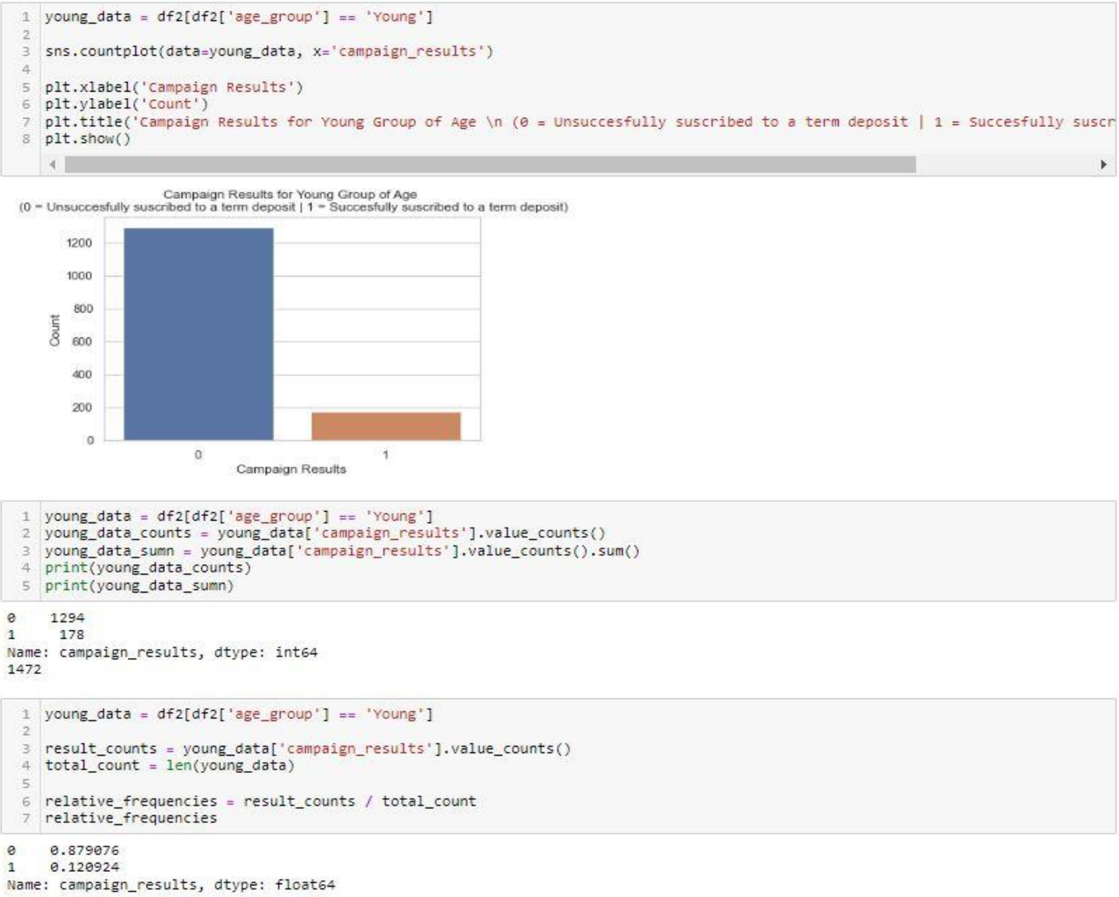


Illustration 8. Campaign results by age group

We have discovered that even if most of customers contacted were adults, it is actually **young customers** who seems to be **proportionately more likely to subscribe to a term deposit.**



```
young_data = df2[df2['age_group'] == 'Young']

sns.countplot(data=young_data, x='campaign_results')

plt.xlabel('Campaign Results')
plt.ylabel('Count')
plt.title('Campaign Results for Young Group of Age \n (0 = Unsuccesfully suscribed to a term deposit | 1 = Succesfully suscr
plt.show()
```

Campaign Results for Young Group of Age
(0 = Unsuccesfully suscribed to a term deposit | 1 = Succesfully suscribed to a term deposit)
1200
1000
800
600
400
200
0
Count
0
1
Campaign Results

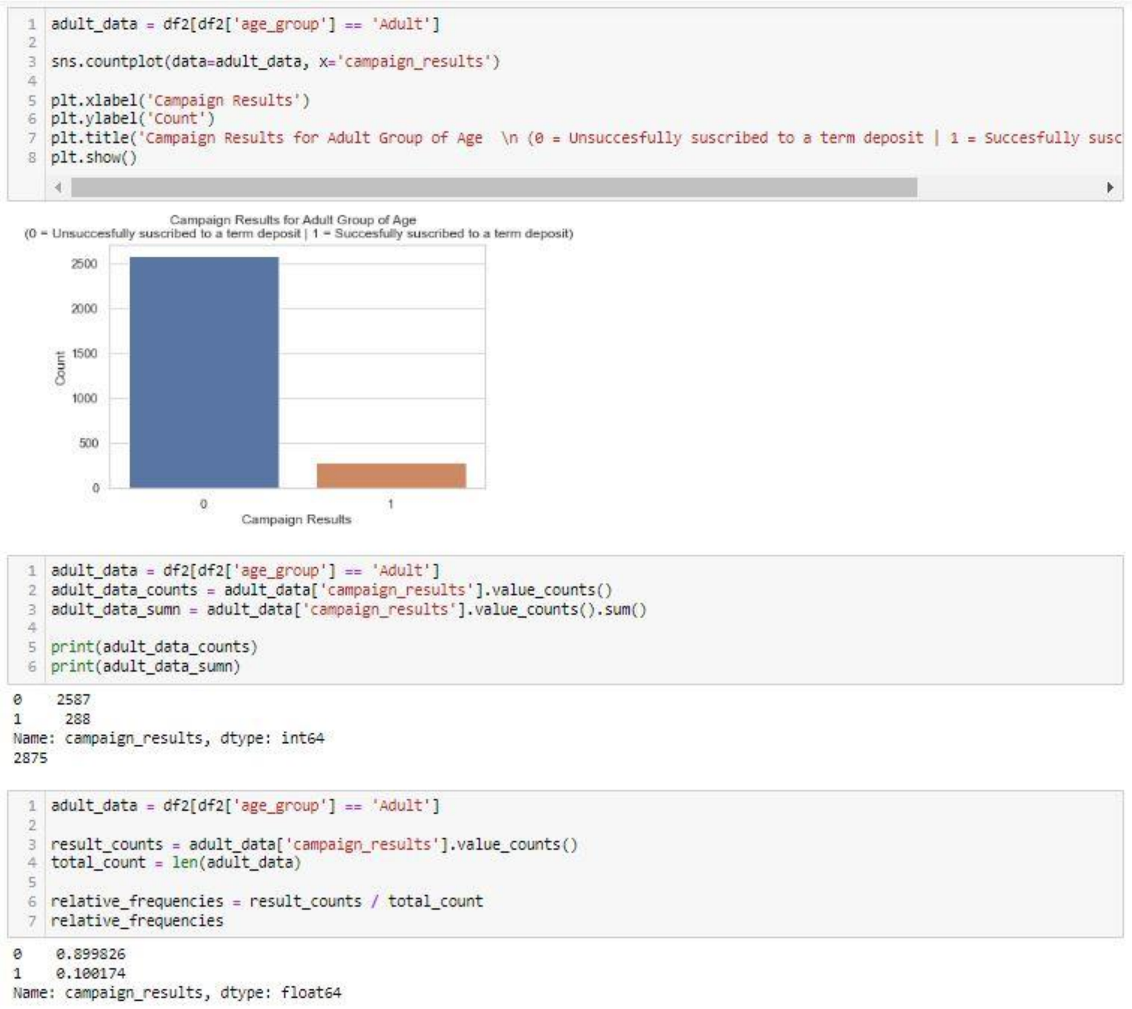
```
young_data = df2[df2['age_group'] == 'Young']
young_data_counts = young_data['campaign_results'].value_counts()
young_data_sumn = young_data['campaign_results'].value_counts().sum()
print(young_data_counts)
print(young_data_sumn)
```

```
0    1294
1     178
Name: campaign_results, dtype: int64
1472
```

```
young_data = df2[df2['age_group'] == 'Young']

result_counts = young_data['campaign_results'].value_counts()
total_count = len(young_data)

relative_frequencies = result_counts / total_count
relative_frequencies
```

```
0    0.879076
1    0.120924
Name: campaign_results, dtype: float64
```

Illustration 9. Campaign results by young group



```
adult_data = df2[df2['age_group'] == 'Adult']

sns.countplot(data=adult_data, x='campaign_results')

plt.xlabel('Campaign Results')
plt.ylabel('Count')
plt.title('Campaign Results for Adult Group of Age  \n (0 = Unsuccesfully suscribed to a term deposit | 1 = Succesfully susc
plt.show()
```

Campaign Results for Adult Group of Age
(0 = Unsuccesfully suscribed to a term deposit | 1 = Succesfully suscribed to a term deposit)
2500
2000
1500
1000
500
0
Count
0
1
Campaign Results

```
adult_data = df2[df2['age_group'] == 'Adult']
adult_data_counts = adult_data['campaign_results'].value_counts()
adult_data_sumn = adult_data['campaign_results'].value_counts().sum()

print(adult_data_counts)
print(adult_data_sumn)
```

```
0    2587
1     288
Name: campaign_results, dtype: int64
2875
```

```
adult_data = df2[df2['age_group'] == 'Adult']

result_counts = adult_data['campaign_results'].value_counts()
total_count = len(adult_data)

relative_frequencies = result_counts / total_count
relative_frequencies
```

```
0    0.899826
1    0.100174
Name: campaign_results, dtype: float64
```

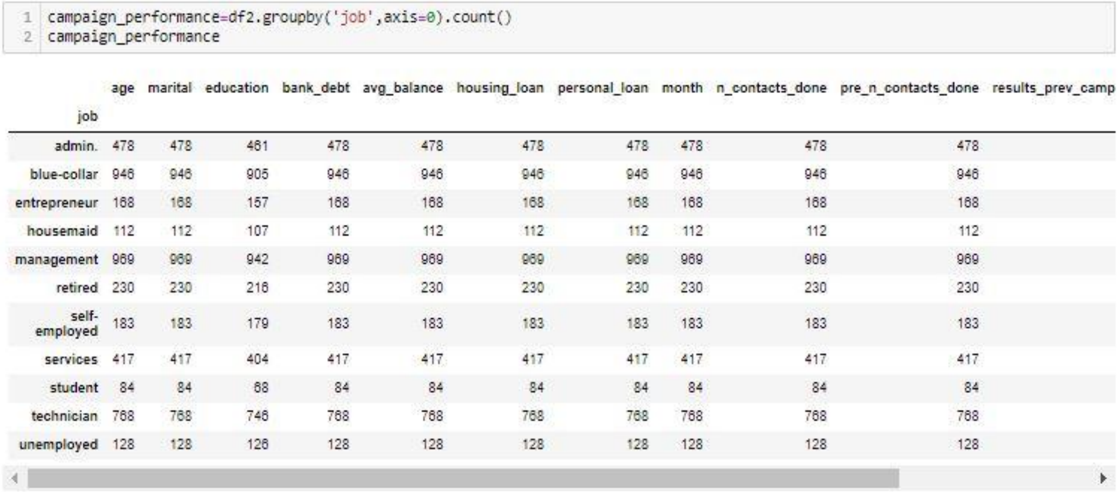
Illustration 10. Campaign results by adult group

Conducting a more thorough analysis of young and adult clients, it was observed that out of 1472 young customers contacted, **12% were successfully subscribed to a term deposit**. Which is proportionally a higher proportion of customers if we compare with **adults** where it was the **10%** of them who did it.

However, adult customers are considered more suitable for term deposit offers, as young customers are perceived as vulnerable. This is why we consider this group of people to be best suited to better contact and persuasion strategies.

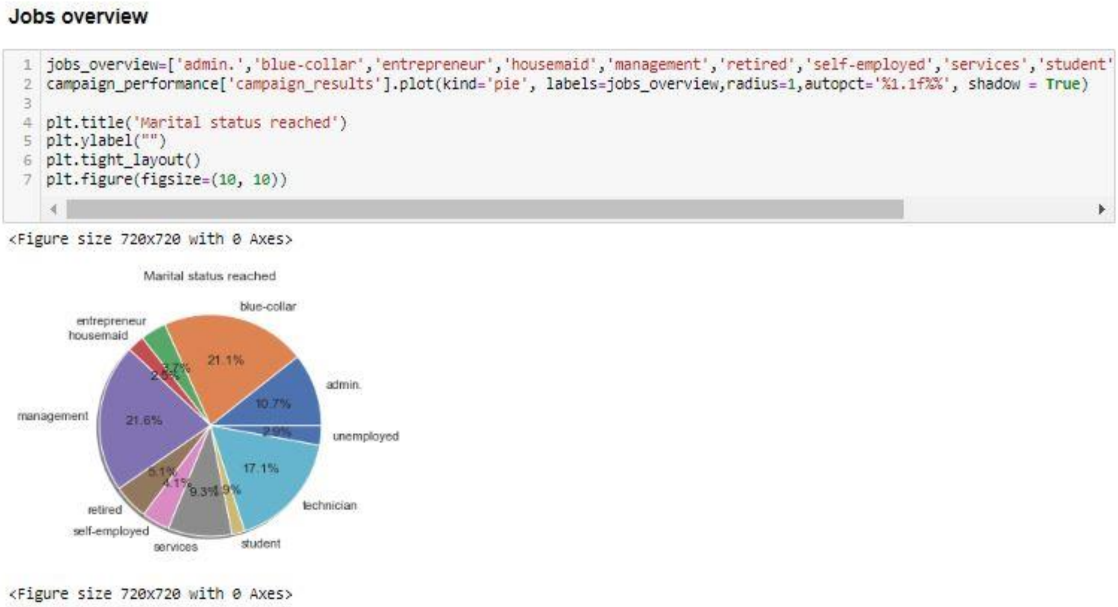
- It is worth noting that, due to company strategies, further analysis of elder customers was not conducted in this study, although they may be potential clients. However, it is outside the scope of this analysis to consider them.

Campaign Results by Job



```
campaign_performance=df2.groupby('job',axis=0).count()
campaign_performance
```

job	age	marital	education	bank_debt	avg_balance	housing_loan	personal_loan	month	n_contacts_done	pre_n_contacts_done	results_prev_camp
admin.	478	478	461	478	478	478	478	478	478	478	
blue-collar	946	946	905	946	946	946	946	946	946	946	
entrepreneur	168	168	157	168	168	168	168	168	168	168	
housemaid	112	112	107	112	112	112	112	112	112	112	
management	969	969	942	969	969	969	969	969	969	969	
retired	230	230	216	230	230	230	230	230	230	230	
self-employed	183	183	179	183	183	183	183	183	183	183	
services	417	417	404	417	417	417	417	417	417	417	
student	84	84	68	84	84	84	84	84	84	84	
technician	768	768	746	768	768	768	768	768	768	768	
unemployed	128	128	126	128	128	128	128	128	128	128	



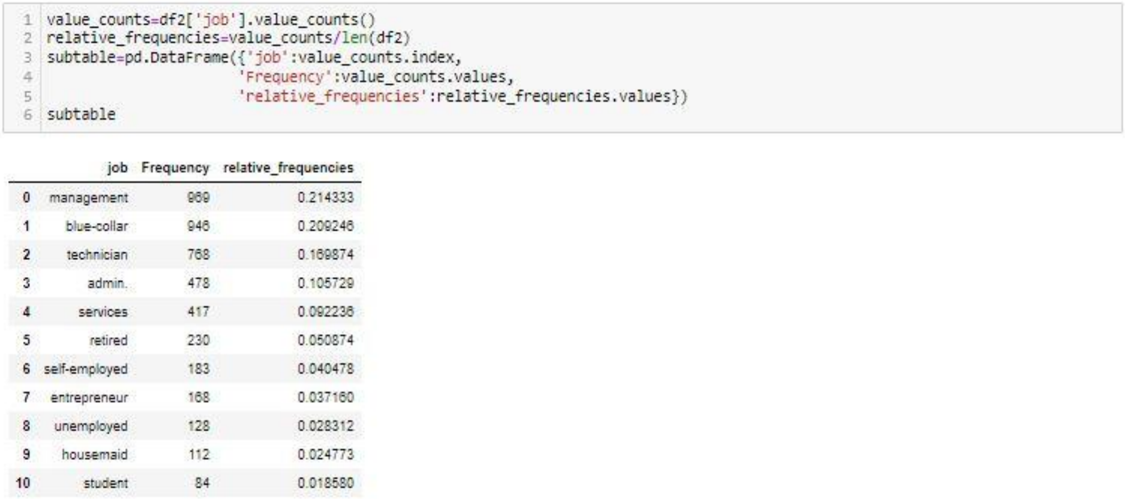
Jobs overview

```
jobs_overview=['admin.','blue-collar','entrepreneur','housemaid','management','retired','self-employed','services','student'
campaign_performance['campaign_results'].plot(kind='pie', labels=jobs_overview,radius=1,autopct='%1.1f%%', shadow = True)

plt.title('Marital status reached')
plt.ylabel("")
plt.tight_layout()
plt.figure(figsize=(10, 10))
```

```
<Figure size 720x720 with 0 Axes>
```

```
<Figure size 720x720 with 0 Axes>
```

Illustration 11. Customers contacted by job.



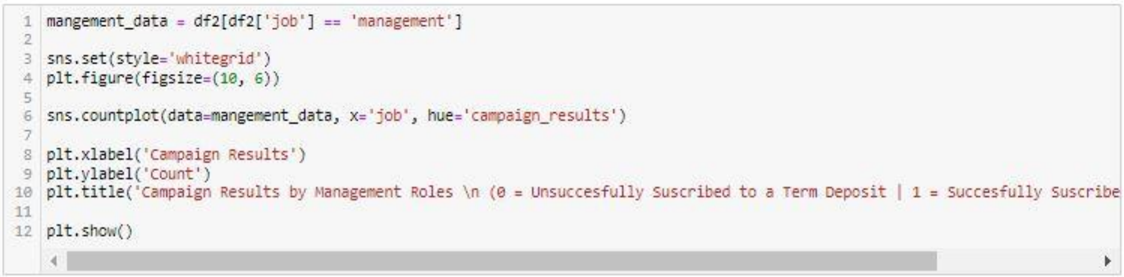
```
value_counts=df2['job'].value_counts()
relative_frequencies=value_counts/len(df2)
subtable=pd.DataFrame({'job':value_counts.index,
                       'Frequency':value_counts.values,
                       'relative_frequencies':relative_frequencies.values})
subtable
```

	job	Frequency	relative_frequencies
0	management	969	0.214333
1	blue-collar	946	0.209246
2	technician	768	0.169874
3	admin.	478	0.105729
4	services	417	0.092236
5	retired	230	0.050874
6	self-employed	183	0.040478
7	entrepreneur	168	0.037160
8	unemployed	128	0.028312
9	housemaid	112	0.024773
10	student	84	0.018580

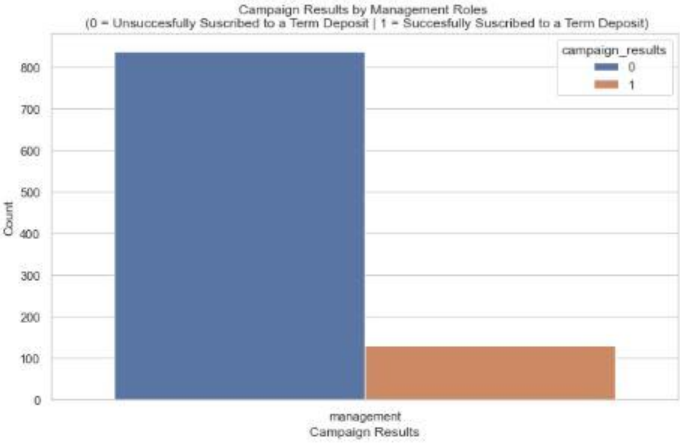
Illustration 12. Relative frequencies of customers contacted by their job.

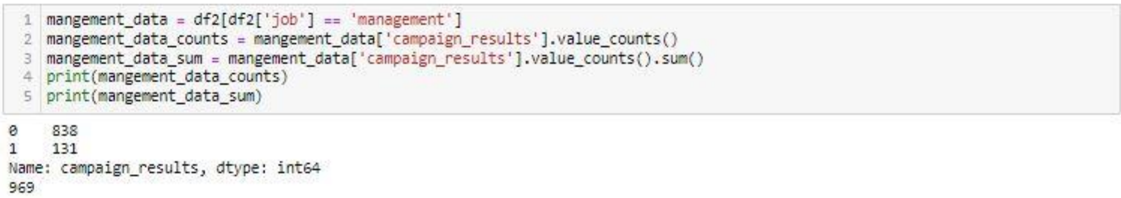
In previous campaigns:

- 21.4% of customers contacted were managers.
- 20.9% of customers contacted were single.



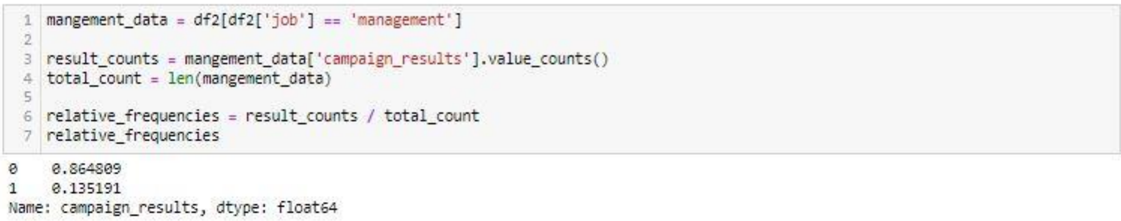
```
mangement_data = df2[df2['job'] == 'management']

sns.set(style='whitegrid')
plt.figure(figsize=(10, 6))

sns.countplot(data=mangement_data, x='job', hue='campaign_results')

plt.xlabel('Campaign Results')
plt.ylabel('Count')
plt.title('Campaign Results by Management Roles \n (0 = Unsuccesfully Suscribed to a Term Deposit | 1 = Succesfully Suscribe

plt.show()
```





```
mangement_data = df2[df2['job'] == 'management']
mangement_data_counts = mangement_data['campaign_results'].value_counts()
mangement_data_sum = mangement_data['campaign_results'].value_counts().sum()
print(mangement_data_counts)
print(mangement_data_sum)
```

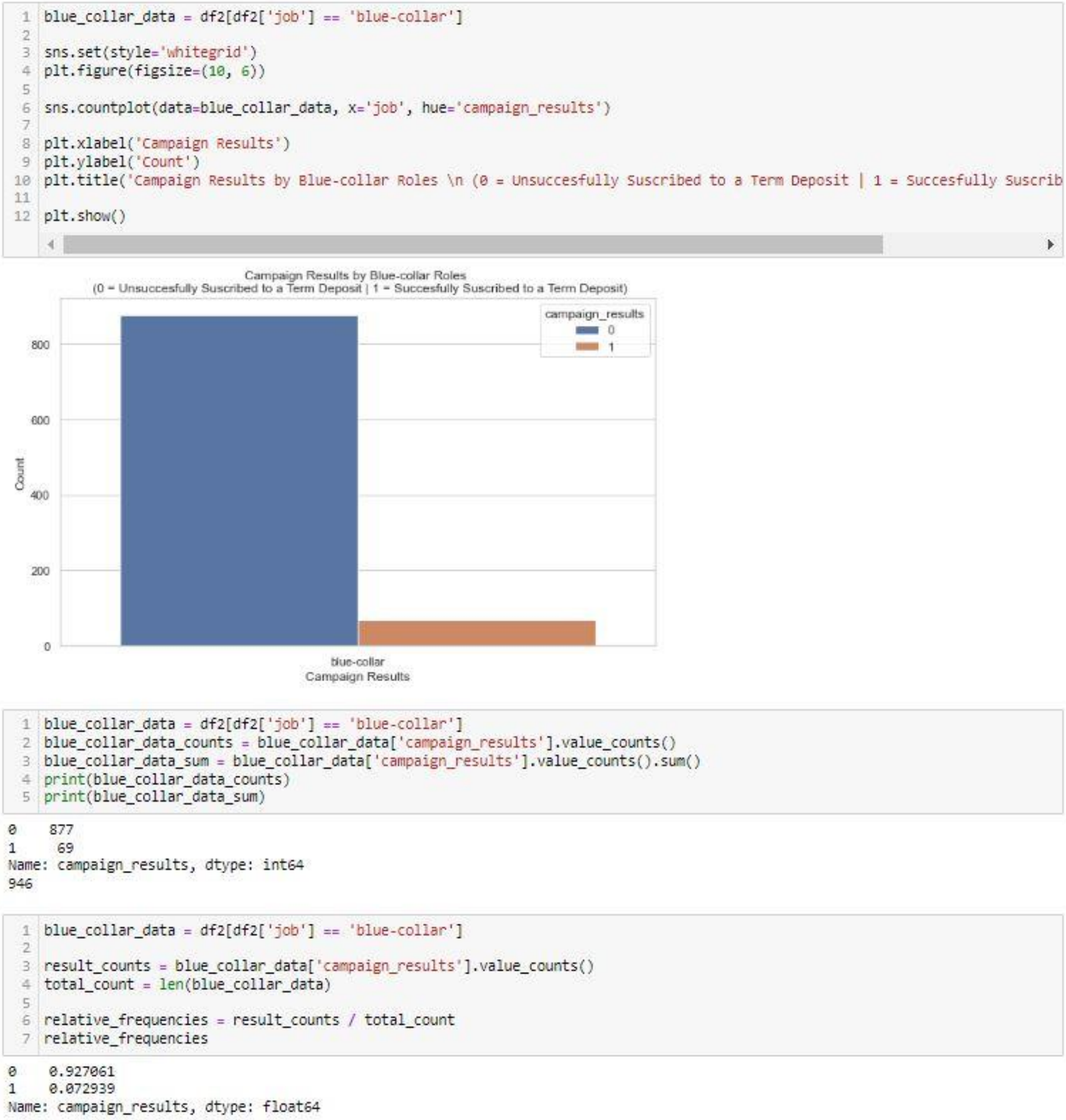
```
0    838
1    131
Name: campaign_results, dtype: int64
969
```



```
mangement_data = df2[df2['job'] == 'management']

result_counts = mangement_data['campaign_results'].value_counts()
total_count = len(mangement_data)

relative_frequencies = result_counts / total_count
relative_frequencies
```

```
0    0.864809
1    0.135191
Name: campaign_results, dtype: float64
```

Illustration 13. Campaign results by management group



```
blue_collar_data = df2[df2['job'] == 'blue-collar']

sns.set(style='whitegrid')
plt.figure(figsize=(10, 6))

sns.countplot(data=blue_collar_data, x='job', hue='campaign_results')

plt.xlabel('Campaign Results')
plt.ylabel('Count')
plt.title('Campaign Results by Blue-collar Roles \n (0 = Unsuccesfully Suscribed to a Term Deposit | 1 = Succesfully Suscrib

plt.show()
```

Campaign Results by Blue-collar Roles
(0 = Unsuccesfully Suscribed to a Term Deposit | 1 = Succesfully Suscribed to a Term Deposit)
campaign_results
0
1
800
600
400
200
0
Count
blue-collar
Campaign Results

```
blue_collar_data = df2[df2['job'] == 'blue-collar']
blue_collar_data_counts = blue_collar_data['campaign_results'].value_counts()
blue_collar_data_sum = blue_collar_data['campaign_results'].value_counts().sum()
print(blue_collar_data_counts)
print(blue_collar_data_sum)
```

```
0    877
1     69
Name: campaign_results, dtype: int64
946
```

```
blue_collar_data = df2[df2['job'] == 'blue-collar']

result_counts = blue_collar_data['campaign_results'].value_counts()
total_count = len(blue_collar_data)

relative_frequencies = result_counts / total_count
relative_frequencies
```

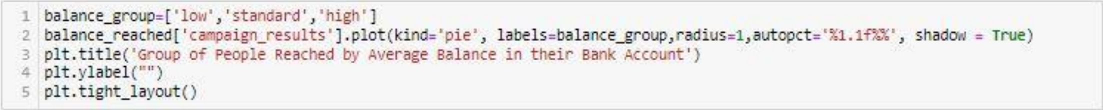
```
0    0.927061
1    0.072939
Name: campaign_results, dtype: float64
```

Illustration 14. Campaign results by blue-collar group

A more detailed analysis of customers working as managers and blue-collar shows that 131 out of 969 **managers contacted**, or **13.5%**, **successfully subscribed to a term deposit**. This is a proportionally higher percentage of customers compared to **blue-collar customers**, where it was **7%**. Therefore, it is recommended to increase efforts to reach customers with management roles.

Campaign Results by Average Balance in the Bank

In order to study how successful was the campaign by the average balance in the bank account, we have decided grouping customers by their average balance and where low is population from -3333 to 1421€, standard from 1422 to 4000€ and high from 4001 to 71188€.



```
balance_group=['low','standard','high']
balance_reached['campaign_results'].plot(kind='pie', labels=balance_group,radius=1,autopct='%1.1f%%', shadow = True)
plt.title('Group of People Reached by Average Balance in their Bank Account')
plt.ylabel("")
plt.tight_layout()
```

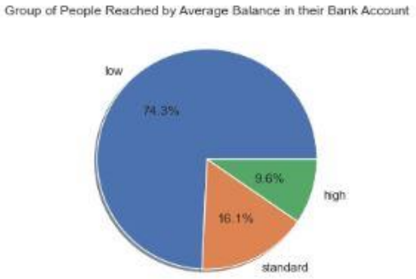
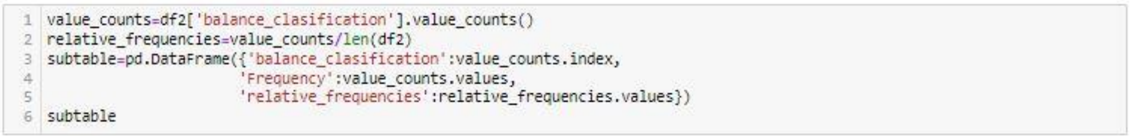
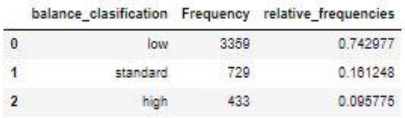


Illustration 15. Customers contacted by Average Balance in their Bank Account.



```
value_counts=df2['balance_clasification'].value_counts()
relative_frequencies=value_counts/len(df2)
subtable=pd.DataFrame({'balance_clasification':value_counts.index,
                       'Frequency':value_counts.values,
                       'relative_frequencies':relative_frequencies.values})
subtable
```

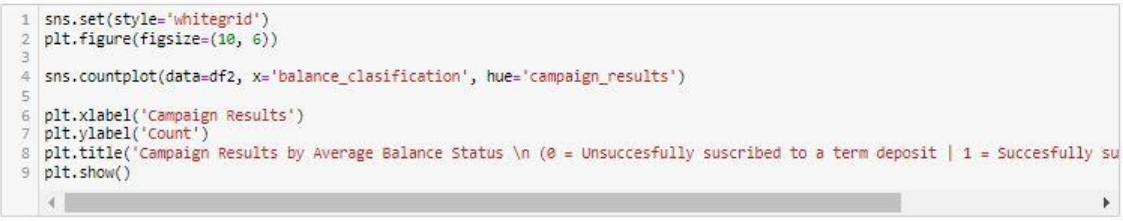


	balance_clasification	Frequency	relative_frequencies
0	low	3359	0.742977
1	standard	729	0.161248
2	high	433	0.095775

Illustration 16. Relative frequencies of customers contacted by Average Balance in their Bank Account.

In previous campaigns:

- 74.2% of customers contacted had a low average balance in their bank account.
- 16.1% of customers contacted had a standard average balance in their bank account.
- 9.7% of customers contacted had a high average balance in their bank account.



```
sns.set(style='whitegrid')
plt.figure(figsize=(10, 6))

sns.countplot(data=df2, x='balance_clasification', hue='campaign_results')

plt.xlabel('Campaign Results')
plt.ylabel('Count')
plt.title('Campaign Results by Average Balance Status \n (0 = Unsuccesfully suscribed to a term deposit | 1 = Succesfully su
plt.show()
```

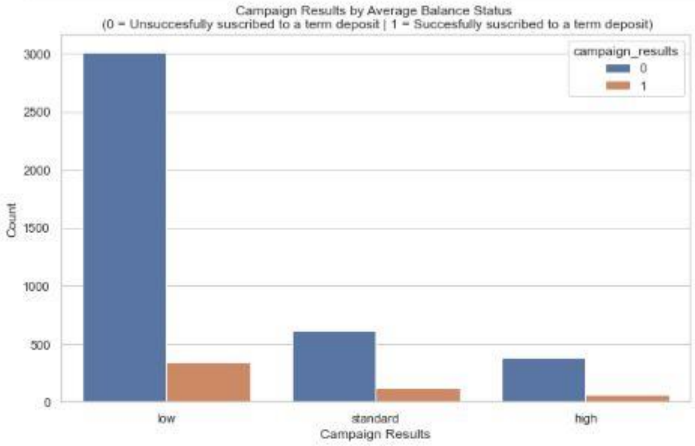
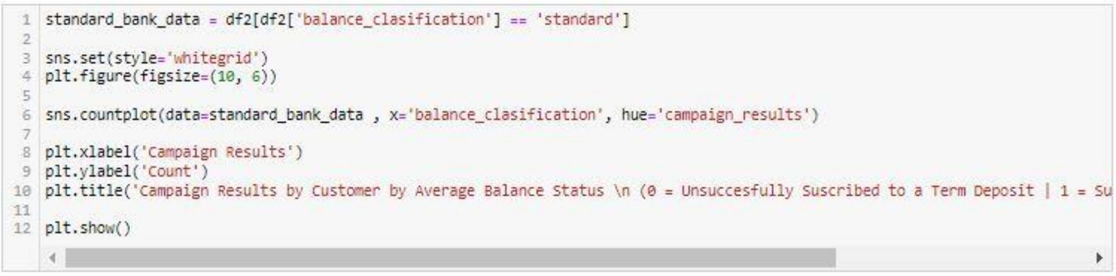


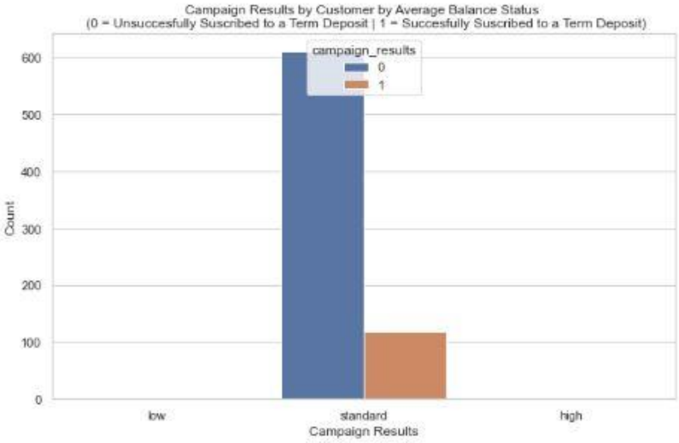
Illustration 17. Campaign results by Average Balance in their Bank Account.

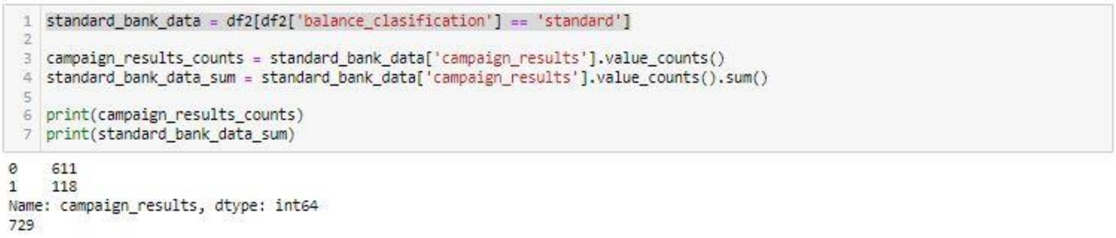
Although it seems obvious that most customers have a low average balance, it may be recommended to make a better effort to reach people with a standard and high average balance in order to take a safer position.

- As it may be obvious that people with a high average balance in their bank account are a target for further campaigns, we may not consider them for further analysis.



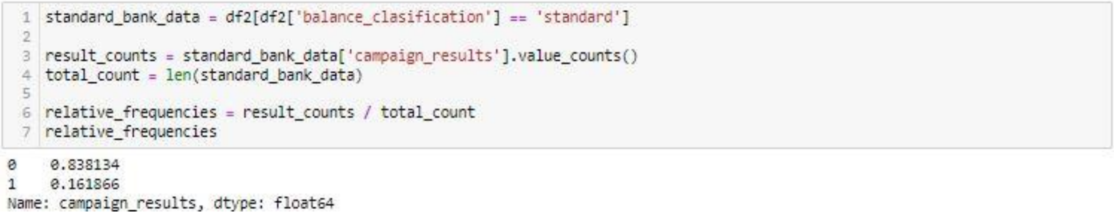
```
standard_bank_data = df2[df2['balance_clasification'] == 'standard']

sns.set(style='whitegrid')
plt.figure(figsize=(10, 6))

sns.countplot(data=standard_bank_data , x='balance_clasification', hue='campaign_results')

plt.xlabel('Campaign Results')
plt.ylabel('Count')
plt.title('Campaign Results by Customer by Average Balance Status \n (0 = Unsuccesfully Suscribed to a Term Deposit | 1 = Su

plt.show()
```





```
standard_bank_data = df2[df2['balance_clasification'] == 'standard']

campaign_results_counts = standard_bank_data['campaign_results'].value_counts()
standard_bank_data_sum = standard_bank_data['campaign_results'].value_counts().sum()

print(campaign_results_counts)
print(standard_bank_data_sum)
```

```
0    611
1    118
Name: campaign_results, dtype: int64
729
```



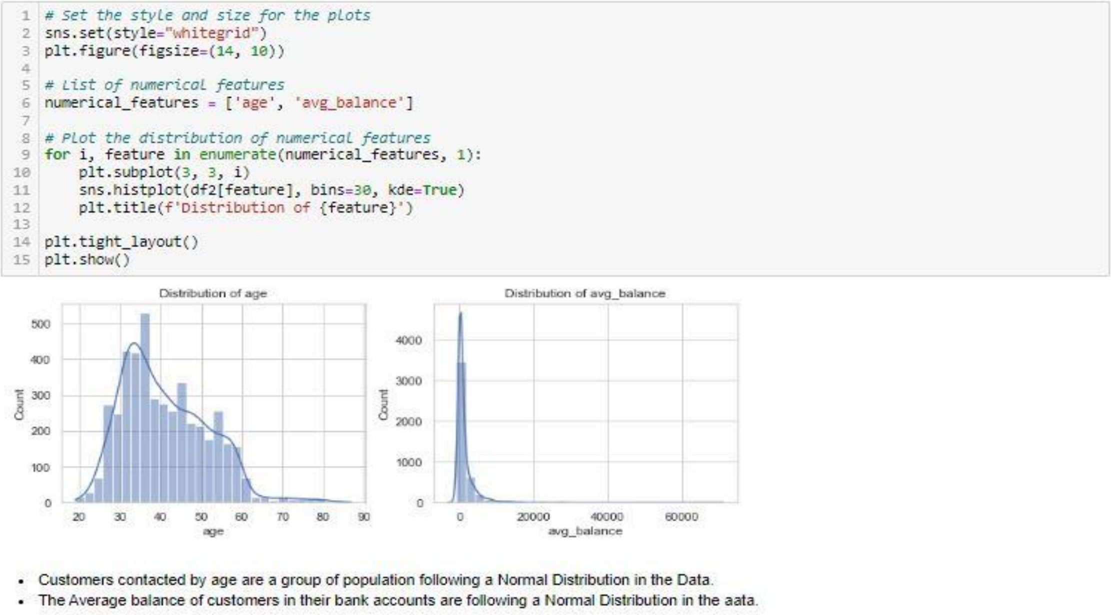
```
standard_bank_data = df2[df2['balance_clasification'] == 'standard']

result_counts = standard_bank_data['campaign_results'].value_counts()
total_count = len(standard_bank_data)

relative_frequencies = result_counts / total_count
relative_frequencies
```

```
0    0.838134
1    0.161866
Name: campaign_results, dtype: float64
```

Illustration 18. Campaign results by customers with a standard Balance in their Bank Account.

Conducting a more thorough analysis customers with a standard Balance in their Bank Account, it was observed that 118 out of 729, **16% were successfully subscribed to a term deposit**.

Summary Analysis

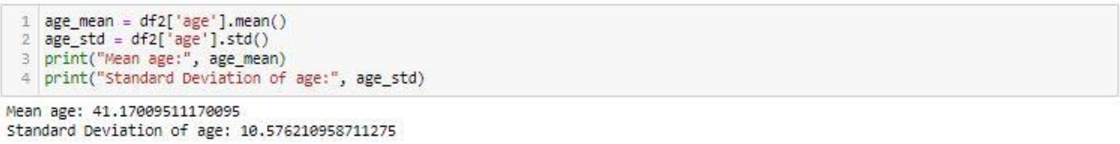


```python
# Set the style and size for the plots
sns.set(style="whitegrid")
plt.figure(figsize=(14, 10))

# List of numerical features
numerical_features = ['age', 'avg_balance']

# Plot the distribution of numerical features
for i, feature in enumerate(numerical_features, 1):
    plt.subplot(3, 3, i)
    sns.histplot(df2[feature], bins=30, kde=True)
    plt.title(f'Distribution of {feature}')

plt.tight_layout()
plt.show()
```

- Customers contacted by age are a group of population following a Normal Distribution in the Data.
- The Average balance of customers in their bank accounts are following a Normal Distribution in the aata.

Illustration 19. Distribution of age and average balance.

- **Age and average balance** of customers in their bank accounts are following a **Normal Distribution** in the data.

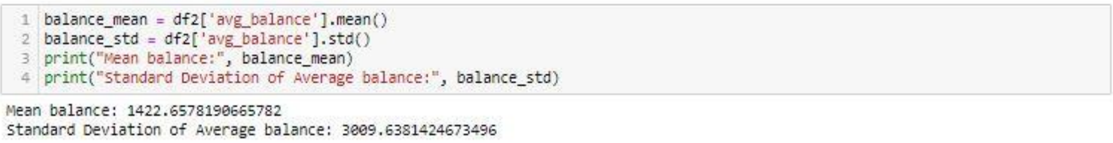


```python
age_mean = df2['age'].mean()
age_std = df2['age'].std()
print("Mean age:", age_mean)
print("Standard Deviation of age:", age_std)
```

```
Mean age: 41.17009511170095
Standard Deviation of age: 10.576210958711275
```

Illustration 20. Mean and standard deviation of customers age.

- **The average age of customers reached during the campaign is 41**, with the majority of pupation analysed within +/- 10 years of this age.

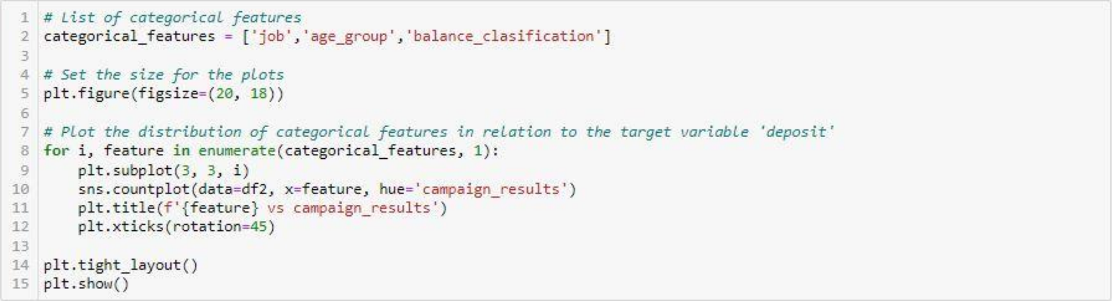


```python
balance_mean = df2['avg_balance'].mean()
balance_std = df2['avg_balance'].std()
print("Mean balance:", balance_mean)
print("Standard Deviation of Average balance:", balance_std)
```

```
Mean balance: 1422.6578190665782
Standard Deviation of Average balance: 3009.6381424673496
```

Illustration 21. Mean and standard deviation of the customers average balance.

The balance average of customers reached during the campaign is 1422€, with the majority of pupation analysed within +/- 3000€ of this amount.



```python
# List of categorical features
categorical_features = ['job','age_group','balance_clasification']

# Set the size for the plots
plt.figure(figsize=(20, 18))

# Plot the distribution of categorical features in relation to the target variable 'deposit'
for i, feature in enumerate(categorical_features, 1):
    plt.subplot(3, 3, i)
    sns.countplot(data=df2, x=feature, hue='campaign_results')
    plt.title(f'{feature} vs campaign_results')
    plt.xticks(rotation=45)

plt.tight_layout()
plt.show()
```

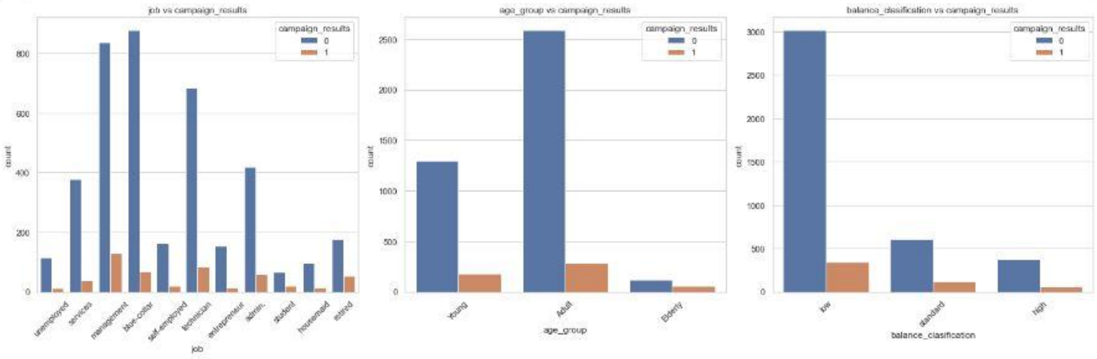


Illustration 22. Distribution of age and average balance.

- People in management roles are more likely to subscribe to a term deposit followed by technicians.
    - 131 out of 969 **managers** contacted, or **13.5%, successfully subscribed** to a term deposit.
- Adult customers are deemed more suitable for term deposit offers since young customers are perceived as precarious prospects, and the elderly may not be inclined to show interest for commercial strategic reasons.
    - 288 out of 2875 **adult** customers contacted, the **10% were successfully subscribed** to a term deposit.
- Although we could adopt a more conservative approach and target customers with a high average balance, we have decided to concentrate on clients with a regular balance who are more statistically inclined to apply for a term deposit.
    - 118 out of 729 **customers** with a **standard** average balance, **16% were successfully subscribed** to a term deposit.

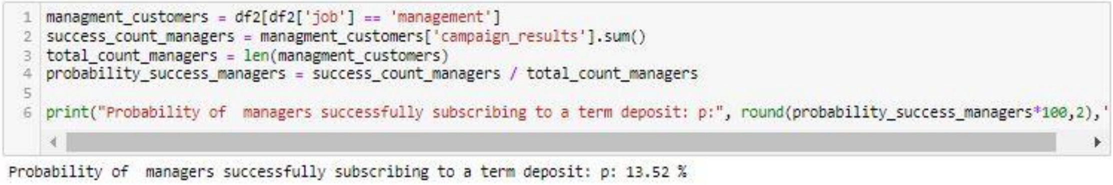
Based on the study delivered in previous steps we have defined our potential client more likely to subscribe a term deposit a customer with:

- **A Management role**
- **Adult**
- **In possession of a Standard Average balance account**.

Recommendations to help future campaigns succeed

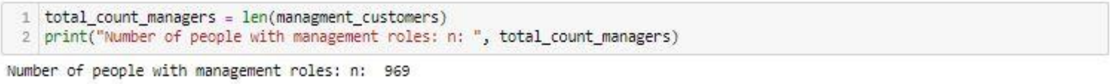
- It may be recommended to make a better effort to reach people with a standard and high average balance in order to take a safer position.
- It is recommended to increase efforts to reach customers with management roles.
- We have discovered that even if most of customers contacted were adults, it is actually young customers who seems to be proportionately more likely to subscribe to a term deposit.

Probability of managers successfully subscribing to a term deposit



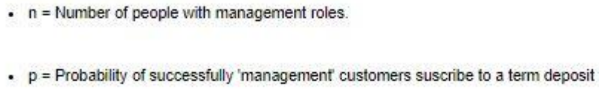
```
managment_customers = df2[df2['job'] == 'management']
success_count_managers = managment_customers['campaign_results'].sum()
total_count_managers = len(managment_customers)
probability_success_managers = success_count_managers / total_count_managers

print("Probability of  managers successfully subscribing to a term deposit: p:", round(probability_success_managers*100,2),'
```

```
Probability of  managers successfully subscribing to a term deposit: p: 13.52 %
```



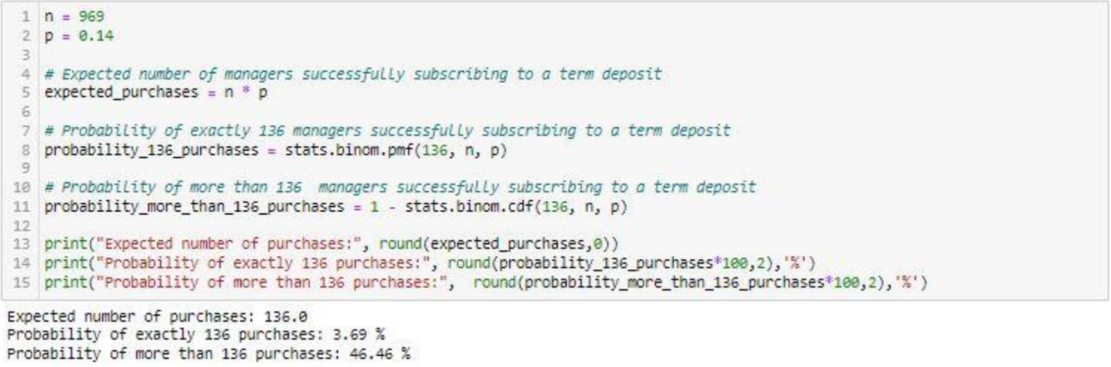
```
total_count_managers = len(managment_customers)
print("Number of people with management roles: n: ", total_count_managers)
```

```
Number of people with management roles: n:  969
```



- n = Number of people with management roles.

- p = Probability of successfully 'management' customers suscribe to a term deposit



```
n = 969
p = 0.14

# Expected number of managers successfully subscribing to a term deposit
expected_purchases = n * p

# Probability of exactly 136 managers successfully subscribing to a term deposit
probability_136_purchases = stats.binom.pmf(136, n, p)

# Probability of more than 136  managers successfully subscribing to a term deposit
probability_more_than_136_purchases = 1 - stats.binom.cdf(136, n, p)

print("Expected number of purchases:", round(expected_purchases,0))
print("Probability of exactly 136 purchases:", round(probability_136_purchases*100,2),'%')
print("Probability of more than 136 purchases:",  round(probability_more_than_136_purchases*100,2),'%')
```

```
Expected number of purchases: 136.0
Probability of exactly 136 purchases: 3.69 %
Probability of more than 136 purchases: 46.46 %
```

In order to study the **probability of a manager successfully subscribing to a term deposit**, we decided to perform a **binomial distribution analysis**.

To develop this analysis:

1. We have study which is the probability a manager successfully subscribed to a term deposit, studying their relative frequencies.
2. We have taken the total amount of managers we have in our dataset.
3. With those values
   a. Number of managers: **969.**
   b. Probability of successfully 'management' customers subscribe to a term deposit: **14%.**

- We have studied:
  - The total amount of managers we are expecting to successfully subscribe to a term deposit is **136**.
  - The probability of exactly 136 managers successfully subscribe to a term deposit is **3.69%.**
  - The probability of more than 136 managers successfully subscribe to a term deposit is **46.46%.**