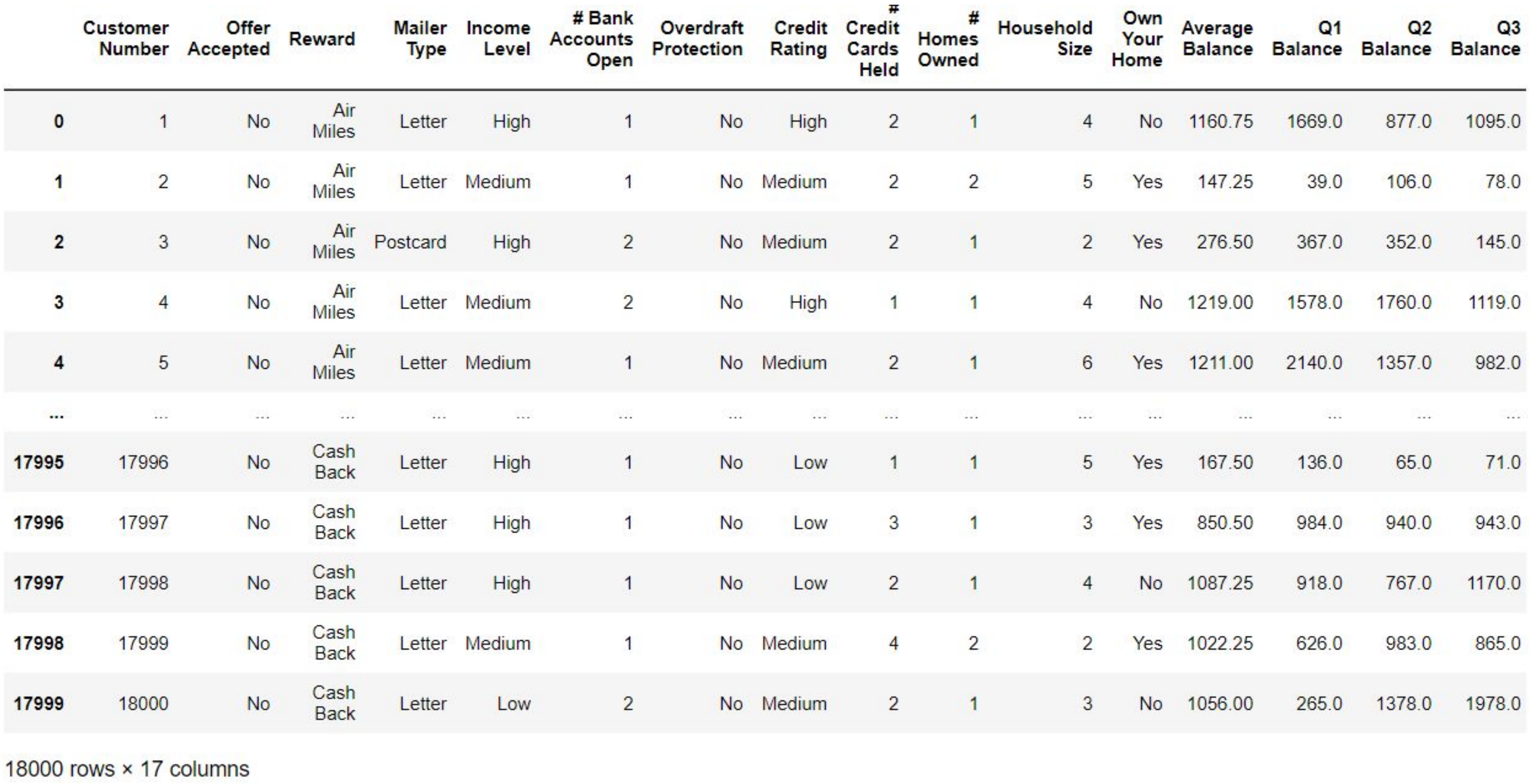
I. Glavatskyi @ Ironhack

Mid BootCamp Project „**Credit card customers classification**“

Extracting criteria for acceptance of the credit card offer to better target the banks politics and identify potential customers

Uses: Python.Pandas, SQL, Matplotlib, Seaborn, Numpy, Getpass, SQLAlchemy

Data overview:

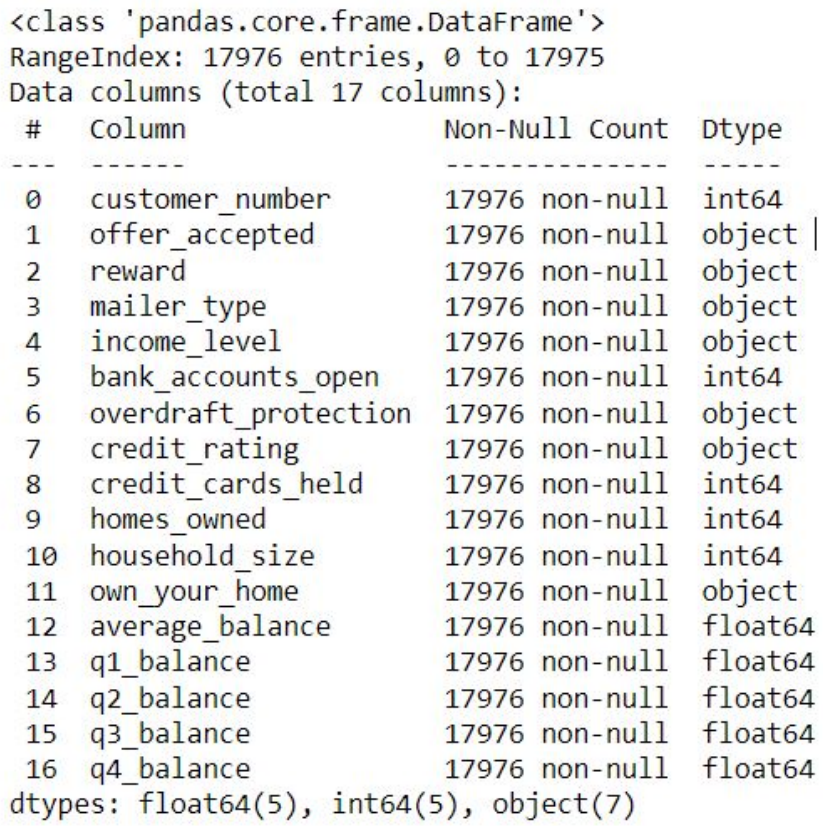


	Customer Number	Offer Accepted	Reward	Mailer Type	Income Level	# Bank Accounts Open	Overdraft Protection	Credit Rating	# Credit Cards Held	# Homes Owned	Household Size	Own Your Home	Average Balance	Q1 Balance	Q2 Balance	Q3 Balance
0	1	No	Air Miles	Letter	High	1	No	High	2	1	4	No	1160.75	1669.0	877.0	1095.0
1	2	No	Air Miles	Letter	Medium	1	No	Medium	2	2	5	Yes	147.25	39.0	106.0	78.0
2	3	No	Air Miles	Postcard	High	2	No	Medium	2	1	2	Yes	276.50	367.0	352.0	145.0
3	4	No	Air Miles	Letter	Medium	2	No	High	1	1	4	No	1219.00	1578.0	1760.0	1119.0
4	5	No	Air Miles	Letter	Medium	1	No	Medium	2	1	6	Yes	1211.00	2140.0	1357.0	982.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
17995	17996	No	Cash Back	Letter	High	1	No	Low	1	1	5	Yes	167.50	136.0	65.0	71.0
17996	17997	No	Cash Back	Letter	High	1	No	Low	3	1	3	Yes	850.50	984.0	940.0	943.0
17997	17998	No	Cash Back	Letter	High	1	No	Low	2	1	4	No	1087.25	918.0	767.0	1170.0
17998	17999	No	Cash Back	Letter	Medium	1	No	Medium	4	2	2	Yes	1022.25	626.0	983.0	865.0
17999	18000	No	Cash Back	Letter	Low	2	No	Medium	2	1	3	No	1056.00	265.0	1378.0	1978.0

18000 rows × 17 columns

Initial preparation

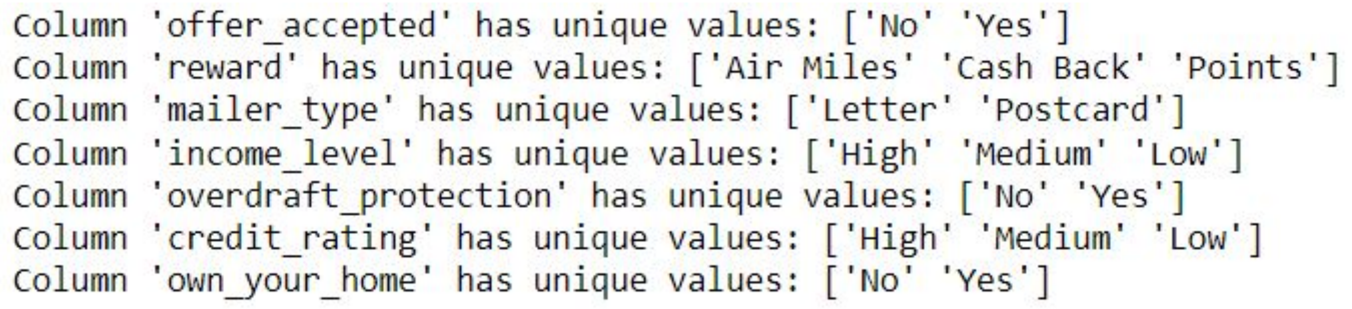
- nan_percentage = (df.isna().sum() / len(df)) * 100 -> average_balance 0.13% - Dropped.
- Checked column types and adjusted to proper ->



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17976 entries, 0 to 17975
Data columns (total 17 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   customer_number       17976 non-null  int64
 1   offer_accepted        17976 non-null  object
 2   reward                17976 non-null  object
 3   mailer_type           17976 non-null  object
 4   income_level          17976 non-null  object
 5   bank_accounts_open    17976 non-null  int64
 6   overdraft_protection  17976 non-null  object
 7   credit_rating         17976 non-null  object
 8   credit_cards_held     17976 non-null  int64
 9   homes_owned           17976 non-null  int64
 10  household_size        17976 non-null  int64
 11  own_your_home         17976 non-null  object
 12  average_balance       17976 non-null  float64
 13  q1_balance            17976 non-null  float64
 14  q2_balance            17976 non-null  float64
 15  q3_balance            17976 non-null  float64
 16  q4_balance            17976 non-null  float64
dtypes: float64(5), int64(5), object(7)
```

- Created the DB with a table and dropped irrelevant data (Q4 balance)
- Checked the values of ordinal columns to ensure if standardization is necessary:

  unique_values_per_column =
  {column: df[column].unique() for column in
  df.select_dtypes(include=['object']).columns}



```
Column 'offer_accepted' has unique values: ['No' 'Yes']
Column 'reward' has unique values: ['Air Miles' 'Cash Back' 'Points']
Column 'mailer_type' has unique values: ['Letter' 'Postcard']
Column 'income_level' has unique values: ['High' 'Medium' 'Low']
Column 'overdraft_protection' has unique values: ['No' 'Yes']
Column 'credit_rating' has unique values: ['High' 'Medium' 'Low']
Column 'own_your_home' has unique values: ['No' 'Yes']
```

Investigating the data

- #10.1 Average Balance of All Customers by Income:
  [('High', 942.6), ('Medium', 940.9), ('Low', 937.7)] – *difference neglectable*

- #10.2 Average balance of customers grouped by Income Level:
  Average Balance of All Customers by # Bank Accs: [(1, 941.5), (2, 936.5), (3, 948.3)] - *insignificant*

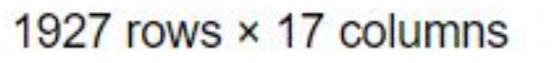
- #10.3 Average balance of customers grouped by Income Level:
  Average Balance of All Customers by # Bank Accs: [(1, 941.5), (2, 936.5), (3, 948.3)] – *insignificant*

- *Selected a view of customers with the following properties: (4949 rows × 17 columns)*

- Credit rating medium or high &
- Credit cards held 2 or less &
- Owns their own home &
- Household size 3 or more

And selected customers whose average balance is less than that of all the customers in the database:

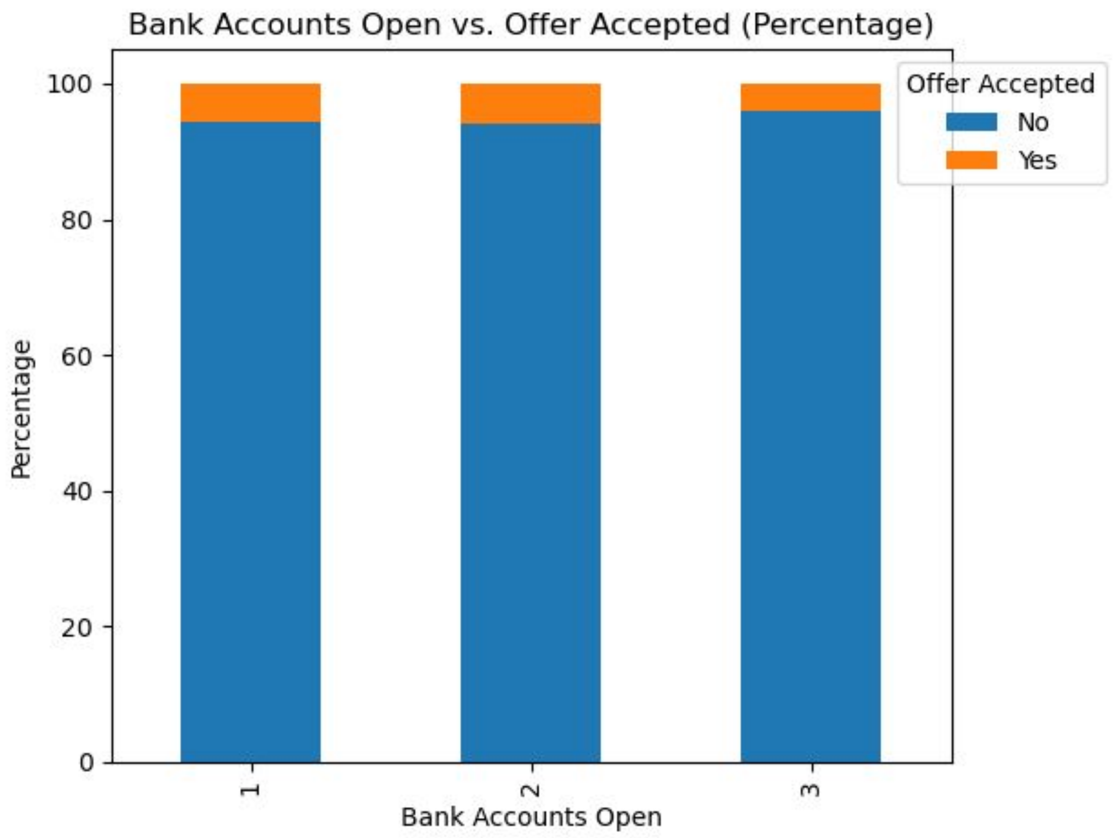
```
query = f"""SELECT * FROM {table_name}
    WHERE (credit_rating = 'Medium' OR credit_rating = 'High') AND
       credit_cards_held <= 2 AND
       own_your_home = 'Yes' AND
       household_size >= 3 AND
       average_balance < (SELECT AVG(average_balance)
    FROM {table_name});"""
```

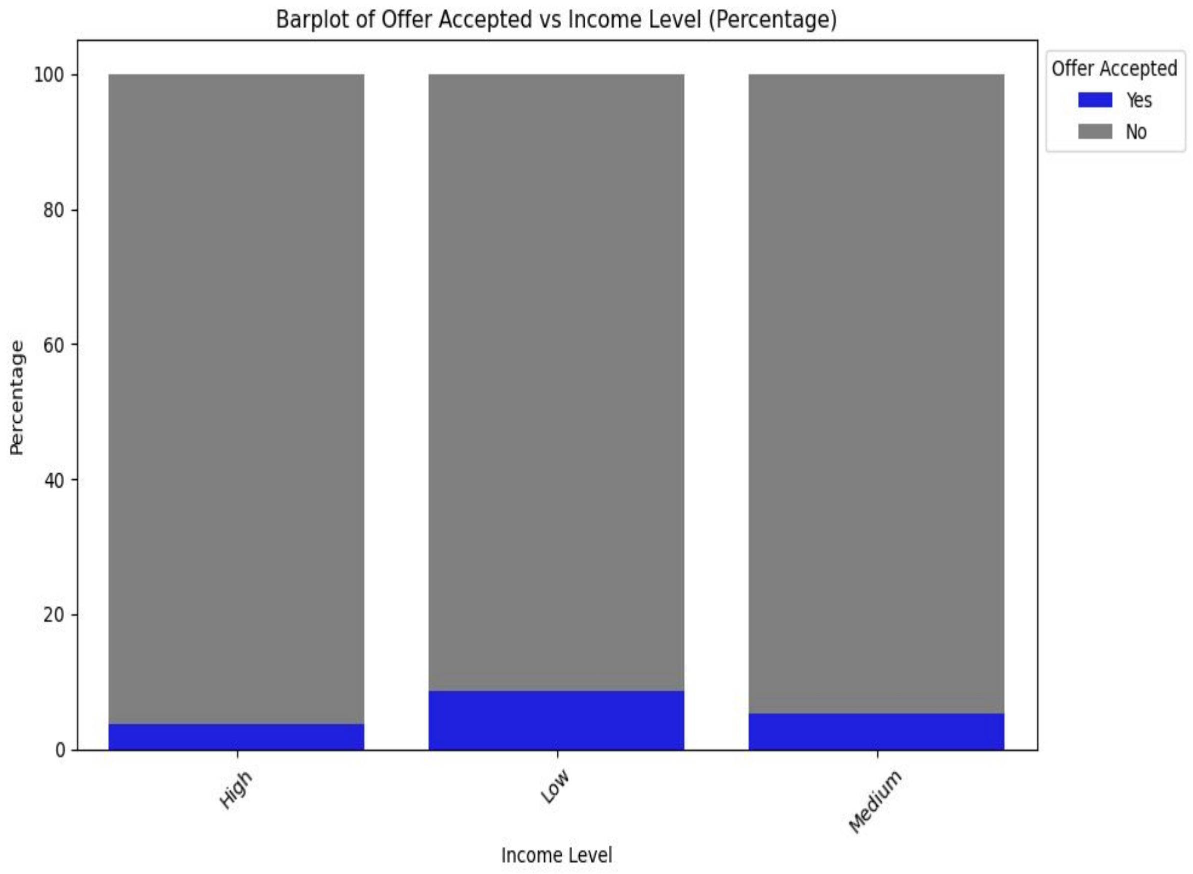


```
1927 rows × 17 columns
```

Find out credit card acceptance criteria

- Filter only those accepted: `df_accepted = df[df["offer_accepted"]=="Yes"] # 1021 entries`
- Customers with medium-high ratings have clearly more money on the balance, *as expected*
- Communication is important: among the customers who **accepted** the offer, **721** were addressed by **Postcards**, while **300** by **Letter**.


Bank Accounts Open vs. Offer Accepted (Percentage)
Percentage
Bank Accounts Open
Offer Accepted
No
Yes


Barplot of Offer Accepted vs Income Level (Percentage)
Percentage
High
Low
Medium
Income Level
Offer Accepted
Yes
No