Lugovtsev Timur, DSBA201, Variant 20 Report

Exploring the dataset

First things first, since the SAS system is no longer available in Russia, I used Python only for dealing with data.

Overview of the data:

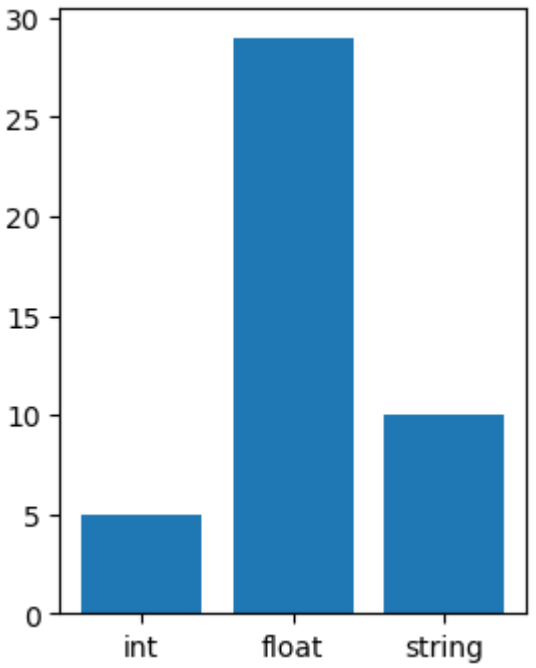
- ☐ 43 Features
- ☐ 34 numerical, as recognised by Python
- ☐ 9 Categorical

Plan for data analysis:

1. Explore the datatypes distribution
2. Exploring the nans percentage in columns
3. Exploring correlation between columns
4. Exploring different properties for numerical columns
5. Exploring the categorical variables' distribution

1. Datatypes distribution:

As can be seen from the data,
The major part of columns have floats as their values. As a result, the resulting dataframe after encoding all of the data would still be reasonably compact.

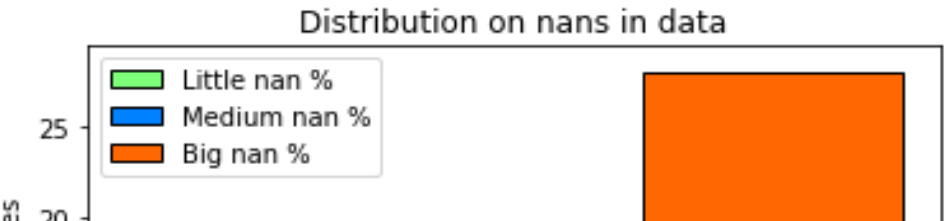


2. Exploring the nans in columns

After exploring the datatypes of the columns, it crossed my mind that a lot of them have the vast majority of data missing. It could lead to severe deterioration of segmentation model's quality and, as a result, to a worse segmentation. Thus, I decided to firstly explore the raw nan counts in the data:

As can be seen, the vast majority of columns have little to no available data. The two possible ways of dealing with it: either leave only those columns with decent (>70% at least) non-null counts in the rows and then impute everything or simply explore which clients can we delete to have more columns but less rows. I chose the latter, since this approach avoids loosing information about clients which can be useful when doing segmentation (For instance, the family status is quite important and it only has 3900 rows available). A more visual approach could be found on the next page.

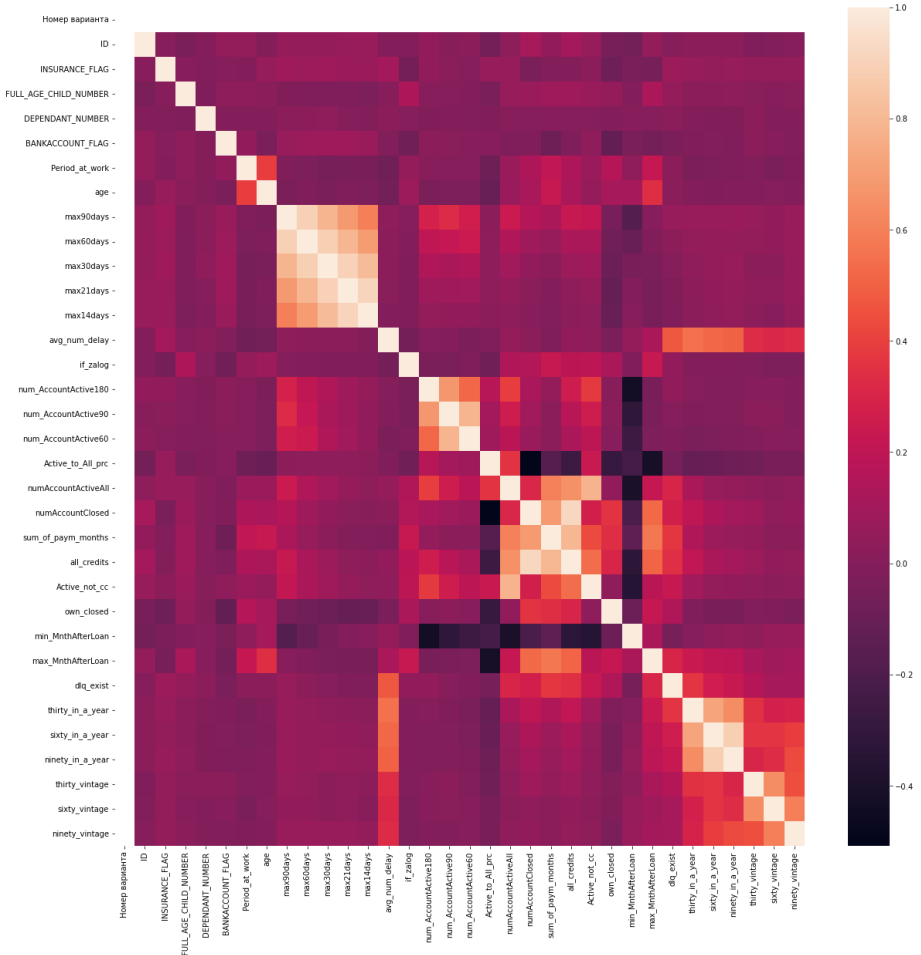
#	Column	Non-Null Count	Dtype
0	Номер варианта	10242 non-null	int64
1	ID	10242 non-null	int64
2	INCOME_BASE_TYPE	10179 non-null	object
3	CREDIT_PURPOSE	10242 non-null	object
4	INSURANCE_FLAG	10242 non-null	int64
5	DTI	10133 non-null	object
6	SEX	10242 non-null	object
7	FULL_AGE_CHILD_NUMBER	10242 non-null	int64
8	DEPENDANT_NUMBER	10242 non-null	int64
9	EDUCATION	10242 non-null	object
10	EMPL_TYPE	10231 non-null	object
11	EMPL_SIZE	10133 non-null	object
12	BANKACCOUNT_FLAG	7964 non-null	float64
13	Period_at_work	7963 non-null	float64
14	age	7964 non-null	float64
15	EMPL_PROPERTY	7963 non-null	object
16	EMPL_FORM	3926 non-null	object
17	FAMILY_STATUS	3926 non-null	object
18	max90days	3877 non-null	float64
19	max60days	3877 non-null	float64
20	max30days	3877 non-null	float64
21	max21days	3877 non-null	float64
22	max14days	3877 non-null	float64
23	avg_num_delay	3597 non-null	float64
24	if_zalog	3609 non-null	float64
25	num_AccountActive180	3609 non-null	float64
26	num_AccountActive90	3609 non-null	float64
27	num_AccountActive60	3609 non-null	float64
28	Active_to_All_prc	3609 non-null	float64
29	numAccountActiveAll	3609 non-null	float64
30	numAccountClosed	3609 non-null	float64
31	sum_of_paym_months	3609 non-null	float64
32	all_credits	3609 non-null	float64
33	Active_not_cc	3609 non-null	float64
34	own_closed	3609 non-null	float64
35	min_MnthAfterLoan	3609 non-null	float64
36	max_MnthAfterLoan	3609 non-null	float64
37	dlq_exist	3609 non-null	float64
38	thirty_in_a_year	3609 non-null	float64
39	sixty_in_a_year	3609 non-null	float64
40	ninety_in_a_year	3609 non-null	float64
41	thirty_vintage	3609 non-null	float64
42	sixty_vintage	3609 non-null	float64
43	ninety_vintage	3609 non-null	float64



From the diagram, It can be clearly seen that the vast majority of data has > 0.6 missing data in it.

3. Exploring feature correlation

As can be seen from the heatmap, overall, the overall correlation between features lies around 0.2-0.4 which is actually decent for machine learning algorithms. The most correlated features are those related to temporal events which makes a lot of sense, if one thinks about it.

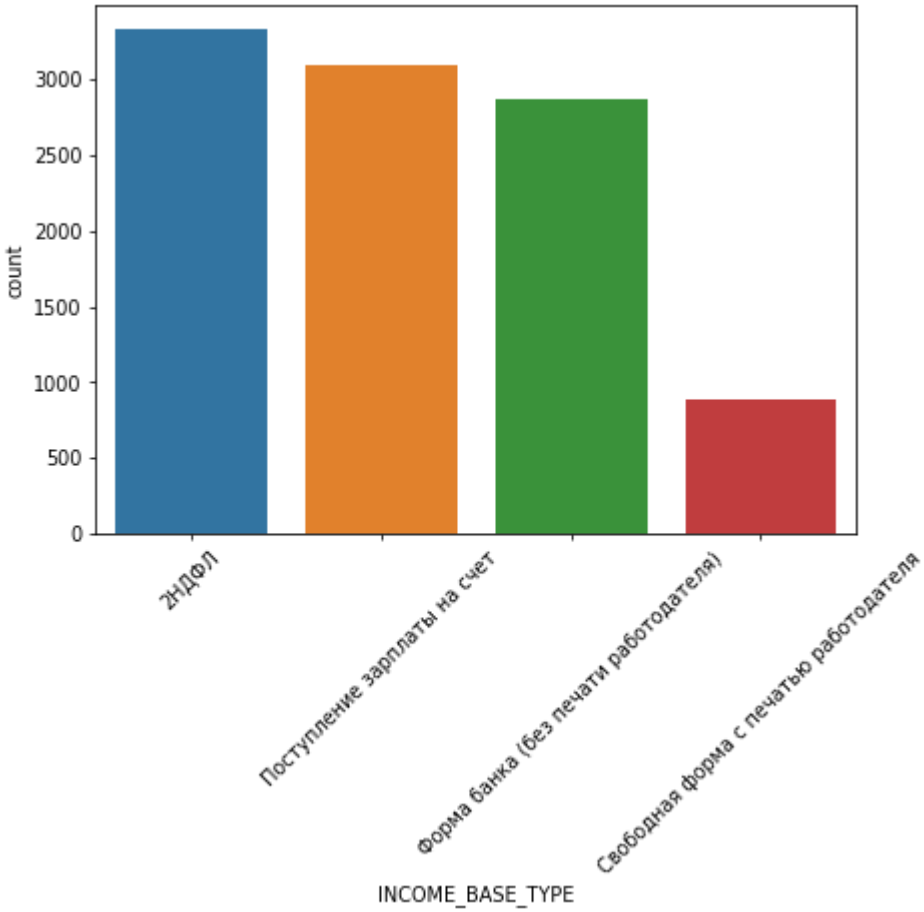


However, in the future it makes sense to use PCA for eliminating of correlation between features, as well as for dimensionality reduction purposes.

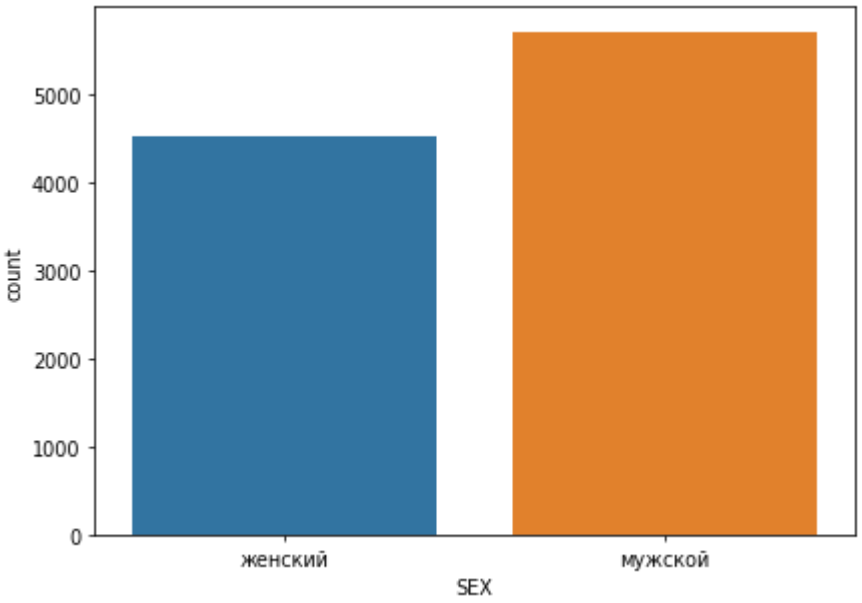
4. Exploring the categorical variables distribution

In order to better know what data you are working with and what to expect and also to build a profile of your typical clients, it makes sense to explore how different characteristics of data are distributed. For this one, I will use histograms and count plots from Python.

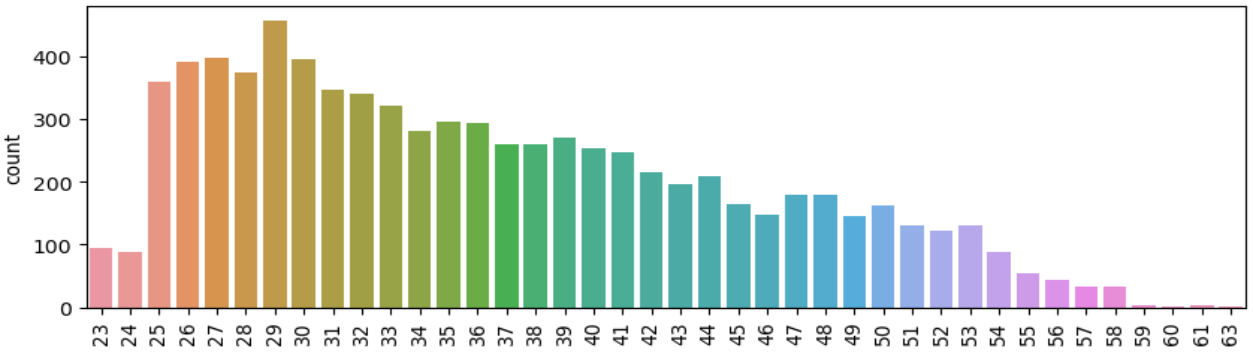
The first one is the income base type - it helps us to get insights on how our clients usually earn their money - do they have a reliable source of income? According to the data - yes, they do. A lot of clients are registered in a company and get their salary from it. The distribution is almost uniform, except for the last category, showing that the majority of our clients work a regular job.



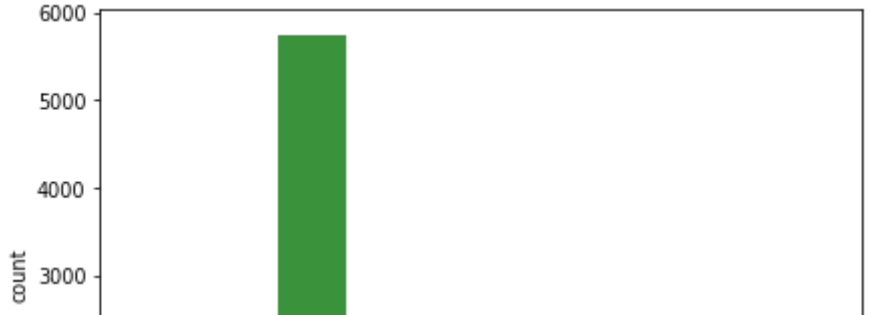
The second feature to explore is the gender of our clients. After plotting, we now know that the majority of people taking clients are male. This is not a surprise, since in Russian and post-soviet countries men are more likely to be the main earner of the family and thus taking the responsibility for the financial actions.



The third feature is the age. From the plot, we can see a strong diminishing trend starting from the year 29 up to 63. The period from 23 to 29 corresponds to younger people taking their first serious financial decisions in life - we can see a spike sound 29 years old which is a typical age for forming a family and getting a mortgage. It can also be concluded, that we should target the middle-aged people in their 30s-40s in our business decisions, since they contribute to around ¾ of the data.

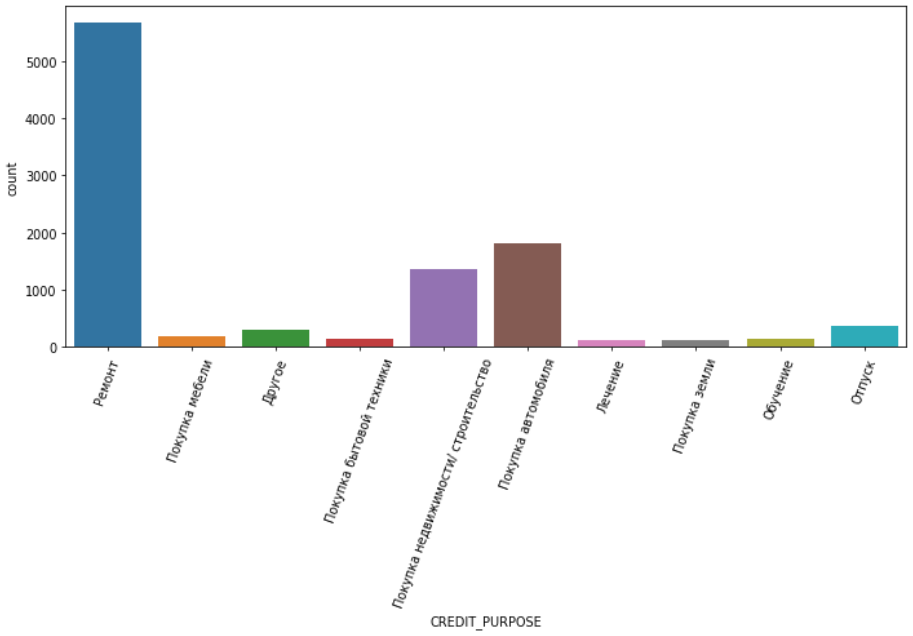


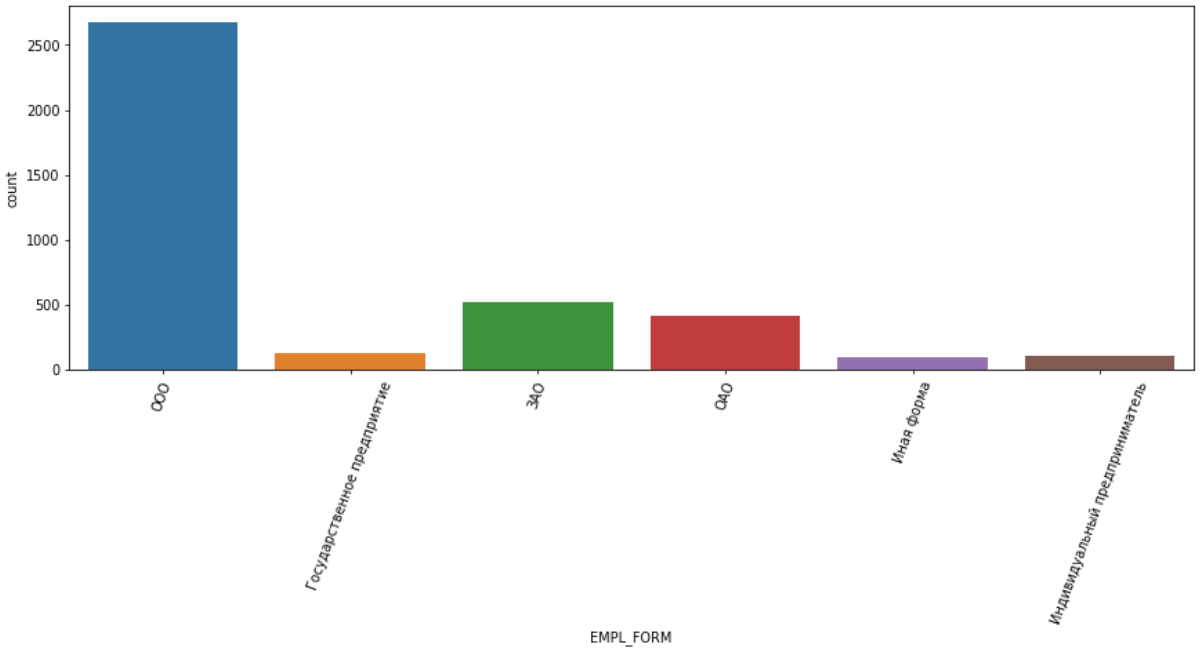
The education distribution is very uneven and is highly



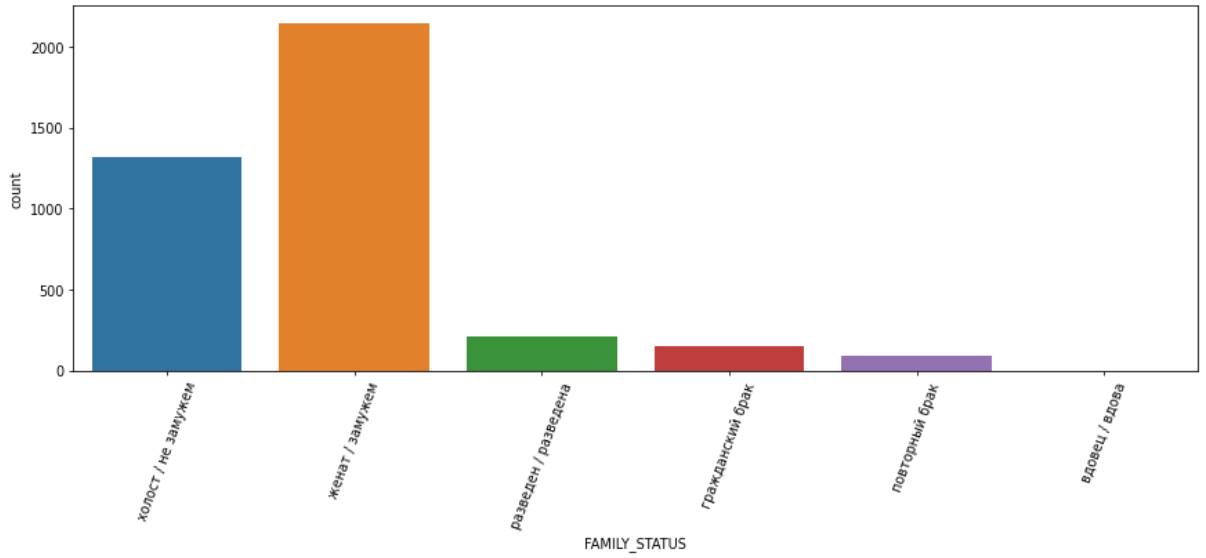
unlikely to be used as a predictor during segmentation. However, it provides insights on out target client - the predominant majority of our clients have higher education and thus are, hopefully, able to finance their credit (coming from a logic where the higher education result in a higher-paying job)

The credit purpose distribution is as disbalanced as the former feature. It also proves a point where the improving of your living condition, combined with property purchase which is also quite popular constitutes the majority of purposes for taking credit. Thus, it could be beneficial to provide special conditions for those if one wants to attract clients





As for the employment, after plotting the histogram, it becomes clear that the vast majority of clients are working for this or that type of LLC. This is not surprising, taking into account smaller salaries in federal companies.



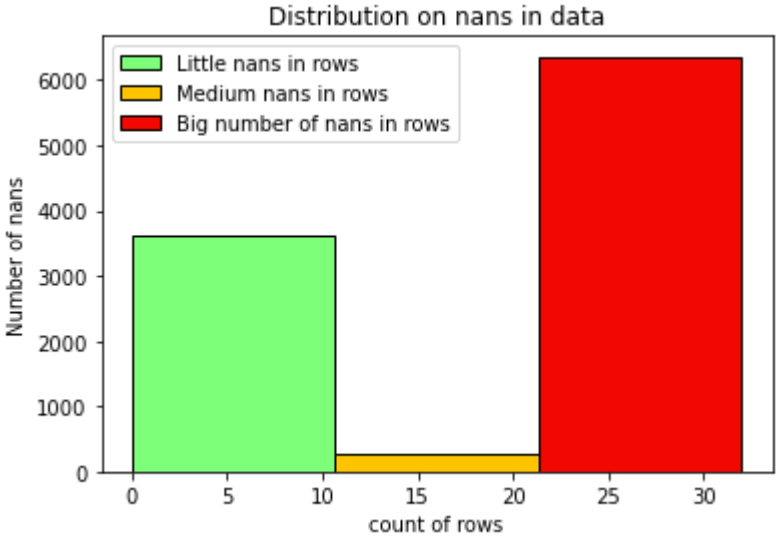
Considering the family status, most of the clients are married. This is not uncommon as it is more frequent that somebody takes credit for the purposes of the whole family, rather then for him/her-self. After these data manipulations, we now have our most frequent client:

- **Man**
- **Working for a LLC**
- **Earning >250k rubles/month**
- **Married**
- **Taking credit for renovations**
- **Having no kids (this can be found in code)**

Data cleaning for the segmentation.

Now, as we explored the entire dataset and decided on how we would decrease it's volume - let us proceed to doing so. Firstly, I built a separate dataframe showing the count of nans in a row and got this distribution.

As can be seen, the vast majority of rows have > 25 nans in them and this simply can't be used or imputed without severely damaging the quality of segmentation. Thus, We can focus on the green area and only include rows with less than 10 nans in them into our final data frame.



However, after some research, it was found that there is no significant difference between leaving rows with no nans and including those with

1-10 nans in them, the volume of the dataframe remained around the same.

```
'''
As can be seen from the figure above, it is reasoneble to choose the subset of data
with little to no nans, since the large proportion of rows have >20 nans in them
Thus, they are physically
Now, let's conclude on the treshhold, starting with 10
(the largest number on the green bin)
'''
for i in range(11):
    print(len([x for x in miss if x<=i]))
```

```
3593
3605
3606
3606
3606
3608
3609
3609
3609
3609
3609
```

```
'''
As can be clearly seen, we can simply use dropna without threshold and get
practically the same results as if we included rows with 10 nans in them
Thus, prepare data
'''
df = data.dropna().reset_index()
```

Now, after cleaning the data from nans, I am going to proceed with cleaning the data by replacing inadequate values with whatever is useful. Also, I will build the typical client of our bank by extracting the most frequent values out of several columns. After that, I will apply PCA for dimensionality reduction and use Different Segmentation techniques.

Data segmentation.

After the data preprocessing, we are left with ±3800 rows to segmentize. Also, another important step before that - encode the categorical variables. I used one-hot encoding for all variables, except for the education which has a clear progression and can be encoded in a growing number fashion: 1 for those finished school and 8 for PhDs.

```
'''
Now, let's use one-hot encoding for all categorical data except
the education - in this case, it is more appropriate to
use ordinal encoding IMO.
'''
Mapping = {'среднее':1, 'среднее-специальное':2,'незаконченное высшее':3,
           'высшее':4,'Высшее/Второе высшее/Ученая степень':5,'второе высшее':6,'ученая степень':7}
df['EDUCATION'] = df['EDUCATION'].map(Mapping)
```

After we are finished with this part, we now have 76 features which we are then going to expose to a dimensionality reduction procedure using Principal Component Analysis.

Now, for the segmentation methods: I used K-means clustering and decision tree classifier. Both of those methods are familiar to me and suit the data well.

K-means:
• This method is excellent for market segmentation, including customer segmentation by loans
• We have a very large number of variables in our sample. In this regard, k-means segmentation will help
to separate clusters according to an unknown feature, which will be set by the algorithm of the method itself.
• The whole idea of the method - maximize the distance between clusters -> improve the quality of segmentation by itself.

Decision tree:
- Fit well to this data, consisting of a large number of features
- If pruned correctly, can provide insights on the presented data
- Does not require scaling
- Agile method able to be customized for the best fit.

K-means clustering:

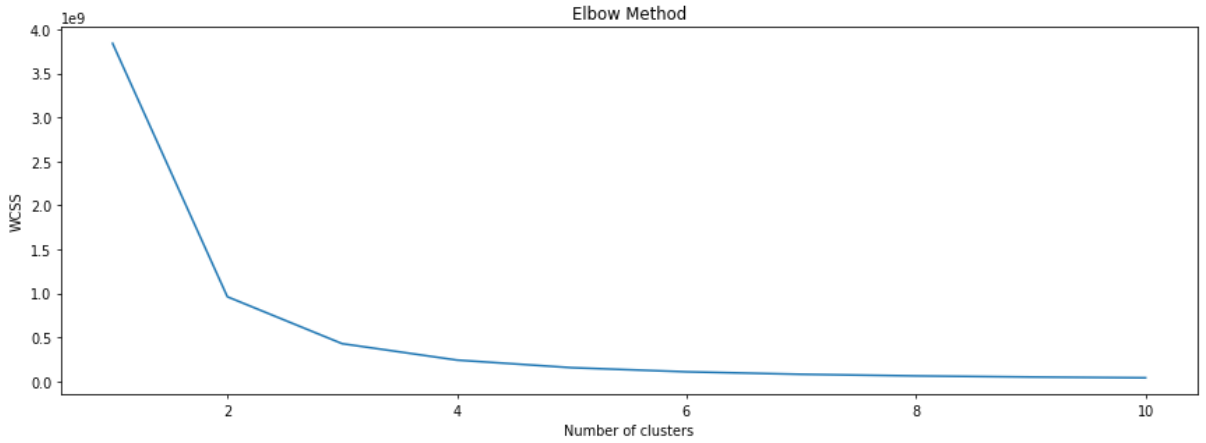
```
'''
Let's now use sklearn scaling, PCA in order to
quickly and conviniently apply kMeans

Firstly - find all continuous features to scale.

UPD - It was found to provide inferior results to unscaled
data -> I did not use it and only used
'''
float_cols = [x for x in df.columns if df[x].dtype==np.float64]
ss = StandardScaler()
for i in float_cols:
    df[i] = ss.fit_transform(np.array(df[i]).reshape(-1, 1))
```

```
'''
Now, let us use PCA for solving such issues as high  dimensionality and mutual
correlation. I will use the number of components saving 0.9 data variance - both reducing
the dimensiontionality and saving decent amount of variance in data.
'''
pc = PCA(n_components = 0.9,svd_solver='full')
new_x = pc.fit_transform(df)
len(new_x[0])
```

```
13
```

After applying PCA for dimensionality reduction, it was found that reducing the dimensions to 13 leaves our data's variance almost untouched which shows the redundancy of the vast majority of data.

In order to determine the number of clusters needed, I used the elbow method which showed me that 3 clusters is the optimal amount.



The final clustering visualization looks as follows:

Now, for each segment:

Analysis of clusters:

```
'''
Cluster one: 'Career people'
- Older age
- Have at least unfinished higher education
- Working for a LLC company
- Having insurance on her credit
- Have 1 children
- Working in commerce
- Planning on doing renoviation in their house
- Working for at least half a year on their current job
- Using online cabinet from at least 1 device
'''
```

Conclusion from this - Very workable.They differ in that they can long work in the new place. Reliable:may be late in payments, but still pay the loan. With a high probability will get a loan again

```
'''
Cluster 2 - Young worker
- Younger people
- Having 0 kids
- Working for LLC
- Just started at their new workplace with 30 days on average
- Having higher education
- Married
- Having insurance
'''
```

Conclusion from this - Young people which just started earning. High chances of delaying the payments. High chances of taking more credits in the future as well as earning more. Not very good credit history, since they only recently joined the labour force.

```
'''
Cluster 3 - Credit takers
- Moderate age
- Working for LLC
- 11 credits on average with decent spread over time
- Having higher education
- Working as managers

'''

for i in cl3_columns:
```

Conclusions from this - our preferable target audience. Middl-aged, decently wealthy and with great credit history. Will have little chance of delaying payment as well as changing their place of work.

Decision trees

Here I used a classic model from SkLearn library. I took the industry as the target, since it had a more balanced distribution and decent number of classes. I also pruned my tree to a length 3 in order to not overfit on the data. As a result, I got these segments:

9]:
```
'''
Preparing the target
'''

new_st = []
for i in range(len(df)):
    if(df['EMPL_PROPERTY_Другое'][i] ==1):
        new_st.append('Другое')
    elif(df['EMPL_PROPERTY_Информационные технологии'][i] ==1):
        new_st.append('Информационные технологии')
    elif(df['EMPL_PROPERTY_Сельское и лесное хозяйство'][i] ==1):
        new_st.append('Сельское и лесное хозяйство')
    elif(df['EMPL_PROPERTY_Торговля'][i] ==1):
        new_st.append('Торговля')
    elif(df['EMPL_PROPERTY_Юридические услуги'][i] ==1):
        new_st.append('Юридические услуги')
    else:
        print(i)

df['Target'] = new_st
y,class_names = pd.factorize(df['Target'])
df.drop('Target',axis=1,inplace=True)
```

4]:
```
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier(criterion='entropy',max_depth=3, random_state=42)
classifier.fit(X_train, y_train)
```

```
'''
The specialist:
- Working for a salary
- Middle age
- Frequently taking credit for car/property purchases
- Working as a specialist
- A lot of people are 2 times married in this category
'''
```

Conclusion: not the most reliable people. Not the most secure credit purposes, as well as getting married 2 times can signal that they are more risky and unstable in their payments.

```
'''
The Commerce people:
Basically, the same as the Kmeans cluster #1
'''
```

```
'''
The wealthy men working in IT:
- Working for a salary
- Young
- A lot of them are  high earners, compared to the rest of dataset
- The purposes are more evenly distributed
- At least higher education
'''
```

Conclusions: Well-educated, well-skilled and stable people. Working in a high-demand and high-paying domain which is good in terms of paying their credits. High chances of taking credit another time.

```
'''
Segment 5 -women working as support staff
- Have good credit history with around 5 years of paying them
- Quite a lot of them earning <50k a month compared to other segments
- Not currently married
- Making a career in Law firms
'''
```

Conclusions: low income as well as the job makes them eager to take credits as well as delay the payments. Moderate risk, mostly taking into account income and job. Since currently making a career - can expect better paying jobs and more stability in the future, thus should also be targeted.