In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import pandas as pd
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
import itertools
```

In [2]:
```
application_data = pd.read_csv(r"F:\stige projects\loan case study\kaggel\application_data.csv")
previous_application = pd.read_csv(r"F:\stige projects\loan case study\kaggel\previous_application.csv")
columns_description = pd.read_csv(r"F:\stige projects\loan case study\kaggel\columns_description.csv",skiprows =1)
```

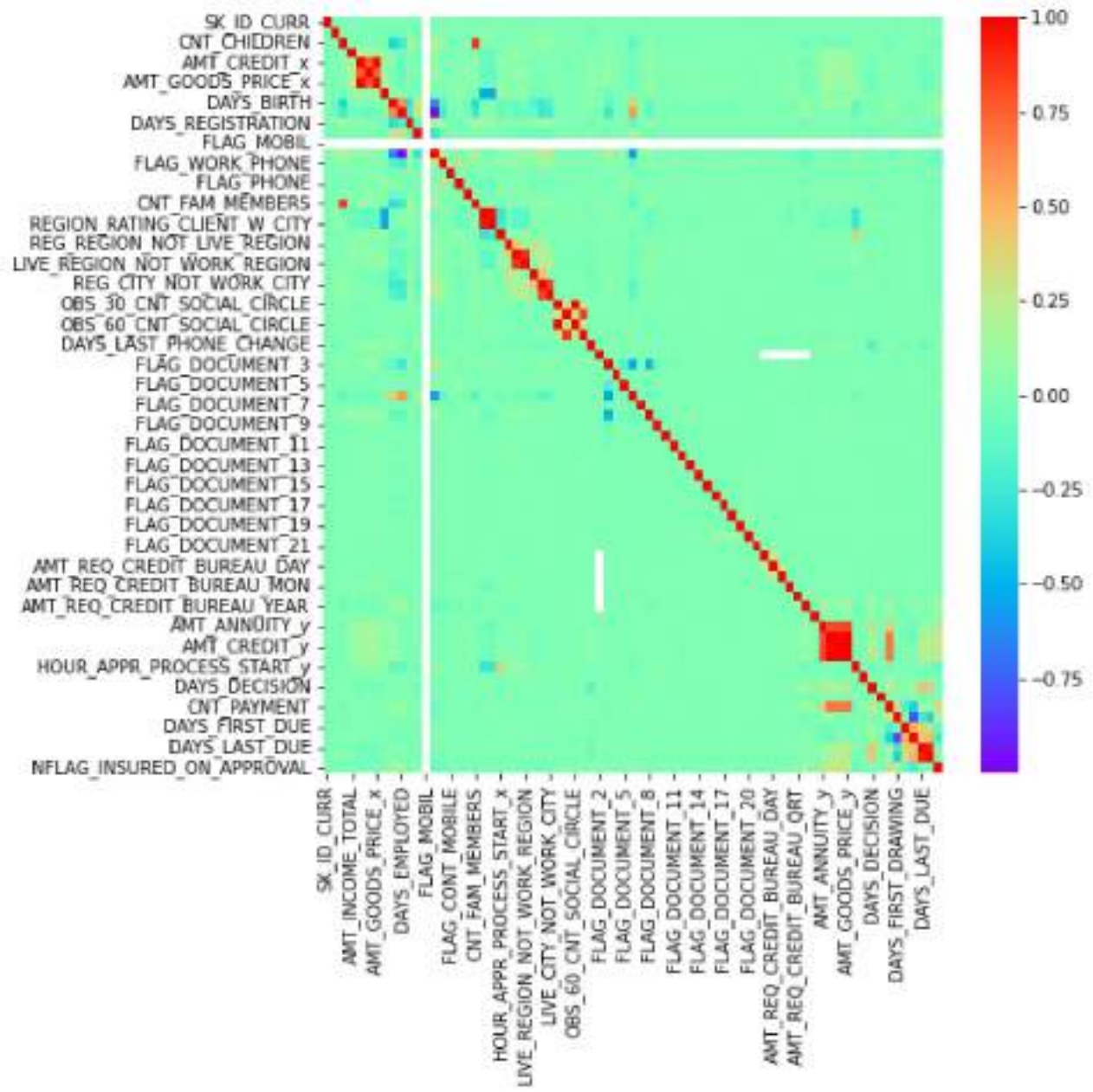
In [3]:
```
application_data=application_data.drop([ 'EXT_SOURCE_1', 'EXT_SOURCE_2', 'EXT_SOURCE_3',
       'APARTMENTS_AVG', 'BASEMENTAREA_AVG', 'YEARS_BEGINEXPLUATATION_AVG',
       'YEARS_BUILD_AVG', 'COMMONAREA_AVG', 'ELEVATORS_AVG', 'ENTRANCES_AVG',
       'FLOORSMAX_AVG', 'FLOORSMIN_AVG', 'LANDAREA_AVG',
       'LIVINGAPARTMENTS_AVG', 'LIVINGAREA_AVG', 'NONLIVINGAPARTMENTS_AVG',
       'NONLIVINGAREA_AVG', 'APARTMENTS_MODE', 'BASEMENTAREA_MODE',
       'YEARS_BEGINEXPLUATATION_MODE', 'YEARS_BUILD_MODE', 'COMMONAREA_MODE',
       'ELEVATORS_MODE', 'ENTRANCES_MODE', 'FLOORSMAX_MODE', 'FLOORSMIN_MODE',
       'LANDAREA_MODE', 'LIVINGAPARTMENTS_MODE', 'LIVINGAREA_MODE',
       'NONLIVINGAPARTMENTS_MODE', 'NONLIVINGAREA_MODE', 'APARTMENTS_MEDI',
       'BASEMENTAREA_MEDI', 'YEARS_BEGINEXPLUATATION_MEDI', 'YEARS_BUILD_MEDI',
       'COMMONAREA_MEDI', 'ELEVATORS_MEDI', 'ENTRANCES_MEDI', 'FLOORSMAX_MEDI',
       'FLOORSMIN_MEDI', 'LANDAREA_MEDI', 'LIVINGAPARTMENTS_MEDI',
       'LIVINGAREA_MEDI', 'NONLIVINGAPARTMENTS_MEDI', 'NONLIVINGAREA_MEDI',
       'FONDKAPREMONT_MODE', 'HOUSETYPE_MODE', 'TOTALAREA_MODE',
       'WALLSMATERIAL_MODE', 'EMERGENCYSTATE_MODE',"OWN_CAR_AGE","OCCUPATION_TYPE"],axis=1)
previous_application=previous_application.drop([ 'AMT_DOWN_PAYMENT', 'RATE_DOWN_PAYMENT', 'RATE_INTEREST_PRIMARY',
       "RATE_INTEREST_PRIVILEGED"],axis=1)
```

In [4]:
```
pd.set_option("display.max_rows", None, "display.max_columns", None)
combined_df= pd.merge(application_data, previous_application, on='SK_ID_CURR', how='inner')

combined_df.sort_values(by=['SK_ID_CURR','SK_ID_PREV'],ascending=[True,True],inplace=True)
display(combined_df.head(10))
```

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE_x	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTA
0	100002	1	Cash loans	M	N	Y	0	202500
1	100003	0	Cash loans	F	N	N	0	270000
3	100003	0	Cash loans	F	N	N	0	270000
2	100003	0	Cash loans	F	N	N	0	270000
4	100004	0	Revolving loans	M	Y	Y	0	67500
9	100006	0	Cash loans	F	N	Y	0	135000
10	100006	0	Cash loans	F	N	Y	0	135000
8	100006	0	Cash loans	F	N	Y	0	135000
13	100006	0	Cash loans	F	N	Y	0	135000
5	100006	0	Cash loans	F	N	Y	0	135000

In [5]:
```
combined_df['DAYS_BIRTH'] = abs(combined_df['DAYS_BIRTH'])
combined_df['DAYS_ID_PUBLISH'] = abs(combined_df['DAYS_ID_PUBLISH'])
combined_df['DAYS_ID_PUBLISH'] = abs(combined_df['DAYS_ID_PUBLISH'])
combined_df['DAYS_LAST_PHONE_CHANGE'] = abs(combined_df['DAYS_LAST_PHONE_CHANGE'])
```

In [6]:
```
corrmat = combined_df.corr()

f, ax = plt.subplots(figsize =(8, 8))
sns.heatmap(corrmat, ax = ax, cmap ="rainbow")
plt.show()
```



In [7]:

```
corrmat = combined_df.corr()
corrdf = corrmat.where(np.triu(np.ones(corrmat.shape), k=1).astype(np.bool))
corrdf = corrdf.unstack().reset_index()
corrdf.columns = ['Var1', 'Var2', 'Correlation']
```

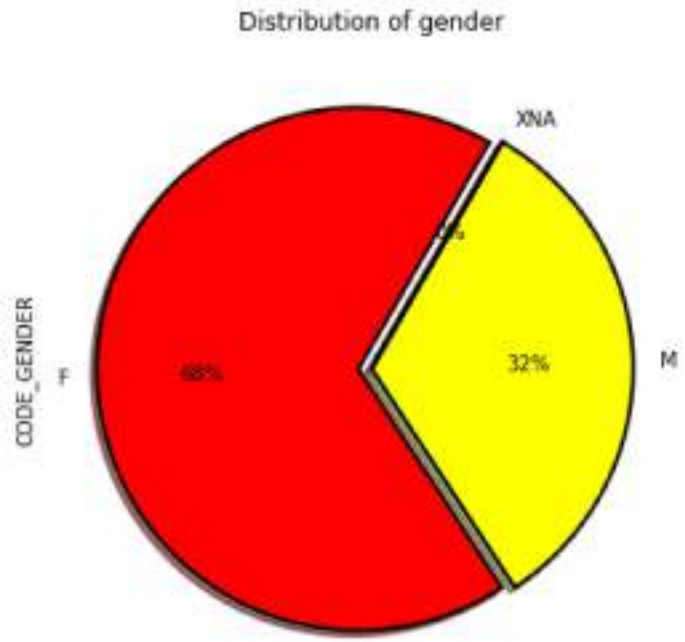
```
corrdf.dropna(subset = ['Correlation'], inplace = True)
corrdf['Correlation'] = round(corrdf['Correlation'], 2)
corrdf['Correlation'] = abs(corrdf['Correlation'])
corrdf.sort_values(by = 'Correlation', ascending = False).head(10)
```

Out[7]:

	Var1	Var2	Correlation
4786	AMT_GOODS_PRICE_y	AMT_APPLICATION	1.00
2278	OBS_60_CNT_SOCIAL_CIRCLE	OBS_30_CNT_SOCIAL_CIRCLE	1.00
984	FLAG_EMP_PHONE	DAYS_EMPLOYED	1.00
4787	AMT_GOODS_PRICE_y	AMT_CREDIT_y	0.99
454	AMT_GOODS_PRICE_x	AMT_CREDIT_x	0.99
4711	AMT_CREDIT_y	AMT_APPLICATION	0.98
1519	REGION_RATING_CLIENT_W_CITY	REGION_RATING_CLIENT	0.95
5547	DAYS_TERMINATION	DAYS_LAST_DUE	0.93
1823	LIVE_REGION_NOT_WORK_REGION	REG_REGION_NOT_WORK_REGION	0.88
1352	CNT_FAM_MEMBERS	CNT_CHILDREN	0.88

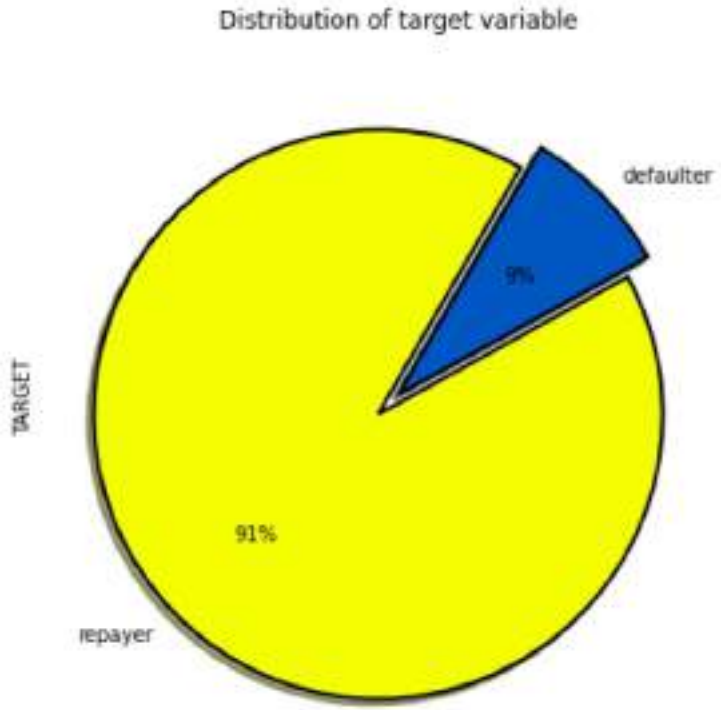
In [8]:

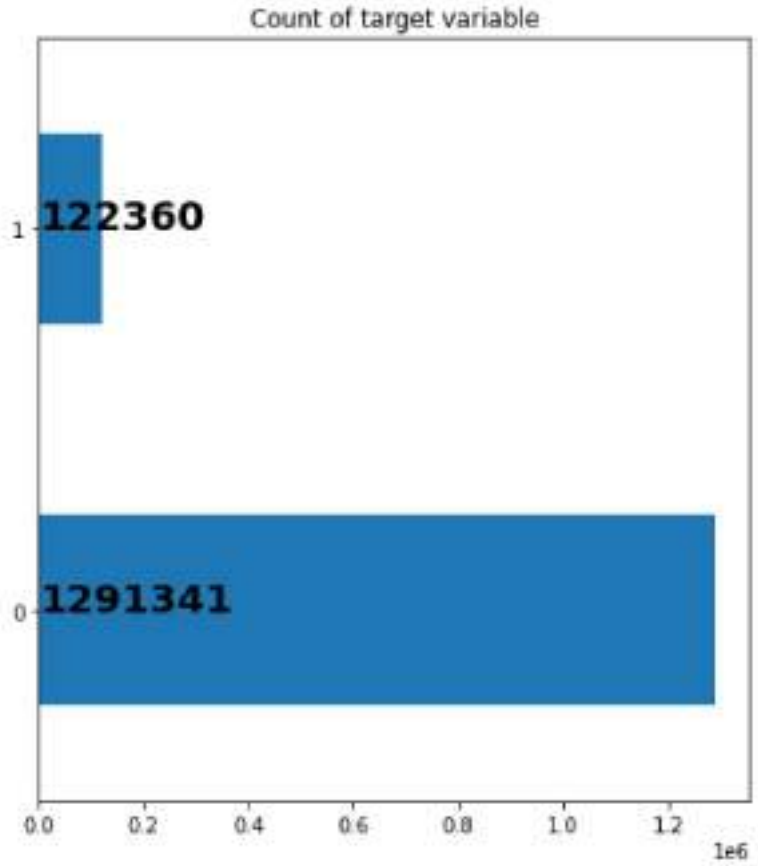
```
fig = plt.figure(figsize=(13,6))
plt.subplot(121)
combined_df["CODE_GENDER"].value_counts().plot.pie(autopct = "%1.0f%%",colors = ["red","yellow"],startangle = 60,
                                                   wedgeprops={"linewidth":2,"edgecolor":"k"},explod
plt.title("Distribution of gender")
plt.show()
```



In [9]:

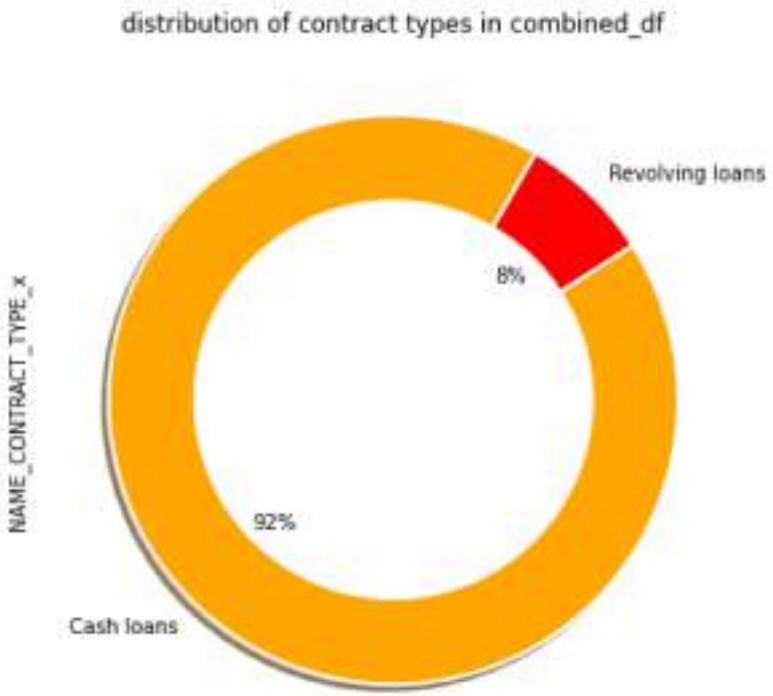
```
plt.figure(figsize=(14,7))
plt.subplot(121)
combined_df["TARGET"].value_counts().plot.pie(autopct = "%1.0f%%",colors = sns.color_palette("prism",7),startangle = 60,l
                                                                    wedgeprops={"linewidth":2,"edgecolor":"k"},explod
plt.title("Distribution of target variable")

plt.subplot(122)
ax = combined_df["TARGET"].value_counts().plot(kind="barh")

for i,j in enumerate(combined_df["TARGET"].value_counts().values):
    ax.text(.7,i,j,weight = "bold",fontsize=20)

plt.title("Count of target variable")
plt.show()
```





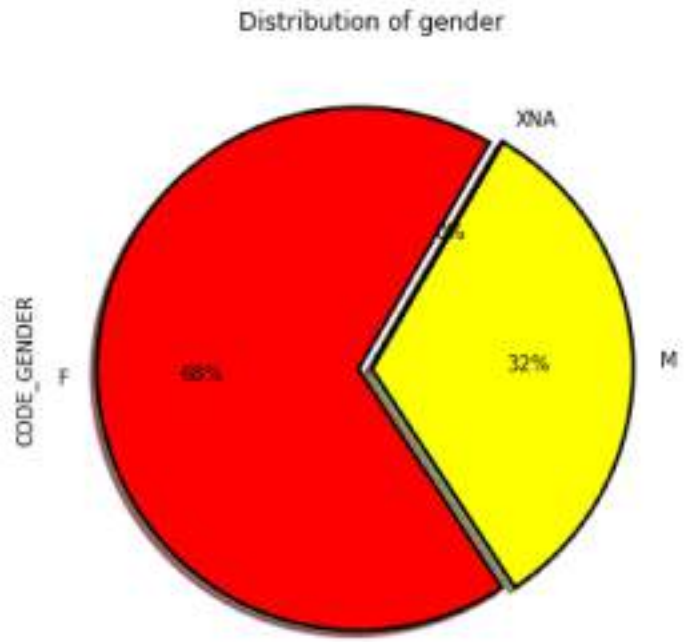
In [10]:

```
plt.figure(figsize=(14,7))
plt.subplot(121)
combined_df["NAME_CONTRACT_TYPE_x"].value_counts().plot.pie(autopct = "%1.0f%%",colors = ["orange","red"],startangle = 60
                                                            wedgeprops={"linewidth":2,"edgecolor":"white"},sh
circ = plt.Circle((0,0),.7,color="white")
plt.gca().add_artist(circ)
plt.title("distribution of contract types in combined_df")

plt.show()
```



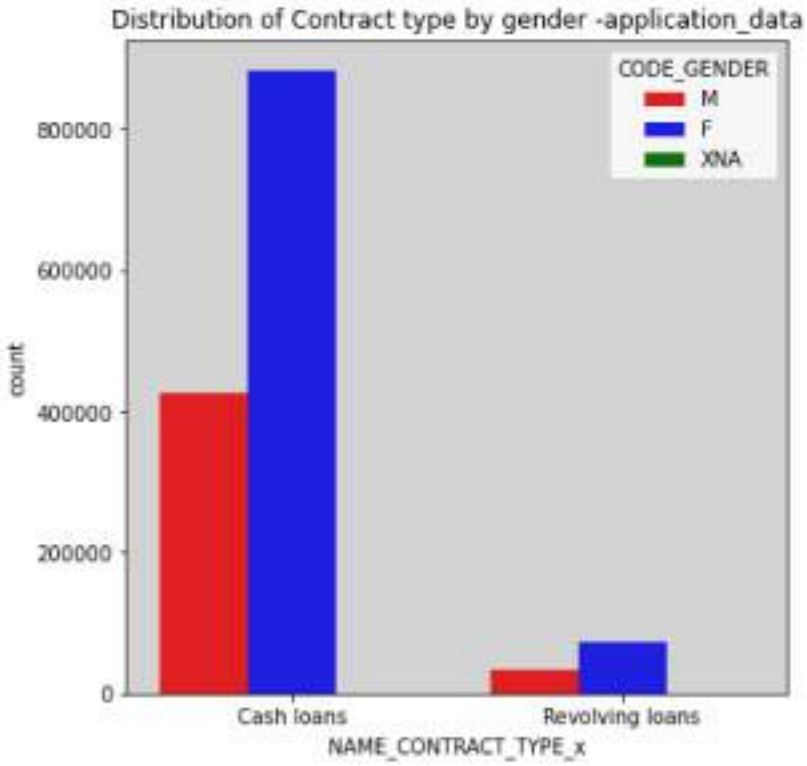
In [11]:

```
fig = plt.figure(figsize=(13,6))
plt.subplot(121)
combined_df["CODE_GENDER"].value_counts().plot.pie(autopct = "%1.0f%%",colors = ["red","yellow"],startangle = 60,
                                                   wedgeprops={"linewidth":2,"edgecolor":"k"},explod
plt.title("Distribution of gender")
plt.show()
```



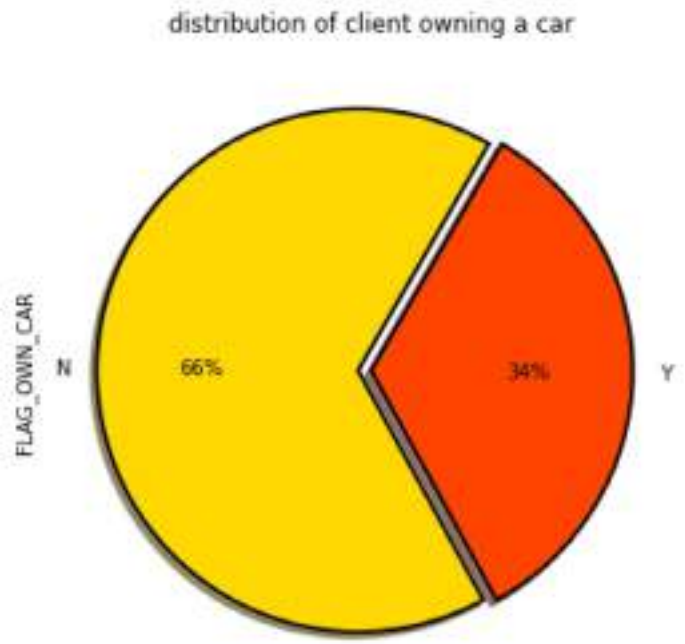
In [12]:

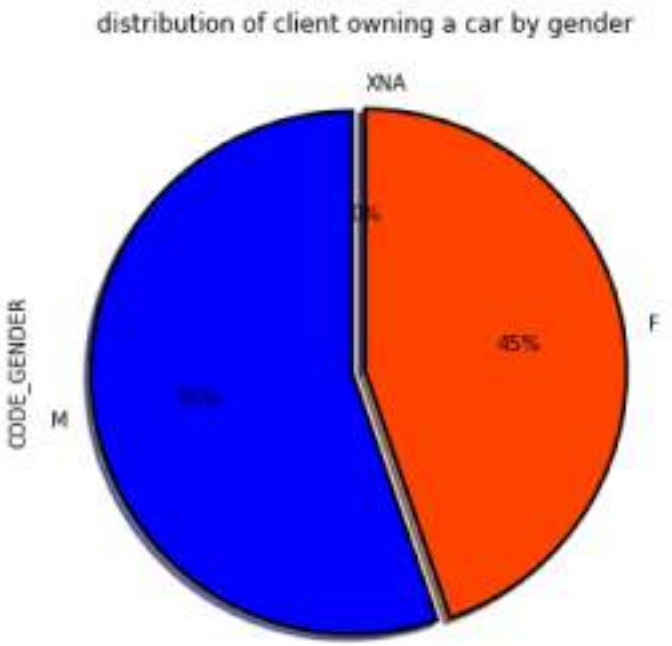
```
fig  = plt.figure(figsize=(13,6))
plt.subplot(121)
ax = sns.countplot("NAME_CONTRACT_TYPE_x",hue="CODE_GENDER",data=combined_df,palette=["r","b","g"])
ax.set_facecolor("lightgrey")
ax.set_title("Distribution of Contract type by gender -application_data")


plt.show()
```



In [13]:

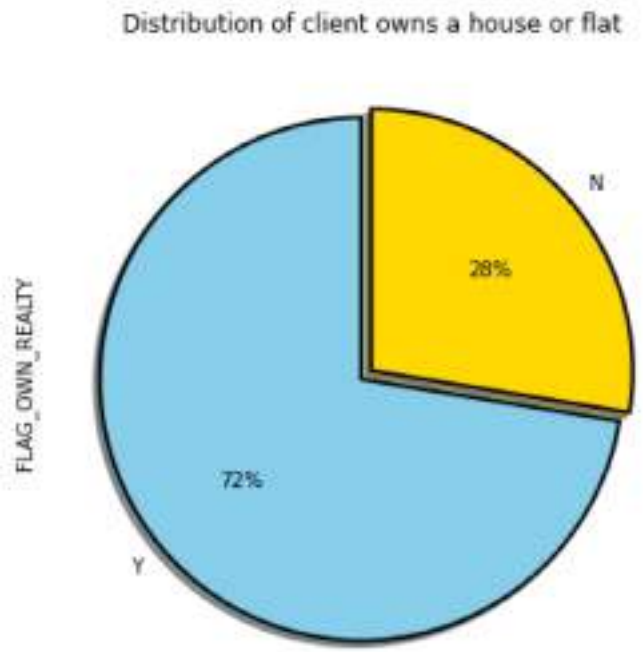
```
fig = plt.figure(figsize=(13,6))

plt.subplot(121)
combined_df["FLAG_OWN_CAR"].value_counts().plot.pie(autopct = "%1.0f%%",colors = ["gold","orangered"],startangle = 60,
                                                    wedgeprops={"linewidth":2,"edgecolor":"k"},explod
plt.title("distribution of client owning a car")

plt.subplot(122)
combined_df[combined_df["FLAG_OWN_CAR"] == "Y"]["CODE_GENDER"].value_counts().plot.pie(autopct = "%1.0f%%",colors = ["b",
                                                    wedgeprops={"linewidth":2,"edgecolor":"k"},explod
plt.title("distribution of client owning a car by gender")

plt.show()
```

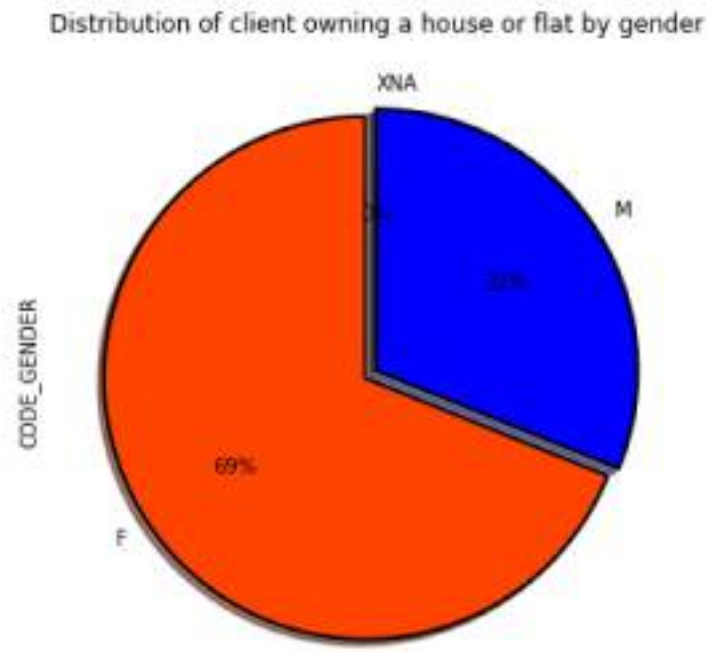




In [14]:

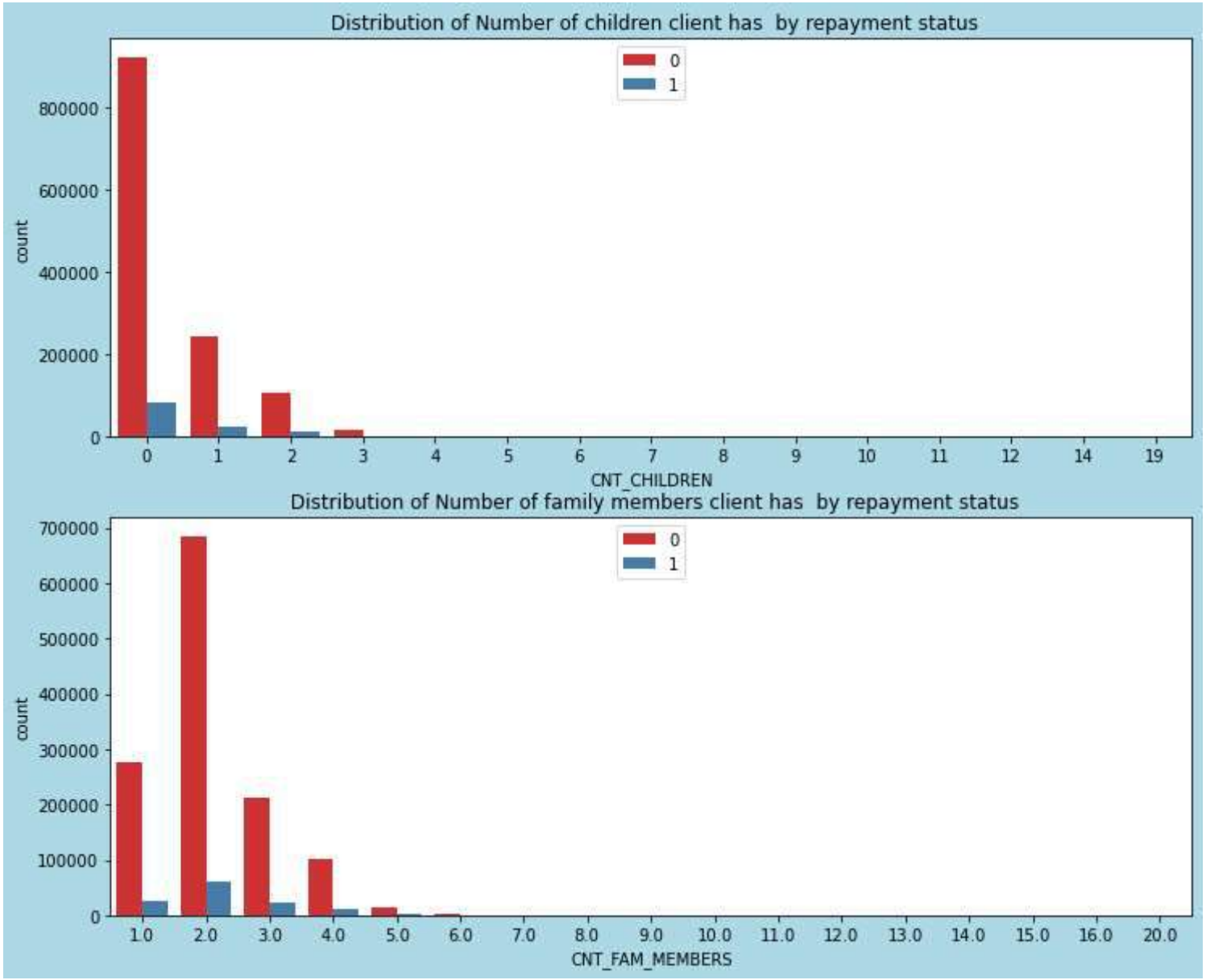
```
plt.figure(figsize=(13,6))
plt.subplot(121)
combined_df["FLAG_OWN_REALTY"].value_counts().plot.pie(autopct = "%1.0f%%",colors = ["skyblue","gold"],startangle = 90,
                                                       wedgeprops={"linewidth":2,"edgecolor":"k"},explode=[0.05,0],shadow =True)
plt.title("Distribution of client owns a house or flat")

plt.subplot(122)
combined_df[combined_df["FLAG_OWN_REALTY"] == "Y"]["CODE_GENDER"].value_counts().plot.pie(autopct = "%1.0f%%",colors = ["
                                                                                          wedgeprops={"linewidth":2,"edgecolor":"k"},explod
plt.title("Distribution of client owning a house or flat by gender")
plt.show()
```





In [15]:

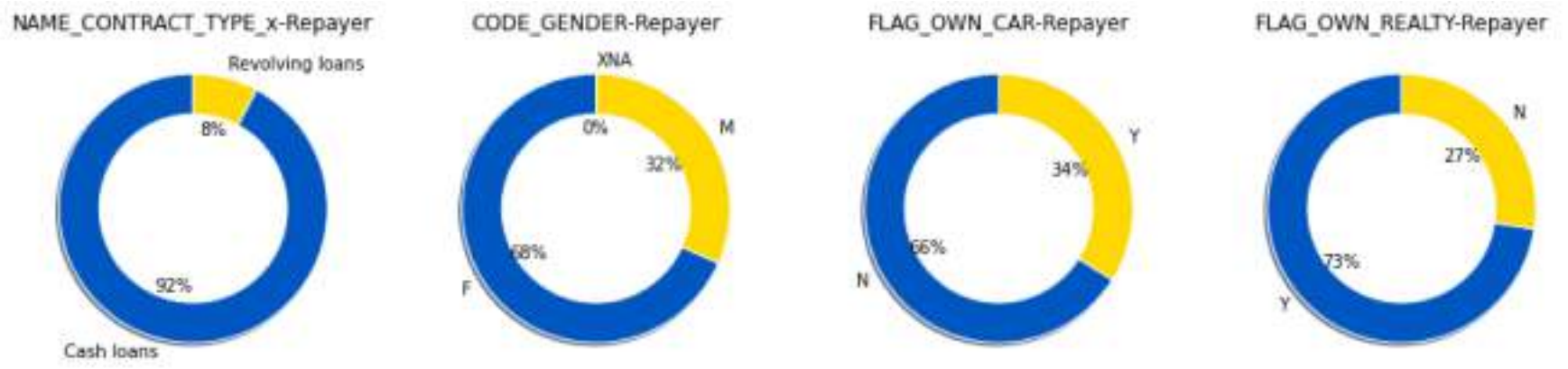
```
fig = plt.figure(figsize=(12,10))
plt.subplot(211)
sns.countplot(combined_df["CNT_CHILDREN"],palette="Set1",hue=combined_df["TARGET"])
plt.legend(loc="upper center")
plt.title(" Distribution of Number of children client has  by repayment status")
plt.subplot(212)
sns.countplot(combined_df["CNT_FAM_MEMBERS"],palette="Set1",hue=combined_df["TARGET"])
plt.legend(loc="upper center")
plt.title(" Distribution of Number of family members client has  by repayment status")
fig.set_facecolor("lightblue")
```



In [16]:

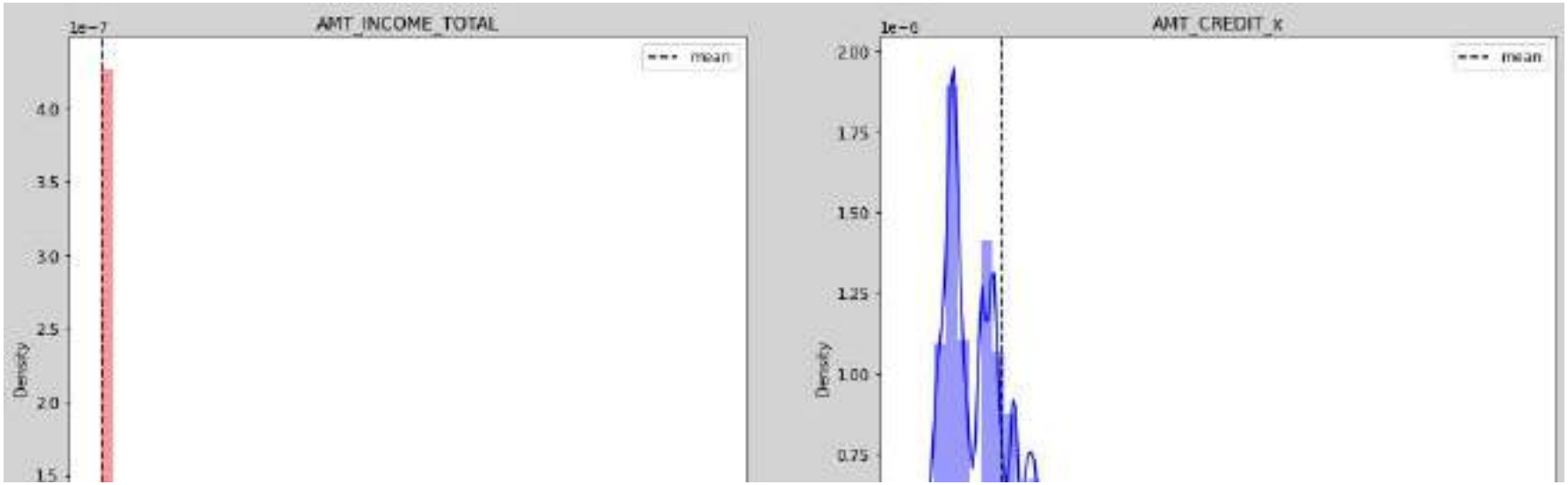
```
default = combined_df[combined_df["TARGET"]==1][[ 'NAME_CONTRACT_TYPE_x', 'CODE_GENDER','FLAG_OWN_CAR', 'FLAG_OWN_REALTY'
non_default = combined_df[combined_df["TARGET"]==0][[ 'NAME_CONTRACT_TYPE_x', 'CODE_GENDER','FLAG_OWN_CAR', 'FLAG_OWN_REA

d_cols = ['NAME_CONTRACT_TYPE_x', 'CODE_GENDER','FLAG_OWN_CAR', 'FLAG_OWN_REALTY']
d_length = len(d_cols)
```

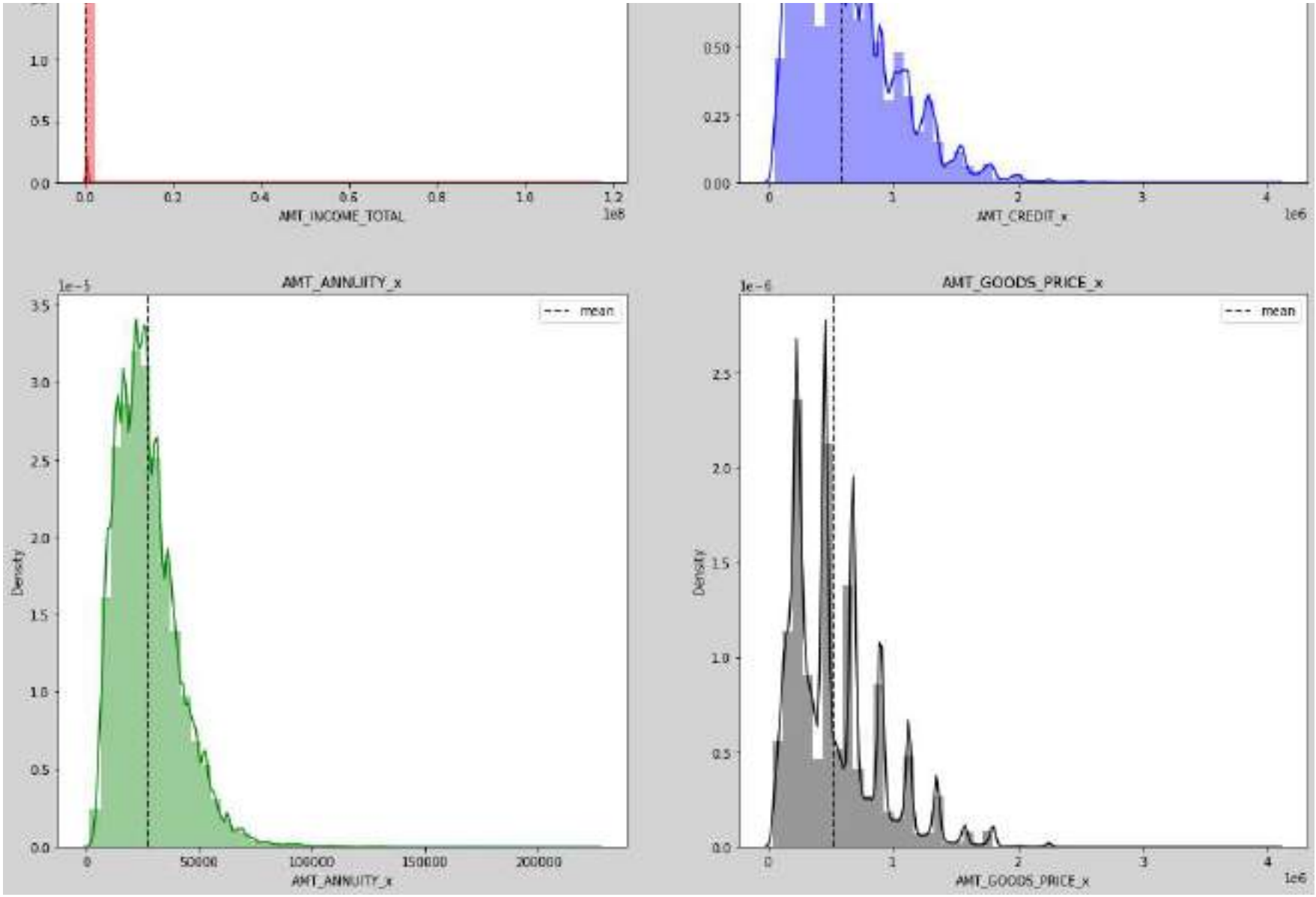
```
fig = plt.figure(figsize=(16,4))
for i,j in itertools.zip_longest(d_cols,range(d_length)):
    plt.subplot(1,4,j+1)
    default[i].value_counts().plot.pie(autopct = "%1.0f%%",colors = sns.color_palette("prism"),startangle = 90,
                                       wedgeprops={"linewidth":1,"edgecolor":"white"},shadow =True)
    circ = plt.Circle((0,0),.7,color="white")
    plt.gca().add_artist(circ)
    plt.ylabel("")
    plt.title(i+"-Defaulter")


fig = plt.figure(figsize=(16,4))
for i,j in itertools.zip_longest(d_cols,range(d_length)):
    plt.subplot(1,4,j+1)
    non_default[i].value_counts().plot.pie(autopct = "%1.0f%%",colors = sns.color_palette("prism",3),startangle = 90,
                                           wedgeprops={"linewidth":1,"edgecolor":"white"},shadow =True)
    circ = plt.Circle((0,0),.7,color="white")
    plt.gca().add_artist(circ)
    plt.ylabel("")
    plt.title(i+"-Repayer")
```



In [17]:

```
cols = [ 'AMT_INCOME_TOTAL', 'AMT_CREDIT_x','AMT_ANNUITY_x', 'AMT_GOODS_PRICE_x']
length = len(cols)
cs = ["r","b","g","k"]

ax = plt.figure(figsize=(18,18))
ax.set_facecolor("lightgrey")
for i,j,k in itertools.zip_longest(cols,range(length),cs):
    plt.subplot(2,2,j+1)
    sns.distplot(combined_df[combined_df[i].notnull()][i],color=k)
    plt.axvline(combined_df[i].mean(),label = "mean",linestyle="dashed",color="k")
    plt.legend(loc="best")
    plt.title(i)
    plt.subplots_adjust(hspace = .2)
```

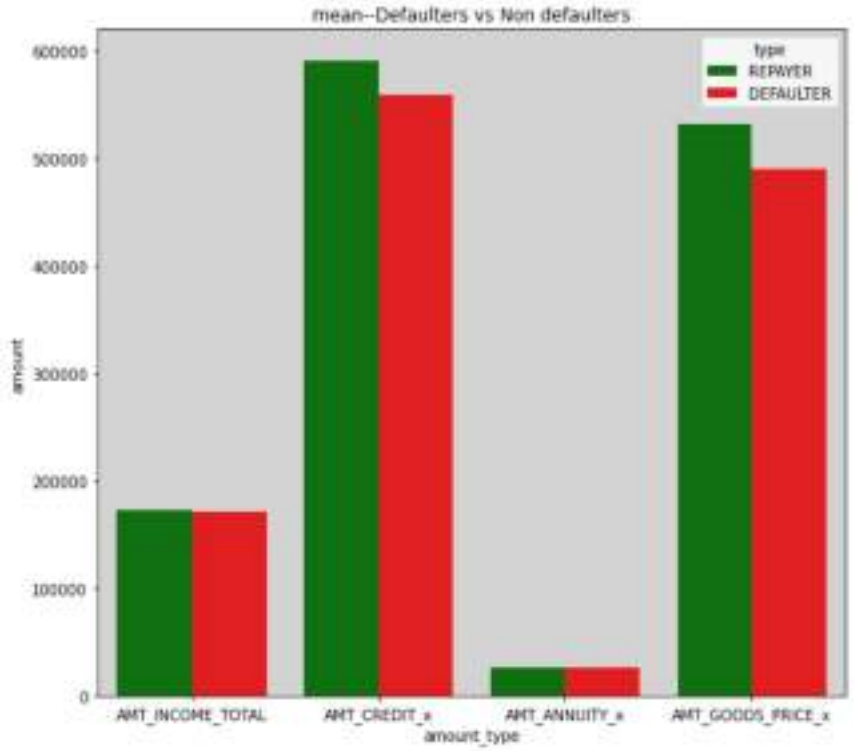


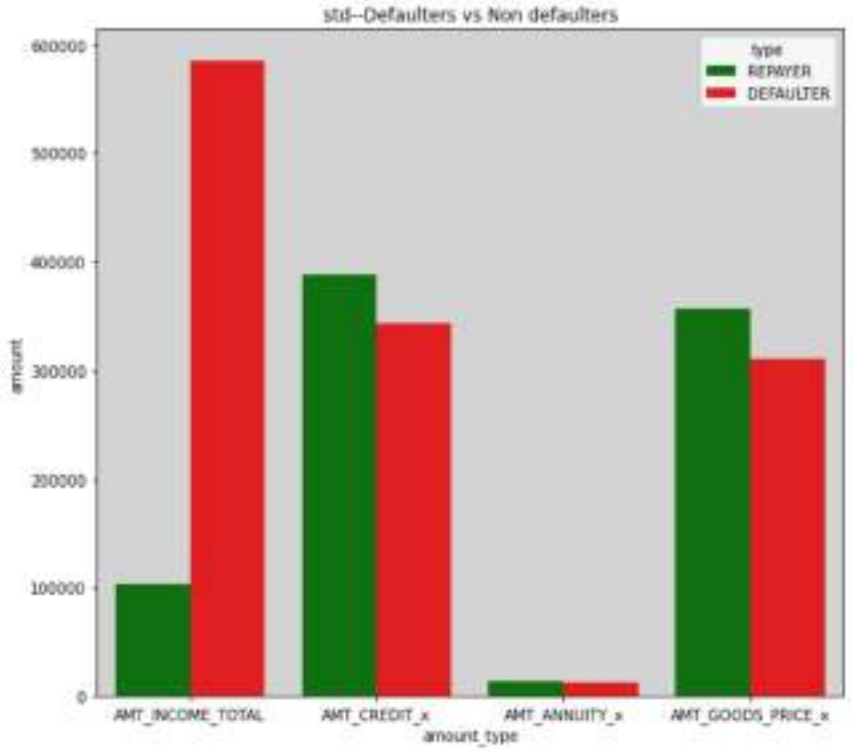


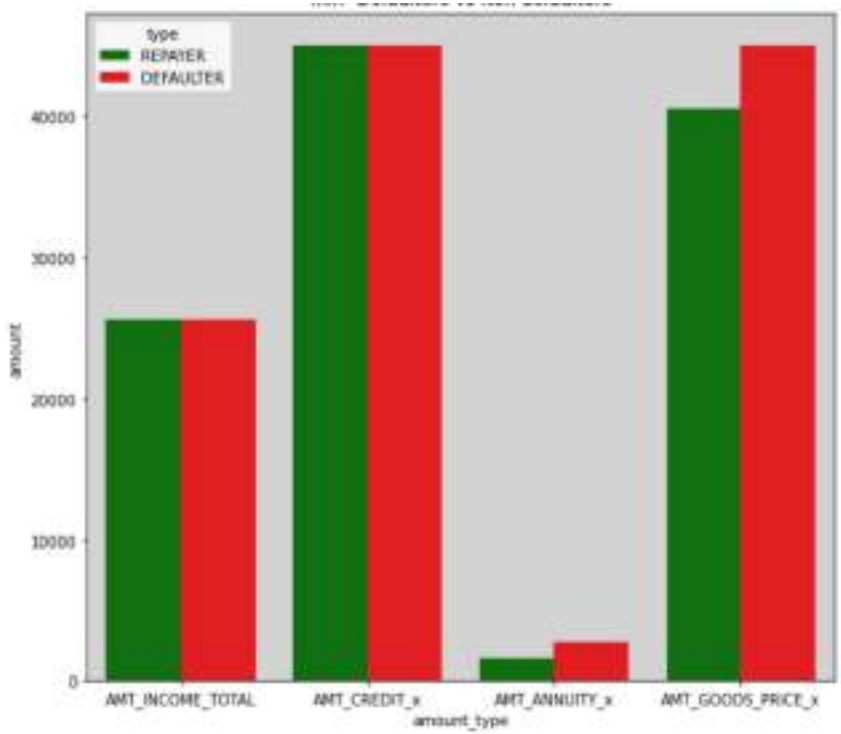
In [18]:

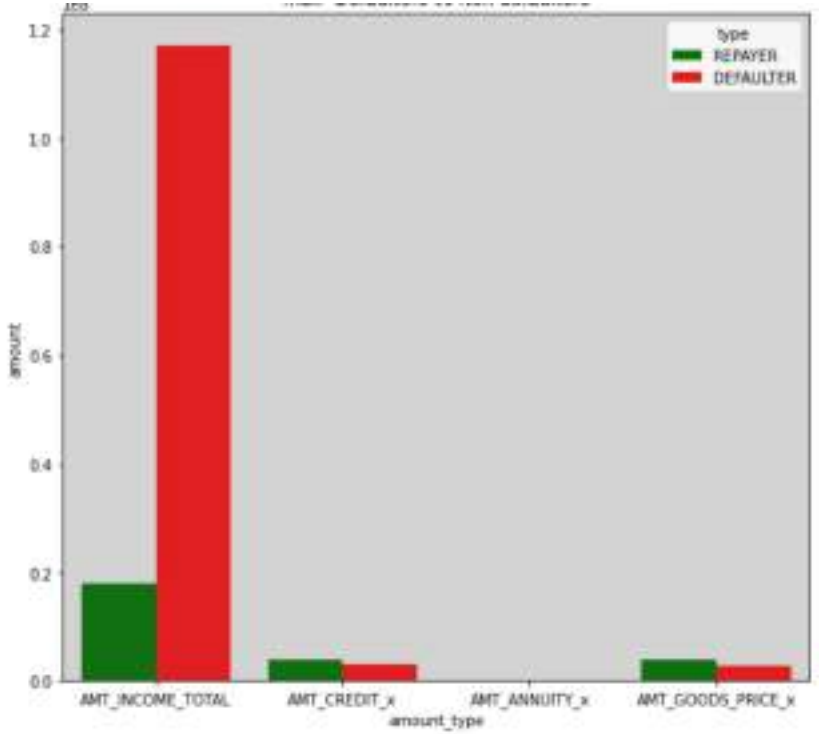
```
df = combined_df.groupby("TARGET")[cols].describe().transpose().reset_index()
df = df[df["level_1"].isin([ 'mean', 'std', 'min', 'max'])]
df_x = df[["level_0","level_1",0]]
df_y = df[["level_0","level_1",1]]
df_x = df_x.rename(columns={'level_0':"amount_type", 'level_1':"statistic", 0:"amount"})
df_x["type"] = "REPAYER"
df_y = df_y.rename(columns={'level_0':"amount_type", 'level_1':"statistic", 1:"amount"})
```

```
df_y["type"] = "DEFAULTER"
df_new = pd.concat([df_x,df_y],axis = 0)

stat = df_new["statistic"].unique().tolist()
length = len(stat)

plt.figure(figsize=(20,20))

for i,j in itertools.zip_longest(stat,range(length)):
    plt.subplot(2,2,j+1)
    fig = sns.barplot(df_new[df_new["statistic"] == i]["amount_type"],df_new[df_new["statistic"] == i]["amount"],
                hue=df_new[df_new["statistic"] == i]["type"],palette=["g","r"])
    plt.title(i + "--Defaulters vs Non defaulters")
    plt.subplots_adjust(hspace = .4)
    fig.set_facecolor("lightgrey")
```



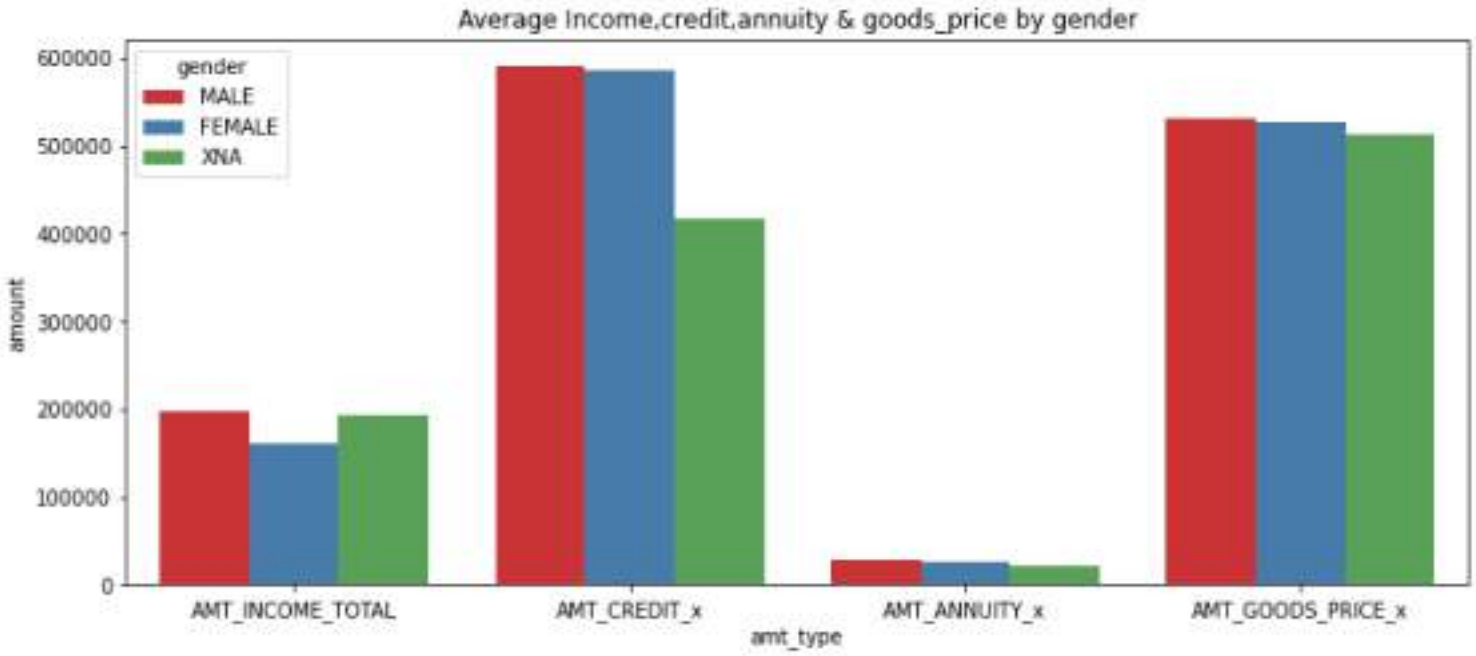






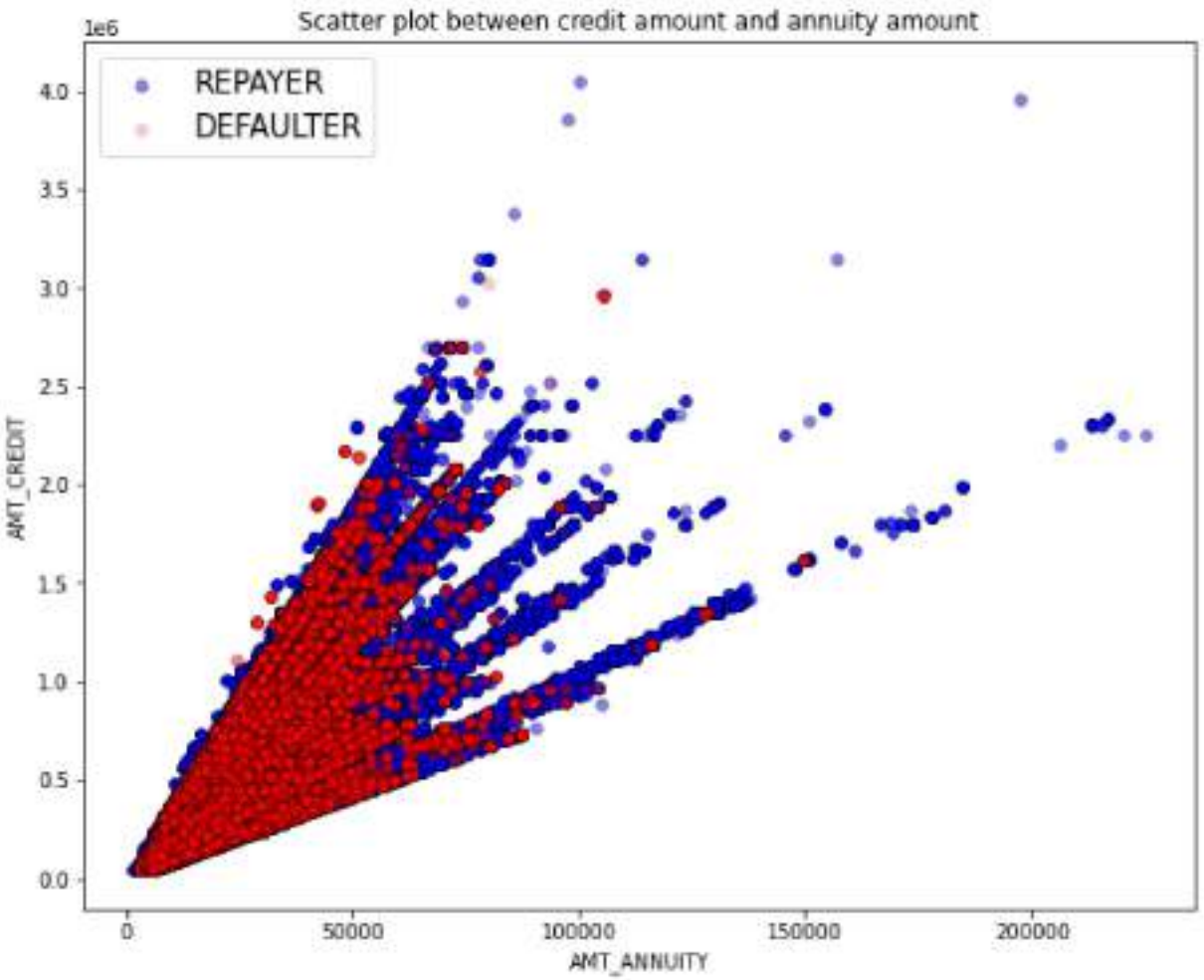
In [19]:

```
cols = [ 'AMT_INCOME_TOTAL', 'AMT_CREDIT_x','AMT_ANNUITY_x', 'AMT_GOODS_PRICE_x']

df1 = combined_df.groupby("CODE_GENDER")[cols].mean().transpose().reset_index()

df_f   = df1[["index","F"]]
df_f   = df_f.rename(columns={'index':"amt_type", 'F':"amount"})
df_f["gender"] = "FEMALE"
df_m   = df1[["index","M"]]
df_m   = df_m.rename(columns={'index':"amt_type", 'M':"amount"})
df_m["gender"] = "MALE"
df_xna = df1[["index","XNA"]]
df_xna = df_xna.rename(columns={'index':"amt_type", 'XNA':"amount"})
df_xna["gender"] = "XNA"

df_gen = pd.concat([df_m,df_f,df_xna],axis=0)

plt.figure(figsize=(12,5))
ax = sns.barplot("amt_type","amount",data=df_gen,hue="gender",palette="Set1")
plt.title("Average Income,credit,annuity & goods_price by gender")
plt.show()
```



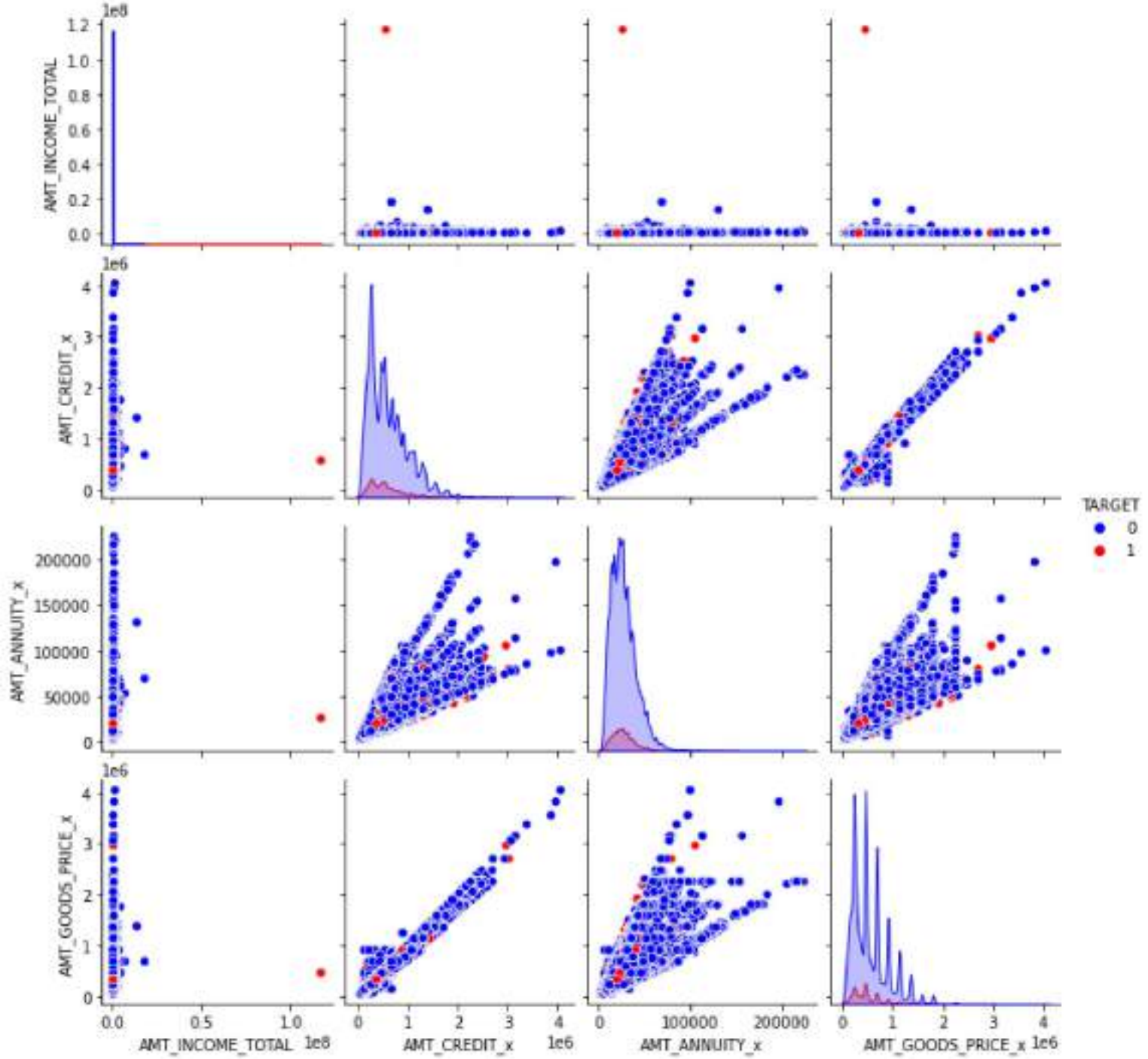
In [20]:

```
fig = plt.figure(figsize=(10,8))
plt.scatter(combined_df[combined_df["TARGET"]==0]['AMT_ANNUITY_x'],combined_df[combined_df["TARGET"]==0]['AMT_CREDIT_x'],
            color="b",alpha=.5,label="REPAYER",linewidth=.5,edgecolor="k")
plt.scatter(combined_df[combined_df["TARGET"]==1]['AMT_ANNUITY_x'],combined_df[combined_df["TARGET"]==1]['AMT_CREDIT_x'],
            color="r",alpha=.2,label="DEFAULTER",linewidth=.5,edgecolor="k")
plt.legend(loc="best",prop={"size":15})
plt.xlabel("AMT_ANNUITY")
plt.ylabel("AMT_CREDIT")
plt.title("Scatter plot between credit amount and annuity amount")
plt.show()
```

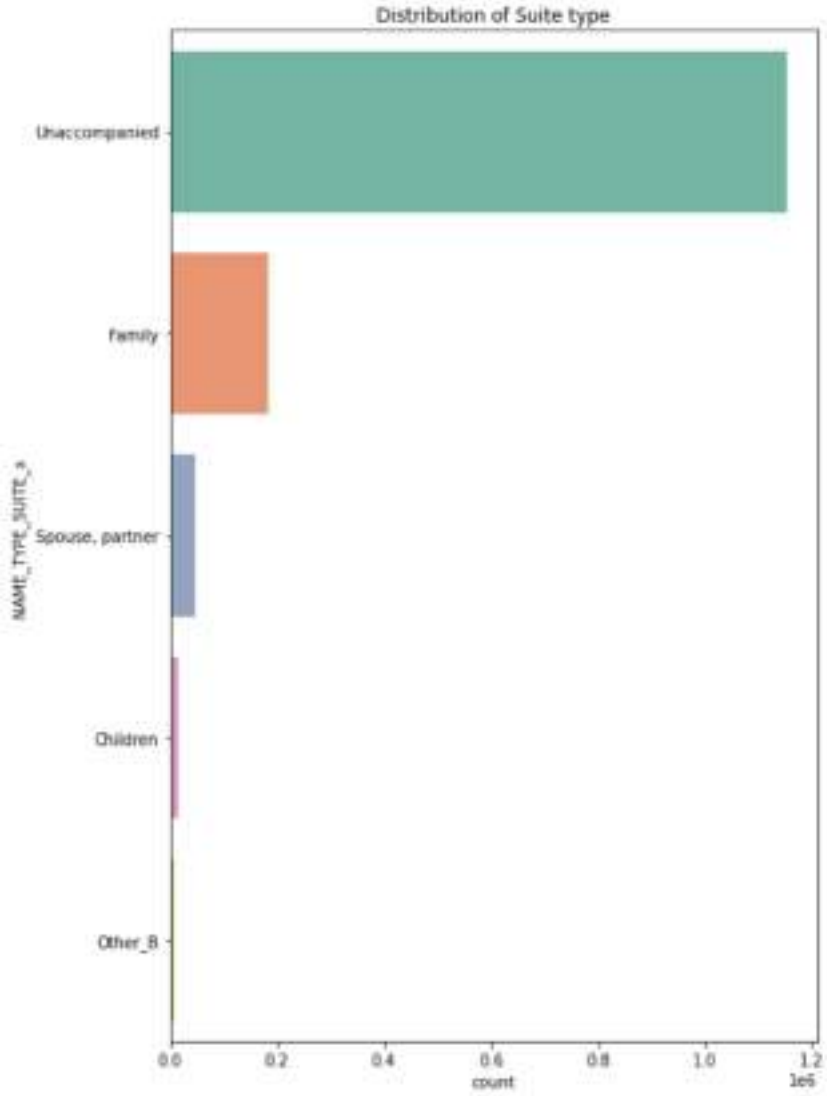


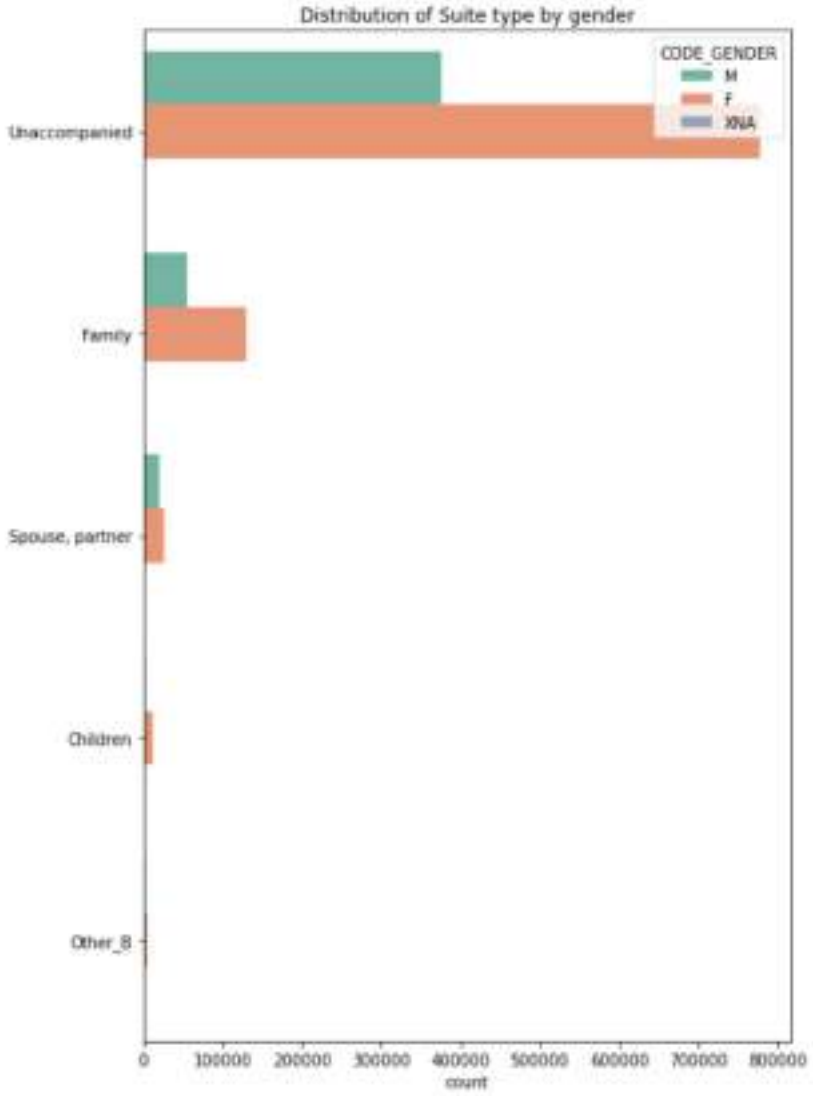
In [21]:

```
amt = combined_df[[ 'AMT_INCOME_TOTAL','AMT_CREDIT_x',
                    'AMT_ANNUITY_x', 'AMT_GOODS_PRICE_x',"TARGET"]]
amt = amt[(amt["AMT_GOODS_PRICE_x"].notnull()) & (amt["AMT_ANNUITY_x"].notnull())]
sns.pairplot(amt,hue="TARGET",palette=["b","r"])
plt.show()
```


1e8
AMT_INCOME_TOTAL
1e6
AMT_CREDIT_x
AMT_ANNUITY_x
1e6
AMT_GOODS_PRICE_x
AMT_INCOME_TOTAL
1e8
AMT_CREDIT_x
1e6
AMT_ANNUITY_x
AMT_GOODS_PRICE_x
1e6
TARGET
0
1

In [22]:
```
plt.figure(figsize=(18,12))
plt.subplot(121)
sns.countplot(y=combined_df["NAME_TYPE_SUITE_x"],
              palette="Set2",
              order=combined_df["NAME_TYPE_SUITE_x"].value_counts().index[:5])
plt.title("Distribution of Suite type")

plt.subplot(122)
sns.countplot(y=combined_df["NAME_TYPE_SUITE_x"],
              hue=combined_df["CODE_GENDER"],palette="Set2",
              order=combined_df["NAME_TYPE_SUITE_x"].value_counts().index[:5])
plt.ylabel("")
plt.title("Distribution of Suite type by gender")
plt.subplots_adjust(wspace = .4)
```

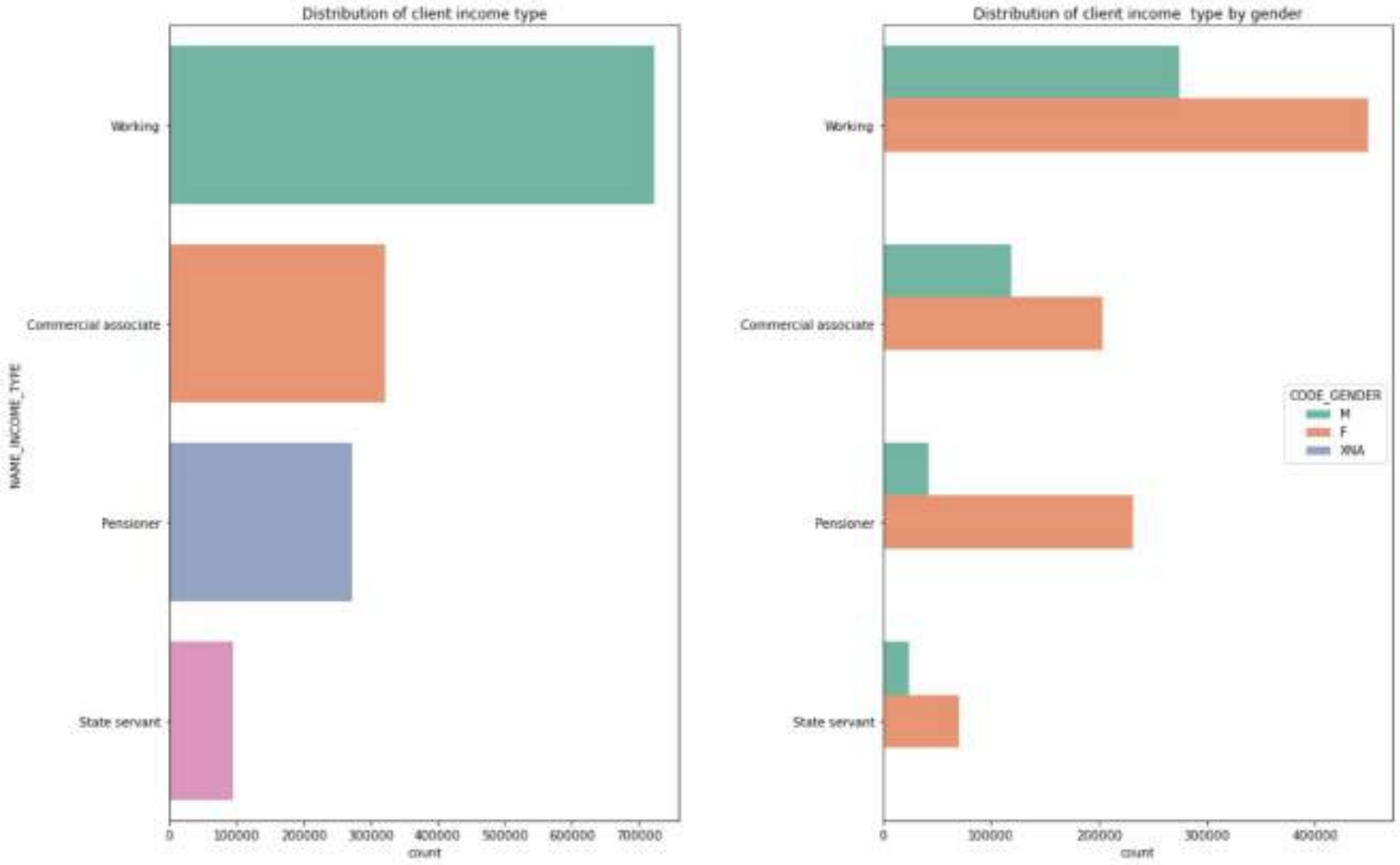




In [23]:

```
plt.figure(figsize=(18,12))
plt.subplot(121)
sns.countplot(y=combined_df["NAME_INCOME_TYPE"],
              palette="Set2",
              order=combined_df["NAME_INCOME_TYPE"].value_counts().index[:4])
plt.title("Distribution of client income type")

plt.subplot(122)
sns.countplot(y=combined_df["NAME_INCOME_TYPE"],
```

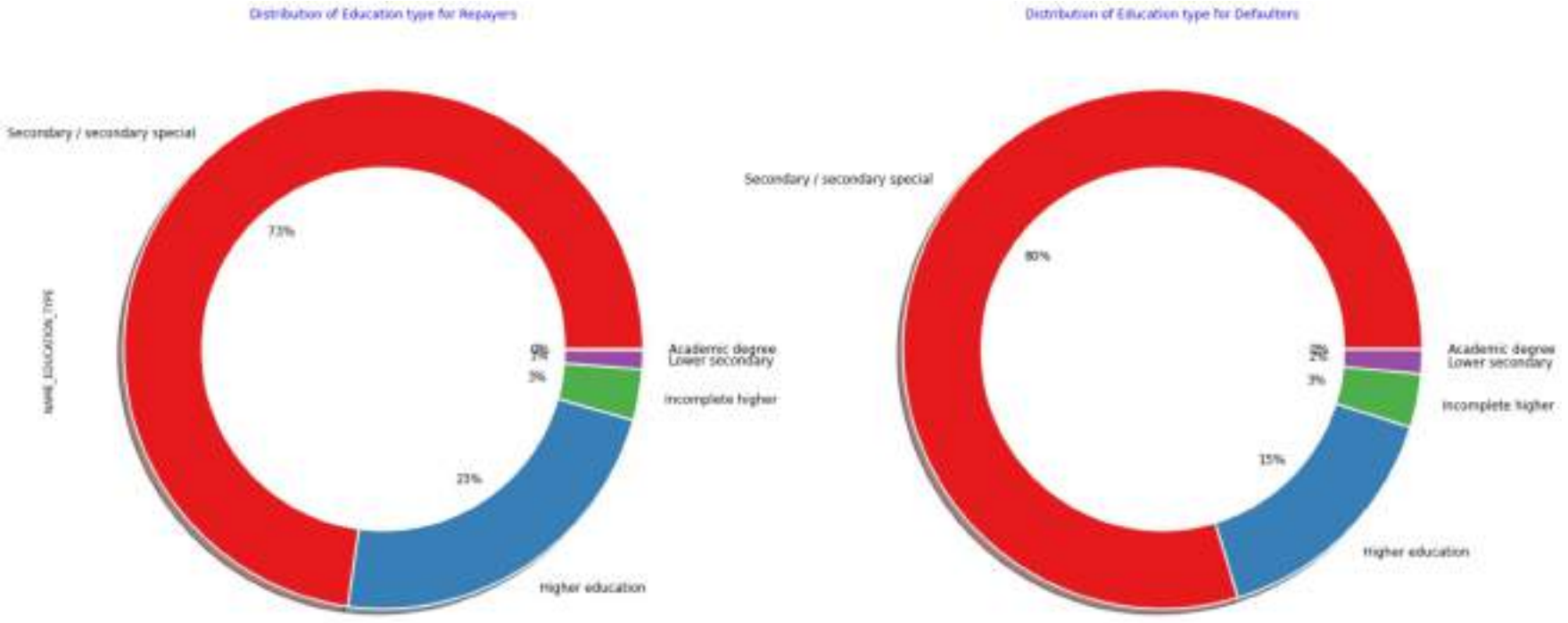
```
                hue=combined_df["CODE_GENDER"],
                palette="Set2",
                order=combined_df["NAME_INCOME_TYPE"].value_counts().index[:4])
plt.ylabel("")
plt.title("Distribution of client income  type by gender")
plt.subplots_adjust(wspace = .4)
```



In [24]:

```
plt.figure(figsize=(25,25))
plt.subplot(121)
combined_df[combined_df["TARGET"]==0]["NAME_EDUCATION_TYPE"].value_counts().plot.pie(fontsize=12,autopct = "%1.0f%%",
```

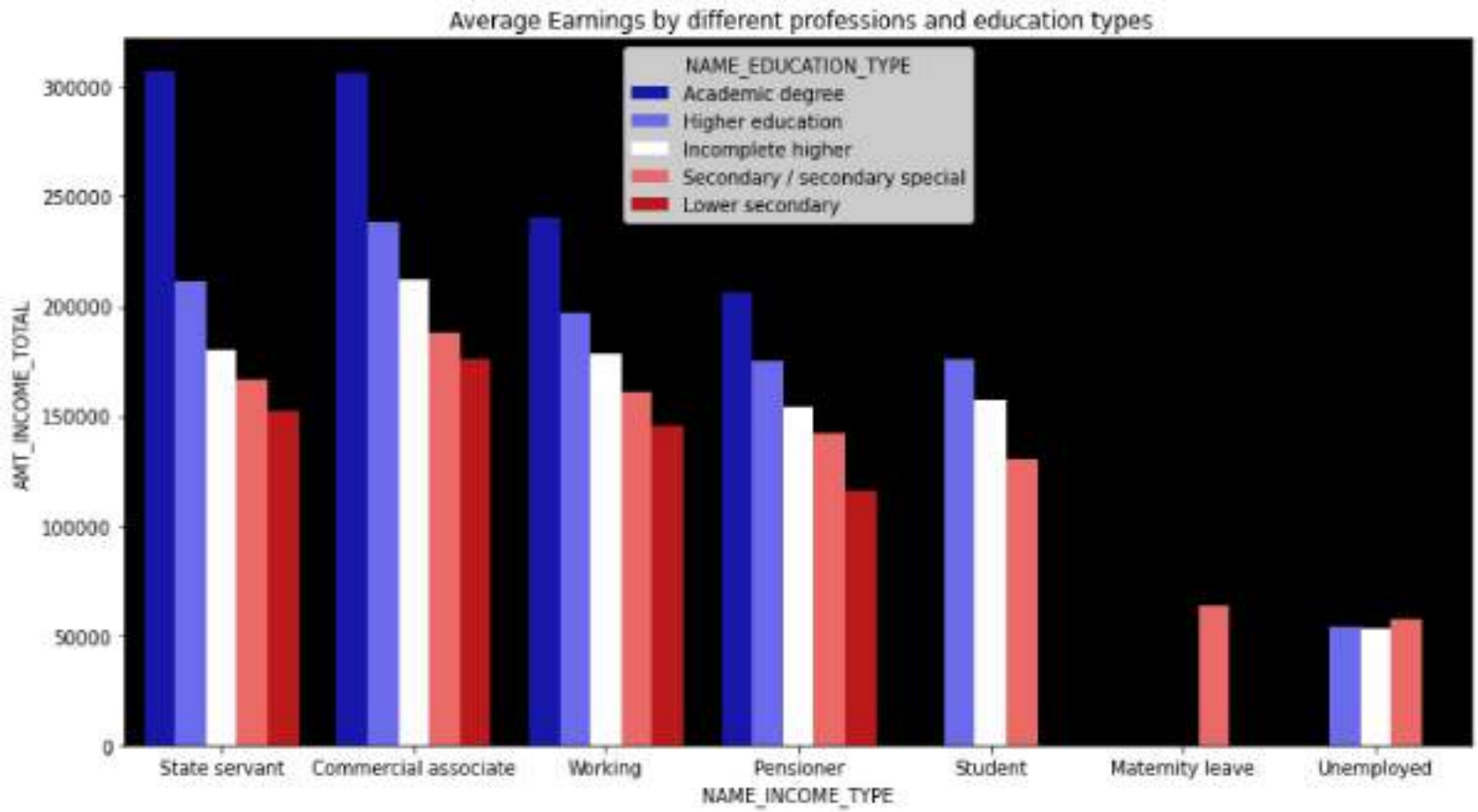
```
                                                                    colors = sns.color_palet
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)
circ = plt.Circle((0,0),.7,color="white")
plt.gca().add_artist(circ)
plt.title("Distribution of Education type for Repayers",color="b")

plt.subplot(122)
combined_df[combined_df["TARGET"]==1]["NAME_EDUCATION_TYPE"].value_counts().plot.pie(fontsize=12,autopct = "%1.0f%%",
                                                                    colors = sns.color_palet
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)
circ = plt.Circle((0,0),.7,color="white")
plt.gca().add_artist(circ)
plt.title("Distribution of Education type for Defaulters",color="b")
plt.ylabel("")
plt.show()
```



In [25]:

```
edu = combined_df.groupby(['NAME_EDUCATION_TYPE','NAME_INCOME_TYPE'])['AMT_INCOME_TOTAL'].mean().reset_index().sort_value
fig = plt.figure(figsize=(13,7))
ax = sns.barplot('NAME_INCOME_TYPE','AMT_INCOME_TOTAL',data=edu,hue='NAME_EDUCATION_TYPE',palette="seismic")
ax.set_facecolor("k")
```

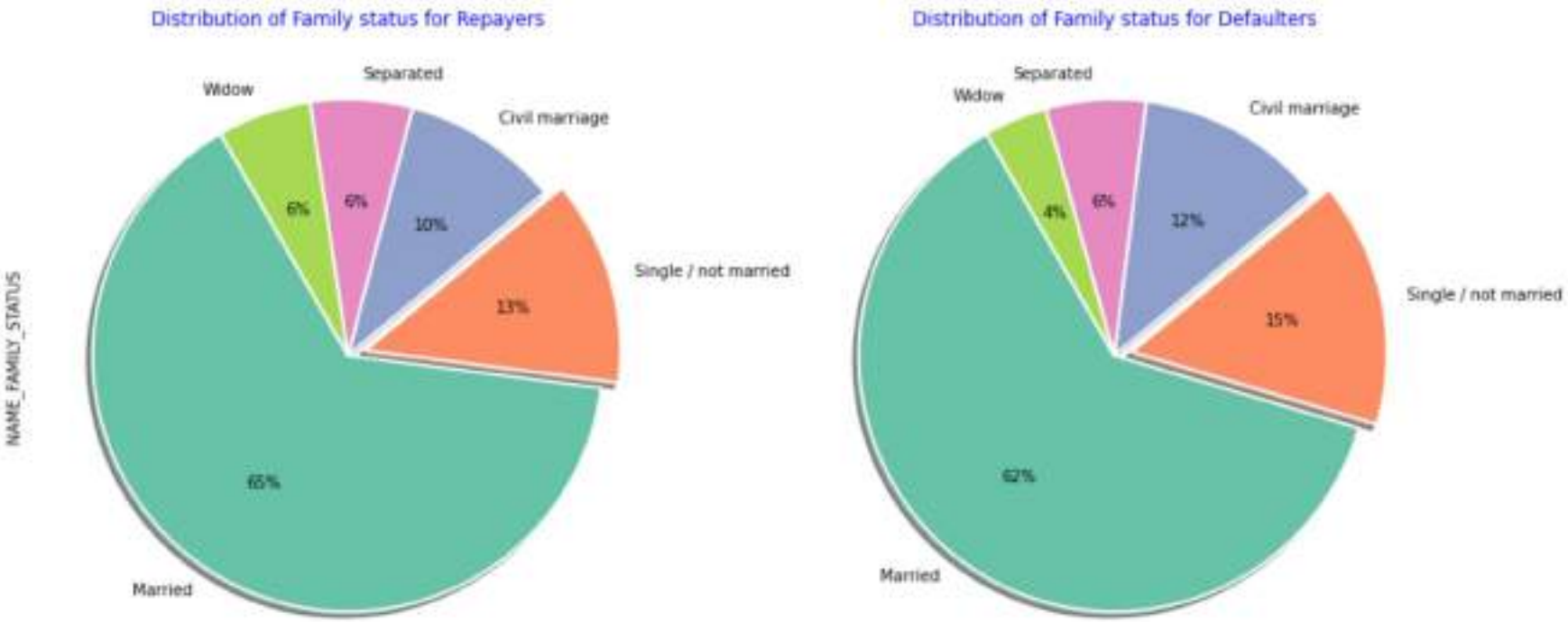
```
plt.title(" Average Earnings by different professions and education types")
plt.show()
```



In [26]:

```
plt.figure(figsize=(16,8))
plt.subplot(121)
combined_df[combined_df["TARGET"]==0]["NAME_FAMILY_STATUS"].value_counts().plot.pie(autopct = "%1.0f%%",
                                                                                    startangle=120,colors = sns.color_palette("Set2",7),
                                                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True,explode=[0,.07,

plt.title("Distribution of Family status for Repayers",color="b")

plt.subplot(122)
combined_df[combined_df["TARGET"]==1]["NAME_FAMILY_STATUS"].value_counts().plot.pie(autopct = "%1.0f%%",
                                                                                    startangle=120,colors = sns.color_palette("Set2",7),
                                                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True,explode=[0,.07,
```

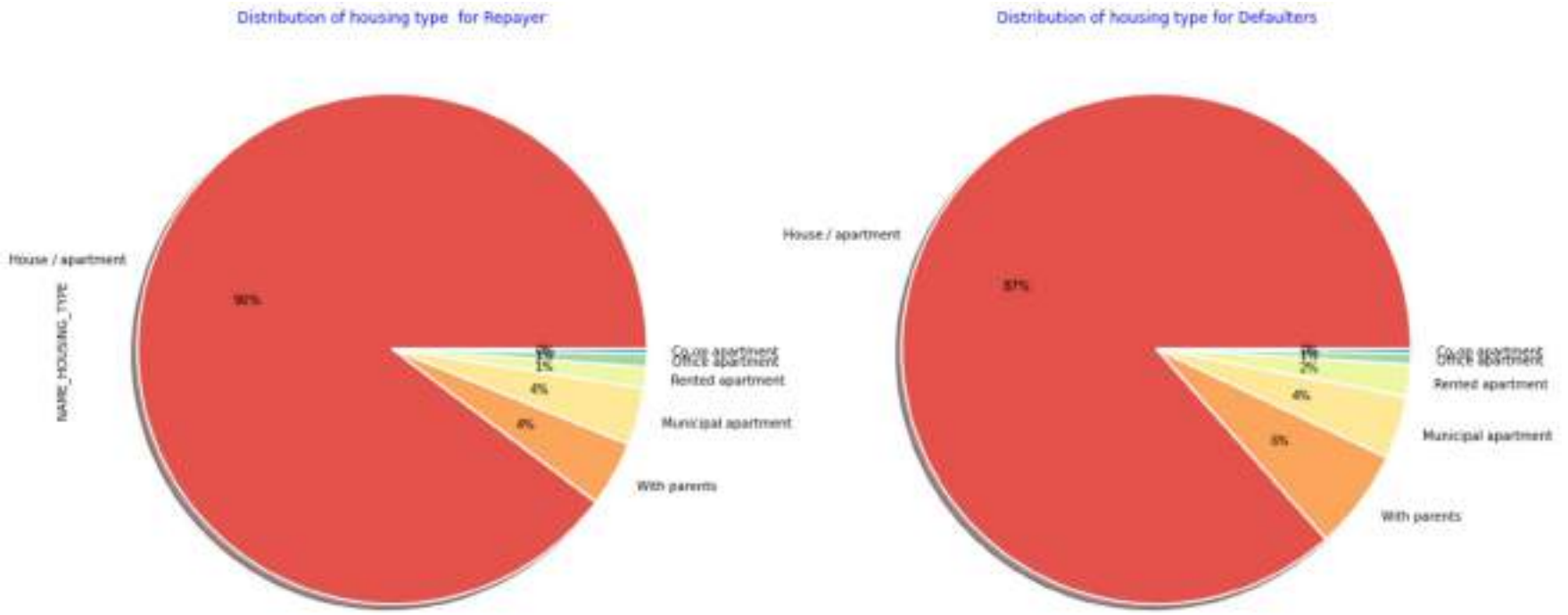
```
plt.title("Distribution of Family status for Defaulters",color="b")
plt.ylabel("")
plt.show()
```



In [27]:

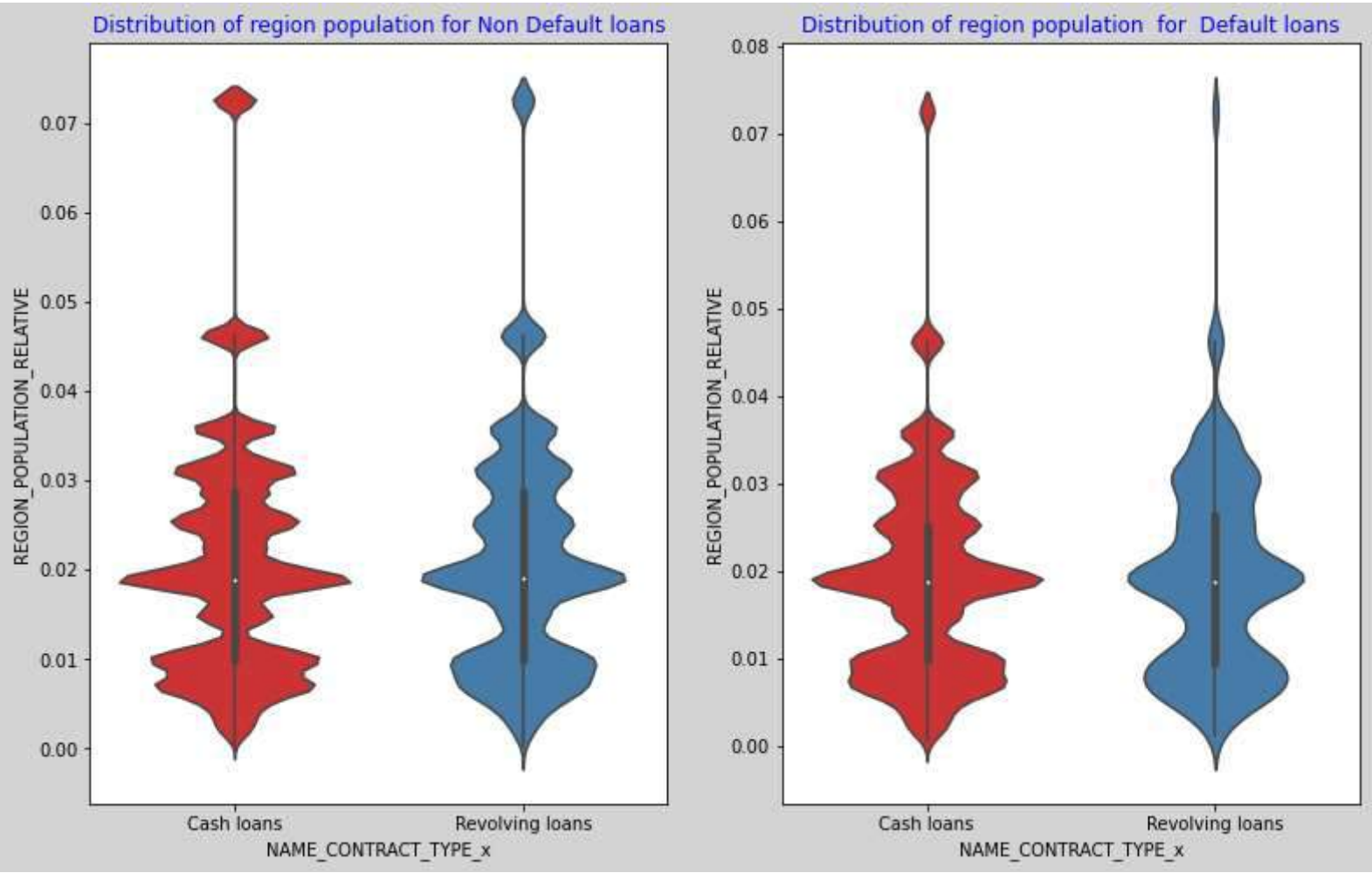
```
plt.figure(figsize=(20,20))
plt.subplot(121)
combined_df[combined_df["TARGET"]==0]["NAME_HOUSING_TYPE"].value_counts().plot.pie(autopct = "%1.0f%%",fontsize=10,
                                                    colors = sns.color_palette("Spectral"),
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)

plt.title("Distribution of housing type  for Repayer",color="b")

plt.subplot(122)
combined_df[combined_df["TARGET"]==1]["NAME_HOUSING_TYPE"].value_counts().plot.pie(autopct = "%1.0f%%",fontsize=10,
                                                colors = sns.color_palette("Spectral"),
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)


plt.title("Distribution of housing type for Defaulters",color="b")
```

```
plt.ylabel("")
plt.show()
```



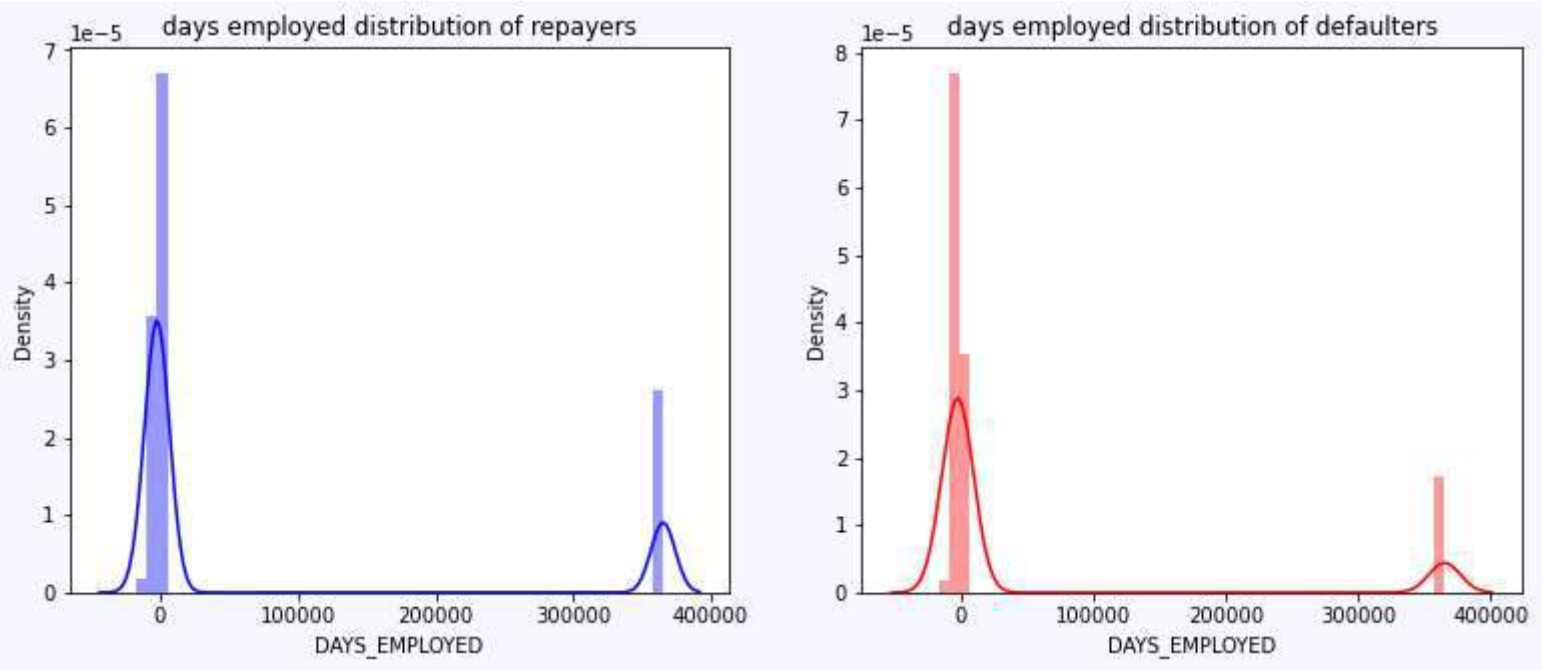
In [28]:

```
fig = plt.figure(figsize=(13,8))

plt.subplot(121)
sns.violinplot(y=combined_df[combined_df["TARGET"]==0]["REGION_POPULATION_RELATIVE"]
               ,x=combined_df[combined_df["TARGET"]==0]["NAME_CONTRACT_TYPE_x"],
               palette="Set1")
plt.title("Distribution of region population for Non Default loans",color="b")
plt.subplot(122)
sns.violinplot(y = combined_df[combined_df["TARGET"]==1]["REGION_POPULATION_RELATIVE"]
               ,x=combined_df[combined_df["TARGET"]==1]["NAME_CONTRACT_TYPE_x"]
               ,palette="Set1")
plt.title("Distribution of region population  for  Default loans",color="b")

plt.subplots_adjust(wspace = .2)
fig.set_facecolor("lightgrey")
```



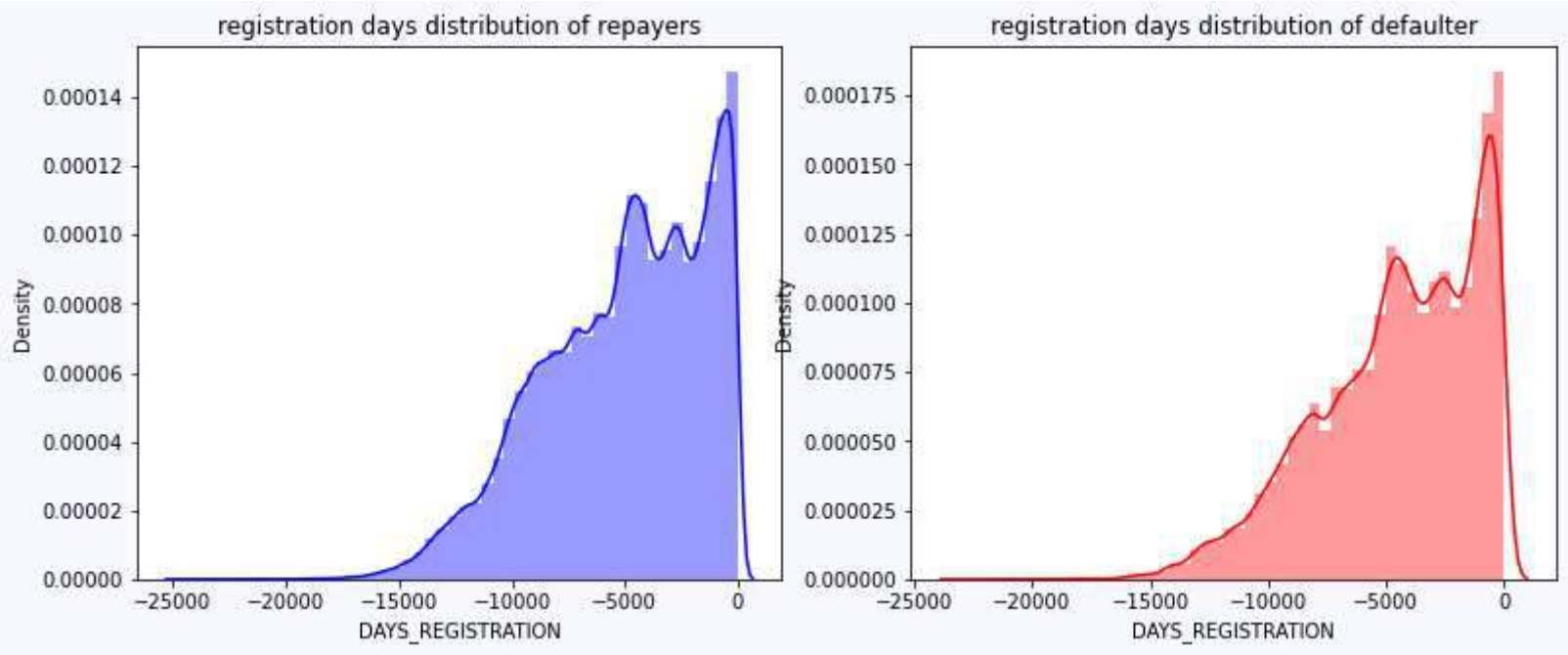
In [30]:

```
fig = plt.figure(figsize=(13,5))

plt.subplot(121)
sns.distplot(combined_df[combined_df["TARGET"]==0]["DAYS_EMPLOYED"],color="b")
plt.title("days employed distribution of repayers")

plt.subplot(122)
sns.distplot(combined_df[combined_df["TARGET"]==1]["DAYS_EMPLOYED"],color="r")
plt.title("days employed distribution of defaulters")

fig.set_facecolor("ghostwhite")
```



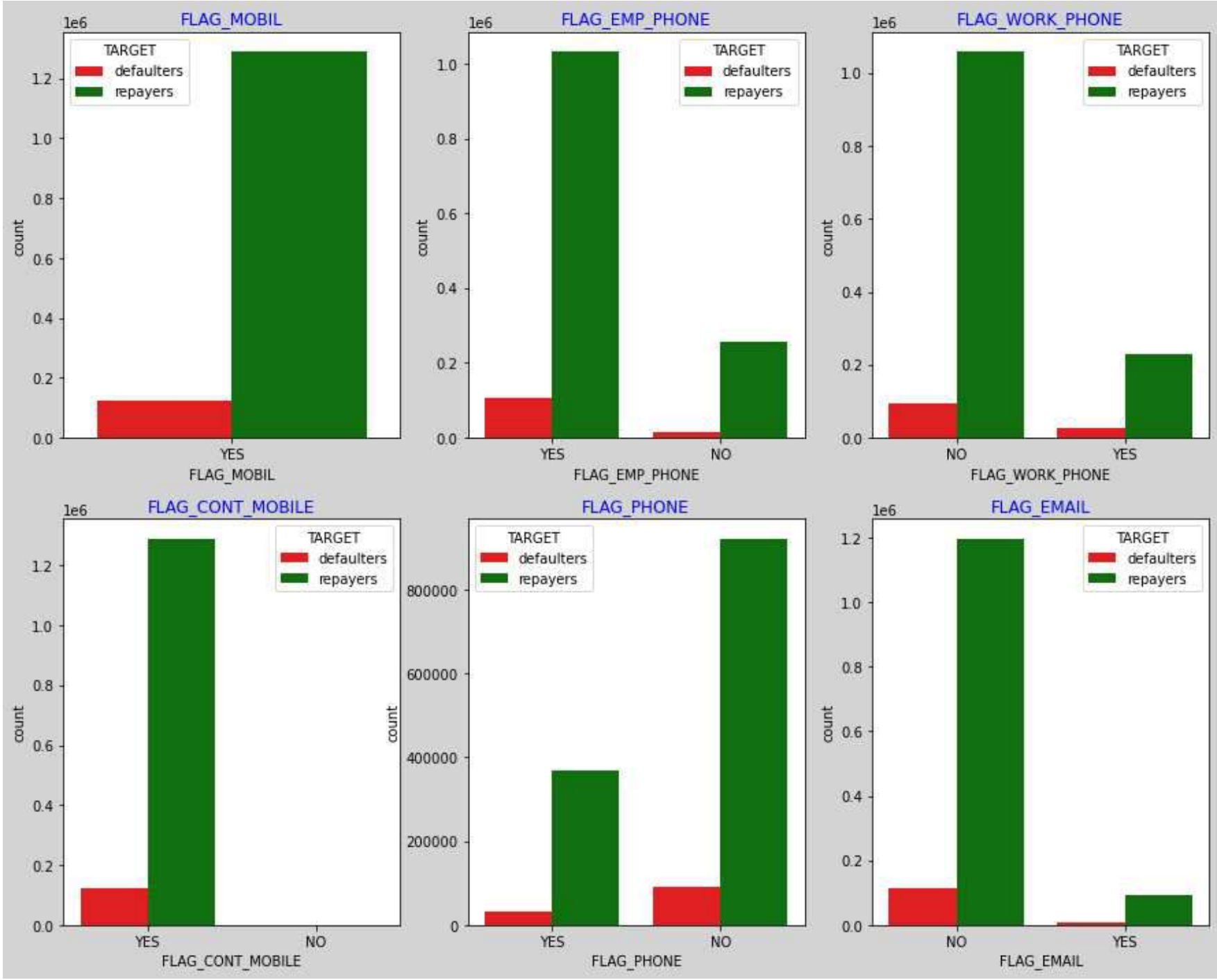
In [31]:

```
fig = plt.figure(figsize=(13,5))

plt.subplot(121)
sns.distplot(combined_df[combined_df["TARGET"]==0]["DAYS_REGISTRATION"],color="b")
plt.title("registration days distribution of repayers")

plt.subplot(122)
sns.distplot(combined_df[combined_df["TARGET"]==1]["DAYS_REGISTRATION"],color="r")
plt.title("registration days distribution of defaulter")

fig.set_facecolor("ghostwhite")
```



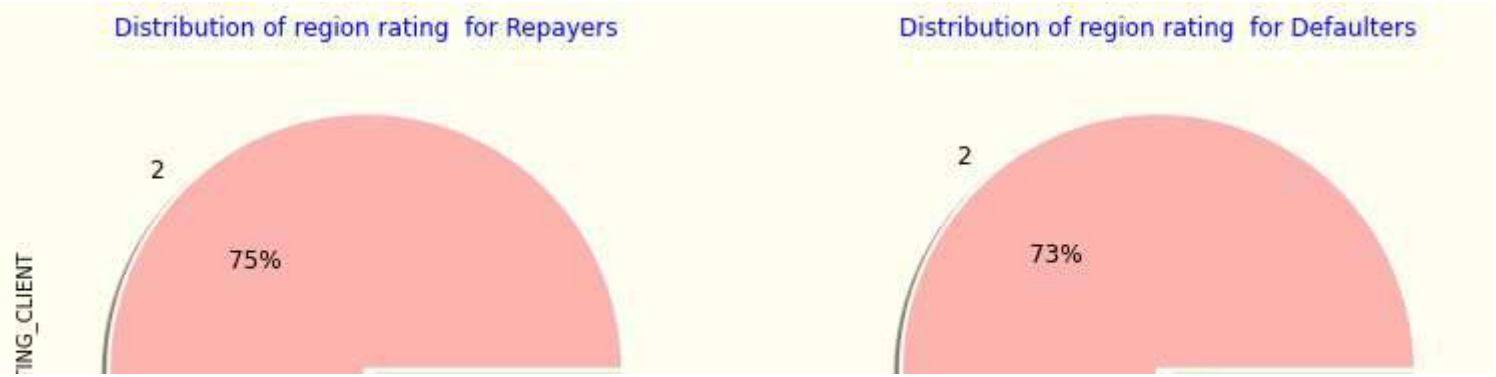
In [32]:

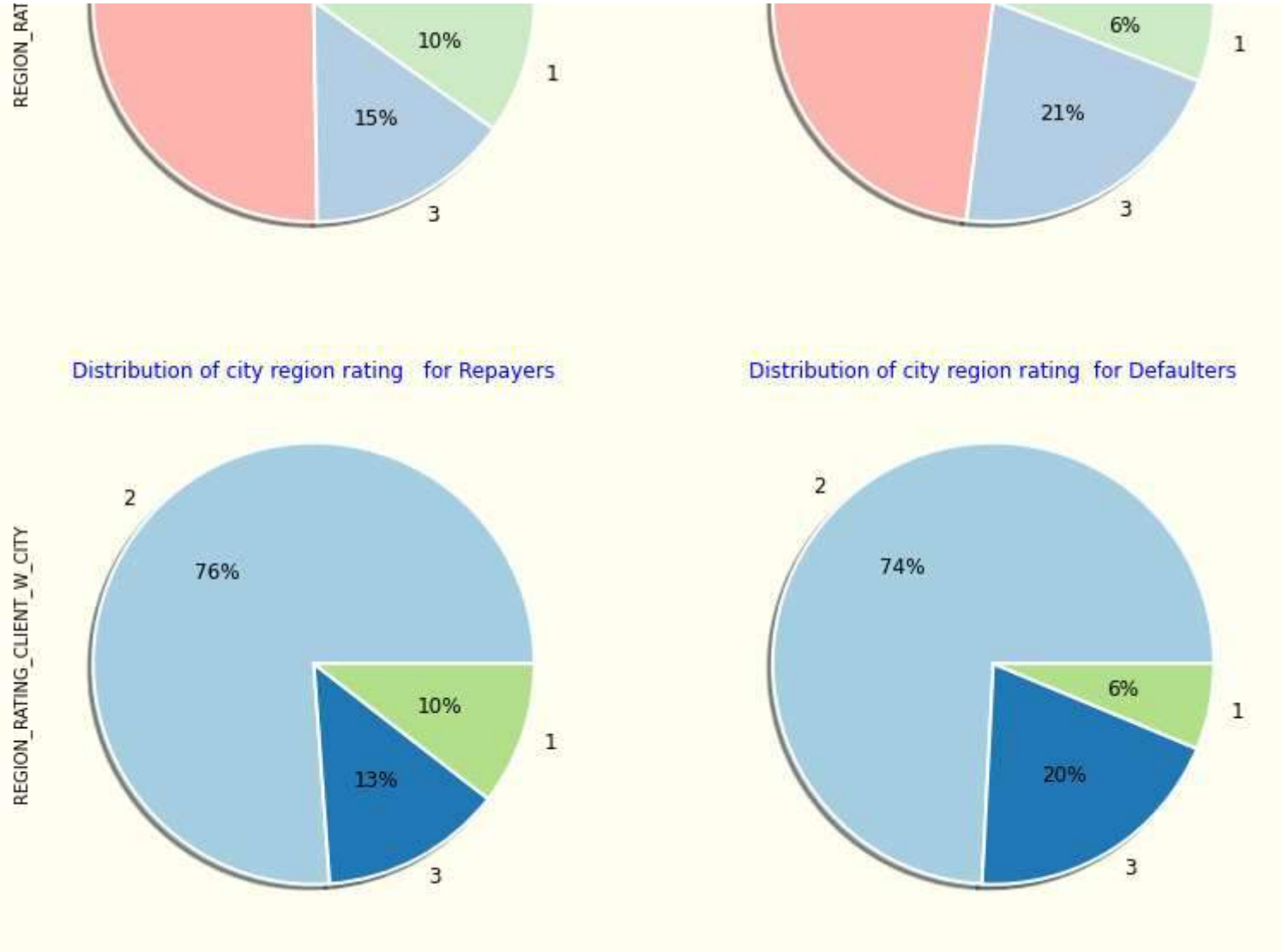
```
x   = combined_df[['FLAG_MOBIL', 'FLAG_EMP_PHONE', 'FLAG_WORK_PHONE', 'FLAG_CONT_MOBILE',
       'FLAG_PHONE', 'FLAG_EMAIL',"TARGET"]]
x["TARGET"] = x["TARGET"].replace({0:"repayers",1:"defaulters"})
x  = x.replace({1:"YES",0:"NO"})

cols = ['FLAG_MOBIL', 'FLAG_EMP_PHONE', 'FLAG_WORK_PHONE', 'FLAG_CONT_MOBILE',
       'FLAG_PHONE', 'FLAG_EMAIL']
length = len(cols)

fig = plt.figure(figsize=(15,12))
fig.set_facecolor("lightgrey")

for i,j in itertools.zip_longest(cols,range(length)):
    plt.subplot(2,3,j+1)
    sns.countplot(x[i],hue=x["TARGET"],palette=["r","g"])
    plt.title(i,color="b")
```


FLAG_MOBIL
1e6
TARGET
defaulters
repayers
count
YES
FLAG_MOBIL
FLAG_EMP_PHONE
1e6
TARGET
defaulters
repayers
count
YES
NO
FLAG_EMP_PHONE
FLAG_WORK_PHONE
1e6
TARGET
defaulters
repayers
count
NO
YES
FLAG_WORK_PHONE
FLAG_CONT_MOBILE
1e6
TARGET
defaulters
repayers
count
YES
NO
FLAG_CONT_MOBILE
FLAG_PHONE
TARGET
defaulters
repayers
count
YES
NO
FLAG_PHONE
FLAG_EMAIL
1e6
TARGET
defaulters
repayers
count
NO
YES
FLAG_EMAIL

In [33]:

```
fig = plt.figure(figsize=(13,13))
plt.subplot(221)
combined_df[combined_df["TARGET"]==0]["REGION_RATING_CLIENT"].value_counts().plot.pie(autopct = "%1.0f%%",fontsize=12,
                                                        colors = sns.color_palette("Pastel1"),
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)

plt.title("Distribution of region rating  for Repayers",color="b")

plt.subplot(222)
combined_df[combined_df["TARGET"]==1]["REGION_RATING_CLIENT"].value_counts().plot.pie(autopct = "%1.0f%%",fontsize=12,
                                                      colors = sns.color_palette("Pastel1"),
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)


plt.title("Distribution of region rating  for Defaulters",color="b")
plt.ylabel("")

plt.subplot(223)
combined_df[combined_df["TARGET"]==0]["REGION_RATING_CLIENT_W_CITY"].value_counts().plot.pie(autopct = "%1.0f%%",fontsize
                                                        colors = sns.color_palette("Paired"),
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)

plt.title("Distribution of city region rating   for Repayers",color="b")

plt.subplot(224)
combined_df[combined_df["TARGET"]==1]["REGION_RATING_CLIENT_W_CITY"].value_counts().plot.pie(autopct = "%1.0f%%",fontsize
                                                      colors = sns.color_palette("Paired"),
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)


plt.title("Distribution of city region rating  for Defaulters",color="b")
plt.ylabel("")
fig.set_facecolor("ivory")
```

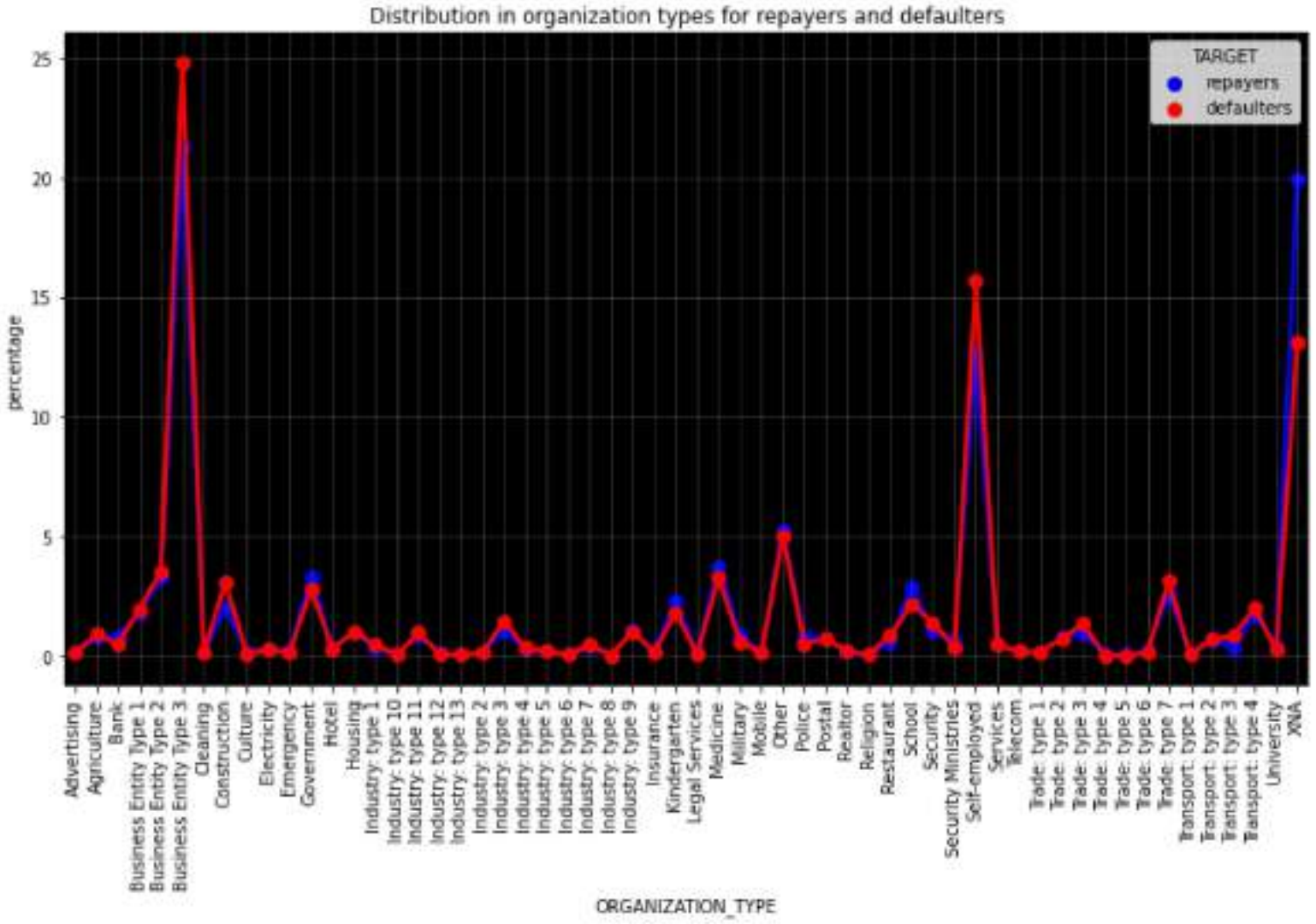




In [34]:

```
org = combined_df.groupby("TARGET").agg({"ORGANIZATION_TYPE":"value_counts"})
org = org.rename(columns = {"ORGANIZATION_TYPE":"value_counts"}).reset_index()
org_0 = org[org["TARGET"] == 0]
org_1 = org[org["TARGET"] == 1]
org_0["percentage"] = org_0["value_counts"]*100/org_0["value_counts"].sum()
org_1["percentage"] = org_1["value_counts"]*100/org_1["value_counts"].sum()

organization = pd.concat([org_0,org_1],axis=0)
organization = organization.sort_values(by="ORGANIZATION_TYPE",ascending=True)
```

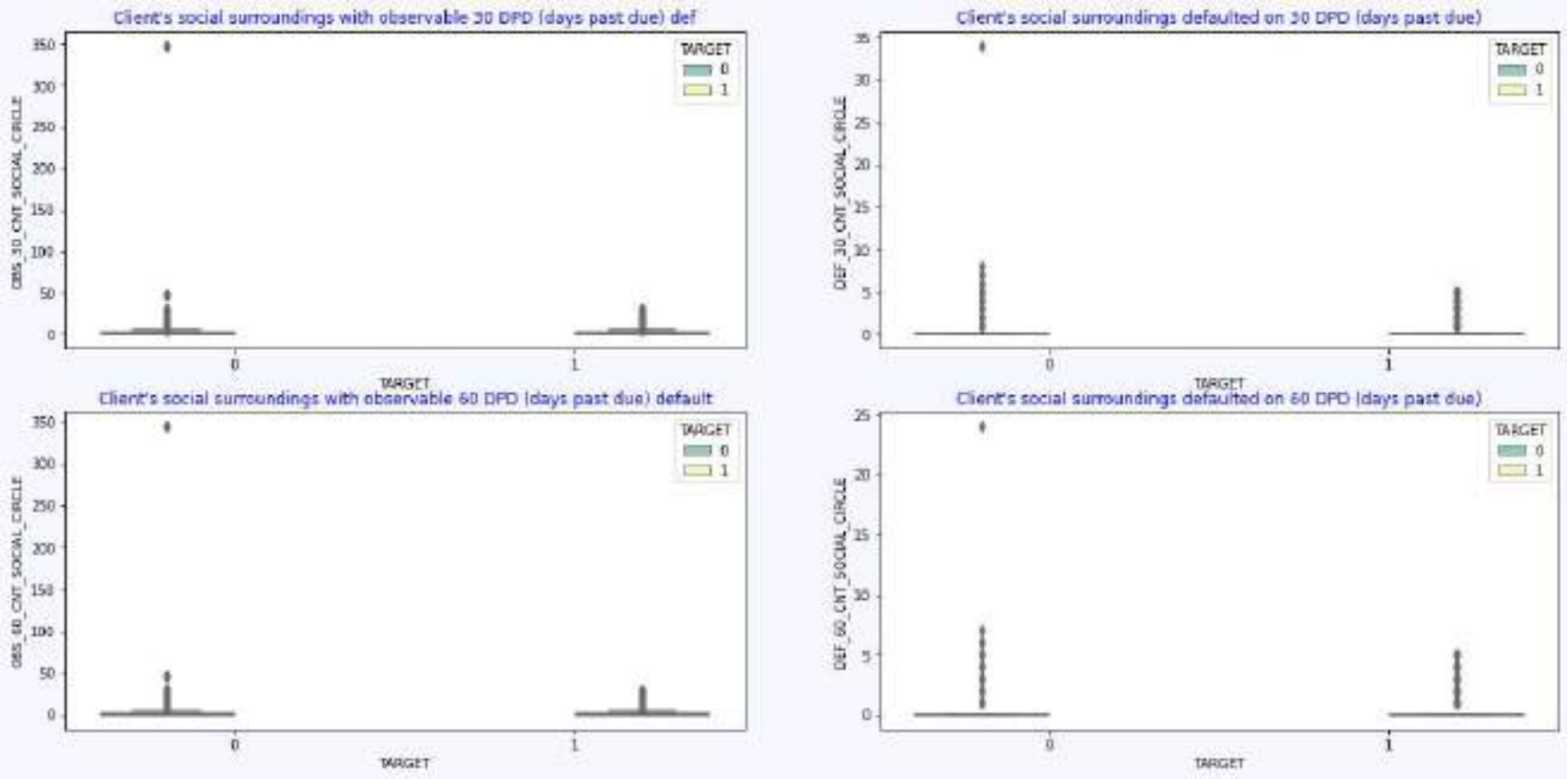
```
organization["TARGET"] = organization["TARGET"].replace({0:"repayers",1:"defaulters"})

organization
plt.figure(figsize=(13,7))
ax = sns.pointplot("ORGANIZATION_TYPE","percentage",
                   data=organization,hue="TARGET",palette=["b","r"])
plt.xticks(rotation=90)
plt.grid(True,alpha=.3)
ax.set_facecolor("k")
ax.set_title("Distribution in organization types for repayers and defaulters")
plt.show()
```



In [35]:

```
fig = plt.figure(figsize=(20,20))
plt.subplot(421)
sns.boxplot(data=combined_df,x='TARGET',y='OBS_30_CNT_SOCIAL_CIRCLE',
            hue="TARGET", palette="Set3")
plt.title("Client's social surroundings with observable 30 DPD (days past due) def",color="b")
plt.subplot(422)
sns.boxplot(data=combined_df,x='TARGET',y='DEF_30_CNT_SOCIAL_CIRCLE',
            hue="TARGET", palette="Set3")
plt.title("Client's social surroundings defaulted on 30 DPD (days past due)",color="b")
plt.subplot(423)
```

```
sns.boxplot(data=combined_df,x='TARGET',y='OBS_60_CNT_SOCIAL_CIRCLE',
            hue="TARGET", palette="Set3")
plt.title("Client's social surroundings with observable 60 DPD (days past due) default",color="b")
plt.subplot(424)
sns.boxplot(data=combined_df,x='TARGET',y='DEF_60_CNT_SOCIAL_CIRCLE',
            hue="TARGET", palette="Set3")
plt.title("Client's social surroundings defaulted on 60 DPD (days past due)",color="b")
fig.set_facecolor("ghostwhite")
```

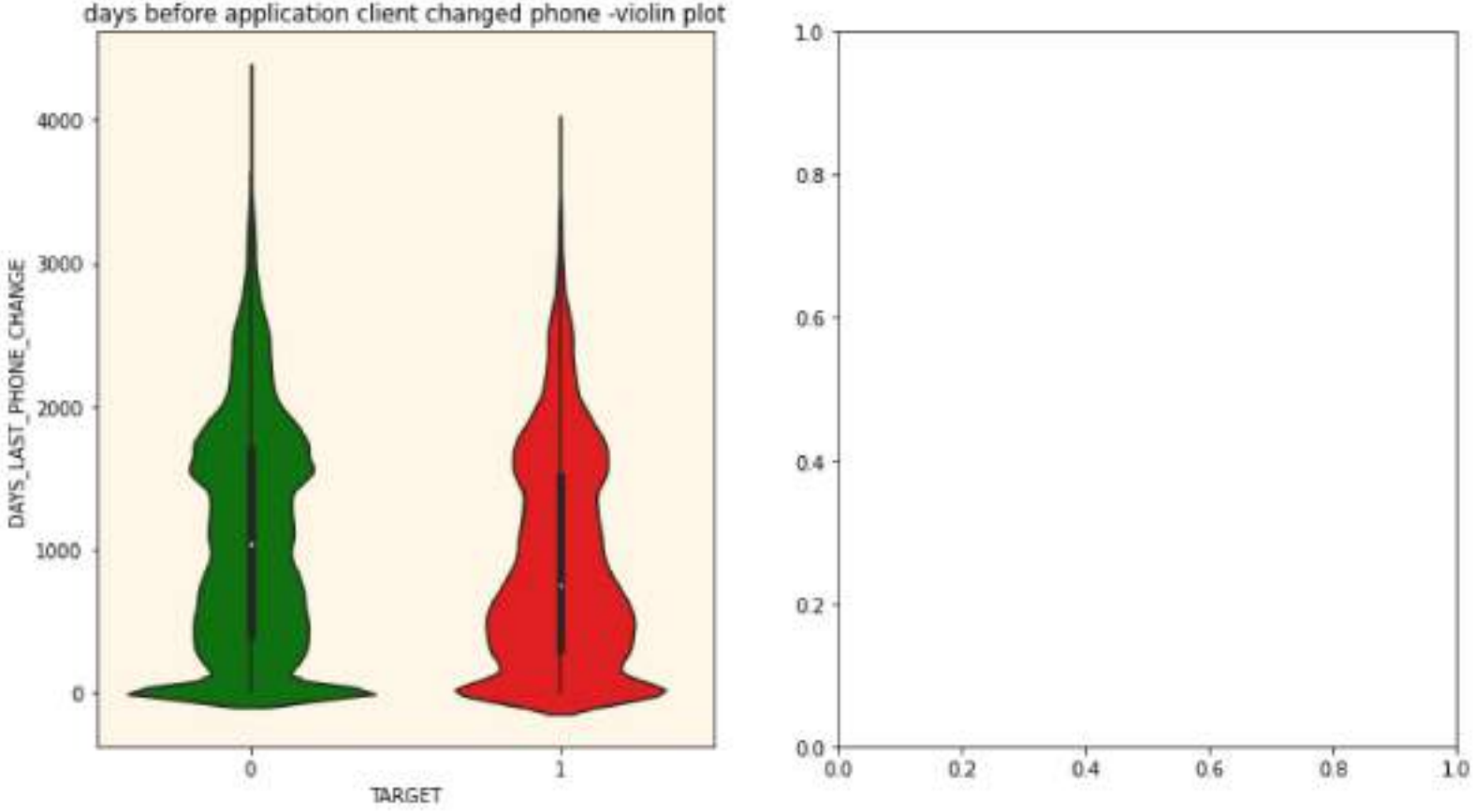


In [36]:

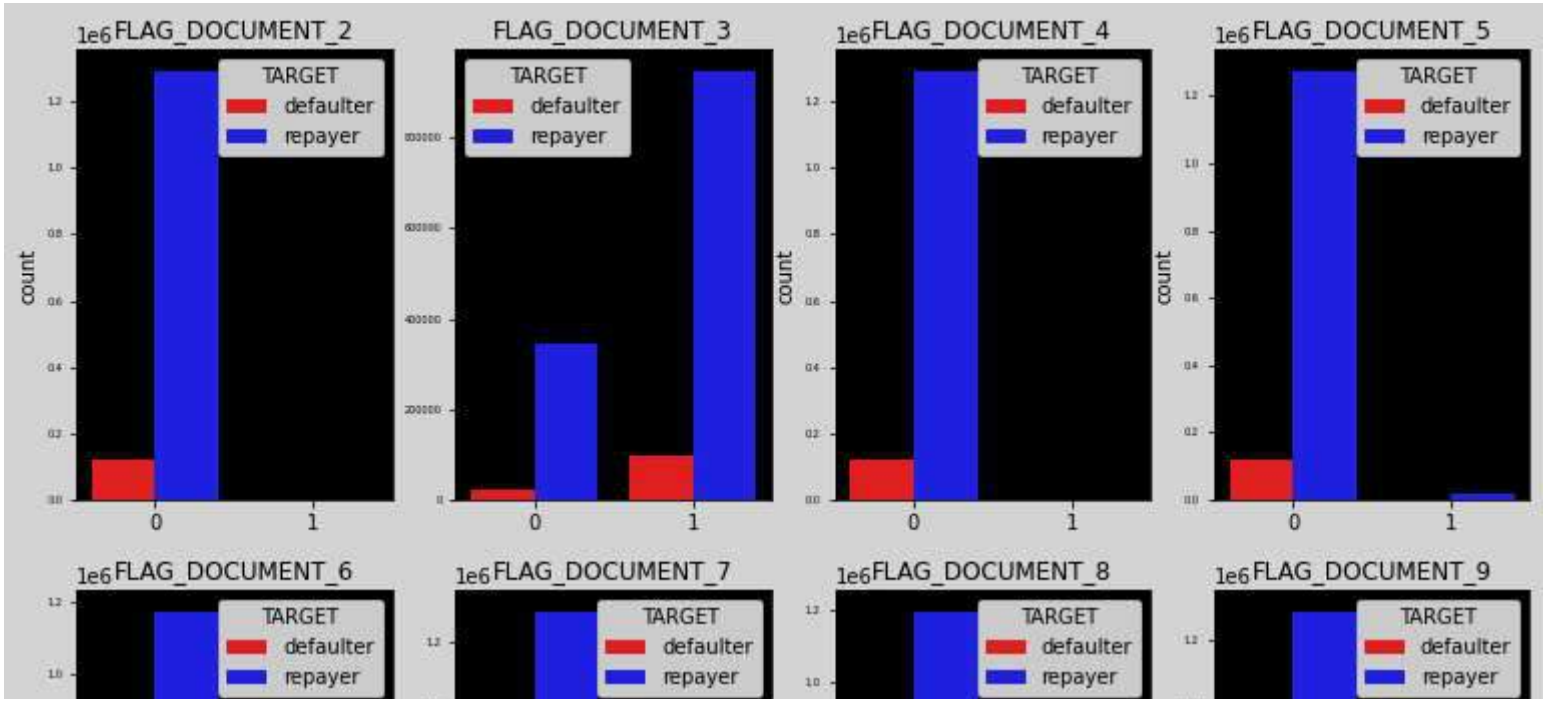
```
plt.figure(figsize=(13,7))
plt.subplot(121)
ax = sns.violinplot(combined_df["TARGET"],
                    combined_df["DAYS_LAST_PHONE_CHANGE"],palette=["g","r"])
ax.set_facecolor("oldlace")
ax.set_title("days before application client changed phone -violin plot")
plt.subplot(122)
```

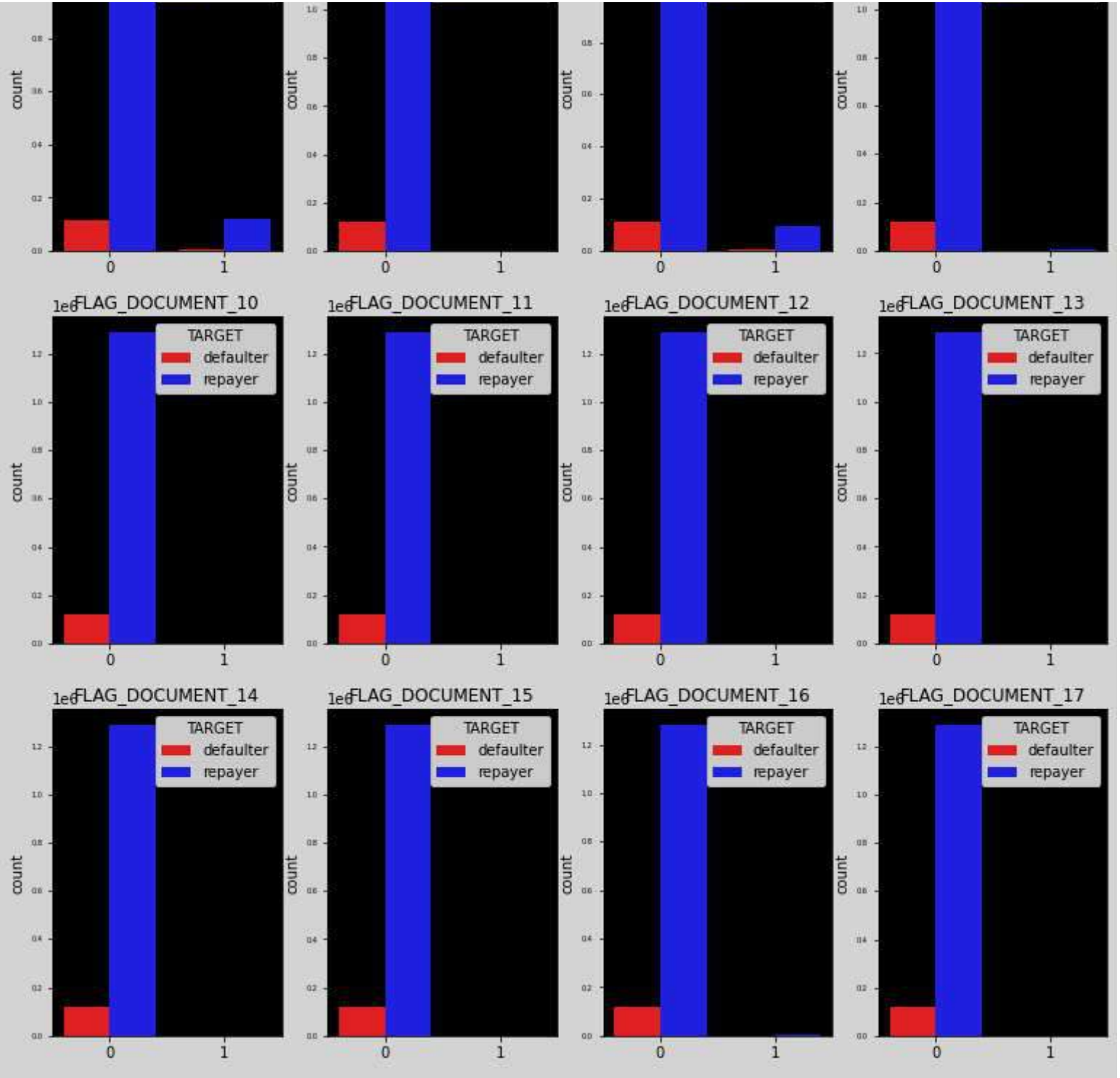
```
ax1 = sns.lvplot(combined_df["TARGET"],
                 combined_df["DAYS_LAST_PHONE_CHANGE"],palette=["g","r"])
ax1.set_facecolor("oldlace")
ax1.set_ylabel("")
ax1.set_title("days before application client changed phone -box plot")
plt.subplots_adjust(wspace = .2)
```

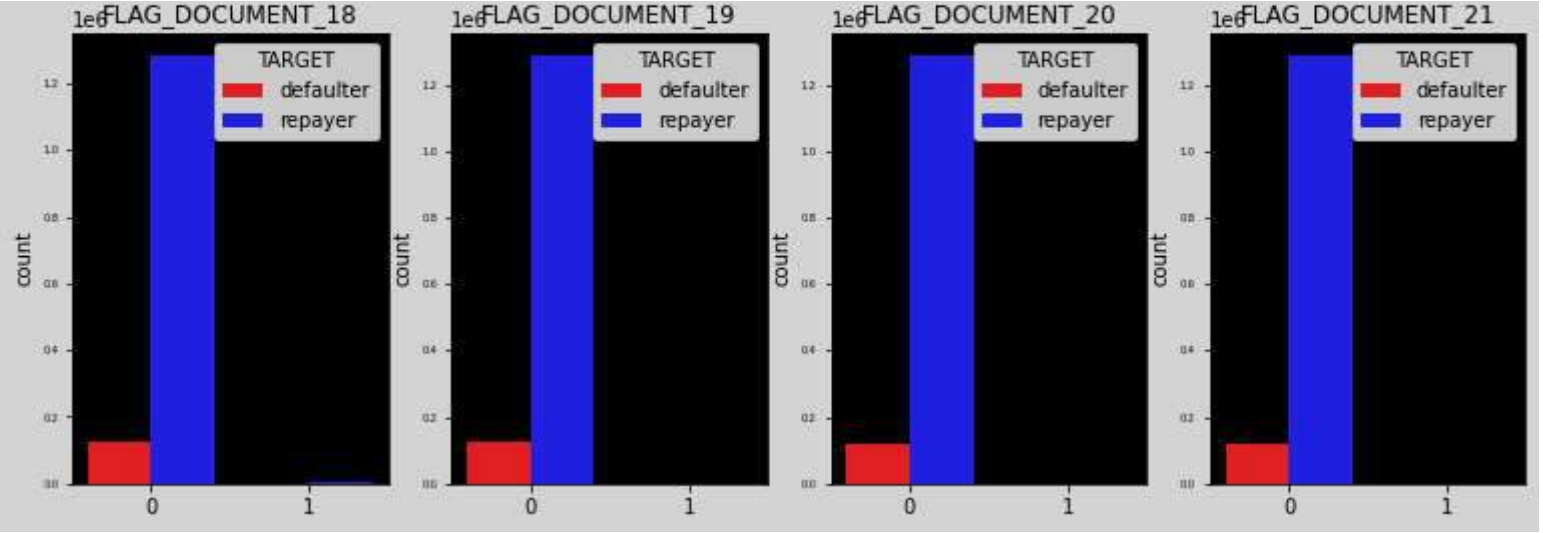
```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_17048/3391209563.py in <module>
      6 ax.set_title("days before application client changed phone -violin plot")
      7 plt.subplot(122)
----> 8 ax1 = sns.lvplot(combined_df["TARGET"],
      9                  combined_df["DAYS_LAST_PHONE_CHANGE"],palette=["g","r"])
     10 ax1.set_facecolor("oldlace")

AttributeError: module 'seaborn' has no attribute 'lvplot'
```



In [37]:
```
cols = [ 'FLAG_DOCUMENT_2', 'FLAG_DOCUMENT_3',
       'FLAG_DOCUMENT_4', 'FLAG_DOCUMENT_5', 'FLAG_DOCUMENT_6',
       'FLAG_DOCUMENT_7', 'FLAG_DOCUMENT_8', 'FLAG_DOCUMENT_9',
       'FLAG_DOCUMENT_10', 'FLAG_DOCUMENT_11', 'FLAG_DOCUMENT_12',
       'FLAG_DOCUMENT_13', 'FLAG_DOCUMENT_14', 'FLAG_DOCUMENT_15',
       'FLAG_DOCUMENT_16', 'FLAG_DOCUMENT_17', 'FLAG_DOCUMENT_18',
       'FLAG_DOCUMENT_19', 'FLAG_DOCUMENT_20', 'FLAG_DOCUMENT_21']

df_flag = combined_df[cols+["TARGET"]]

length = len(cols)

df_flag["TARGET"] = df_flag["TARGET"].replace({1:"defaulter",0:"repayer"})

fig = plt.figure(figsize=(13,24))
fig.set_facecolor("lightgrey")
for i,j in itertools.zip_longest(cols,range(length)):
    plt.subplot(5,4,j+1)
    ax = sns.countplot(df_flag[i],hue=df_flag["TARGET"],palette=["r","b"])
    plt.yticks(fontsize=5)
    plt.xlabel("")
    plt.title(i)
    ax.set_facecolor("k")
```

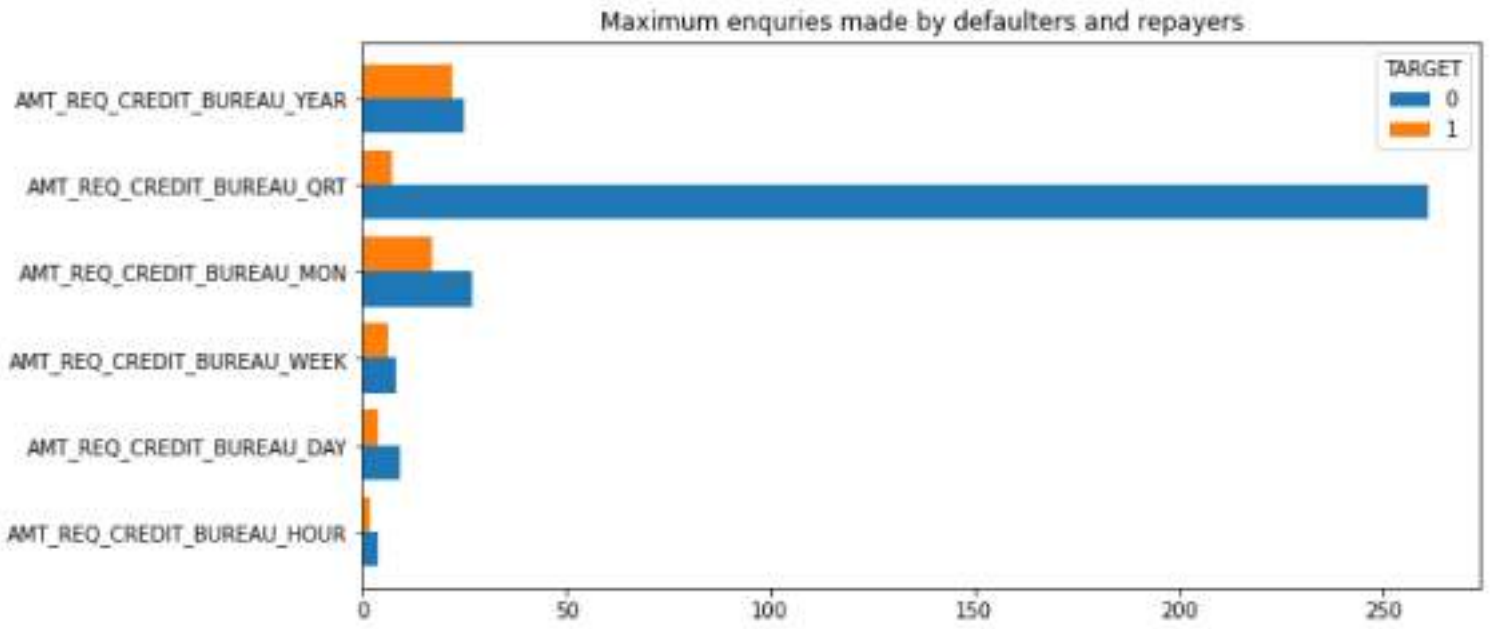


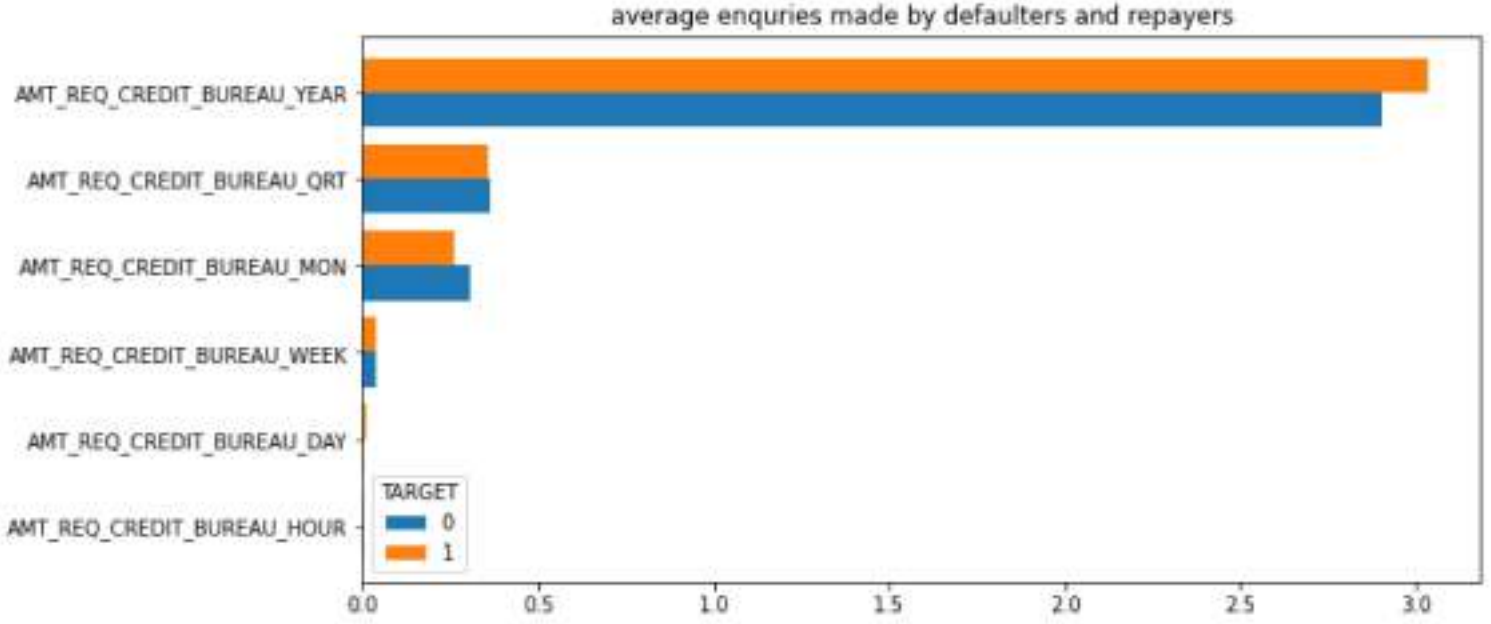

count
0
1
1e6FLAG_DOCUMENT_10
1e6FLAG_DOCUMENT_11
1e6FLAG_DOCUMENT_12
1e6FLAG_DOCUMENT_13
1e6FLAG_DOCUMENT_14
1e6FLAG_DOCUMENT_15
1e6FLAG_DOCUMENT_16
1e6FLAG_DOCUMENT_17
TARGET
defaulter
repayer

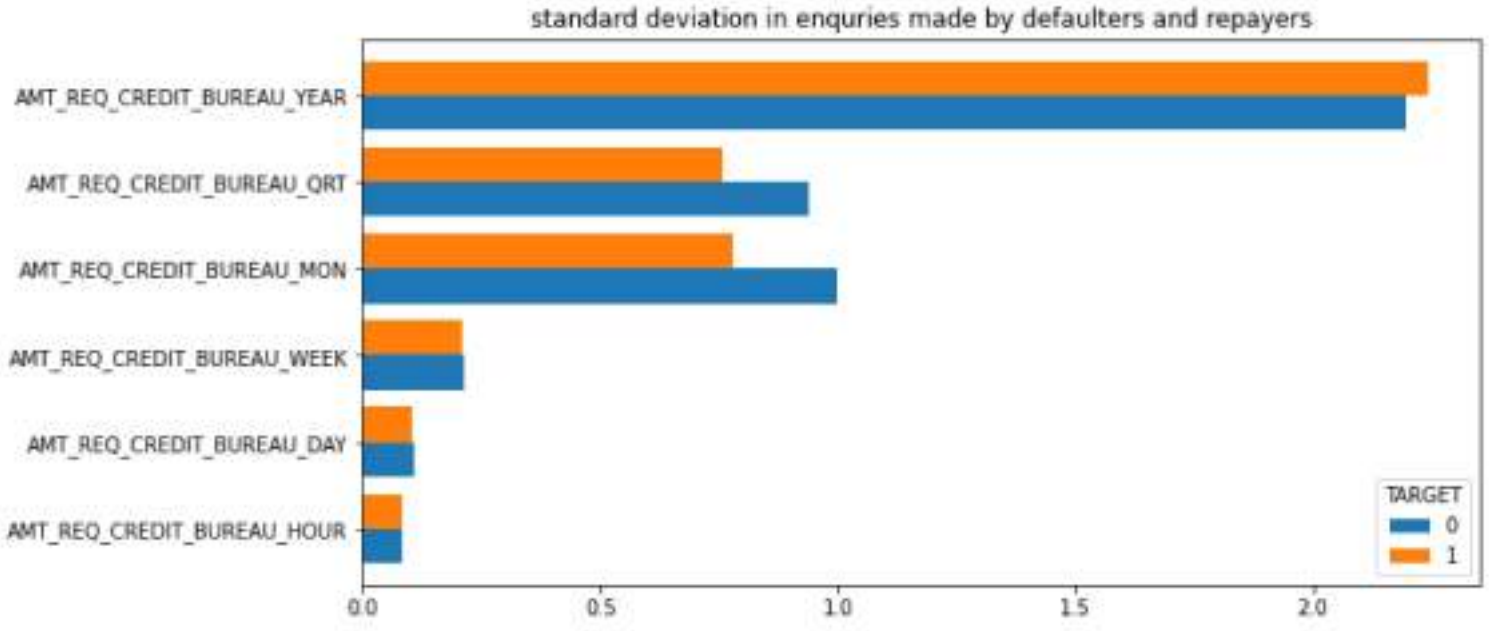


In [38]:

```
cols = ['AMT_REQ_CREDIT_BUREAU_HOUR', 'AMT_REQ_CREDIT_BUREAU_DAY',
       'AMT_REQ_CREDIT_BUREAU_WEEK', 'AMT_REQ_CREDIT_BUREAU_MON',
       'AMT_REQ_CREDIT_BUREAU_QRT', 'AMT_REQ_CREDIT_BUREAU_YEAR']
combined_df.groupby("TARGET")[cols].max().transpose().plot(kind="barh",
                                                           figsize=(10,5),width=.8)
plt.title("Maximum enquries made by defaulters and repayers")
combined_df.groupby("TARGET")[cols].mean().transpose().plot(kind="barh",
                                                            figsize=(10,5),width=.8)
plt.title("average enquries made by defaulters and repayers")
combined_df.groupby("TARGET")[cols].std().transpose().plot(kind="barh",
                                                           figsize=(10,5),width=.8)
plt.title("standard deviation in enquries made by defaulters and repayers")
plt.show()
```

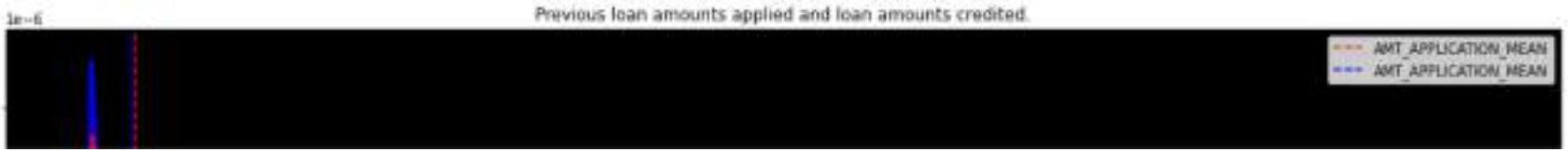

Maximum enquries made by defaulters and repayers
AMT_REQ_CREDIT_BUREAU_YEAR
AMT_REQ_CREDIT_BUREAU_QRT
AMT_REQ_CREDIT_BUREAU_MON
AMT_REQ_CREDIT_BUREAU_WEEK
AMT_REQ_CREDIT_BUREAU_DAY
AMT_REQ_CREDIT_BUREAU_HOUR
TARGET
0
1
0
50
100
150
200
250

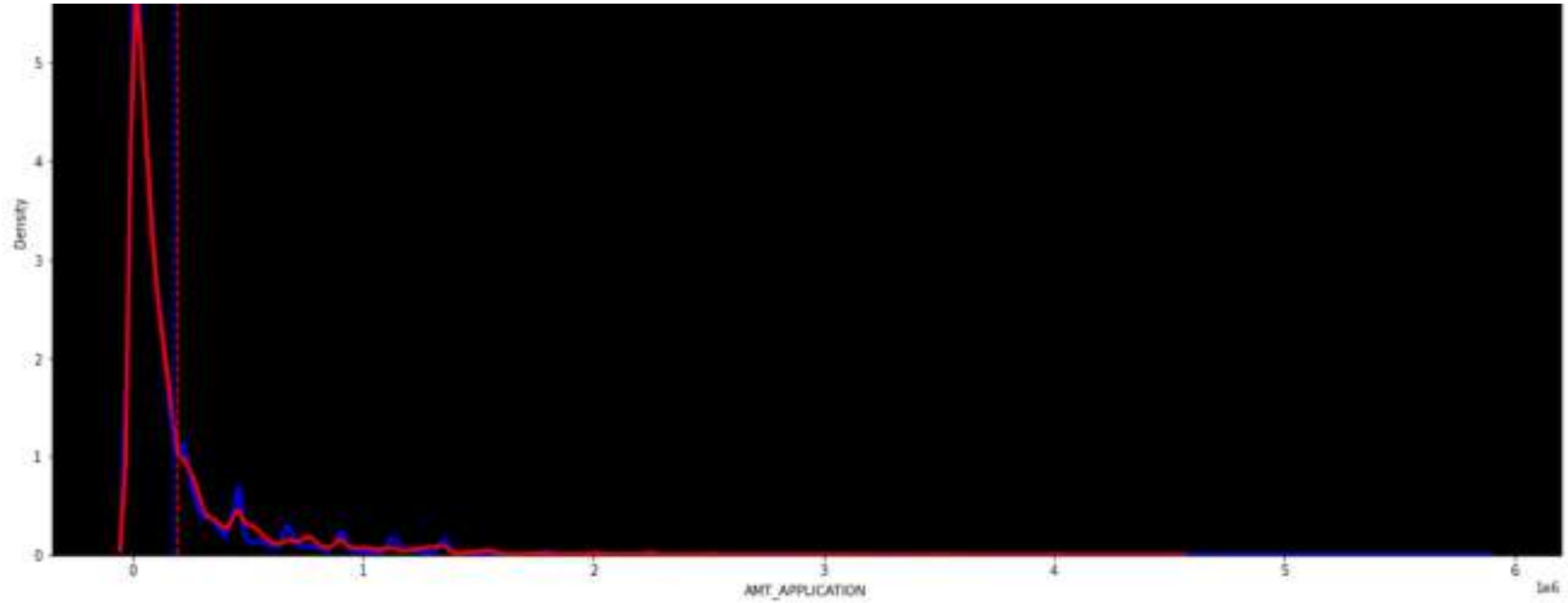

average enquries made by defaulters and repayers
AMT_REQ_CREDIT_BUREAU_YEAR
AMT_REQ_CREDIT_BUREAU_QRT
AMT_REQ_CREDIT_BUREAU_MON
AMT_REQ_CREDIT_BUREAU_WEEK
AMT_REQ_CREDIT_BUREAU_DAY
AMT_REQ_CREDIT_BUREAU_HOUR
TARGET
0
1
0.0
0.5
1.0
1.5
2.0
2.5
3.0

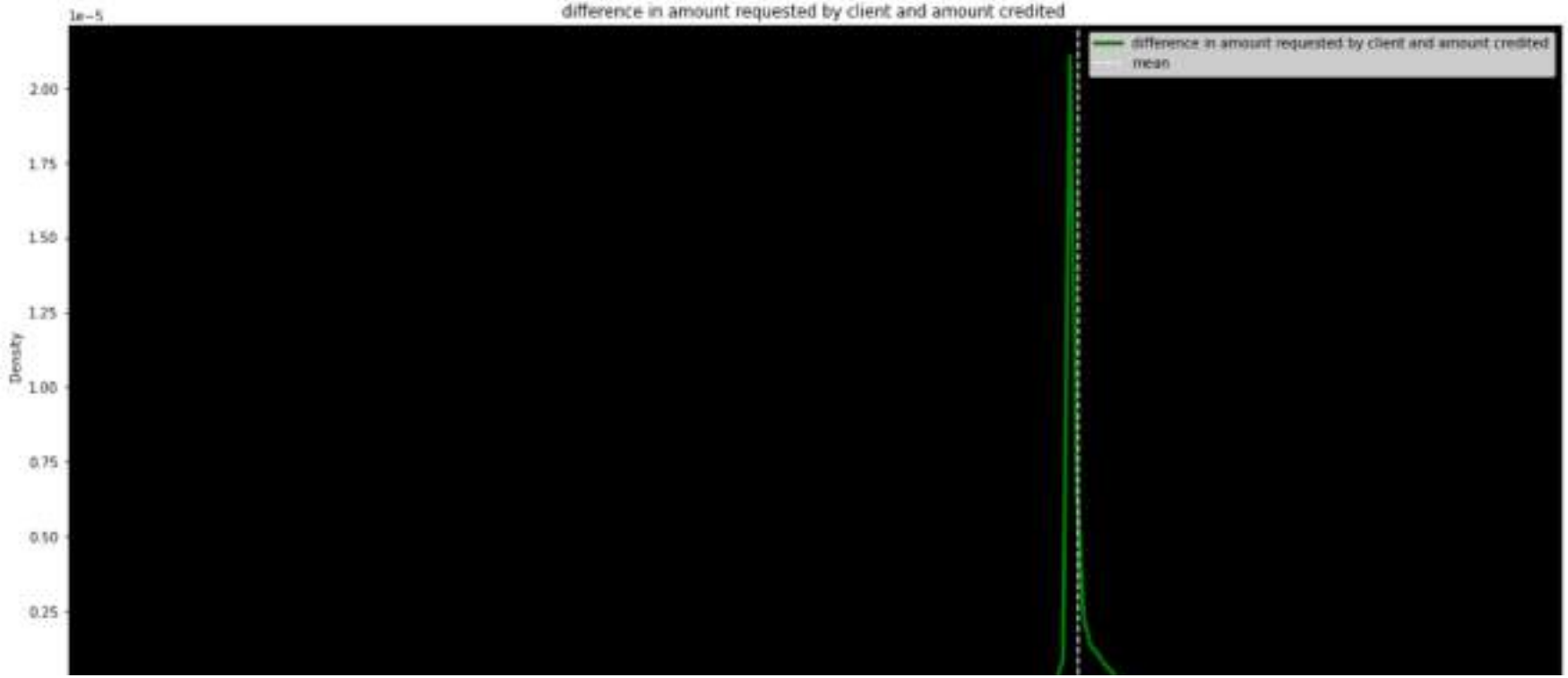


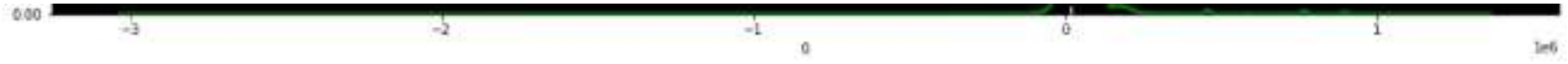
In [39]:

```
plt.figure(figsize=(20,20))
plt.subplot(211)
ax = sns.kdeplot(combined_df["AMT_APPLICATION"],color="b",linewidth=3)
ax = sns.kdeplot(combined_df[combined_df["AMT_CREDIT_y"].notnull()]["AMT_CREDIT_y"],color="r",linewidth=3)
plt.axvline(combined_df[combined_df["AMT_CREDIT_y"].notnull()]["AMT_CREDIT_y"].mean(),color="r",linestyle="dashed",label=
plt.axvline(combined_df["AMT_APPLICATION"].mean(),color="b",linestyle="dashed",label="AMT_APPLICATION_MEAN")
plt.legend(loc="best")
plt.title("Previous loan amounts applied and loan amounts credited.")
ax.set_facecolor("k")

plt.subplot(212)
diff = (combined_df["AMT_CREDIT_y"] - combined_df["AMT_APPLICATION"]).reset_index()
diff = diff[diff[0].notnull()]
ax1 = sns.kdeplot(diff[0],color="g",linewidth=3,label = "difference in amount requested by client and amount credited")
plt.axvline(diff[0].mean(),color="white",linestyle="dashed",label = "mean")
plt.title("difference in amount requested by client and amount credited")
ax1.legend(loc="best")
ax1.set_facecolor("k")
```



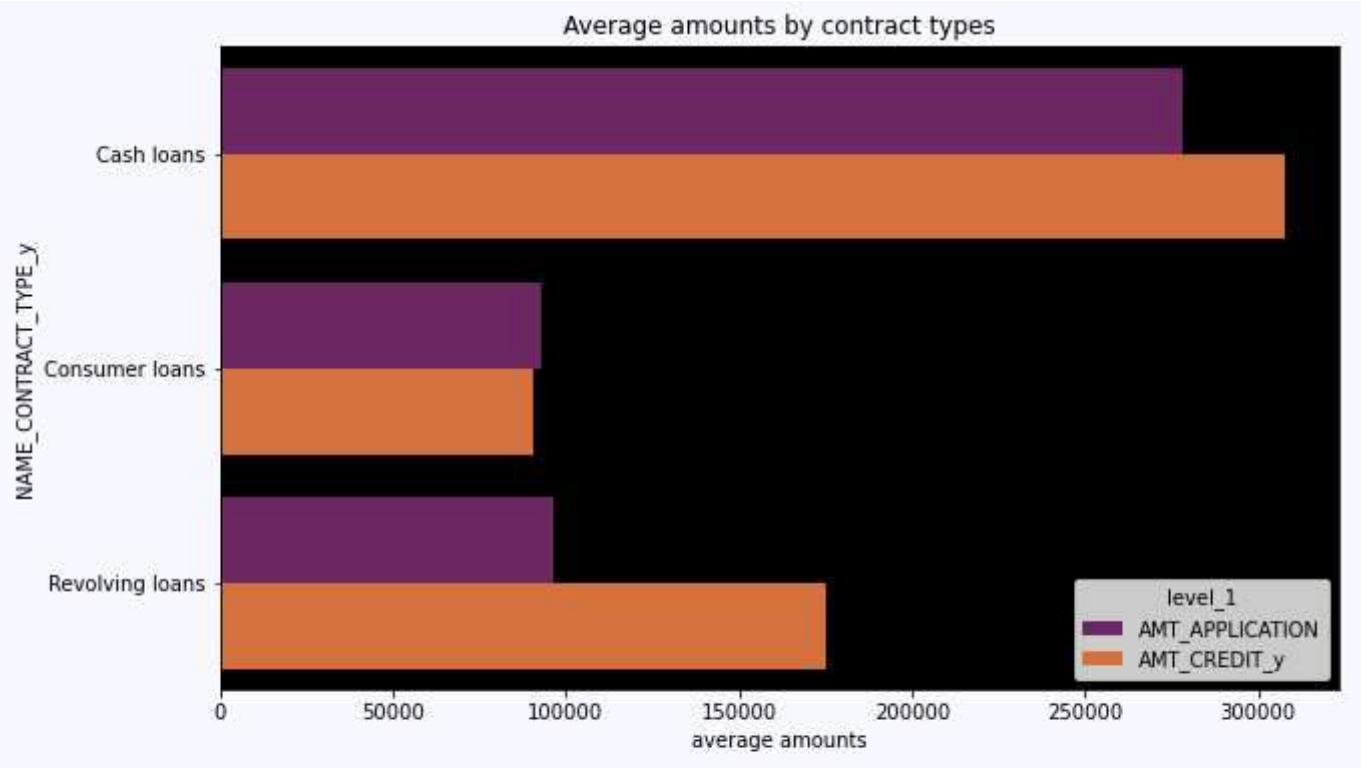

Density
AMT_APPLICATION
1e6

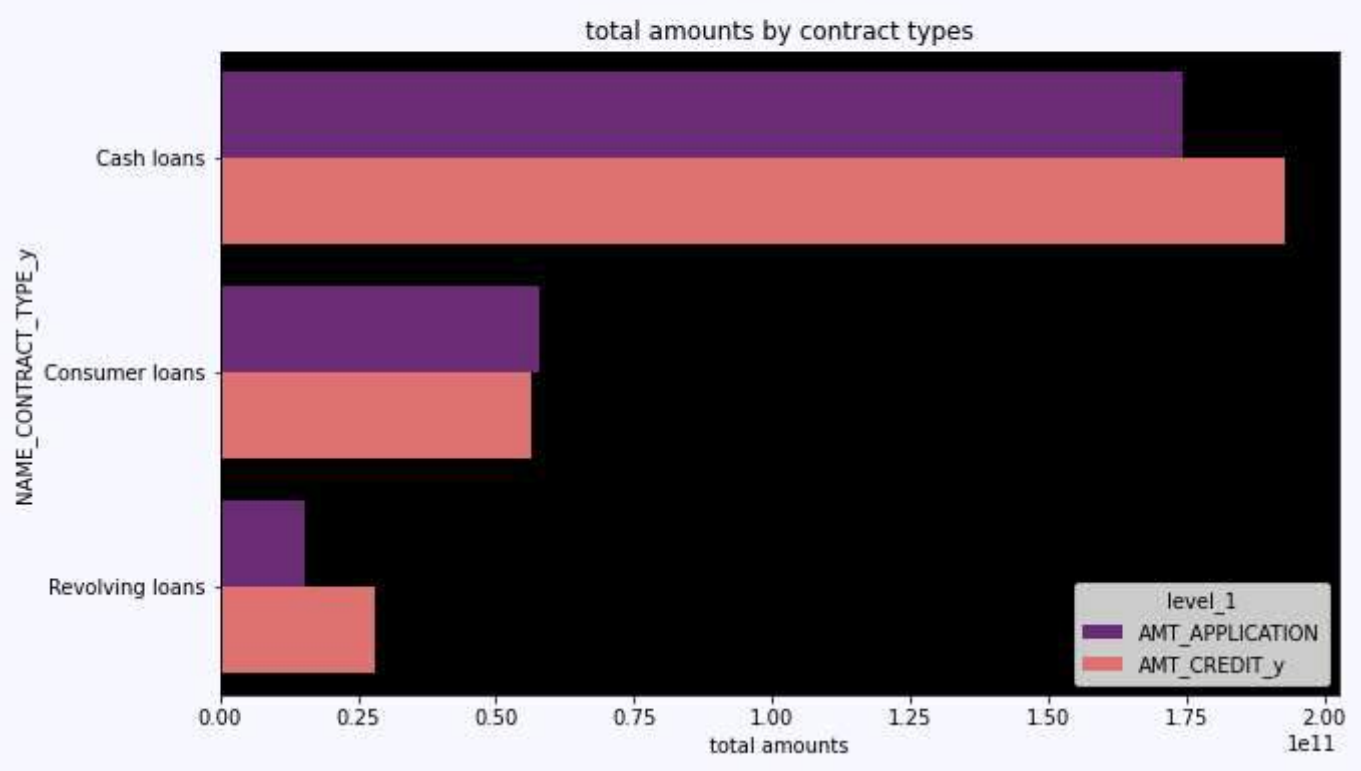

difference in amount requested by client and amount credited
1e−5
Density
difference in amount requested by client and amount credited
mean



In [40]:

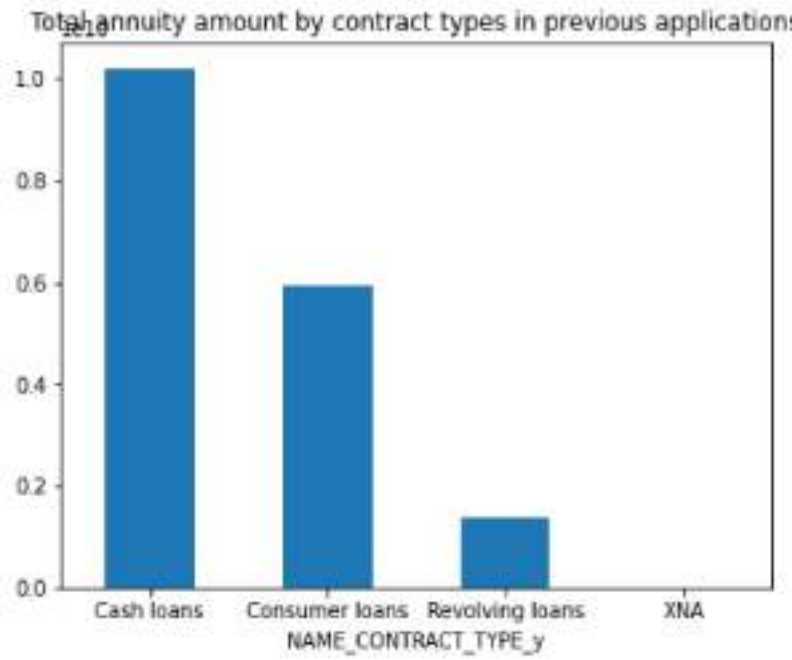
```
mn = combined_df.groupby("NAME_CONTRACT_TYPE_y")[["AMT_APPLICATION","AMT_CREDIT_y"]].mean().stack().reset_index()
tt = combined_df.groupby("NAME_CONTRACT_TYPE_y")[["AMT_APPLICATION","AMT_CREDIT_y"]].sum().stack().reset_index()
fig = plt.figure(figsize=(10,13))
fig.set_facecolor("ghostwhite")
plt.subplot(211)
ax = sns.barplot(0,"NAME_CONTRACT_TYPE_y",data=mn[:6],hue="level_1",palette="inferno")
ax.set_facecolor("k")
ax.set_xlabel("average amounts")
ax.set_title("Average amounts by contract types")

plt.subplot(212)
ax1 = sns.barplot(0,"NAME_CONTRACT_TYPE_y",data=tt[:6],hue="level_1",palette="magma")
ax1.set_facecolor("k")
ax1.set_xlabel("total amounts")
ax1.set_title("total amounts by contract types")
plt.subplots_adjust(hspace = .2)
plt.show()
```

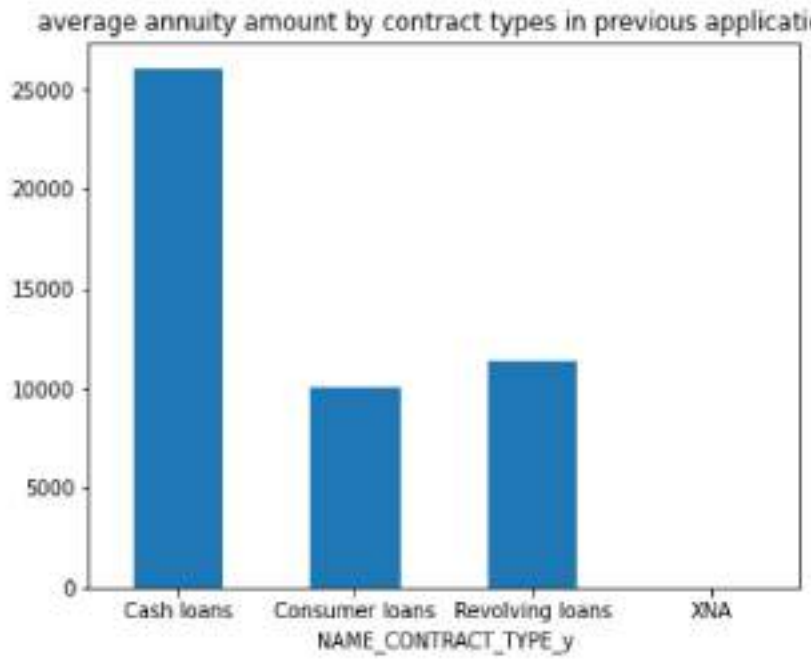




In [41]:

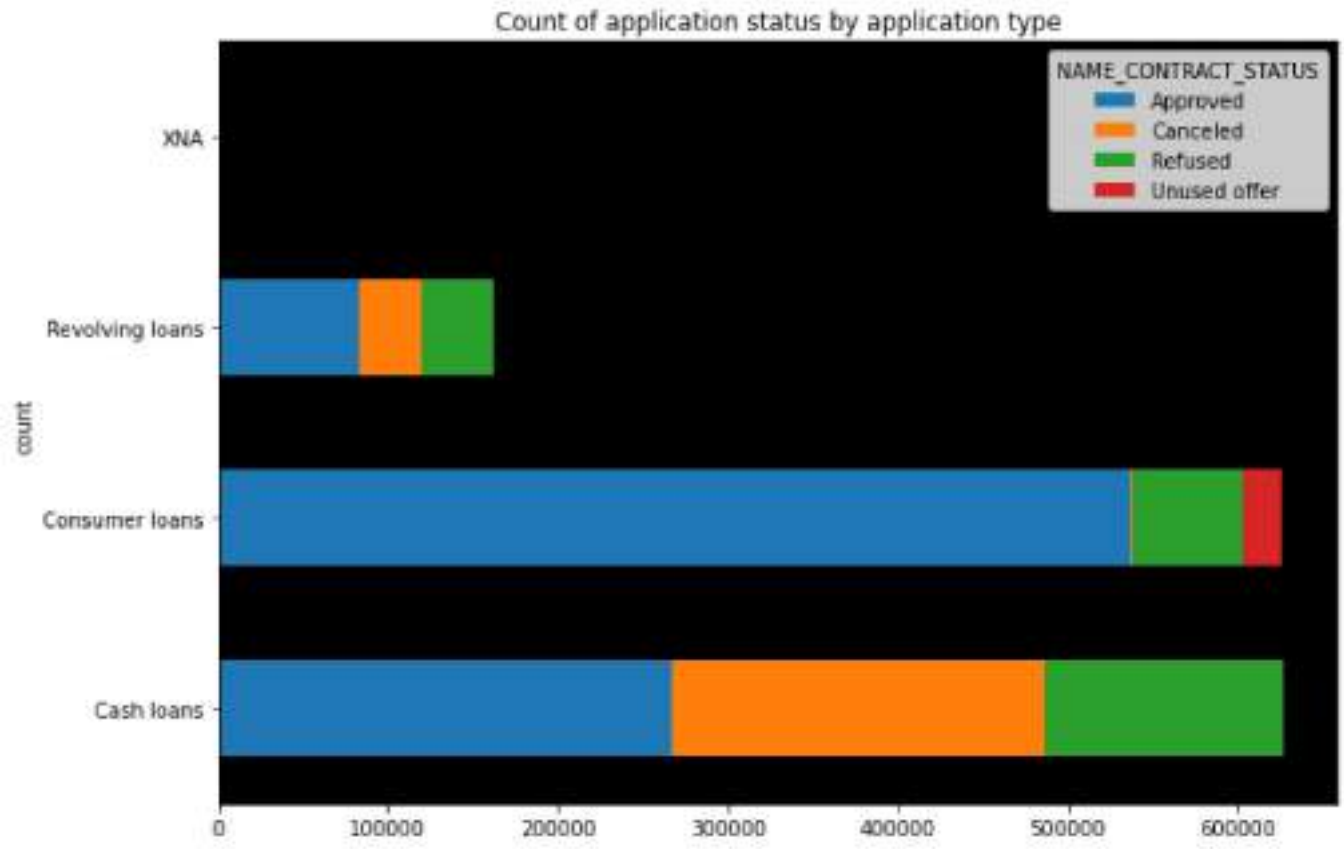
```
plt.figure(figsize=(14,5))
plt.subplot(121)
combined_df.groupby("NAME_CONTRACT_TYPE_y")["AMT_ANNUITY_y"].sum().plot(kind="bar")
plt.xticks(rotation=0)
plt.title("Total annuity amount by contract types in previous applications")
plt.subplot(122)
combined_df.groupby("NAME_CONTRACT_TYPE_y")["AMT_ANNUITY_y"].mean().plot(kind="bar")
plt.title("average annuity amount by contract types in previous applications")
plt.xticks(rotation=0)
plt.show()
```





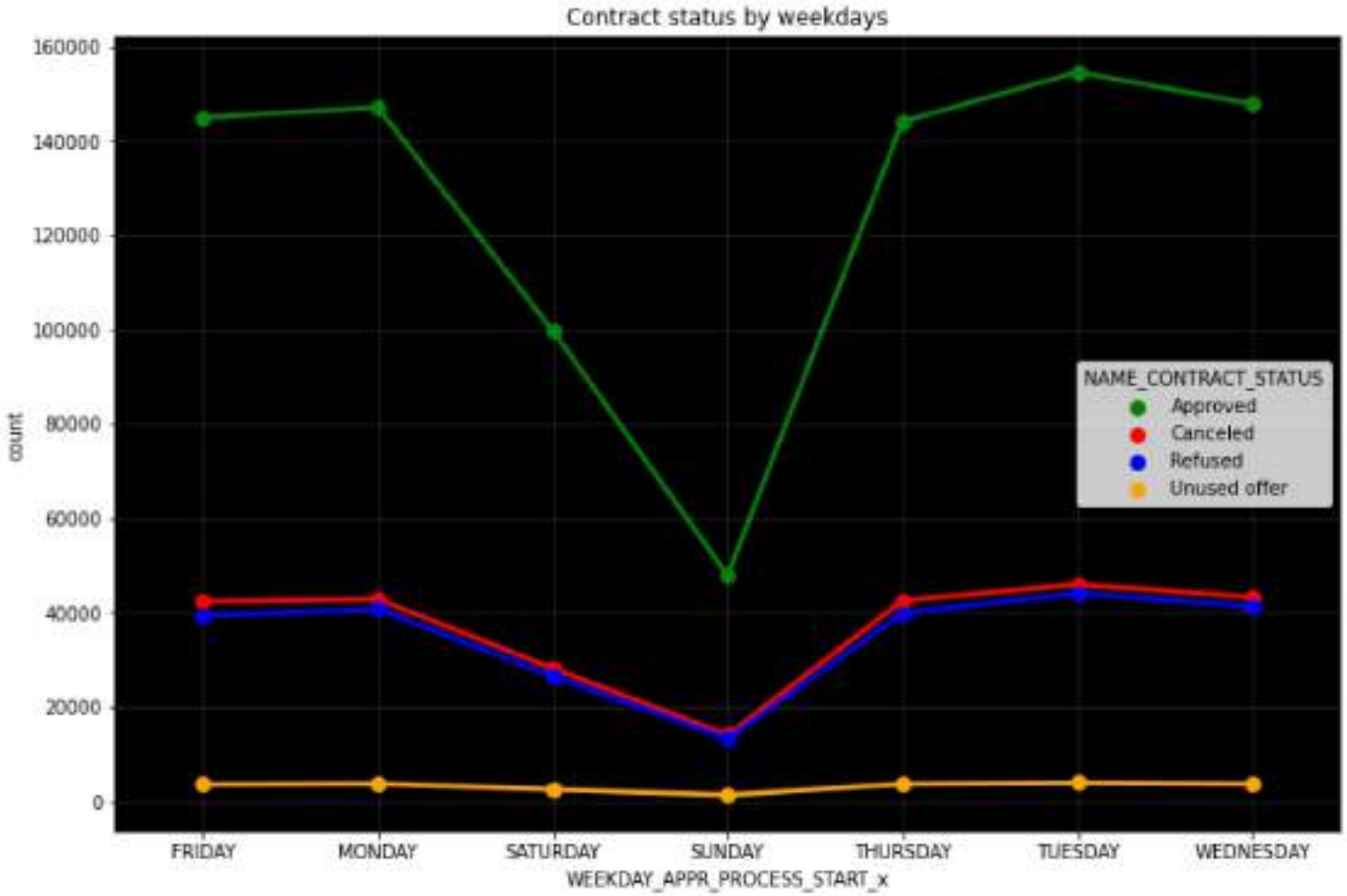
In [42]:

```
ax = pd.crosstab(combined_df["NAME_CONTRACT_TYPE_y"],combined_df["NAME_CONTRACT_STATUS"]).plot(kind="barh",figsize=(10,7)
plt.xticks(rotation =0)
plt.ylabel("count")
plt.title("Count of application status by application type")
ax.set_facecolor("k")
```



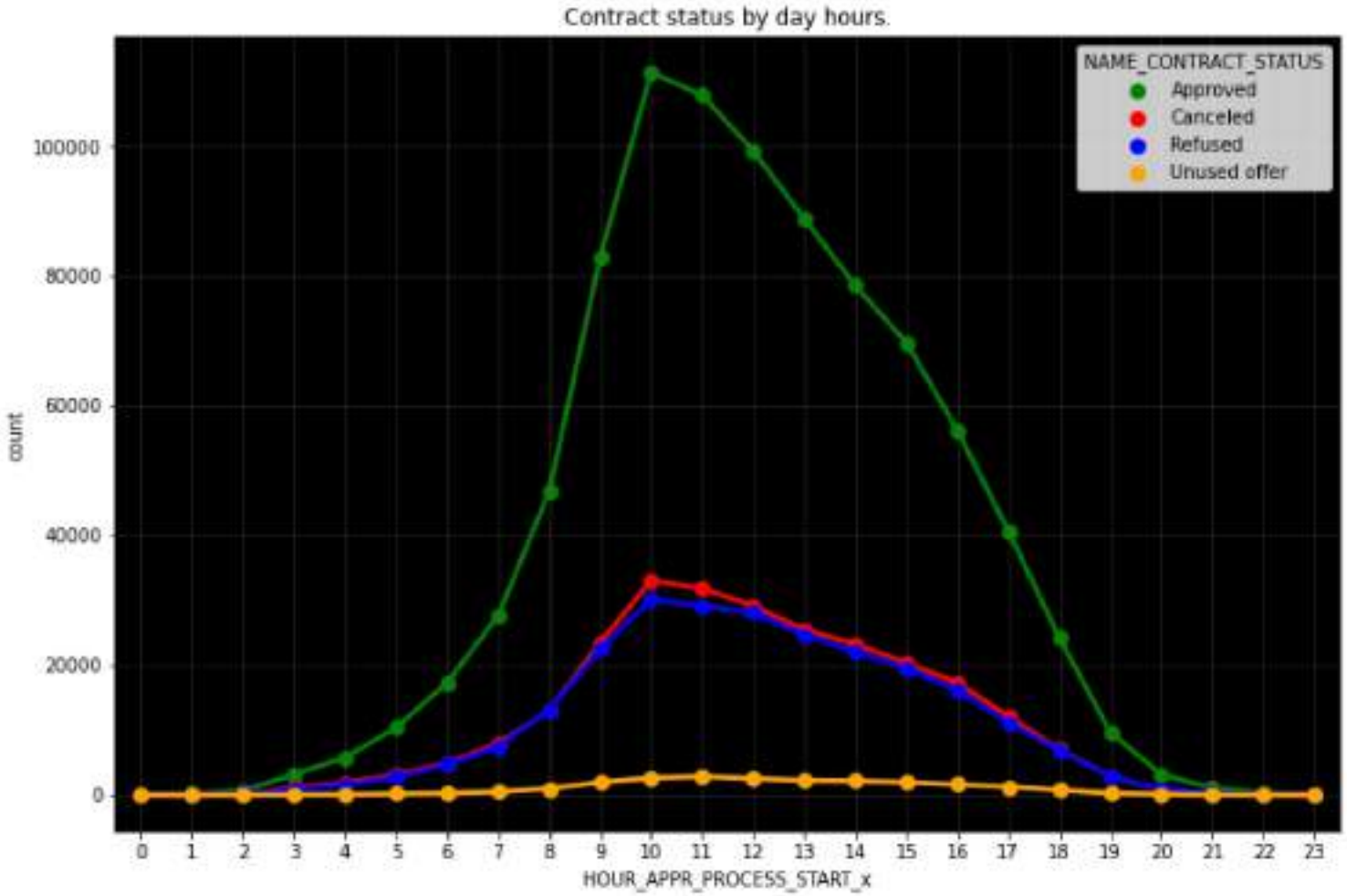
In [43]:

```
hr = pd.crosstab(combined_df["WEEKDAY_APPR_PROCESS_START_x"],combined_df["NAME_CONTRACT_STATUS"]).stack().reset_index()
plt.figure(figsize=(12,8))
ax = sns.pointplot(hr["WEEKDAY_APPR_PROCESS_START_x"],hr[0],hue=hr["NAME_CONTRACT_STATUS"],palette=["g","r","b","orange"]
ax.set_facecolor("k")
ax.set_ylabel("count")
ax.set_title("Contract status by weekdays")
plt.grid(True,alpha=.2)
```



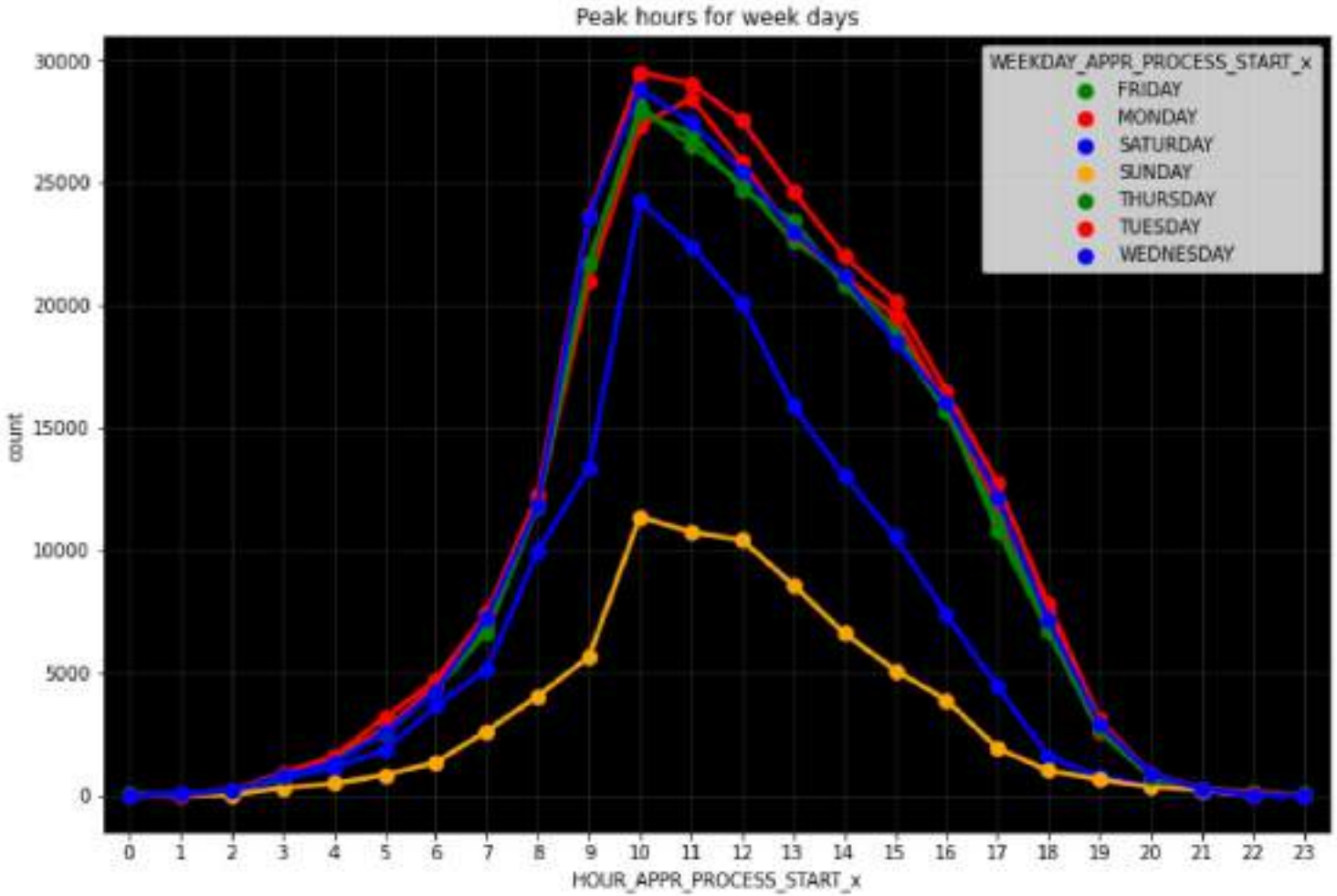
In [44]:

```
hr = pd.crosstab(combined_df["HOUR_APPR_PROCESS_START_x"],combined_df["NAME_CONTRACT_STATUS"]).stack().reset_index()
plt.figure(figsize=(12,8))
ax = sns.pointplot(hr["HOUR_APPR_PROCESS_START_x"],hr[0],hue=hr["NAME_CONTRACT_STATUS"],palette=["g","r","b","orange"],sc
ax.set_facecolor("k")
ax.set_ylabel("count")
ax.set_title("Contract status by day hours.")
plt.grid(True,alpha=.2)
```



In [45]:

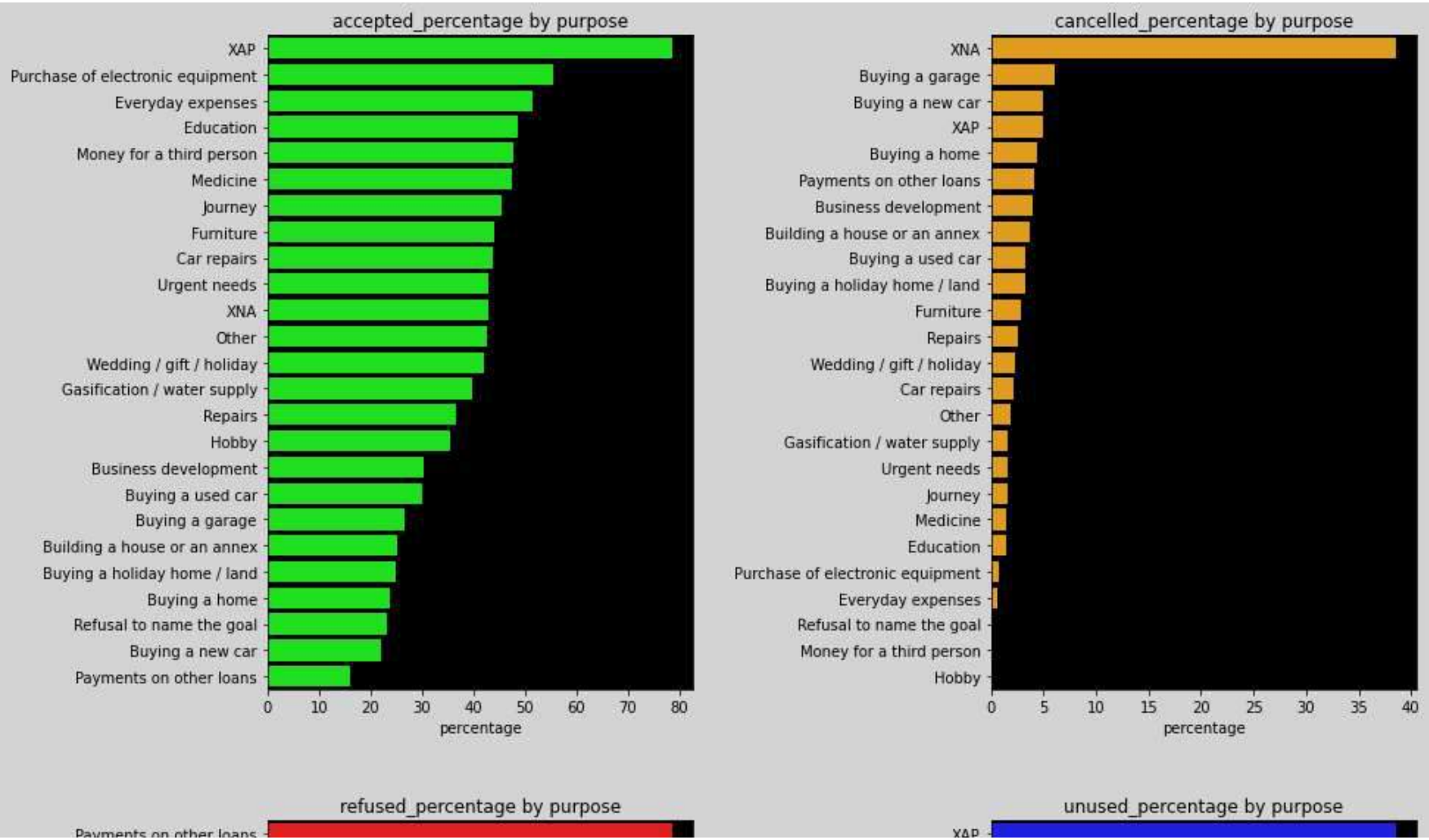
```
hr = pd.crosstab(combined_df["HOUR_APPR_PROCESS_START_x"],combined_df["WEEKDAY_APPR_PROCESS_START_x"]).stack().reset_inde
plt.figure(figsize=(12,8))
ax = sns.pointplot(hr["HOUR_APPR_PROCESS_START_x"],hr[0],hue=hr["WEEKDAY_APPR_PROCESS_START_x"],palette=["g","r","b","ora
ax.set_facecolor("k")
ax.set_ylabel("count")
ax.set_title("Peak hours for week days")
plt.grid(True,alpha=.2)
```

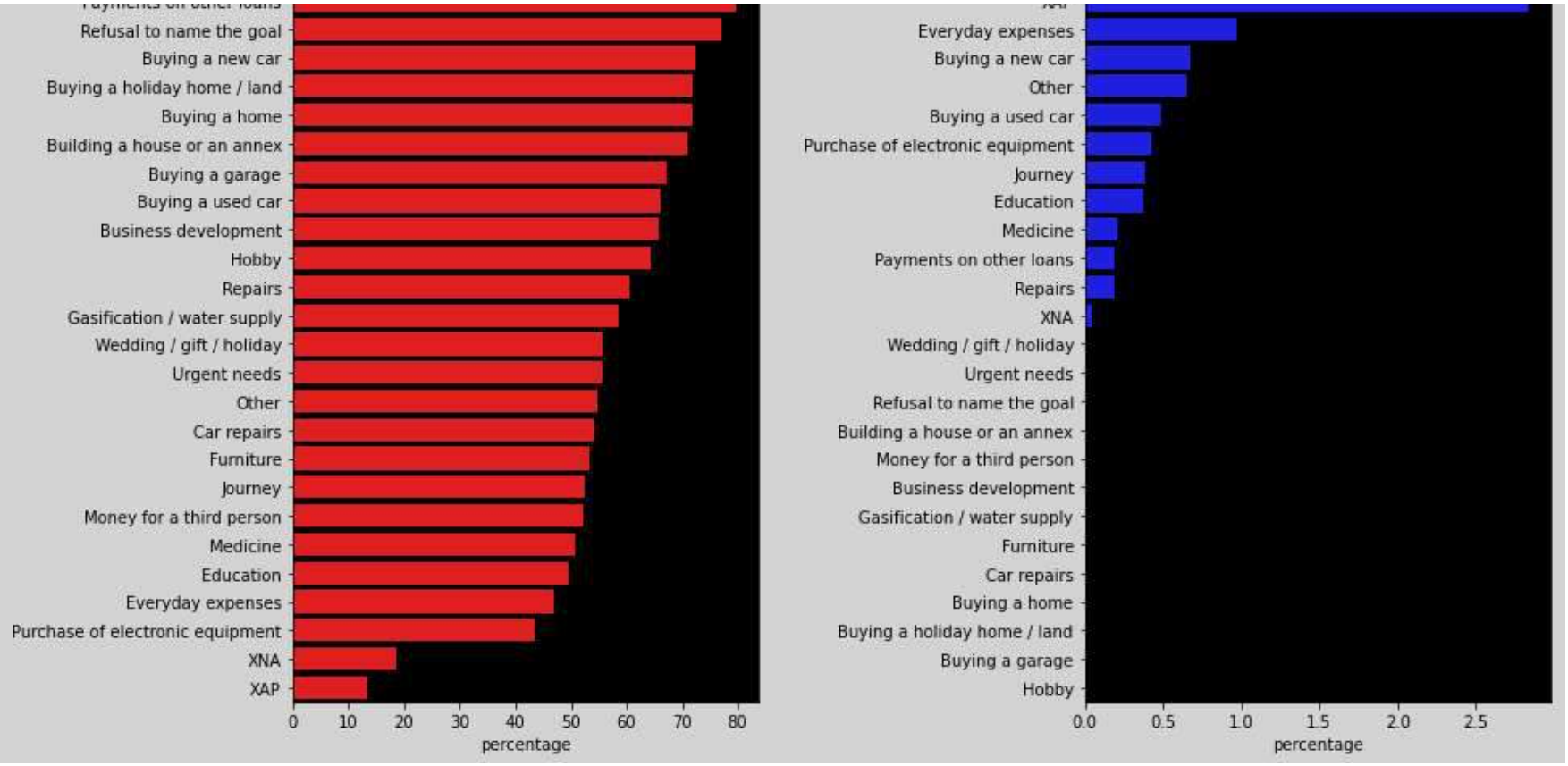


In [46]:

```
combined_df[["NAME_CASH_LOAN_PURPOSE","NAME_CONTRACT_STATUS"]]
purpose = pd.crosstab(combined_df["NAME_CASH_LOAN_PURPOSE"],combined_df["NAME_CONTRACT_STATUS"])
purpose["a"] = (purpose["Approved"]*100)/(purpose["Approved"]+purpose["Canceled"]+purpose["Refused"]+purpose["Unused offe
purpose["c"] = (purpose["Canceled"]*100)/(purpose["Approved"]+purpose["Canceled"]+purpose["Refused"]+purpose["Unused offe
purpose["r"] = (purpose["Refused"]*100)/(purpose["Approved"]+purpose["Canceled"]+purpose["Refused"]+purpose["Unused offer
purpose["u"] = (purpose["Unused offer"]*100)/(purpose["Approved"]+purpose["Canceled"]+purpose["Refused"]+purpose["Unused
purpose_new = purpose[["a","c","r","u"]]
purpose_new = purpose_new.stack().reset_index()
purpose_new["NAME_CONTRACT_STATUS"] = purpose_new["NAME_CONTRACT_STATUS"].replace({"a":"accepted_percentage","c":"cancell
                                                                                   "r":"refused_percentage","u":"unused_percentage"})

lst = purpose_new["NAME_CONTRACT_STATUS"].unique().tolist()
length = len(lst)
```

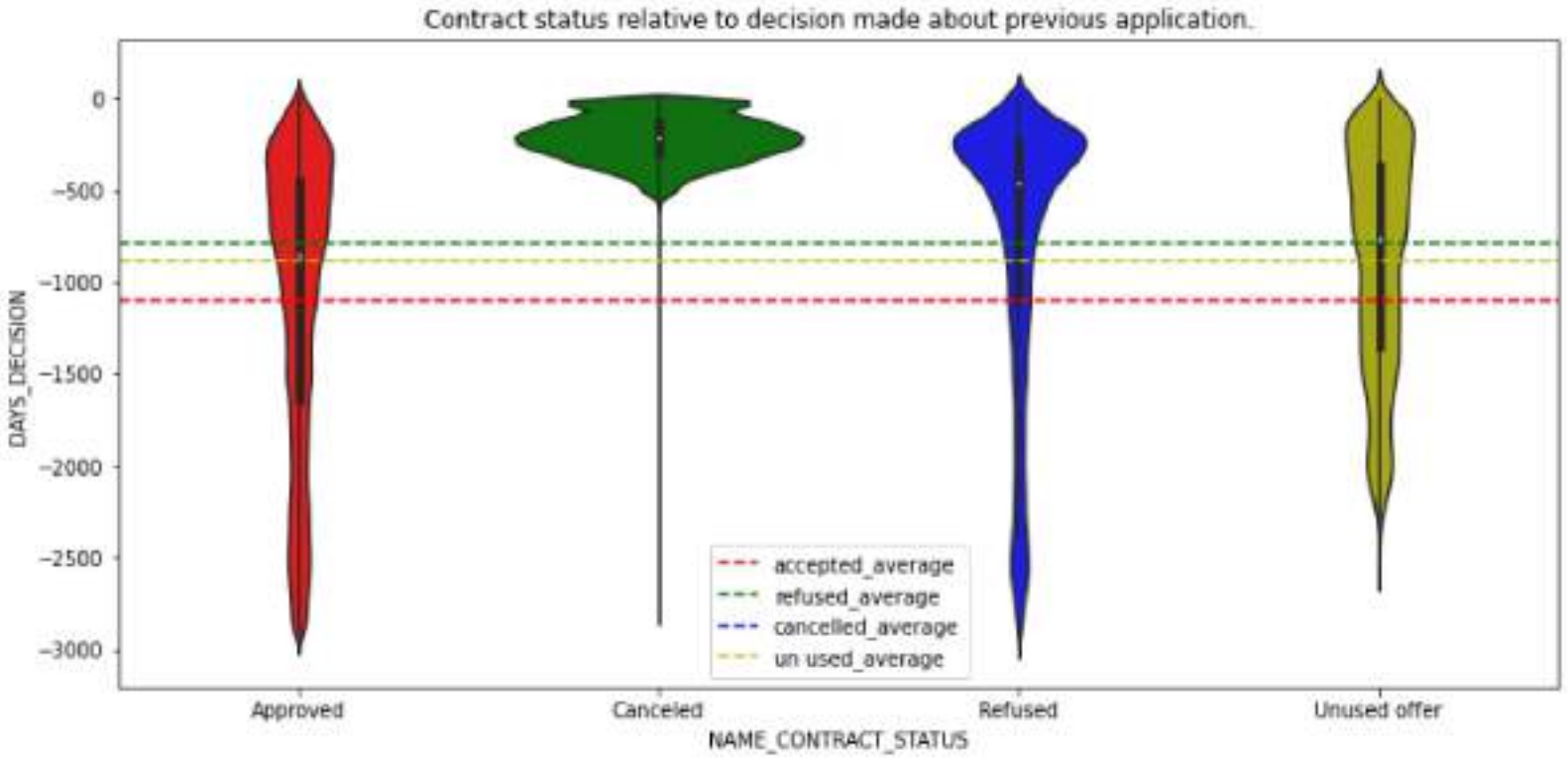
```
cs = ["lime","orange","r","b"]

fig = plt.figure(figsize=(14,18))
fig.set_facecolor("lightgrey")
for i,j,k in itertools.zip_longest(lst,range(length),cs):
    plt.subplot(2,2,j+1)
    dat = purpose_new[purpose_new["NAME_CONTRACT_STATUS"] == i]
    ax = sns.barplot(0,"NAME_CASH_LOAN_PURPOSE",data=dat.sort_values(by=0,ascending=False),color=k)
    plt.ylabel("")
    plt.xlabel("percentage")
    plt.title(i+" by purpose")
    plt.subplots_adjust(wspace = .7)
    ax.set_facecolor("k")
```





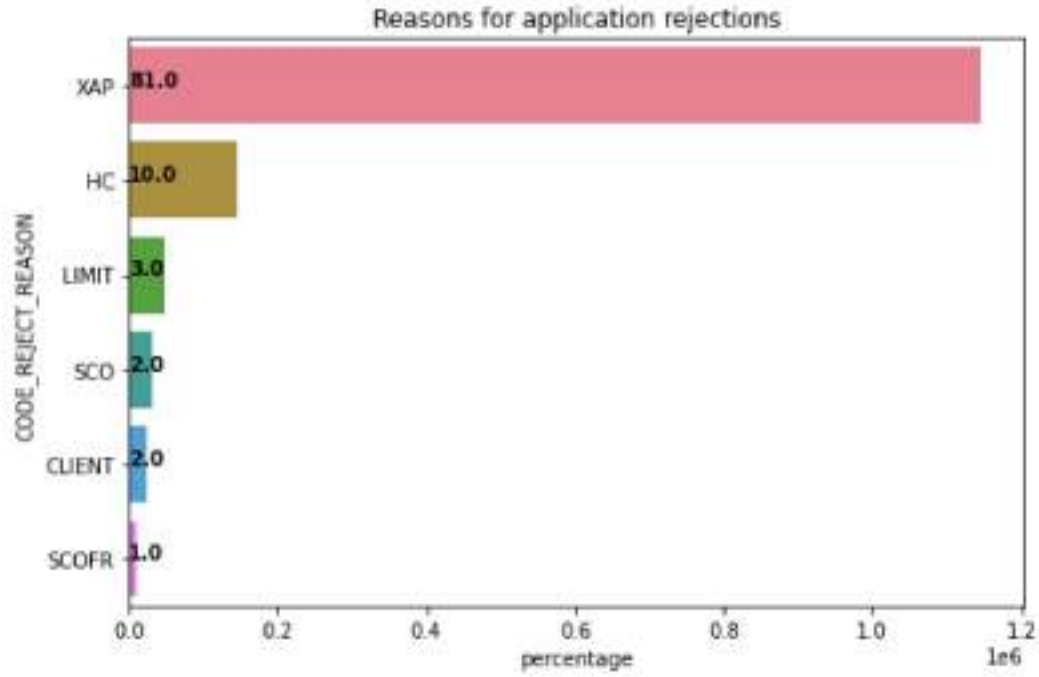
In [47]:

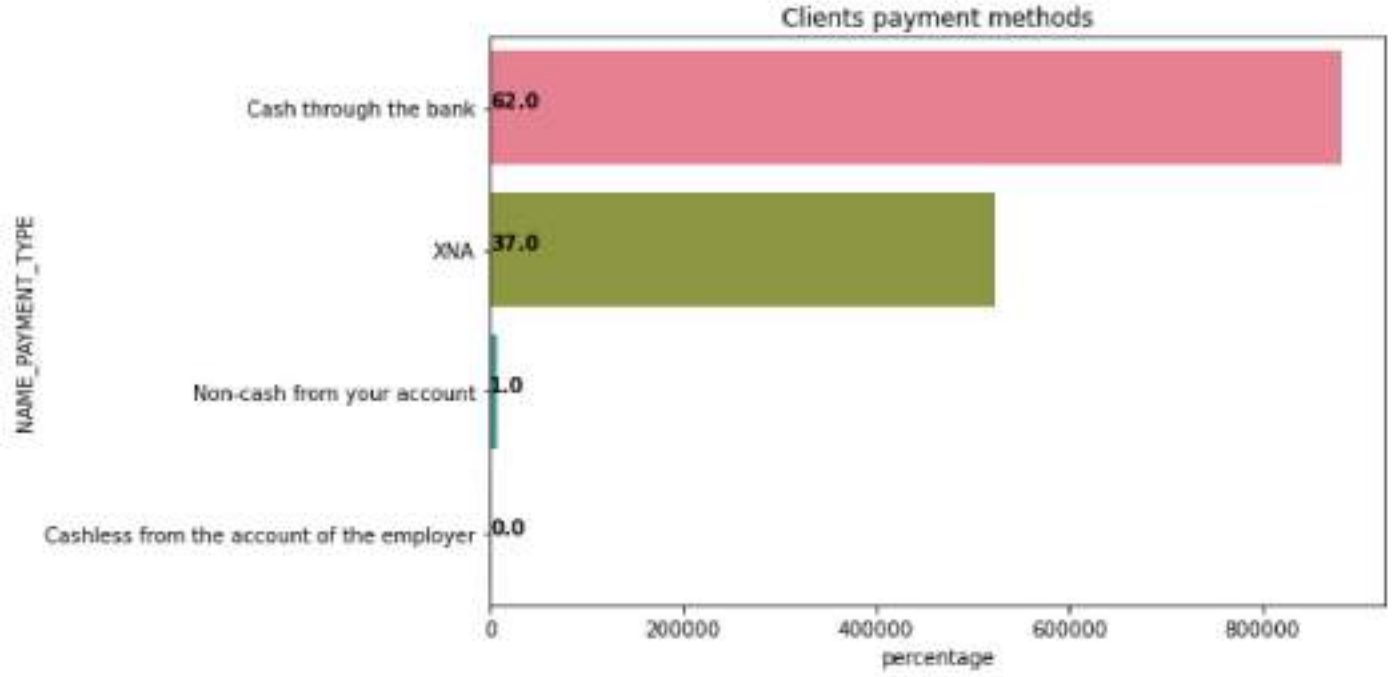
```
plt.figure(figsize=(13,6))
sns.violinplot(y= combined_df["DAYS_DECISION"],
               x = combined_df["NAME_CONTRACT_STATUS"],palette=["r","g","b","y"])
plt.axhline(combined_df[combined_df["NAME_CONTRACT_STATUS"] == "Approved"]["DAYS_DECISION"].mean(),
            color="r",linestyle="dashed",label="accepted_average")
plt.axhline(combined_df[combined_df["NAME_CONTRACT_STATUS"] == "Refused"]["DAYS_DECISION"].mean(),
            color="g",linestyle="dashed",label="refused_average")
plt.axhline(combined_df[combined_df["NAME_CONTRACT_STATUS"] == "Cancelled"]["DAYS_DECISION"].mean(),color="b",
            linestyle="dashed",label="cancelled_average")
plt.axhline(combined_df[combined_df["NAME_CONTRACT_STATUS"] == "Unused offer"]["DAYS_DECISION"].mean(),color="y",
            linestyle="dashed",label="un used_average")
plt.legend(loc="best")

plt.title("Contract status relative to decision made about previous application.")
plt.show()
```



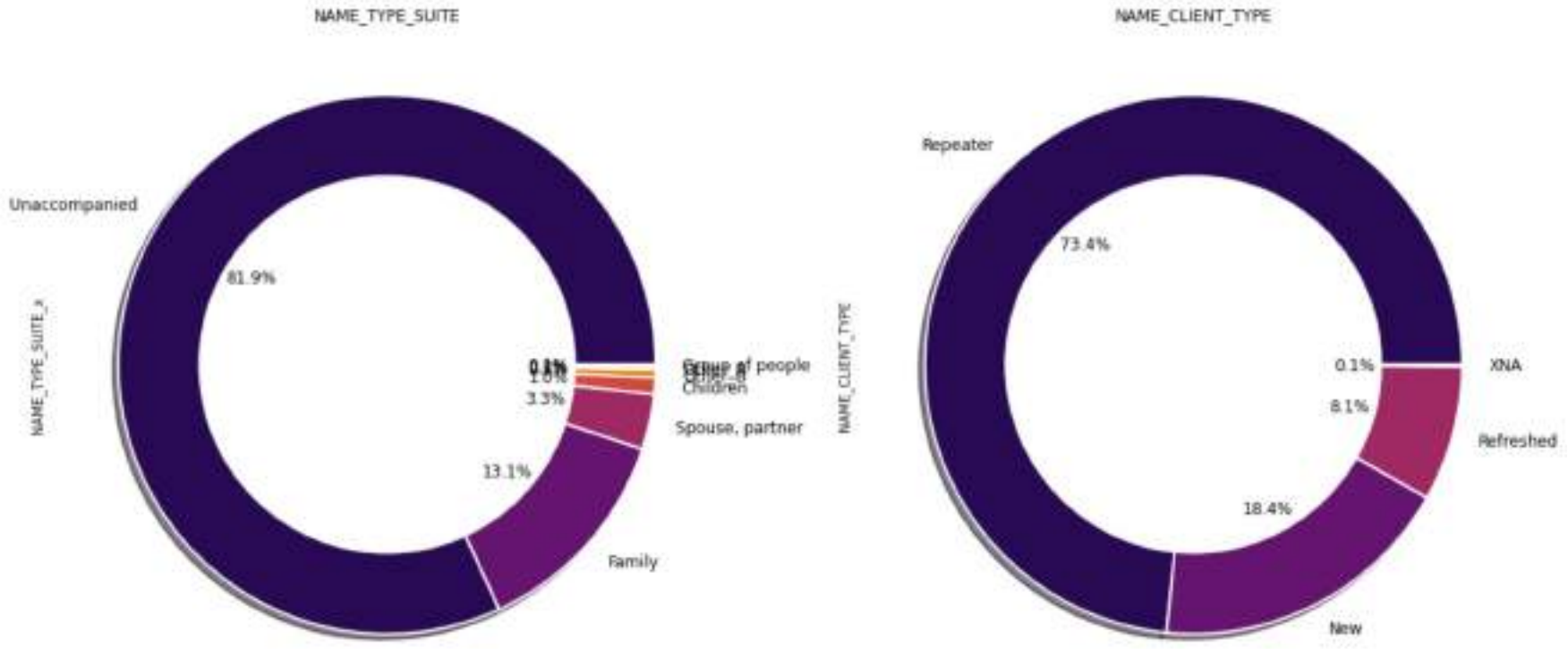
In [48]:

```
plt.figure(figsize=(8,12))
plt.subplot(211)
rej = combined_df["CODE_REJECT_REASON"].value_counts().reset_index()
ax = sns.barplot("CODE_REJECT_REASON","index",data=rej[:6],palette="husl")
for i,j in enumerate(np.around((rej["CODE_REJECT_REASON"][:6].values*100/(rej["CODE_REJECT_REASON"][:6].sum())))):
    ax.text(.7,i,j,weight="bold")
plt.xlabel("percentage")
plt.ylabel("CODE_REJECT_REASON")
plt.title("Reasons for application rejections")

plt.subplot(212)
pay = combined_df["NAME_PAYMENT_TYPE"].value_counts().reset_index()
ax1 = sns.barplot("NAME_PAYMENT_TYPE","index",data=pay,palette="husl")
for i,j in enumerate(np.around((pay["NAME_PAYMENT_TYPE"].values*100/(pay["NAME_PAYMENT_TYPE"].sum())))):
    ax1.text(.7,i,j,weight="bold")
plt.xlabel("percentage")
plt.ylabel("NAME_PAYMENT_TYPE")
plt.title("Clients payment methods")
plt.subplots_adjust(hspace = .3)
```


Reasons for application rejections
CODE_REJECT_REASON
XAP
HC
LIMIT
SCO
CLIENT
SCOFR
81.0
10.0
3.0
2.0
2.0
1.0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
percentage
1e6

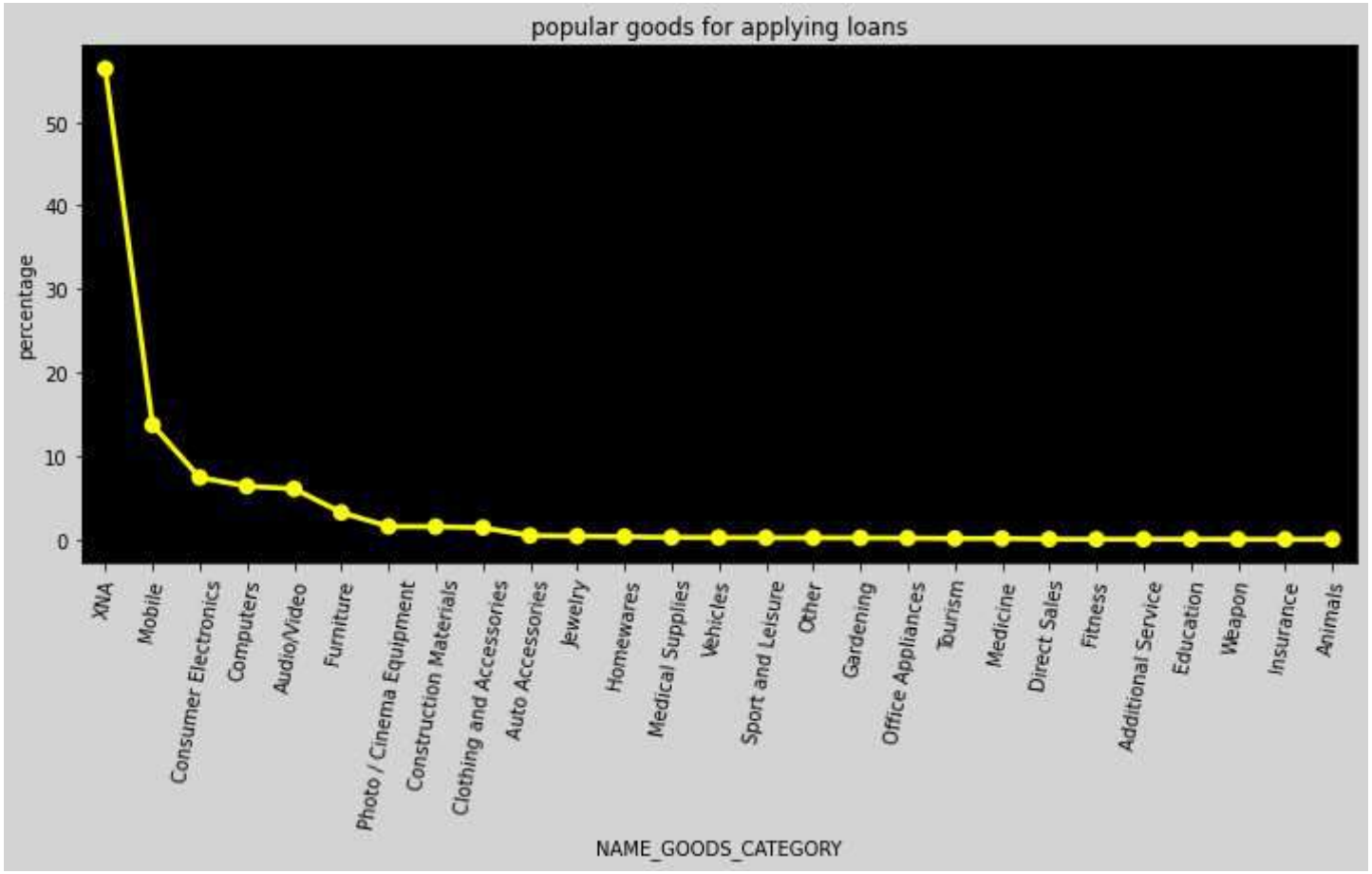

Clients payment methods
NAME_PAYMENT_TYPE
Cash through the bank
XNA
Non-cash from your account
Cashless from the account of the employer
62.0
37.0
1.0
0.0
0
200000
400000
600000
800000
percentage

In [49]:
```
plt.figure(figsize=(20,20))
plt.subplot(121)
combined_df["NAME_TYPE_SUITE_x"].value_counts().plot.pie(autopct = "%1.1f%%",fontsize=12,
                                                         colors = sns.color_palette("inferno"),
                                                    wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)
circ = plt.Circle((0,0),.7,color="white")
plt.gca().add_artist(circ)
plt.title("NAME_TYPE_SUITE")

plt.subplot(122)
combined_df["NAME_CLIENT_TYPE"].value_counts().plot.pie(autopct = "%1.1f%%",fontsize=12,
                                                         colors = sns.color_palette("inferno"),
                                                    wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)
circ = plt.Circle((0,0),.7,color="white")
plt.gca().add_artist(circ)
plt.title("NAME_CLIENT_TYPE")
plt.show()
```



In [50]:
```
goods = combined_df["NAME_GOODS_CATEGORY"].value_counts().reset_index()
```

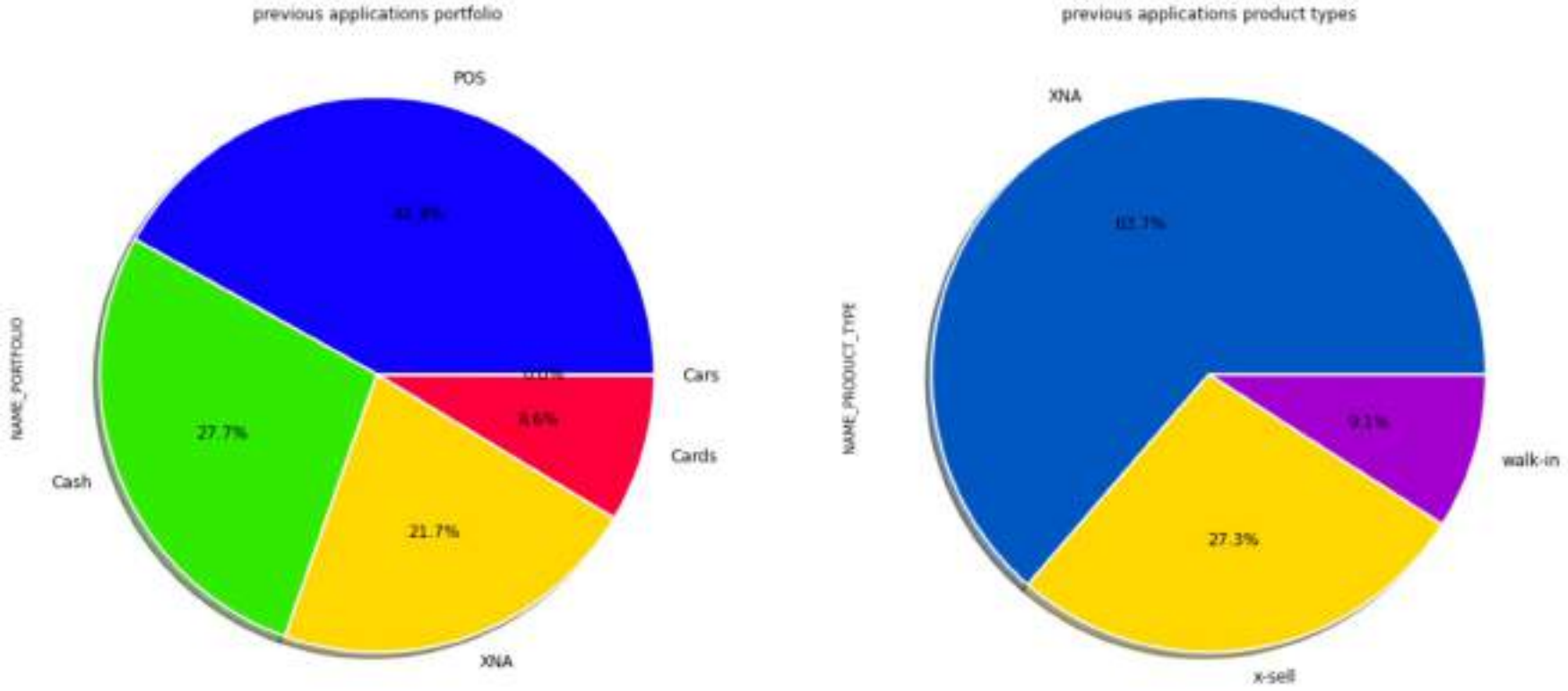
```
goods["percentage"] = round(goods["NAME_GOODS_CATEGORY"]*100/goods["NAME_GOODS_CATEGORY"].sum(),2)
fig = plt.figure(figsize=(12,5))
ax = sns.pointplot("index","percentage",data=goods,color="yellow")
plt.xticks(rotation = 80)
plt.xlabel("NAME_GOODS_CATEGORY")
plt.ylabel("percentage")
plt.title("popular goods for applying loans")
ax.set_facecolor("k")
fig.set_facecolor('lightgrey')
```



In [51]:

```
plt.figure(figsize=(20,20))
plt.subplot(121)
combined_df["NAME_PORTFOLIO"].value_counts().plot.pie(autopct = "%1.1f%%",fontsize=12,
                                                      colors = sns.color_palette("prism",5),
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},
```

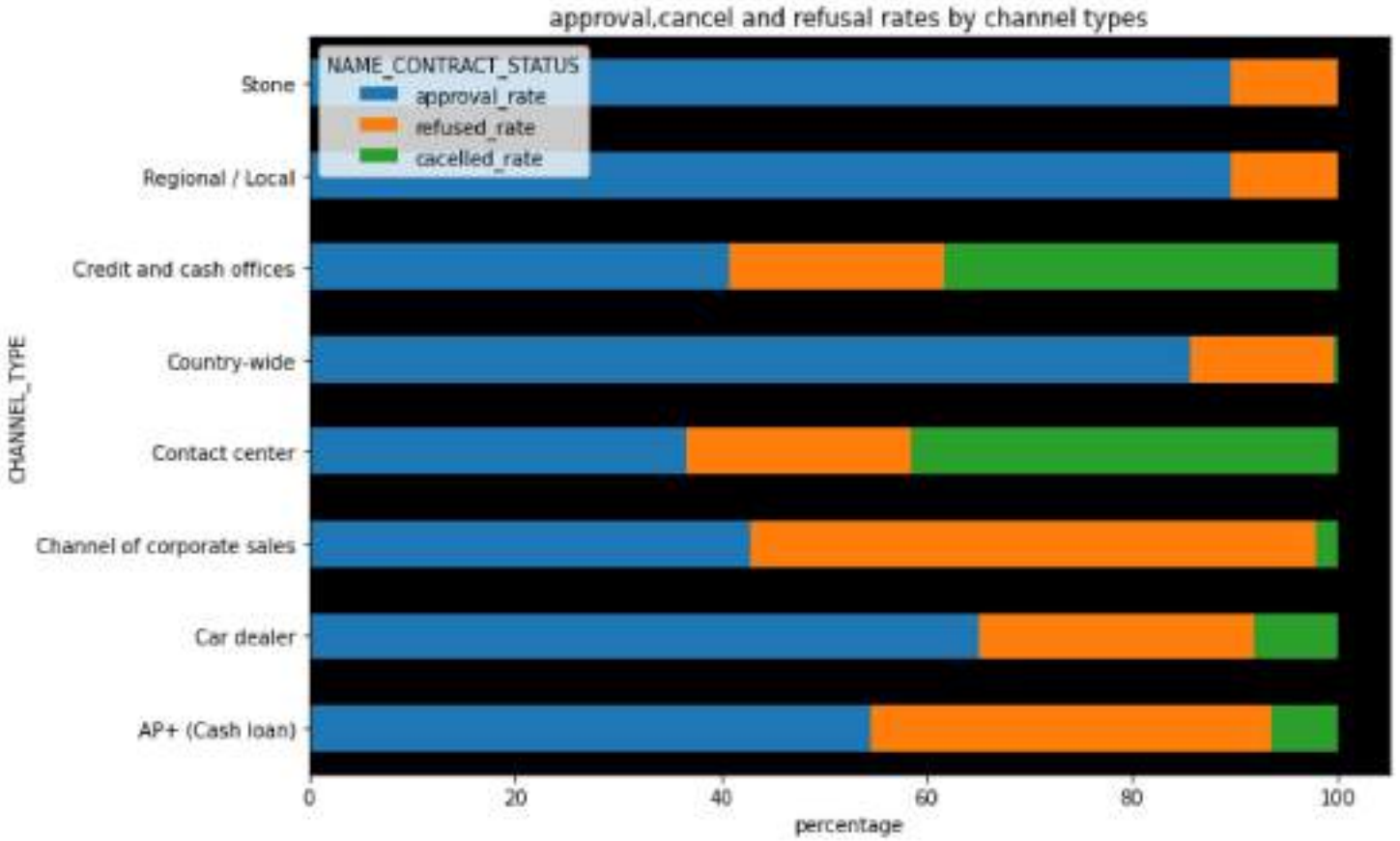
```
                                                        shadow =True)
plt.title("previous applications portfolio")
plt.subplot(122)
combined_df["NAME_PRODUCT_TYPE"].value_counts().plot.pie(autopct = "%1.1f%%",fontsize=12,
                                                         colors = sns.color_palette("prism",3),
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},
                                                         shadow =True)
plt.title("previous applications product types")
plt.show()
```



In [52]:

```
app = pd.crosstab(combined_df["CHANNEL_TYPE"],combined_df["NAME_CONTRACT_STATUS"])
app1 = app
app1["approval_rate"] = app1["Approved"]*100/(app1["Approved"]+app1["Refused"]+app1["Canceled"])
app1["refused_rate"]  = app1["Refused"]*100/(app1["Approved"]+app1["Refused"]+app1["Canceled"])
app1["cacelled_rate"] = app1["Canceled"]*100/(app1["Approved"]+app1["Refused"]+app1["Canceled"])
app2 = app[["approval_rate","refused_rate","cacelled_rate"]]
ax = app2.plot(kind="barh",stacked=True,figsize=(10,7))
ax.set_facecolor("k")
```

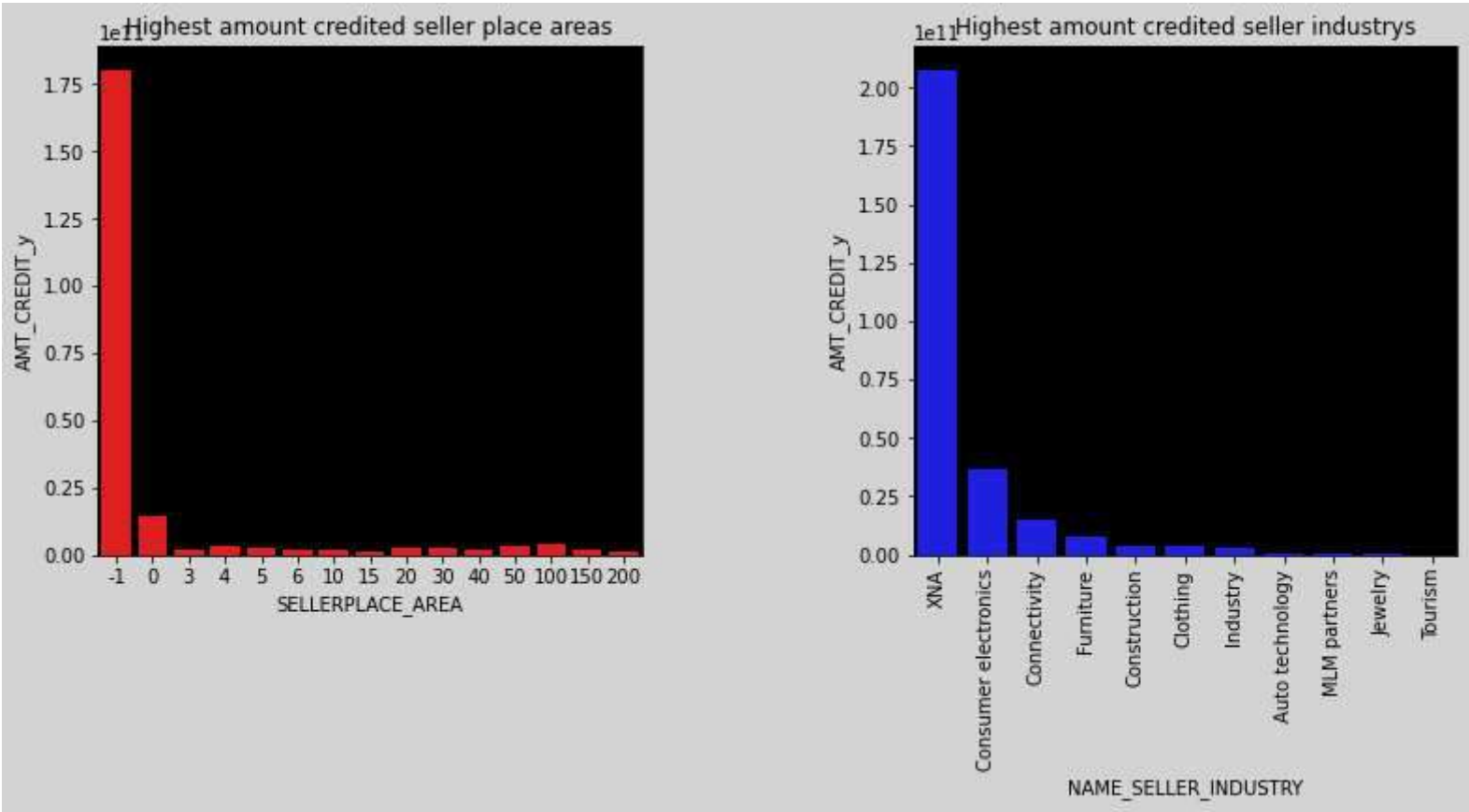
```
ax.set_xlabel("percentage")
ax.set_title("approval,cancel and refusal rates by channel types")
plt.show()
```



In [53]:

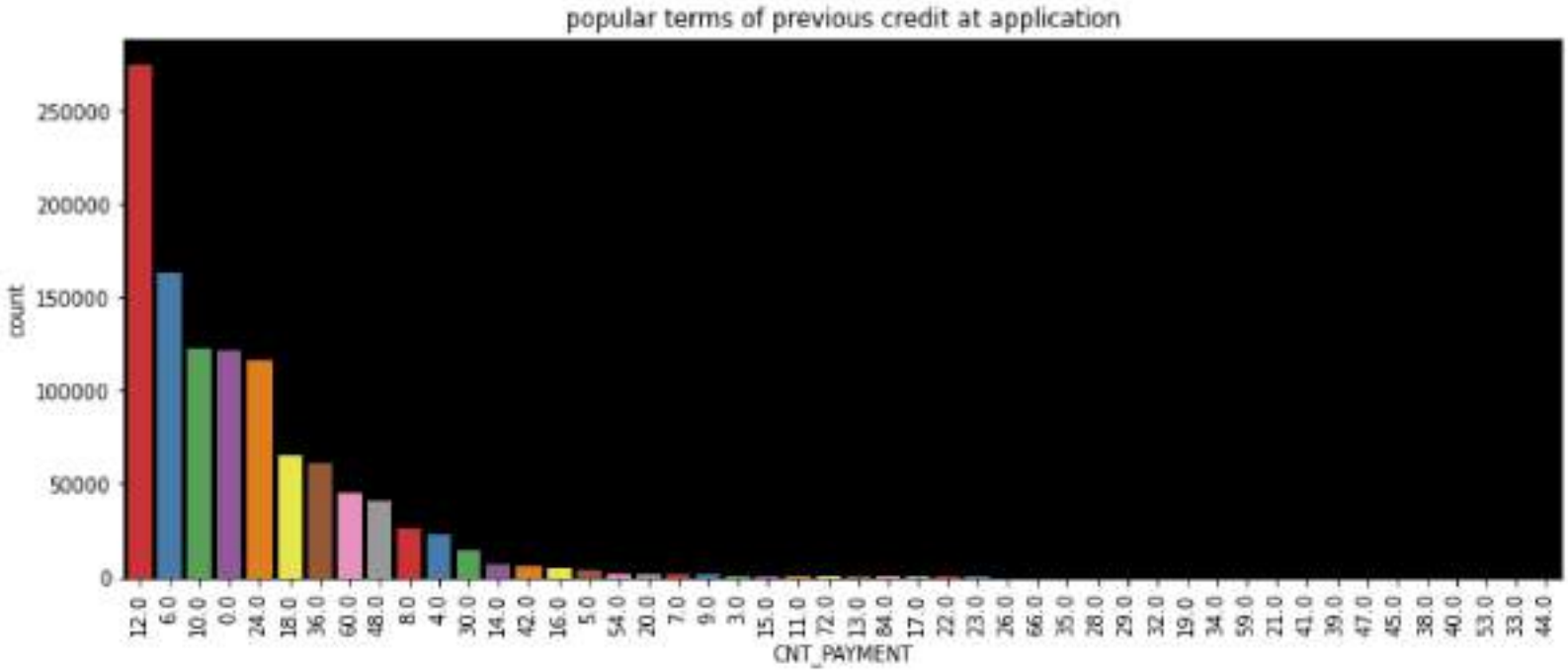
```
fig = plt.figure(figsize=(13,5))
plt.subplot(121)
are = combined_df.groupby("SELLERPLACE_AREA")["AMT_CREDIT_y"].sum().reset_index()
are = are.sort_values(by ="AMT_CREDIT_y",ascending = False)
ax = sns.barplot(y= "AMT_CREDIT_y",x ="SELLERPLACE_AREA",data=are[:15],color="r")
ax.set_facecolor("k")
ax.set_title("Highest amount credited seller place areas")

plt.subplot(122)
sell = combined_df.groupby("NAME_SELLER_INDUSTRY")["AMT_CREDIT_y"].sum().reset_index().sort_values(by = "AMT_CREDIT_y",as
ax1=sns.barplot(y = "AMT_CREDIT_y",x = "NAME_SELLER_INDUSTRY",data=sell,color="b")
ax1.set_facecolor("k")
```

```
ax1.set_title("Highest amount credited seller industrys")
plt.xticks(rotation=90)
plt.subplots_adjust(wspace = .5)
fig.set_facecolor("lightgrey")
```



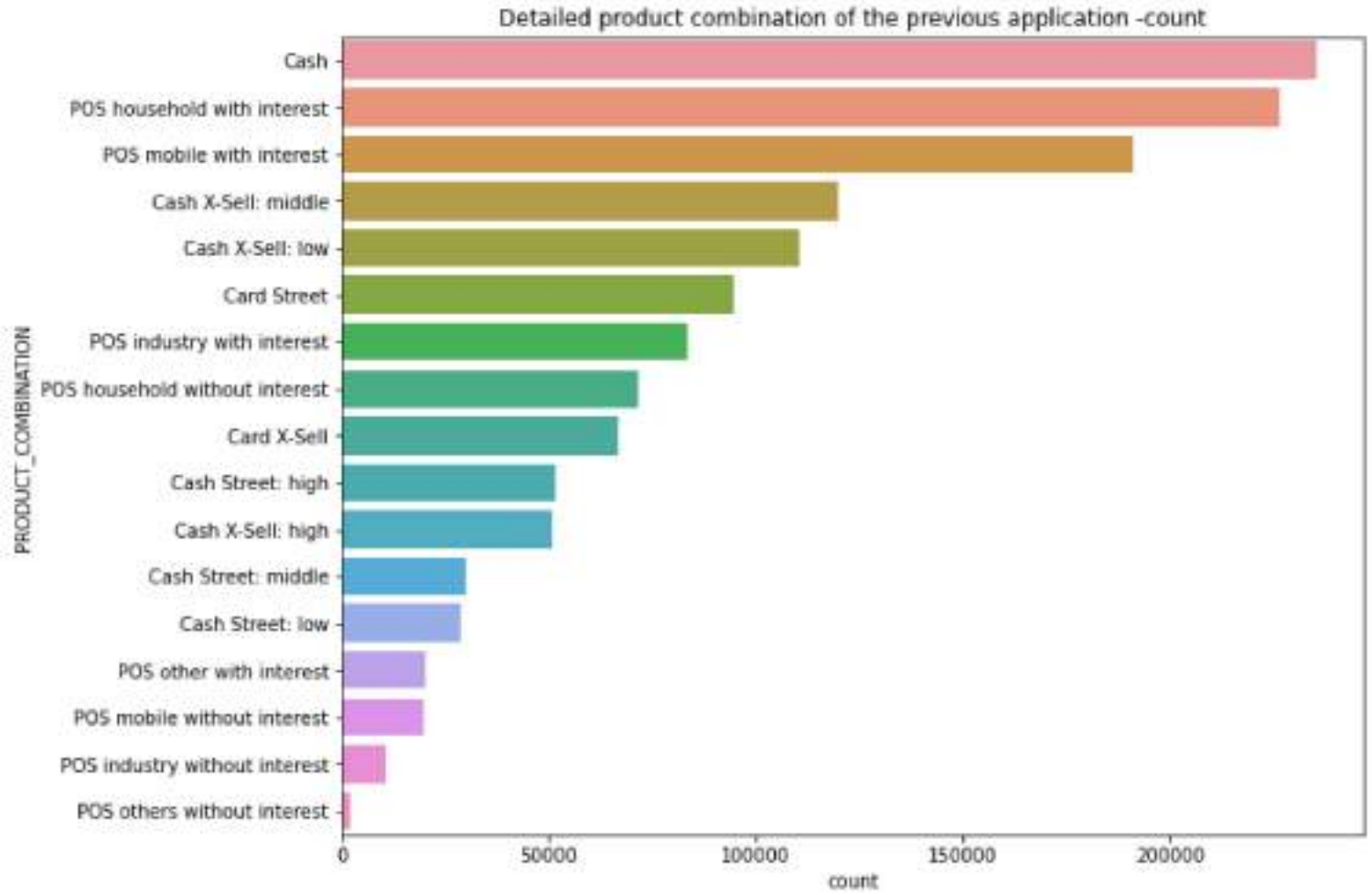
In [54]:

```
plt.figure(figsize=(13,5))
ax = sns.countplot(combined_df["CNT_PAYMENT"],palette="Set1",order=combined_df["CNT_PAYMENT"].value_counts().index)
ax.set_facecolor("k")
plt.xticks(rotation = 90)
plt.title("popular terms of previous credit at application")
plt.show()
```



In [55]:

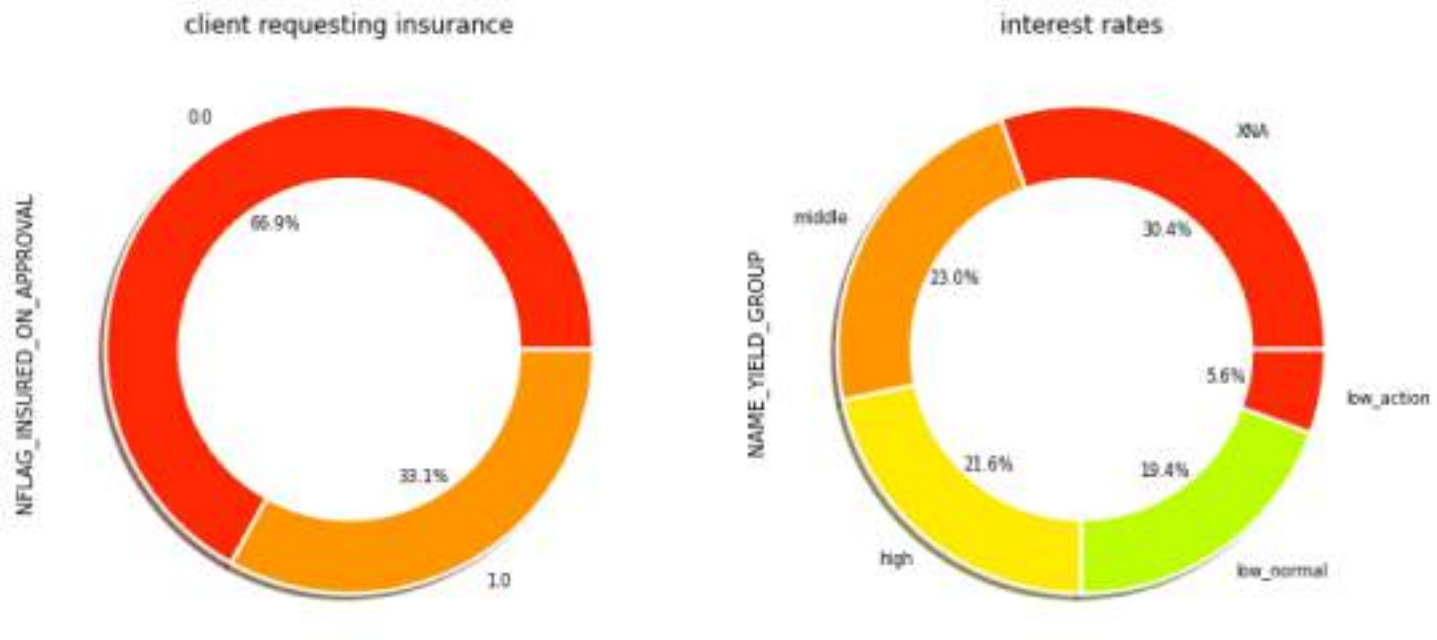
```
plt.figure(figsize=(10,8))
sns.countplot(y = combined_df["PRODUCT_COMBINATION"],order=combined_df["PRODUCT_COMBINATION"].value_counts().index)
plt.title("Detailed product combination of the previous application -count")
plt.show()
```



In [56]:

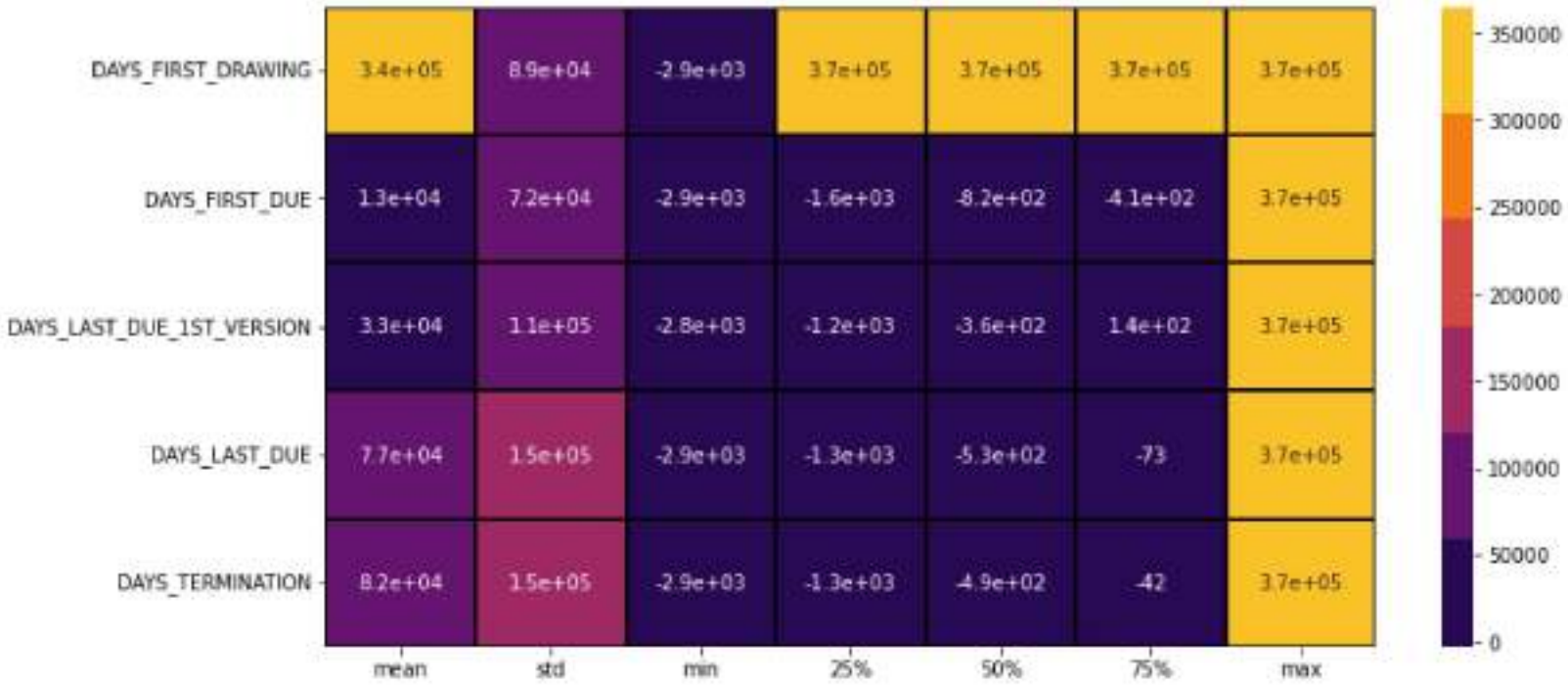
```
plt.figure(figsize=(12,6))
plt.subplot(121)
combined_df["NFLAG_INSURED_ON_APPROVAL"].value_counts().plot.pie(autopct = "%1.1f%%",fontsize=8,
                                                                 colors = sns.color_palette("prism",4),
                                                        wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)
circ = plt.Circle((0,0),.7,color="white")
plt.gca().add_artist(circ)
plt.title("client requesting insurance")

plt.subplot(122)
combined_df["NAME_YIELD_GROUP"].value_counts().plot.pie(autopct = "%1.1f%%",fontsize=8,
                                                                 colors = sns.color_palette("prism",4),
                                                        wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)
```

```
circ = plt.Circle((0,0),.7,color="white")
plt.gca().add_artist(circ)
plt.title("interest rates")
plt.show()
```



In [57]:

```
cols = ['DAYS_FIRST_DRAWING', 'DAYS_FIRST_DUE', 'DAYS_LAST_DUE_1ST_VERSION','DAYS_LAST_DUE', 'DAYS_TERMINATION']
plt.figure(figsize=(12,6))
sns.heatmap(combined_df[cols].describe()[1:].transpose(),
            annot=True,linewidth=2,linecolor="k",cmap=sns.color_palette("inferno"))
plt.show()
```



In [58]:
```
df_repayer = combined_df[combined_df['TARGET'] == 0]
df_defaulter = combined_df[combined_df['TARGET'] == 1]
```

In [59]:
```
corrmat = df_repayer.corr()
corrdf = corrmat.where(np.triu(np.ones(corrmat.shape), k=1).astype(np.bool))
corrdf = corrdf.unstack().reset_index()
corrdf.columns = ['Var1', 'Var2', 'Correlation']
corrdf.dropna(subset = ['Correlation'], inplace = True)
corrdf['Correlation'] = round(corrdf['Correlation'], 2)
corrdf['Correlation'] = abs(corrdf['Correlation'])
corrdf.sort_values(by = 'Correlation', ascending = False).head(10)
```

Out[59]:

	Var1	Var2	Correlation
4786	AMT_GOODS_PRICE_y	AMT_APPLICATION	1.00
984	FLAG_EMP_PHONE	DAYS_EMPLOYED	1.00
2278	OBS_60_CNT_SOCIAL_CIRCLE	OBS_30_CNT_SOCIAL_CIRCLE	1.00

	Var1	Var2	Correlation
454	AMT_GOODS_PRICE_x	AMT_CREDIT_x	0.99
4787	AMT_GOODS_PRICE_y	AMT_CREDIT_y	0.99
4711	AMT_CREDIT_y	AMT_APPLICATION	0.98
1519	REGION_RATING_CLIENT_W_CITY	REGION_RATING_CLIENT	0.94
5547	DAYS_TERMINATION	DAYS_LAST_DUE	0.93
1823	LIVE_REGION_NOT_WORK_REGION	REG_REGION_NOT_WORK_REGION	0.88
1352	CNT_FAM_MEMBERS	CNT_CHILDREN	0.88

In [60]:

```
corrmat = df_defaulter.corr()
corrdf = corrmat.where(np.triu(np.ones(corrmat.shape), k=1).astype(np.bool))
corrdf = corrdf.unstack().reset_index()
corrdf.columns = ['Var1', 'Var2', 'Correlation']
corrdf.dropna(subset = ['Correlation'], inplace = True)
corrdf['Correlation'] = round(corrdf['Correlation'], 2)
corrdf['Correlation'] = abs(corrdf['Correlation'])
corrdf.sort_values(by = 'Correlation', ascending = False).head(10)
```

Out[60]:

	Var1	Var2	Correlation
2278	OBS_60_CNT_SOCIAL_CIRCLE	OBS_30_CNT_SOCIAL_CIRCLE	1.00
4786	AMT_GOODS_PRICE_y	AMT_APPLICATION	1.00
984	FLAG_EMP_PHONE	DAYS_EMPLOYED	1.00
4787	AMT_GOODS_PRICE_y	AMT_CREDIT_y	0.99
454	AMT_GOODS_PRICE_x	AMT_CREDIT_x	0.98
4711	AMT_CREDIT_y	AMT_APPLICATION	0.98
1519	REGION_RATING_CLIENT_W_CITY	REGION_RATING_CLIENT	0.96
5547	DAYS_TERMINATION	DAYS_LAST_DUE	0.94
5394	DAYS_LAST_DUE_1ST_VERSION	DAYS_FIRST_DRAWING	0.89
1352	CNT_FAM_MEMBERS	CNT_CHILDREN	0.89

In [2]:
```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import pandas as pd
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
import itertools
```

In [ ]:
```
```

In [9]:
```
application_data = pd.read_csv(r"F:\stige projects\loan case study\kaggel\application_data.csv")
previous_application = pd.read_csv(r"F:\stige projects\loan case study\kaggel\previous_application.csv")
columns_description = pd.read_csv(r"F:\stige projects\loan case study\kaggel\columns_description.csv",skiprows =1)
```

In [10]:
```
print("applicaiton_data: " ,application_data.shape)
print("previous_application: ",previous_application.shape)
print("columns_description: ",columns_description.shape)
```

```
applicaiton_data:  (307511, 122)
previous_application:  (1670214, 37)
columns_description:  (159, 5)
```

In [11]:
```
pd.set_option("display.max_rows", None, "display.max_columns", None)
display("application_data")
display(application_data.head(3))
```

'application_data'

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL
0	100002	1	Cash loans	M	N	Y	0	202500.0
1	100003	0	Cash loans	F	N	N	0	270000.0
2	100004	0	Revolving loans	M	Y	Y	0	67500.0

In [12]:
```
display("previous_application ")
display(previous_application.head(3))
```

'previous_application '

	SK_ID_PREV	SK_ID_CURR	NAME_CONTRACT_TYPE	AMT_ANNUITY	AMT_APPLICATION	AMT_CREDIT	AMT_DOWN_PAYMENT	AMT_GOODS_P
0	2030495	271877	Consumer loans	1730.430	17145.0	17145.0	0.0	17
1	2802425	108129	Cash loans	25188.615	607500.0	679671.0	NaN	607!
2	2523466	122040	Cash loans	15060.735	112500.0	136444.5	NaN	112!

In [13]:
```
display("columns_description")
columns_description=columns_description.drop(['1'],axis=1)
display(columns_description)
```

'columns_description'

	application_data	SK_ID_CURR	ID of loan in our sample	Unnamed: 4
0	application_data	TARGET	Target variable (1 - client with payment diffi...	NaN
1	application_data	NAME_CONTRACT_TYPE	Identification if loan is cash or revolving	NaN
2	application_data	CODE_GENDER	Gender of the client	NaN
3	application_data	FLAG_OWN_CAR	Flag if the client owns a car	NaN
4	application_data	FLAG_OWN_REALTY	Flag if client owns a house or flat	NaN
5	application_data	CNT_CHILDREN	Number of children the client has	NaN
6	application_data	AMT_INCOME_TOTAL	Income of the client	NaN
7	application_data	AMT_CREDIT	Credit amount of the loan	NaN
8	application_data	AMT_ANNUITY	Loan annuity	NaN
9	application_data	AMT_GOODS_PRICE	For consumer loans it is the price of the good...	NaN
10	application_data	NAME_TYPE_SUITE	Who was accompanying client when he was applyi...	NaN

	application_data	SK_ID_CURR	ID of loan in our sample	Unnamed: 4
11	application_data	NAME_INCOME_TYPE	Clients income type (businessman, working, mat...	NaN
12	application_data	NAME_EDUCATION_TYPE	Level of highest education the client achieved	NaN
13	application_data	NAME_FAMILY_STATUS	Family status of the client	NaN
14	application_data	NAME_HOUSING_TYPE	What is the housing situation of the client (r...	NaN
15	application_data	REGION_POPULATION_RELATIVE	Normalized population of region where client l...	normalized
16	application_data	DAYS_BIRTH	Client's age in days at the time of application	time only relative to the application
17	application_data	DAYS_EMPLOYED	How many days before the application the perso...	time only relative to the application
18	application_data	DAYS_REGISTRATION	How many days before the application did clien...	time only relative to the application
19	application_data	DAYS_ID_PUBLISH	How many days before the application did clien...	time only relative to the application
20	application_data	OWN_CAR_AGE	Age of client's car	NaN
21	application_data	FLAG_MOBIL	Did client provide mobile phone (1=YES, 0=NO)	NaN
22	application_data	FLAG_EMP_PHONE	Did client provide work phone (1=YES, 0=NO)	NaN
23	application_data	FLAG_WORK_PHONE	Did client provide home phone (1=YES, 0=NO)	NaN
24	application_data	FLAG_CONT_MOBILE	Was mobile phone reachable (1=YES, 0=NO)	NaN
25	application_data	FLAG_PHONE	Did client provide home phone (1=YES, 0=NO)	NaN
26	application_data	FLAG_EMAIL	Did client provide email (1=YES, 0=NO)	NaN
27	application_data	OCCUPATION_TYPE	What kind of occupation does the client have	NaN
28	application_data	CNT_FAM_MEMBERS	How many family members does client have	NaN
29	application_data	REGION_RATING_CLIENT	Our rating of the region where client lives (1...	NaN
30	application_data	REGION_RATING_CLIENT_W_CITY	Our rating of the region where client lives wi...	NaN
31	application_data	WEEKDAY_APPR_PROCESS_START	On which day of the week did the client apply ...	NaN
32	application_data	HOUR_APPR_PROCESS_START	Approximately at what hour did the client appl...	rounded
33	application_data	REG_REGION_NOT_LIVE_REGION	Flag if client's permanent address does not ma...	NaN

	application_data	SK_ID_CURR	ID of loan in our sample	Unnamed: 4
34	application_data	REG_REGION_NOT_WORK_REGION	Flag if client's permanent address does not ma...	NaN
35	application_data	LIVE_REGION_NOT_WORK_REGION	Flag if client's contact address does not matc...	NaN
36	application_data	REG_CITY_NOT_LIVE_CITY	Flag if client's permanent address does not ma...	NaN
37	application_data	REG_CITY_NOT_WORK_CITY	Flag if client's permanent address does not ma...	NaN
38	application_data	LIVE_CITY_NOT_WORK_CITY	Flag if client's contact address does not matc...	NaN
39	application_data	ORGANIZATION_TYPE	Type of organization where client works	NaN
40	application_data	EXT_SOURCE_1	Normalized score from external data source	normalized
41	application_data	EXT_SOURCE_2	Normalized score from external data source	normalized
42	application_data	EXT_SOURCE_3	Normalized score from external data source	normalized
43	application_data	APARTMENTS_AVG	Normalized information about building where th...	normalized
44	application_data	BASEMENTAREA_AVG	Normalized information about building where th...	normalized
45	application_data	YEARS_BEGINEXPLUATATION_AVG	Normalized information about building where th...	normalized
46	application_data	YEARS_BUILD_AVG	Normalized information about building where th...	normalized
47	application_data	COMMONAREA_AVG	Normalized information about building where th...	normalized
48	application_data	ELEVATORS_AVG	Normalized information about building where th...	normalized
49	application_data	ENTRANCES_AVG	Normalized information about building where th...	normalized
50	application_data	FLOORSMAX_AVG	Normalized information about building where th...	normalized
51	application_data	FLOORSMIN_AVG	Normalized information about building where th...	normalized
52	application_data	LANDAREA_AVG	Normalized information about building where th...	normalized
53	application_data	LIVINGAPARTMENTS_AVG	Normalized information about building where th...	normalized
54	application_data	LIVINGAREA_AVG	Normalized information about building where th...	normalized
55	application_data	NONLIVINGAPARTMENTS_AVG	Normalized information about building where th...	normalized
56	application_data	NONLIVINGAREA_AVG	Normalized information about building where th...	normalized
57	application_data	APARTMENTS_MODE	Normalized information about building where th...	normalized
58	application_data	BASEMENTAREA_MODE	Normalized information about building where th...	normalized

	application_data	SK_ID_CURR	ID of loan in our sample	Unnamed: 4
59	application_data	YEARS_BEGINEXPLUATATION_MODE	Normalized information about building where th...	normalized
60	application_data	YEARS_BUILD_MODE	Normalized information about building where th...	normalized
61	application_data	COMMONAREA_MODE	Normalized information about building where th...	normalized
62	application_data	ELEVATORS_MODE	Normalized information about building where th...	normalized
63	application_data	ENTRANCES_MODE	Normalized information about building where th...	normalized
64	application_data	FLOORSMAX_MODE	Normalized information about building where th...	normalized
65	application_data	FLOORSMIN_MODE	Normalized information about building where th...	normalized
66	application_data	LANDAREA_MODE	Normalized information about building where th...	normalized
67	application_data	LIVINGAPARTMENTS_MODE	Normalized information about building where th...	normalized
68	application_data	LIVINGAREA_MODE	Normalized information about building where th...	normalized
69	application_data	NONLIVINGAPARTMENTS_MODE	Normalized information about building where th...	normalized
70	application_data	NONLIVINGAREA_MODE	Normalized information about building where th...	normalized
71	application_data	APARTMENTS_MEDI	Normalized information about building where th...	normalized
72	application_data	BASEMENTAREA_MEDI	Normalized information about building where th...	normalized
73	application_data	YEARS_BEGINEXPLUATATION_MEDI	Normalized information about building where th...	normalized
74	application_data	YEARS_BUILD_MEDI	Normalized information about building where th...	normalized
75	application_data	COMMONAREA_MEDI	Normalized information about building where th...	normalized
76	application_data	ELEVATORS_MEDI	Normalized information about building where th...	normalized
77	application_data	ENTRANCES_MEDI	Normalized information about building where th...	normalized
78	application_data	FLOORSMAX_MEDI	Normalized information about building where th...	normalized
79	application_data	FLOORSMIN_MEDI	Normalized information about building where th...	normalized
80	application_data	LANDAREA_MEDI	Normalized information about building where th...	normalized
81	application_data	LIVINGAPARTMENTS_MEDI	Normalized information about building where th...	normalized
82	application_data	LIVINGAREA_MEDI	Normalized information about building where th...	normalized
83	application_data	NONLIVINGAPARTMENTS_MEDI	Normalized information about building where th...	normalized

	application_data	SK_ID_CURR	ID of loan in our sample	Unnamed: 4
84	application_data	NONLIVINGAREA_MEDI	Normalized information about building where th...	normalized
85	application_data	FONDKAPREMONT_MODE	Normalized information about building where th...	normalized
86	application_data	HOUSETYPE_MODE	Normalized information about building where th...	normalized
87	application_data	TOTALAREA_MODE	Normalized information about building where th...	normalized
88	application_data	WALLSMATERIAL_MODE	Normalized information about building where th...	normalized
89	application_data	EMERGENCYSTATE_MODE	Normalized information about building where th...	normalized
90	application_data	OBS_30_CNT_SOCIAL_CIRCLE	How many observation of client's social surrou...	NaN
91	application_data	DEF_30_CNT_SOCIAL_CIRCLE	How many observation of client's social surrou...	NaN
92	application_data	OBS_60_CNT_SOCIAL_CIRCLE	How many observation of client's social surrou...	NaN
93	application_data	DEF_60_CNT_SOCIAL_CIRCLE	How many observation of client's social surrou...	NaN
94	application_data	DAYS_LAST_PHONE_CHANGE	How many days before application did client ch...	NaN
95	application_data	FLAG_DOCUMENT_2	Did client provide document 2	NaN
96	application_data	FLAG_DOCUMENT_3	Did client provide document 3	NaN
97	application_data	FLAG_DOCUMENT_4	Did client provide document 4	NaN
98	application_data	FLAG_DOCUMENT_5	Did client provide document 5	NaN
99	application_data	FLAG_DOCUMENT_6	Did client provide document 6	NaN
100	application_data	FLAG_DOCUMENT_7	Did client provide document 7	NaN
101	application_data	FLAG_DOCUMENT_8	Did client provide document 8	NaN
102	application_data	FLAG_DOCUMENT_9	Did client provide document 9	NaN
103	application_data	FLAG_DOCUMENT_10	Did client provide document 10	NaN
104	application_data	FLAG_DOCUMENT_11	Did client provide document 11	NaN
105	application_data	FLAG_DOCUMENT_12	Did client provide document 12	NaN
106	application_data	FLAG_DOCUMENT_13	Did client provide document 13	NaN
107	application_data	FLAG_DOCUMENT_14	Did client provide document 14	NaN
108	application_data	FLAG_DOCUMENT_15	Did client provide document 15	NaN

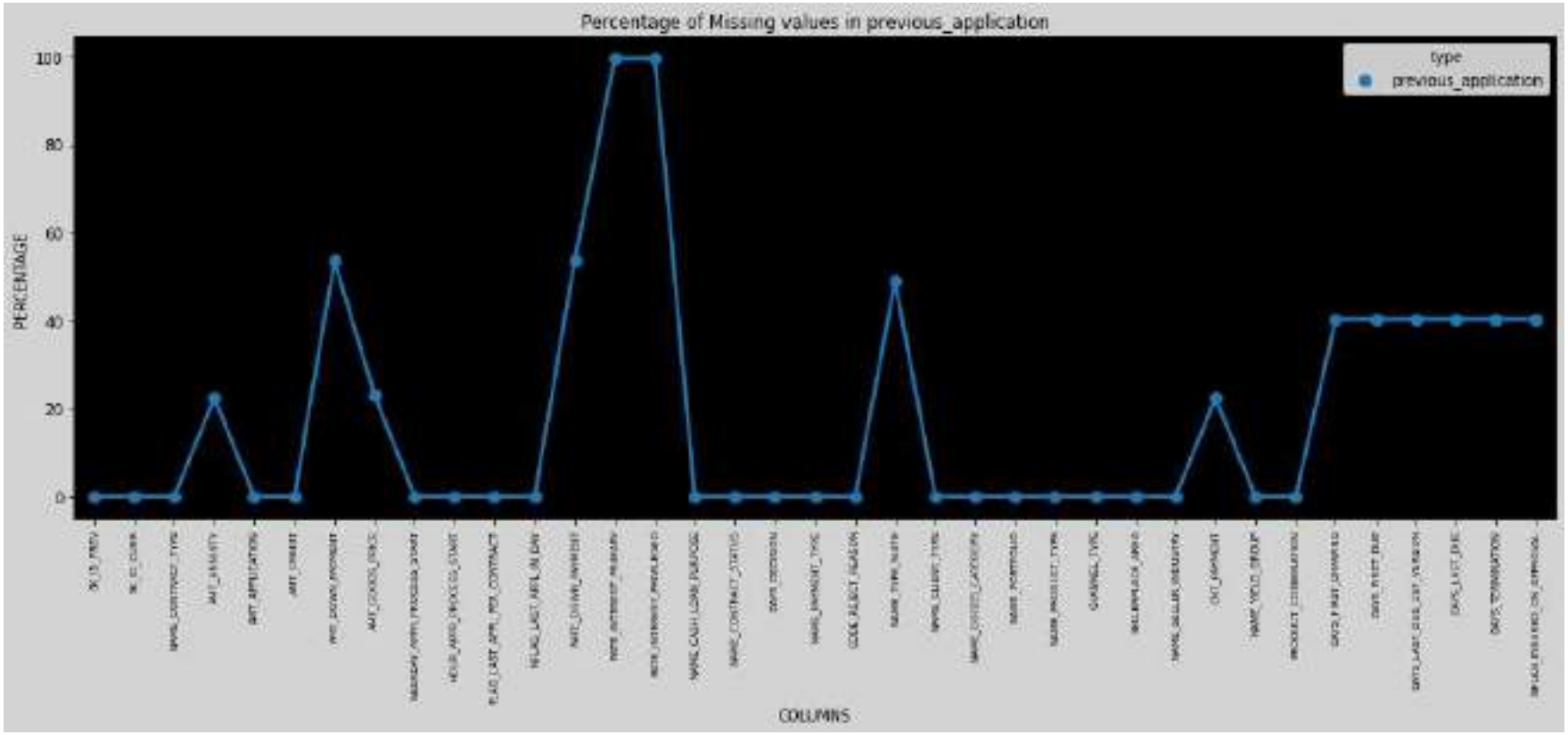
	application_data	SK_ID_CURR	ID of loan in our sample	Unnamed: 4
109	application_data	FLAG_DOCUMENT_16	Did client provide document 16	NaN
110	application_data	FLAG_DOCUMENT_17	Did client provide document 17	NaN
111	application_data	FLAG_DOCUMENT_18	Did client provide document 18	NaN
112	application_data	FLAG_DOCUMENT_19	Did client provide document 19	NaN
113	application_data	FLAG_DOCUMENT_20	Did client provide document 20	NaN
114	application_data	FLAG_DOCUMENT_21	Did client provide document 21	NaN
115	application_data	AMT_REQ_CREDIT_BUREAU_HOUR	Number of enquiries to Credit Bureau about the...	NaN
116	application_data	AMT_REQ_CREDIT_BUREAU_DAY	Number of enquiries to Credit Bureau about the...	NaN
117	application_data	AMT_REQ_CREDIT_BUREAU_WEEK	Number of enquiries to Credit Bureau about the...	NaN
118	application_data	AMT_REQ_CREDIT_BUREAU_MON	Number of enquiries to Credit Bureau about the...	NaN
119	application_data	AMT_REQ_CREDIT_BUREAU_QRT	Number of enquiries to Credit Bureau about the...	NaN
120	application_data	AMT_REQ_CREDIT_BUREAU_YEAR	Number of enquiries to Credit Bureau about the...	NaN
121	previous_application.csv	SK_ID_PREV	ID of previous credit in Home credit related t...	hashed
122	previous_application.csv	SK_ID_CURR	ID of loan in our sample	hashed
123	previous_application.csv	NAME_CONTRACT_TYPE	Contract product type (Cash loan, consumer loa...	NaN
124	previous_application.csv	AMT_ANNUITY	Annuity of previous application	NaN
125	previous_application.csv	AMT_APPLICATION	For how much credit did client ask on the prev...	NaN
126	previous_application.csv	AMT_CREDIT	Final credit amount on the previous applicatio...	NaN
127	previous_application.csv	AMT_DOWN_PAYMENT	Down payment on the previous application	NaN
128	previous_application.csv	AMT_GOODS_PRICE	Goods price of good that client asked for (if ...	NaN
129	previous_application.csv	WEEKDAY_APPR_PROCESS_START	On which day of the week did the client apply ...	NaN
130	previous_application.csv	HOUR_APPR_PROCESS_START	Approximately at what day hour did the client ...	rounded
131	previous_application.csv	FLAG_LAST_APPL_PER_CONTRACT	Flag if it was last application for the previo...	NaN
132	previous_application.csv	NFLAG_LAST_APPL_IN_DAY	Flag if the application was the last applicati...	NaN
133	previous_application.csv	NFLAG_MICRO_CASH	Flag Micro finance loan	NaN

	application_data	SK_ID_CURR	ID of loan in our sample	Unnamed: 4
134	previous_application.csv	RATE_DOWN_PAYMENT	Down payment rate normalized on previous credit	normalized
135	previous_application.csv	RATE_INTEREST_PRIMARY	Interest rate normalized on previous credit	normalized
136	previous_application.csv	RATE_INTEREST_PRIVILEGED	Interest rate normalized on previous credit	normalized
137	previous_application.csv	NAME_CASH_LOAN_PURPOSE	Purpose of the cash loan	NaN
138	previous_application.csv	NAME_CONTRACT_STATUS	Contract status (approved, cancelled, ...) of ...	NaN
139	previous_application.csv	DAYS_DECISION	Relative to current application when was the d...	time only relative to the application
140	previous_application.csv	NAME_PAYMENT_TYPE	Payment method that client chose to pay for th...	NaN
141	previous_application.csv	CODE_REJECT_REASON	Why was the previous application rejected	NaN
142	previous_application.csv	NAME_TYPE_SUITE	Who accompanied client when applying for the p...	NaN
143	previous_application.csv	NAME_CLIENT_TYPE	Was the client old or new client when applying...	NaN
144	previous_application.csv	NAME_GOODS_CATEGORY	What kind of goods did the client apply for in...	NaN
145	previous_application.csv	NAME_PORTFOLIO	Was the previous application for CASH, POS, CA...	NaN
146	previous_application.csv	NAME_PRODUCT_TYPE	Was the previous application x-sell o walk-in	NaN
147	previous_application.csv	CHANNEL_TYPE	Through which channel we acquired the client o...	NaN
148	previous_application.csv	SELLERPLACE_AREA	Selling area of seller place of the previous a...	NaN
149	previous_application.csv	NAME_SELLER_INDUSTRY	The industry of the seller	NaN
150	previous_application.csv	CNT_PAYMENT	Term of previous credit at application of the ...	NaN
151	previous_application.csv	NAME_YIELD_GROUP	Grouped interest rate into small medium and hi...	grouped
152	previous_application.csv	PRODUCT_COMBINATION	Detailed product combination of the previous a...	NaN
153	previous_application.csv	DAYS_FIRST_DRAWING	Relative to application date of current applic...	time only relative to the application
154	previous_application.csv	DAYS_FIRST_DUE	Relative to application date of current applic...	time only relative to the application
155	previous_application.csv	DAYS_LAST_DUE_1ST_VERSION	Relative to application date of current applic...	time only relative to the application

	application_data	SK_ID_CURR	ID of loan in our sample	Unnamed: 4
156	previous_application.csv	DAYS_LAST_DUE	Relative to application date of current applic...	time only relative to the application
157	previous_application.csv	DAYS_TERMINATION	Relative to application date of current applic...	time only relative to the application
158	previous_application.csv	NFLAG_INSURED_ON_APPROVAL	Did the client requested insurance during the ...	NaN

In [14]:

```
fig = plt.figure(figsize=(18,6))
miss_previous_application = pd.DataFrame((previous_application.isnull().sum())*100/previous_application.shape[0]).reset_i
miss_previous_application["type"] = "previous_application"
ax = sns.pointplot("index",0,data=miss_previous_application,hue="type")
plt.xticks(rotation =90,fontsize =7)
plt.title("Percentage of Missing values in previous_application")
plt.ylabel("PERCENTAGE")
plt.xlabel("COLUMNS")
ax.set_facecolor("k")
fig.set_facecolor("lightgrey")
```



In [15]:
```
round(100*(previous_application.isnull().sum()/len(previous_application.index)),2)
```

Out[15]:
```
SK_ID_PREV                    0.00
SK_ID_CURR                    0.00
NAME_CONTRACT_TYPE            0.00
AMT_ANNUITY                  22.29
AMT_APPLICATION               0.00
AMT_CREDIT                    0.00
AMT_DOWN_PAYMENT             53.64
AMT_GOODS_PRICE              23.08
WEEKDAY_APPR_PROCESS_START    0.00
HOUR_APPR_PROCESS_START       0.00
FLAG_LAST_APPL_PER_CONTRACT   0.00
NFLAG_LAST_APPL_IN_DAY        0.00
RATE_DOWN_PAYMENT            53.64
RATE_INTEREST_PRIMARY        99.64
RATE_INTEREST_PRIVILEGED     99.64
NAME_CASH_LOAN_PURPOSE        0.00
```

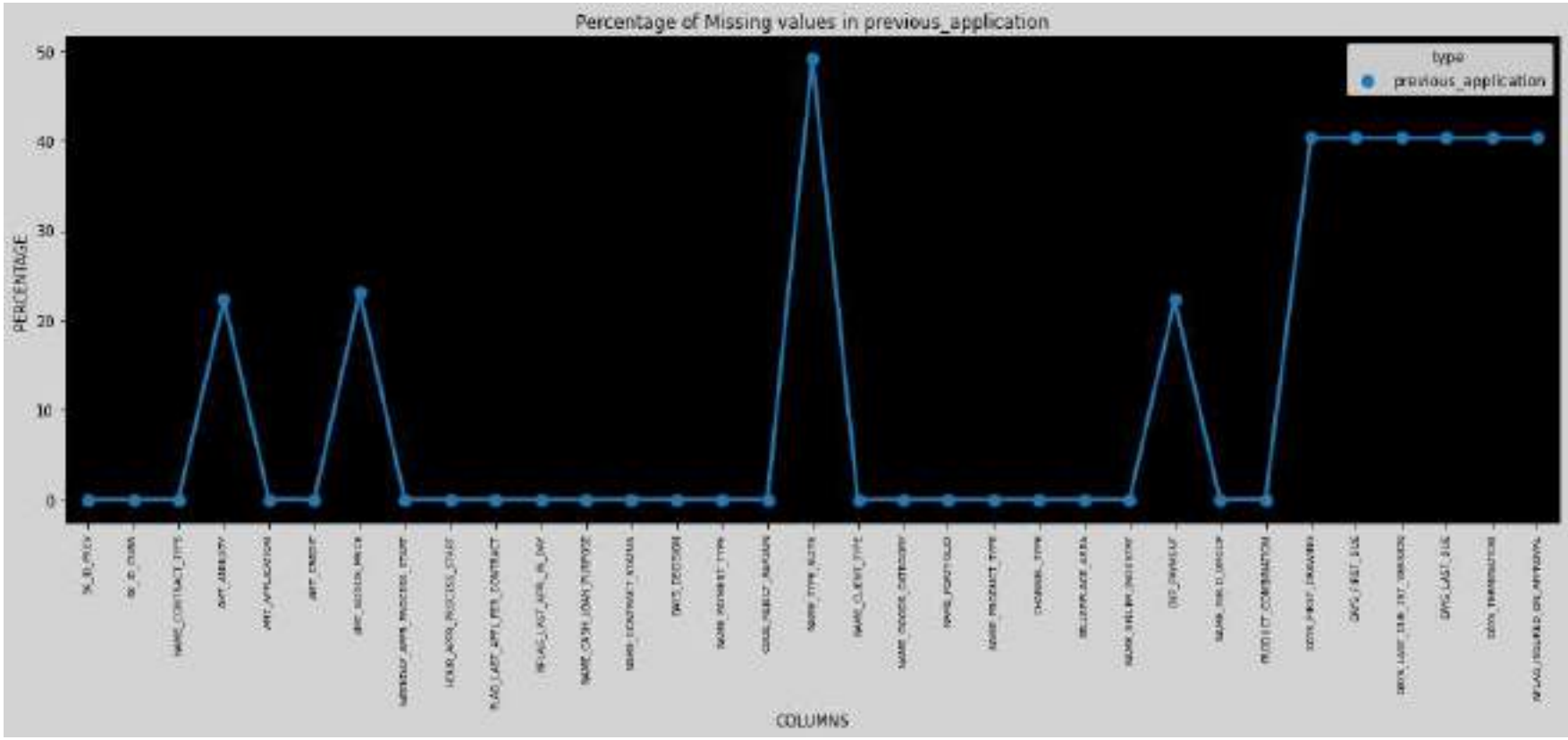
```
NAME_CONTRACT_STATUS           0.00
DAYS_DECISION                  0.00
NAME_PAYMENT_TYPE              0.00
CODE_REJECT_REASON             0.00
NAME_TYPE_SUITE               49.12
NAME_CLIENT_TYPE               0.00
NAME_GOODS_CATEGORY            0.00
NAME_PORTFOLIO                 0.00
NAME_PRODUCT_TYPE              0.00
CHANNEL_TYPE                   0.00
SELLERPLACE_AREA               0.00
NAME_SELLER_INDUSTRY           0.00
CNT_PAYMENT                   22.29
NAME_YIELD_GROUP               0.00
PRODUCT_COMBINATION            0.02
DAYS_FIRST_DRAWING            40.30
DAYS_FIRST_DUE                40.30
DAYS_LAST_DUE_1ST_VERSION     40.30
DAYS_LAST_DUE                 40.30
DAYS_TERMINATION              40.30
NFLAG_INSURED_ON_APPROVAL     40.30
dtype: float64
```

In [16]:

```
previous_application=previous_application.drop([ 'AMT_DOWN_PAYMENT', 'RATE_DOWN_PAYMENT', 'RATE_INTEREST_PRIMARY',
       "RATE_INTEREST_PRIVILEGED"],axis=1)
```

In [17]:

```
fig = plt.figure(figsize=(18,6))
miss_previous_application = pd.DataFrame((previous_application.isnull().sum())*100/previous_application.shape[0]).reset_i
miss_previous_application["type"] = "previous_application"
ax = sns.pointplot("index",0,data=miss_previous_application,hue="type")
plt.xticks(rotation =90,fontsize =7)
plt.title("Percentage of Missing values in previous_application")
plt.ylabel("PERCENTAGE")
plt.xlabel("COLUMNS")
ax.set_facecolor("k")
fig.set_facecolor("lightgrey")
```



In [18]:
```
print("AMT_ANNUITY NULL COUNT:" ,previous_application['AMT_ANNUITY'].isnull().sum())
```

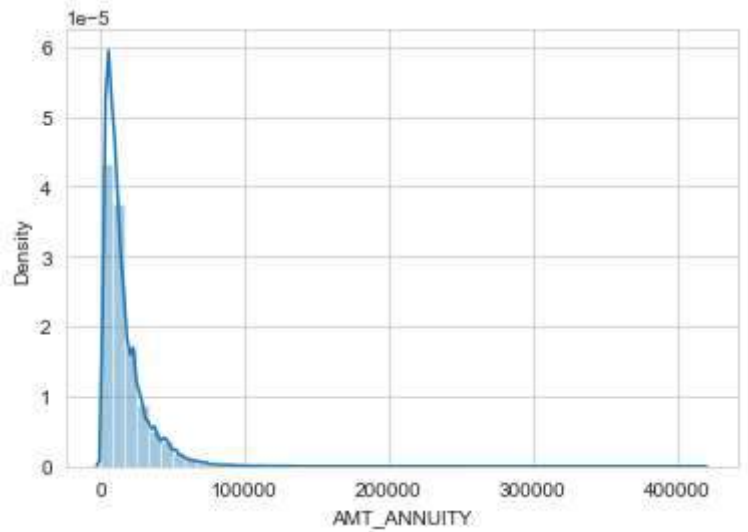
```
AMT_ANNUITY NULL COUNT: 372235
```

In [19]:
```
previous_application['AMT_ANNUITY'].describe()
```

Out[19]:
```
count    1.297979e+06
mean     1.595512e+04
std      1.478214e+04
min      0.000000e+00
25%      6.321780e+03
50%      1.125000e+04
75%      2.065842e+04
max      4.180581e+05
Name: AMT_ANNUITY, dtype: float64
```

In [20]:
```
sns.set_style('whitegrid')
```

```
sns.distplot(previous_application['AMT_ANNUITY'])
plt.show()
```



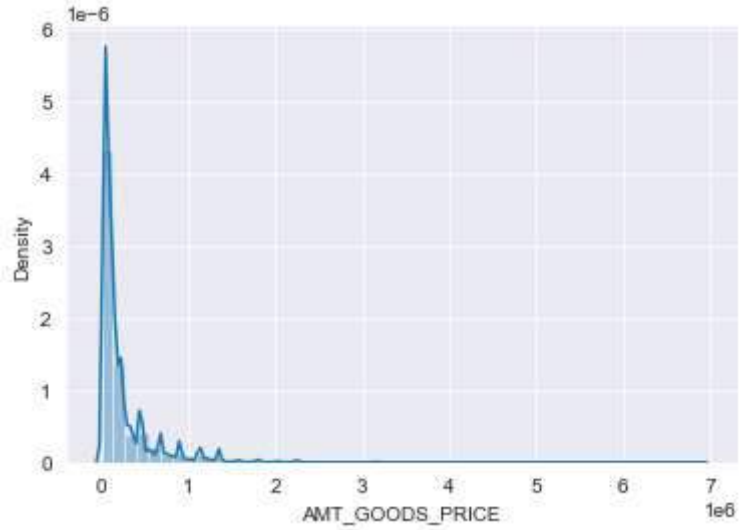
In [21]:
```
print("AMT_GOODS_PRICE NULL COUNT:" ,previous_application['AMT_GOODS_PRICE'].isnull().sum())
```

```
AMT_GOODS_PRICE NULL COUNT: 385515
```

In [22]:
```
previous_application['AMT_GOODS_PRICE'].describe()
```

Out[22]:
```
count    1.284699e+06
mean     2.278473e+05
std      3.153966e+05
min      0.000000e+00
25%      5.084100e+04
50%      1.123200e+05
75%      2.340000e+05
max      6.905160e+06
Name: AMT_GOODS_PRICE, dtype: float64
```

In [23]:
```
sns.set_style('darkgrid')
sns.distplot(previous_application['AMT_GOODS_PRICE'])
plt.show()
```



In [ ]:

In [24]:
```
print("NAME_TYPE_SUITE NULL COUNT:" ,previous_application['NAME_TYPE_SUITE'].isnull().sum())
```

```
NAME_TYPE_SUITE NULL COUNT: 820405
```

In [27]:
```
previous_application['NAME_TYPE_SUITE'].value_counts()
```

Out[27]:
```
Unaccompanied      508970
Family             213263
Spouse, partner     67069
Children            31566
Other_B             17624
Other_A              9077
Group of people      2240
Name: NAME_TYPE_SUITE, dtype: int64
```

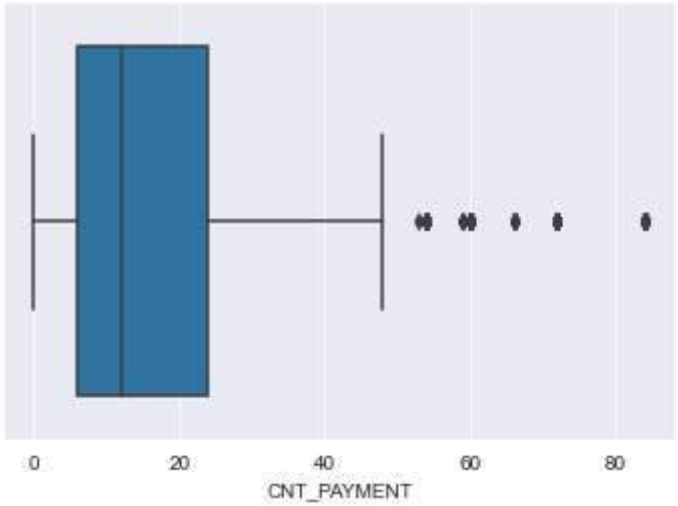
In [28]:
```
print("CNT_PAYMENT NULL COUNT:" ,previous_application['CNT_PAYMENT'].isnull().sum())
```

```
CNT_PAYMENT NULL COUNT: 372230
```

In [25]:
```
previous_application['CNT_PAYMENT'].describe()
```

Out[25]:
```
count    1.297984e+06
mean     1.605408e+01
std      1.456729e+01
min      0.000000e+00
25%      6.000000e+00
50%      1.200000e+01
75%      2.400000e+01
max      8.400000e+01
Name: CNT_PAYMENT, dtype: float64
```

In [26]:
```
sns.set_style('darkgrid')
sns.boxplot(previous_application['CNT_PAYMENT'])
plt.show()
```



In [27]:
```
print("DAYS_FIRST_DRAWING :" ,previous_application['CNT_PAYMENT'].isnull().sum())
```

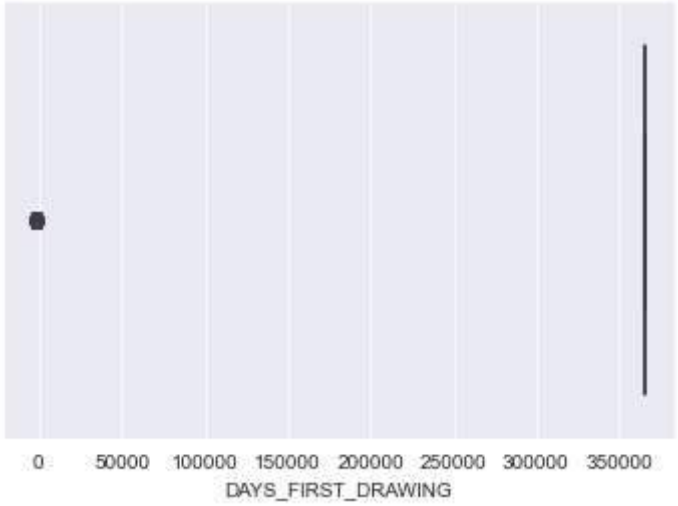
```
DAYS_FIRST_DRAWING : 372230
```

In [28]:
```
previous_application['DAYS_FIRST_DRAWING'].describe()
```

Out[28]:
```
count    997149.000000
mean     342209.855039
std       88916.115834
min       -2922.000000
25%      365243.000000
50%      365243.000000
```

```
75%       365243.000000
max       365243.000000
Name: DAYS_FIRST_DRAWING, dtype: float64
```

In [29]:
```
sns.set_style('darkgrid')
sns.boxplot(previous_application['DAYS_FIRST_DRAWING'])
plt.show()
```



In [30]:
```
print("DAYS_FIRST_DUE :" ,previous_application['DAYS_FIRST_DUE'].isnull().sum())
```

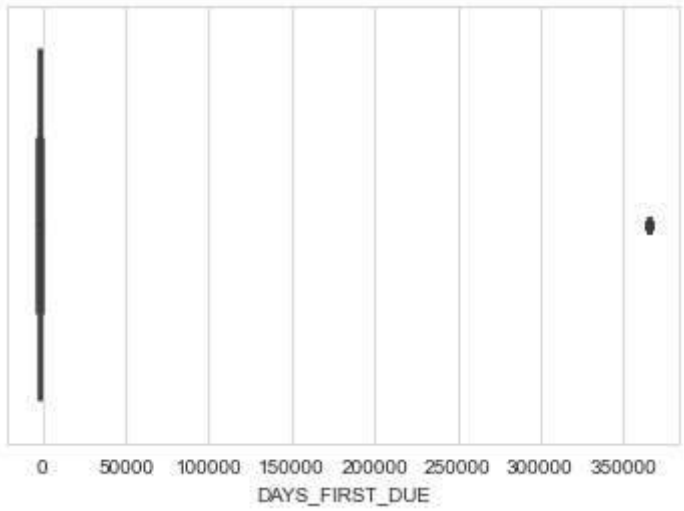
```
DAYS_FIRST_DUE : 673065
```

In [31]:
```
previous_application['DAYS_FIRST_DUE'].describe()
```

Out[31]:
```
count    997149.000000
mean      13826.269337
std       72444.869708
min       -2892.000000
25%       -1628.000000
50%        -831.000000
75%        -411.000000
max      365243.000000
Name: DAYS_FIRST_DUE, dtype: float64
```

In [32]:
```
sns.set_style('whitegrid')
```

```
sns.boxplot(previous_application['DAYS_FIRST_DUE'])
plt.show()
```



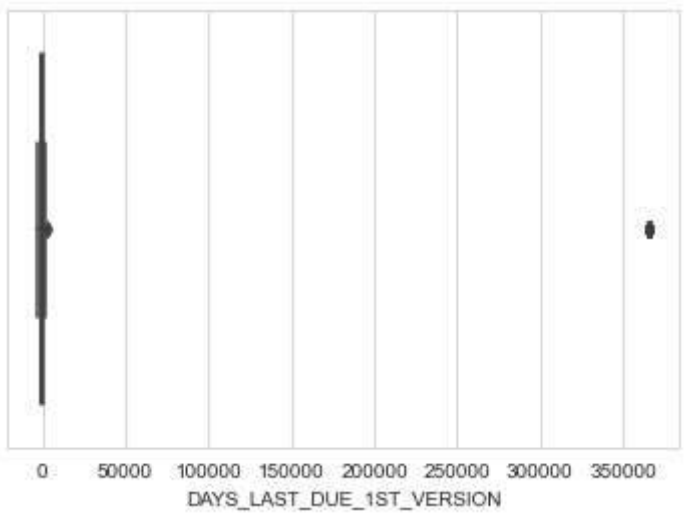
In [33]:
```
print("DAYS_LAST_DUE_1ST_VERSION :" ,previous_application['DAYS_LAST_DUE_1ST_VERSION'].isnull().sum())
```

```
DAYS_LAST_DUE_1ST_VERSION : 673065
```

In [34]:
```
previous_application['DAYS_LAST_DUE_1ST_VERSION'].describe()
```

Out[34]:
```
count    997149.000000
mean      33767.774054
std      106857.034789
min       -2801.000000
25%       -1242.000000
50%        -361.000000
75%         129.000000
max      365243.000000
Name: DAYS_LAST_DUE_1ST_VERSION, dtype: float64
```

In [35]:
```
sns.set_style('whitegrid')
sns.boxplot(previous_application['DAYS_LAST_DUE_1ST_VERSION'])
plt.show()
```



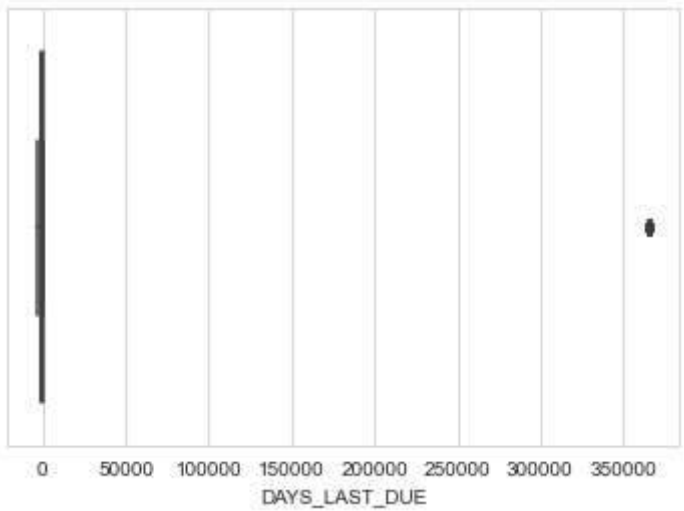
In [36]:
```
print("DAYS_LAST_DUE:" ,previous_application['DAYS_LAST_DUE'].isnull().sum())
```

```
DAYS_LAST_DUE: 673065
```

In [37]:
```
previous_application['DAYS_LAST_DUE'].describe()
```

Out[37]:
```
count    997149.000000
mean      76582.403064
std      149647.415123
min       -2889.000000
25%       -1314.000000
50%        -537.000000
75%         -74.000000
max      365243.000000
Name: DAYS_LAST_DUE, dtype: float64
```

In [38]:
```
sns.set_style('whitegrid')
sns.boxplot(previous_application['DAYS_LAST_DUE'])
plt.show()
```



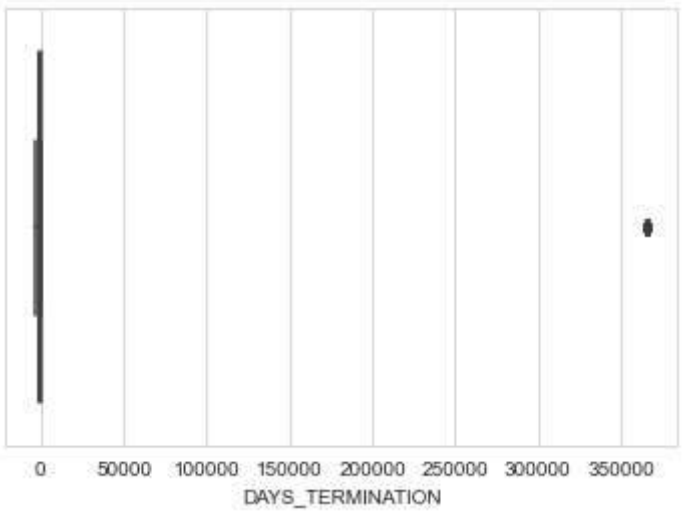
In [39]:
```
print("DAYS_TERMINATION :" ,previous_application['DAYS_TERMINATION'].isnull().sum())
```

```
DAYS_TERMINATION : 673065
```

In [40]:
```
previous_application['DAYS_TERMINATION'].describe()
```

Out[40]:
```
count    997149.000000
mean      81992.343838
std      153303.516729
min       -2874.000000
25%       -1270.000000
50%        -499.000000
75%         -44.000000
max      365243.000000
Name: DAYS_TERMINATION, dtype: float64
```

In [41]:
```
sns.set_style('whitegrid')
sns.boxplot(previous_application['DAYS_TERMINATION'])
plt.show()
```



In [42]:
```
print("NFLAG_INSURED_ON_APPROVAL:" ,previous_application['NFLAG_INSURED_ON_APPROVAL'].isnull().sum())
```

```
NFLAG_INSURED_ON_APPROVAL: 673065
```

In [43]:
```
previous_application['NFLAG_INSURED_ON_APPROVAL'].value_counts()
```

Out[43]:
```
0.0    665527
1.0    331622
Name: NFLAG_INSURED_ON_APPROVAL, dtype: int64
```

In [44]:
```
previous_application.isnull().sum()
```

Out[44]:
```
SK_ID_PREV                        0
SK_ID_CURR                        0
NAME_CONTRACT_TYPE                0
AMT_ANNUITY                  372235
AMT_APPLICATION                   0
AMT_CREDIT                        1
AMT_GOODS_PRICE              385515
WEEKDAY_APPR_PROCESS_START        0
HOUR_APPR_PROCESS_START           0
FLAG_LAST_APPL_PER_CONTRACT       0
NFLAG_LAST_APPL_IN_DAY            0
NAME_CASH_LOAN_PURPOSE            0
NAME_CONTRACT_STATUS              0
DAYS_DECISION                     0
```

```
NAME_PAYMENT_TYPE                 0
CODE_REJECT_REASON                0
NAME_TYPE_SUITE              820405
NAME_CLIENT_TYPE                  0
NAME_GOODS_CATEGORY               0
NAME_PORTFOLIO                    0
NAME_PRODUCT_TYPE                 0
CHANNEL_TYPE                      0
SELLERPLACE_AREA                  0
NAME_SELLER_INDUSTRY              0
CNT_PAYMENT                  372230
NAME_YIELD_GROUP                  0
PRODUCT_COMBINATION             346
DAYS_FIRST_DRAWING           673065
DAYS_FIRST_DUE               673065
DAYS_LAST_DUE_1ST_VERSION    673065
DAYS_LAST_DUE                673065
DAYS_TERMINATION             673065
NFLAG_INSURED_ON_APPROVAL    673065
dtype: int64
```

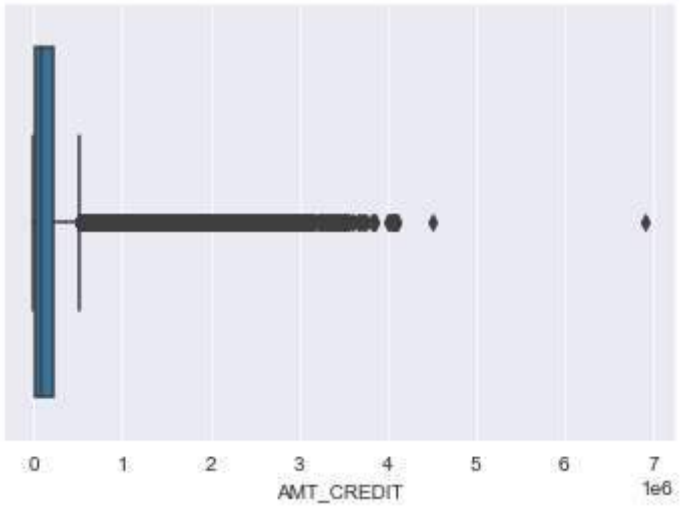
In [45]:
```
print("AMT_CREDIT :" ,previous_application['AMT_CREDIT'].isnull().sum())
```

```
AMT_CREDIT : 1
```

In [46]:
```
previous_application['AMT_CREDIT'].describe()
```

Out[46]:
```
count    1.670213e+06
mean     1.961140e+05
std      3.185746e+05
min      0.000000e+00
25%      2.416050e+04
50%      8.054100e+04
75%      2.164185e+05
max      6.905160e+06
Name: AMT_CREDIT, dtype: float64
```

In [47]:
```
sns.set_style('darkgrid')
sns.boxplot(previous_application['AMT_CREDIT'])
plt.show()
```



In [48]:
```
print("PRODUCT_COMBINATION :" ,previous_application['PRODUCT_COMBINATION'].isnull().sum())
```

```
PRODUCT_COMBINATION : 346
```

In [49]:
```
previous_application['PRODUCT_COMBINATION'].value_counts()
```

Out[49]:
```
Cash                                285990
POS household with interest         263622
POS mobile with interest            220670
Cash X-Sell: middle                 143883
Cash X-Sell: low                    130248
Card Street                         112582
POS industry with interest           98833
POS household without interest       82908
Card X-Sell                          80582
Cash Street: high                    59639
Cash X-Sell: high                    59301
Cash Street: middle                  34658
Cash Street: low                     33834
POS mobile without interest          24082
POS other with interest              23879
POS industry without interest        12602
POS others without interest           2555
Name: PRODUCT_COMBINATION, dtype: int64
```

In [50]:
```
class color:
```

```
    PURPLE = '\033[95m'
    CYAN = '\033[96m'
    DARKCYAN = '\033[36m'
    BLUE = '\033[94m'
    GREEN = '\033[92m'
    YELLOW = '\033[93m'
    RED = '\033[91m'
    BOLD = '\033[1m'
    UNDERLINE = '\033[4m'
    END = '\033[0m'
```

In [51]:
```
obj_dtypes = [i for i in previous_application.select_dtypes(include=np.object).columns if i not in ["type"] ]
num_dtypes = [i for i in previous_application.select_dtypes(include = np.number).columns if i not in ['SK_ID_CURR'] + [ '
```

In [52]:
```
print(color.BOLD + color.PURPLE + 'Categorical Columns' + color.END, "\n")
for x in range(len(obj_dtypes)):
    print(obj_dtypes[x])
```

```
Categorical Columns

NAME_CONTRACT_TYPE
WEEKDAY_APPR_PROCESS_START
FLAG_LAST_APPL_PER_CONTRACT
NAME_CASH_LOAN_PURPOSE
NAME_CONTRACT_STATUS
NAME_PAYMENT_TYPE
CODE_REJECT_REASON
NAME_TYPE_SUITE
NAME_CLIENT_TYPE
NAME_GOODS_CATEGORY
NAME_PORTFOLIO
NAME_PRODUCT_TYPE
CHANNEL_TYPE
NAME_SELLER_INDUSTRY
NAME_YIELD_GROUP
PRODUCT_COMBINATION
```

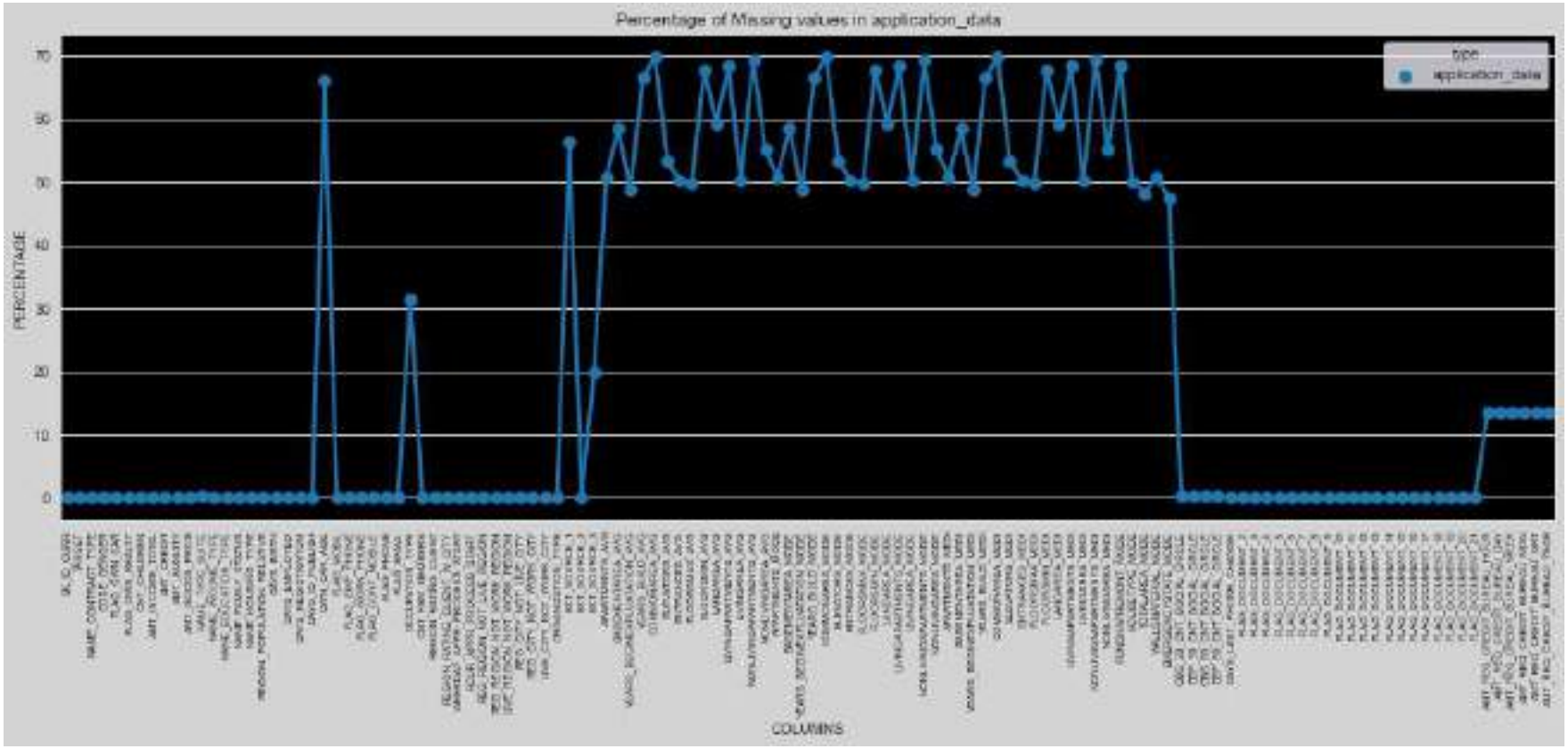
In [53]:
```
print(color.BOLD + color.PURPLE + 'Numerical' + color.END, "\n")
for x in range(len(obj_dtypes)):
    print(obj_dtypes[x])
```

```
Numerical
```

```
NAME_CONTRACT_TYPE
WEEKDAY_APPR_PROCESS_START
FLAG_LAST_APPL_PER_CONTRACT
NAME_CASH_LOAN_PURPOSE
NAME_CONTRACT_STATUS
NAME_PAYMENT_TYPE
CODE_REJECT_REASON
NAME_TYPE_SUITE
NAME_CLIENT_TYPE
NAME_GOODS_CATEGORY
NAME_PORTFOLIO
NAME_PRODUCT_TYPE
CHANNEL_TYPE
NAME_SELLER_INDUSTRY
NAME_YIELD_GROUP
PRODUCT_COMBINATION
```

In [54]:

```
fig = plt.figure(figsize=(18,6))
miss_application_data = pd.DataFrame((application_data.isnull().sum())*100/application_data.shape[0]).reset_index()
miss_application_data["type"] = "application_data"
ax = sns.pointplot("index",0,data=miss_application_data,hue="type")
plt.xticks(rotation =90,fontsize =7)
plt.title("Percentage of Missing values in application_data")
plt.ylabel("PERCENTAGE")
plt.xlabel("COLUMNS")
ax.set_facecolor("k")
fig.set_facecolor("lightgrey")
```



In [55]:
```
round(100*(application_data.isnull().sum()/len(application_data.index)),2)
```

Out[55]:
```
SK_ID_CURR                    0.00
TARGET                        0.00
NAME_CONTRACT_TYPE            0.00
CODE_GENDER                   0.00
FLAG_OWN_CAR                  0.00
FLAG_OWN_REALTY               0.00
CNT_CHILDREN                  0.00
AMT_INCOME_TOTAL              0.00
AMT_CREDIT                    0.00
AMT_ANNUITY                   0.00
AMT_GOODS_PRICE               0.09
NAME_TYPE_SUITE               0.42
NAME_INCOME_TYPE              0.00
NAME_EDUCATION_TYPE           0.00
NAME_FAMILY_STATUS            0.00
NAME_HOUSING_TYPE             0.00
```

REGION_POPULATION_RELATIVE	0.00
DAYS_BIRTH	0.00
DAYS_EMPLOYED	0.00
DAYS_REGISTRATION	0.00
DAYS_ID_PUBLISH	0.00
OWN_CAR_AGE	65.99
FLAG_MOBIL	0.00
FLAG_EMP_PHONE	0.00
FLAG_WORK_PHONE	0.00
FLAG_CONT_MOBILE	0.00
FLAG_PHONE	0.00
FLAG_EMAIL	0.00
OCCUPATION_TYPE	31.35
CNT_FAM_MEMBERS	0.00
REGION_RATING_CLIENT	0.00
REGION_RATING_CLIENT_W_CITY	0.00
WEEKDAY_APPR_PROCESS_START	0.00
HOUR_APPR_PROCESS_START	0.00
REG_REGION_NOT_LIVE_REGION	0.00
REG_REGION_NOT_WORK_REGION	0.00
LIVE_REGION_NOT_WORK_REGION	0.00
REG_CITY_NOT_LIVE_CITY	0.00
REG_CITY_NOT_WORK_CITY	0.00
LIVE_CITY_NOT_WORK_CITY	0.00
ORGANIZATION_TYPE	0.00
EXT_SOURCE_1	56.38
EXT_SOURCE_2	0.21
EXT_SOURCE_3	19.83
APARTMENTS_AVG	50.75
BASEMENTAREA_AVG	58.52
YEARS_BEGINEXPLUATATION_AVG	48.78
YEARS_BUILD_AVG	66.50
COMMONAREA_AVG	69.87
ELEVATORS_AVG	53.30
ENTRANCES_AVG	50.35
FLOORSMAX_AVG	49.76
FLOORSMIN_AVG	67.85
LANDAREA_AVG	59.38
LIVINGAPARTMENTS_AVG	68.35
LIVINGAREA_AVG	50.19
NONLIVINGAPARTMENTS_AVG	69.43
NONLIVINGAREA_AVG	55.18
APARTMENTS_MODE	50.75
BASEMENTAREA_MODE	58.52
YEARS_BEGINEXPLUATATION_MODE	48.78

YEARS_BUILD_MODE	66.50
COMMONAREA_MODE	69.87
ELEVATORS_MODE	53.30
ENTRANCES_MODE	50.35
FLOORSMAX_MODE	49.76
FLOORSMIN_MODE	67.85
LANDAREA_MODE	59.38
LIVINGAPARTMENTS_MODE	68.35
LIVINGAREA_MODE	50.19
NONLIVINGAPARTMENTS_MODE	69.43
NONLIVINGAREA_MODE	55.18
APARTMENTS_MEDI	50.75
BASEMENTAREA_MEDI	58.52
YEARS_BEGINEXPLUATATION_MEDI	48.78
YEARS_BUILD_MEDI	66.50
COMMONAREA_MEDI	69.87
ELEVATORS_MEDI	53.30
ENTRANCES_MEDI	50.35
FLOORSMAX_MEDI	49.76
FLOORSMIN_MEDI	67.85
LANDAREA_MEDI	59.38
LIVINGAPARTMENTS_MEDI	68.35
LIVINGAREA_MEDI	50.19
NONLIVINGAPARTMENTS_MEDI	69.43
NONLIVINGAREA_MEDI	55.18
FONDKAPREMONT_MODE	68.39
HOUSETYPE_MODE	50.18
TOTALAREA_MODE	48.27
WALLSMATERIAL_MODE	50.84
EMERGENCYSTATE_MODE	47.40
OBS_30_CNT_SOCIAL_CIRCLE	0.33
DEF_30_CNT_SOCIAL_CIRCLE	0.33
OBS_60_CNT_SOCIAL_CIRCLE	0.33
DEF_60_CNT_SOCIAL_CIRCLE	0.33
DAYS_LAST_PHONE_CHANGE	0.00
FLAG_DOCUMENT_2	0.00
FLAG_DOCUMENT_3	0.00
FLAG_DOCUMENT_4	0.00
FLAG_DOCUMENT_5	0.00
FLAG_DOCUMENT_6	0.00
FLAG_DOCUMENT_7	0.00
FLAG_DOCUMENT_8	0.00
FLAG_DOCUMENT_9	0.00
FLAG_DOCUMENT_10	0.00
FLAG_DOCUMENT_11	0.00

```
FLAG_DOCUMENT_12                0.00
FLAG_DOCUMENT_13                0.00
FLAG_DOCUMENT_14                0.00
FLAG_DOCUMENT_15                0.00
FLAG_DOCUMENT_16                0.00
FLAG_DOCUMENT_17                0.00
FLAG_DOCUMENT_18                0.00
FLAG_DOCUMENT_19                0.00
FLAG_DOCUMENT_20                0.00
FLAG_DOCUMENT_21                0.00
AMT_REQ_CREDIT_BUREAU_HOUR     13.50
AMT_REQ_CREDIT_BUREAU_DAY      13.50
AMT_REQ_CREDIT_BUREAU_WEEK     13.50
AMT_REQ_CREDIT_BUREAU_MON      13.50
AMT_REQ_CREDIT_BUREAU_QRT      13.50
AMT_REQ_CREDIT_BUREAU_YEAR     13.50
dtype: float64
```

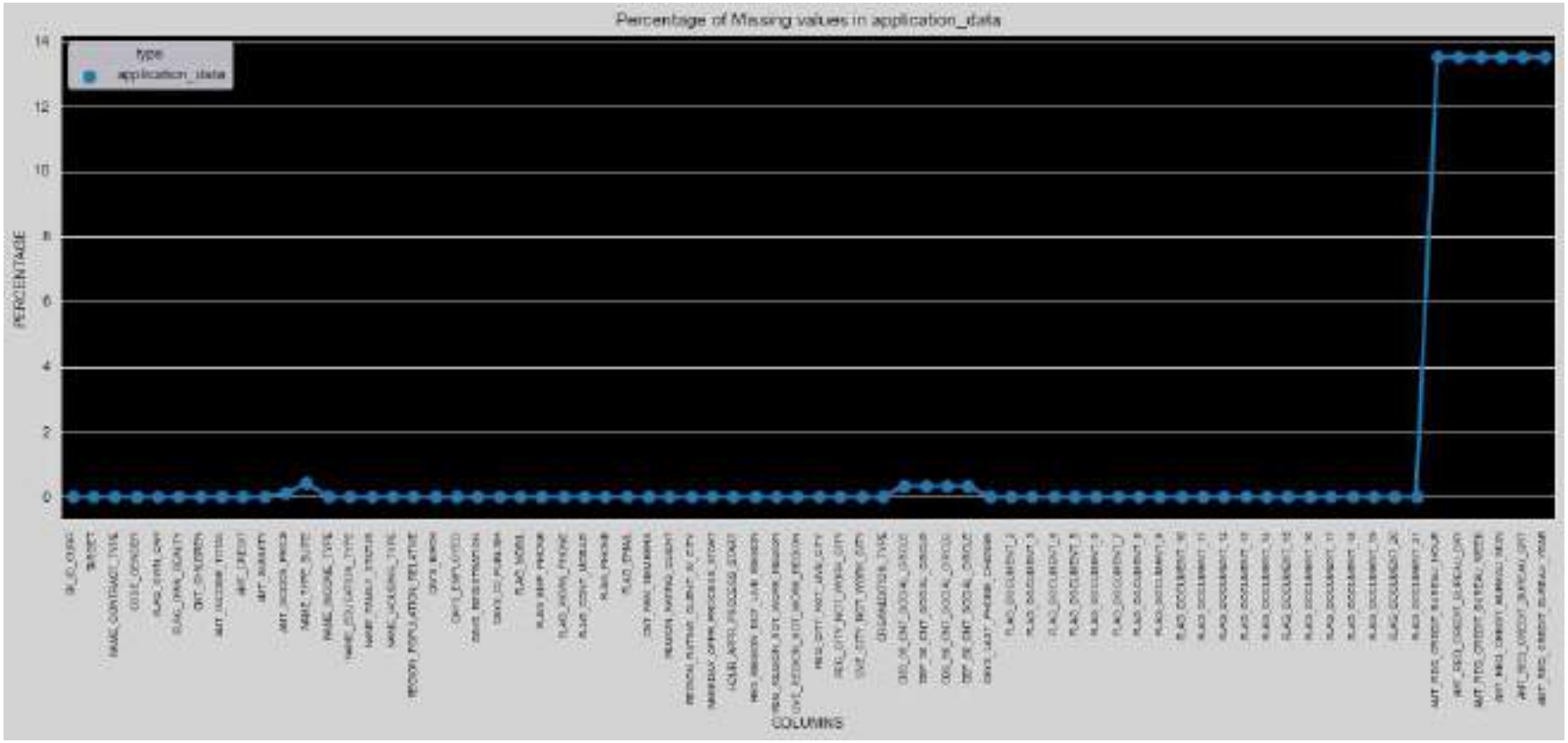
In [56]:

```
application_data=application_data.drop([ 'EXT_SOURCE_1', 'EXT_SOURCE_2', 'EXT_SOURCE_3',
       'APARTMENTS_AVG', 'BASEMENTAREA_AVG', 'YEARS_BEGINEXPLUATATION_AVG',
       'YEARS_BUILD_AVG', 'COMMONAREA_AVG', 'ELEVATORS_AVG', 'ENTRANCES_AVG',
       'FLOORSMAX_AVG', 'FLOORSMIN_AVG', 'LANDAREA_AVG',
       'LIVINGAPARTMENTS_AVG', 'LIVINGAREA_AVG', 'NONLIVINGAPARTMENTS_AVG',
       'NONLIVINGAREA_AVG', 'APARTMENTS_MODE', 'BASEMENTAREA_MODE',
       'YEARS_BEGINEXPLUATATION_MODE', 'YEARS_BUILD_MODE', 'COMMONAREA_MODE',
       'ELEVATORS_MODE', 'ENTRANCES_MODE', 'FLOORSMAX_MODE', 'FLOORSMIN_MODE',
       'LANDAREA_MODE', 'LIVINGAPARTMENTS_MODE', 'LIVINGAREA_MODE',
       'NONLIVINGAPARTMENTS_MODE', 'NONLIVINGAREA_MODE', 'APARTMENTS_MEDI',
       'BASEMENTAREA_MEDI', 'YEARS_BEGINEXPLUATATION_MEDI', 'YEARS_BUILD_MEDI',
       'COMMONAREA_MEDI', 'ELEVATORS_MEDI', 'ENTRANCES_MEDI', 'FLOORSMAX_MEDI',
       'FLOORSMIN_MEDI', 'LANDAREA_MEDI', 'LIVINGAPARTMENTS_MEDI',
       'LIVINGAREA_MEDI', 'NONLIVINGAPARTMENTS_MEDI', 'NONLIVINGAREA_MEDI',
       'FONDKAPREMONT_MODE', 'HOUSETYPE_MODE', 'TOTALAREA_MODE',
       'WALLSMATERIAL_MODE', 'EMERGENCYSTATE_MODE',"OWN_CAR_AGE","OCCUPATION_TYPE"],axis=1)
```

In [57]:

```
fig = plt.figure(figsize=(18,6))
miss_application_data = pd.DataFrame((application_data.isnull().sum())*100/application_data.shape[0]).reset_index()
miss_application_data["type"] = "application_data"
ax = sns.pointplot("index",0,data=miss_application_data,hue="type")
plt.xticks(rotation =90,fontsize =7)
plt.title("Percentage of Missing values in application_data")
plt.ylabel("PERCENTAGE")
plt.xlabel("COLUMNS")
```

```
ax.set_facecolor("k")
fig.set_facecolor("lightgrey")
```



In [58]:
```
round(100*(application_data.isnull().sum()/len(application_data.index)),2)
```

Out[58]:
```
SK_ID_CURR                    0.00
TARGET                        0.00
NAME_CONTRACT_TYPE            0.00
CODE_GENDER                   0.00
FLAG_OWN_CAR                  0.00
FLAG_OWN_REALTY               0.00
CNT_CHILDREN                  0.00
AMT_INCOME_TOTAL              0.00
AMT_CREDIT                    0.00
AMT_ANNUITY                   0.00
AMT_GOODS_PRICE               0.09
NAME_TYPE_SUITE               0.42
NAME_INCOME_TYPE              0.00
```

```
NAME_EDUCATION_TYPE             0.00
NAME_FAMILY_STATUS              0.00
NAME_HOUSING_TYPE               0.00
REGION_POPULATION_RELATIVE      0.00
DAYS_BIRTH                      0.00
DAYS_EMPLOYED                   0.00
DAYS_REGISTRATION               0.00
DAYS_ID_PUBLISH                 0.00
FLAG_MOBIL                      0.00
FLAG_EMP_PHONE                  0.00
FLAG_WORK_PHONE                 0.00
FLAG_CONT_MOBILE                0.00
FLAG_PHONE                      0.00
FLAG_EMAIL                      0.00
CNT_FAM_MEMBERS                 0.00
REGION_RATING_CLIENT            0.00
REGION_RATING_CLIENT_W_CITY     0.00
WEEKDAY_APPR_PROCESS_START      0.00
HOUR_APPR_PROCESS_START         0.00
REG_REGION_NOT_LIVE_REGION      0.00
REG_REGION_NOT_WORK_REGION      0.00
LIVE_REGION_NOT_WORK_REGION     0.00
REG_CITY_NOT_LIVE_CITY          0.00
REG_CITY_NOT_WORK_CITY          0.00
LIVE_CITY_NOT_WORK_CITY         0.00
ORGANIZATION_TYPE               0.00
OBS_30_CNT_SOCIAL_CIRCLE        0.33
DEF_30_CNT_SOCIAL_CIRCLE        0.33
OBS_60_CNT_SOCIAL_CIRCLE        0.33
DEF_60_CNT_SOCIAL_CIRCLE        0.33
DAYS_LAST_PHONE_CHANGE          0.00
FLAG_DOCUMENT_2                 0.00
FLAG_DOCUMENT_3                 0.00
FLAG_DOCUMENT_4                 0.00
FLAG_DOCUMENT_5                 0.00
FLAG_DOCUMENT_6                 0.00
FLAG_DOCUMENT_7                 0.00
FLAG_DOCUMENT_8                 0.00
FLAG_DOCUMENT_9                 0.00
FLAG_DOCUMENT_10                0.00
FLAG_DOCUMENT_11                0.00
FLAG_DOCUMENT_12                0.00
FLAG_DOCUMENT_13                0.00
FLAG_DOCUMENT_14                0.00
FLAG_DOCUMENT_15                0.00
```

```
FLAG_DOCUMENT_16              0.00
FLAG_DOCUMENT_17              0.00
FLAG_DOCUMENT_18              0.00
FLAG_DOCUMENT_19              0.00
FLAG_DOCUMENT_20              0.00
FLAG_DOCUMENT_21              0.00
AMT_REQ_CREDIT_BUREAU_HOUR    13.50
AMT_REQ_CREDIT_BUREAU_DAY     13.50
AMT_REQ_CREDIT_BUREAU_WEEK    13.50
AMT_REQ_CREDIT_BUREAU_MON     13.50
AMT_REQ_CREDIT_BUREAU_QRT     13.50
AMT_REQ_CREDIT_BUREAU_YEAR    13.50
dtype: float64
```

In [59]:
```
print("AMT_REQ_CREDIT_BUREAU_DAY NAN COUNT :" ,application_data['AMT_REQ_CREDIT_BUREAU_DAY'].isnull().sum())
```

```
AMT_REQ_CREDIT_BUREAU_DAY NAN COUNT : 41519
```

In [60]:
```
application_data['AMT_REQ_CREDIT_BUREAU_DAY'].describe()
```

Out[60]:
```
count    265992.000000
mean          0.007000
std           0.110757
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max           9.000000
Name: AMT_REQ_CREDIT_BUREAU_DAY, dtype: float64
```

In [61]:
```
print("AMT_REQ_CREDIT_BUREAU_HOUR NAN COUNT :" ,application_data['AMT_REQ_CREDIT_BUREAU_HOUR'].isnull().sum())
```

```
AMT_REQ_CREDIT_BUREAU_HOUR NAN COUNT : 41519
```

In [62]:
```
application_data['AMT_REQ_CREDIT_BUREAU_HOUR'].describe()
```

Out[62]:
```
count    265992.000000
mean          0.006402
std           0.083849
min           0.000000
25%           0.000000
50%           0.000000
```

```
75%           0.000000
max           4.000000
Name: AMT_REQ_CREDIT_BUREAU_HOUR, dtype: float64
```

In [63]:
```
print("AMT_REQ_CREDIT_BUREAU_MON NAN COUNT :" ,application_data['AMT_REQ_CREDIT_BUREAU_MON'].isnull().sum())
```

```
AMT_REQ_CREDIT_BUREAU_MON NAN COUNT : 41519
```

In [64]:
```
application_data['AMT_REQ_CREDIT_BUREAU_MON'].describe()
```

Out[64]:
```
count    265992.000000
mean          0.267395
std           0.916002
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max          27.000000
Name: AMT_REQ_CREDIT_BUREAU_MON, dtype: float64
```

In [65]:
```
print("AMT_REQ_CREDIT_BUREAU_YEAR NAN COUNT :" ,application_data['AMT_REQ_CREDIT_BUREAU_YEAR'].isnull().sum())
```

```
AMT_REQ_CREDIT_BUREAU_YEAR NAN COUNT : 41519
```

In [66]:
```
application_data['AMT_REQ_CREDIT_BUREAU_YEAR'].describe()
```

Out[66]:
```
count    265992.000000
mean          1.899974
std           1.869295
min           0.000000
25%           0.000000
50%           1.000000
75%           3.000000
max          25.000000
Name: AMT_REQ_CREDIT_BUREAU_YEAR, dtype: float64
```

In [67]:
```
print("DEF_30_CNT_SOCIAL_CIRCLE NAN COUNT :" ,application_data['DEF_30_CNT_SOCIAL_CIRCLE'].isnull().sum())
```

```
DEF_30_CNT_SOCIAL_CIRCLE NAN COUNT : 1021
```

In [68]:
```
application_data['DEF_30_CNT_SOCIAL_CIRCLE'].describe()
```

Out[68]:
```
count    306490.000000
mean          0.143421
std           0.446698
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max          34.000000
Name: DEF_30_CNT_SOCIAL_CIRCLE, dtype: float64
```

In [69]:
```
print("DEF_30_CNT_SOCIAL_CIRCLE :" ,application_data['DEF_30_CNT_SOCIAL_CIRCLE'].isnull().sum())
```

```
DEF_30_CNT_SOCIAL_CIRCLE : 1021
```

In [70]:
```
application_data['DEF_30_CNT_SOCIAL_CIRCLE'].describe()
```

Out[70]:
```
count    306490.000000
mean          0.143421
std           0.446698
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max          34.000000
Name: DEF_30_CNT_SOCIAL_CIRCLE, dtype: float64
```

In [72]:
```
print("OBS_60_CNT_SOCIAL_CIRCLE :" ,application_data['OBS_60_CNT_SOCIAL_CIRCLE'].isnull().sum())
```

```
OBS_60_CNT_SOCIAL_CIRCLE : 1021
```

In [73]:
```
application_data['OBS_60_CNT_SOCIAL_CIRCLE'].describe()
```

Out[73]:
```
count    306490.000000
mean          1.405292
std           2.379803
min           0.000000
25%           0.000000
50%           0.000000
75%           2.000000
max         344.000000
Name: OBS_60_CNT_SOCIAL_CIRCLE, dtype: float64
```

In [74]:
```
print("DEF_60_CNT_SOCIAL_CIRCLE :" ,application_data['DEF_60_CNT_SOCIAL_CIRCLE'].isnull().sum())
```

```
DEF_60_CNT_SOCIAL_CIRCLE : 1021
```

In [75]:
```
application_data['DEF_60_CNT_SOCIAL_CIRCLE'].describe()
```

Out[75]:
```
count    306490.000000
mean          0.100049
std           0.362291
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max          24.000000
Name: DEF_60_CNT_SOCIAL_CIRCLE, dtype: float64
```

In [76]:
```
application_data.isnull().sum()
```

Out[76]:
```
SK_ID_CURR                      0
TARGET                          0
NAME_CONTRACT_TYPE              0
CODE_GENDER                     0
FLAG_OWN_CAR                    0
FLAG_OWN_REALTY                 0
CNT_CHILDREN                    0
AMT_INCOME_TOTAL                0
AMT_CREDIT                      0
AMT_ANNUITY                    12
AMT_GOODS_PRICE               278
NAME_TYPE_SUITE              1292
NAME_INCOME_TYPE                0
NAME_EDUCATION_TYPE             0
NAME_FAMILY_STATUS              0
NAME_HOUSING_TYPE               0
REGION_POPULATION_RELATIVE      0
DAYS_BIRTH                      0
DAYS_EMPLOYED                   0
DAYS_REGISTRATION               0
DAYS_ID_PUBLISH                 0
FLAG_MOBIL                      0
FLAG_EMP_PHONE                  0
FLAG_WORK_PHONE                 0
FLAG_CONT_MOBILE                0
FLAG_PHONE                      0
```

```
FLAG_EMAIL                       0
CNT_FAM_MEMBERS                  2
REGION_RATING_CLIENT             0
REGION_RATING_CLIENT_W_CITY      0
WEEKDAY_APPR_PROCESS_START       0
HOUR_APPR_PROCESS_START          0
REG_REGION_NOT_LIVE_REGION       0
REG_REGION_NOT_WORK_REGION       0
LIVE_REGION_NOT_WORK_REGION      0
REG_CITY_NOT_LIVE_CITY           0
REG_CITY_NOT_WORK_CITY           0
LIVE_CITY_NOT_WORK_CITY          0
ORGANIZATION_TYPE                0
OBS_30_CNT_SOCIAL_CIRCLE      1021
DEF_30_CNT_SOCIAL_CIRCLE      1021
OBS_60_CNT_SOCIAL_CIRCLE      1021
DEF_60_CNT_SOCIAL_CIRCLE      1021
DAYS_LAST_PHONE_CHANGE           1
FLAG_DOCUMENT_2                  0
FLAG_DOCUMENT_3                  0
FLAG_DOCUMENT_4                  0
FLAG_DOCUMENT_5                  0
FLAG_DOCUMENT_6                  0
FLAG_DOCUMENT_7                  0
FLAG_DOCUMENT_8                  0
FLAG_DOCUMENT_9                  0
FLAG_DOCUMENT_10                 0
FLAG_DOCUMENT_11                 0
FLAG_DOCUMENT_12                 0
FLAG_DOCUMENT_13                 0
FLAG_DOCUMENT_14                 0
FLAG_DOCUMENT_15                 0
FLAG_DOCUMENT_16                 0
FLAG_DOCUMENT_17                 0
FLAG_DOCUMENT_18                 0
FLAG_DOCUMENT_19                 0
FLAG_DOCUMENT_20                 0
FLAG_DOCUMENT_21                 0
AMT_REQ_CREDIT_BUREAU_HOUR    41519
AMT_REQ_CREDIT_BUREAU_DAY     41519
AMT_REQ_CREDIT_BUREAU_WEEK    41519
AMT_REQ_CREDIT_BUREAU_MON     41519
AMT_REQ_CREDIT_BUREAU_QRT     41519
AMT_REQ_CREDIT_BUREAU_YEAR    41519
dtype: int64
```

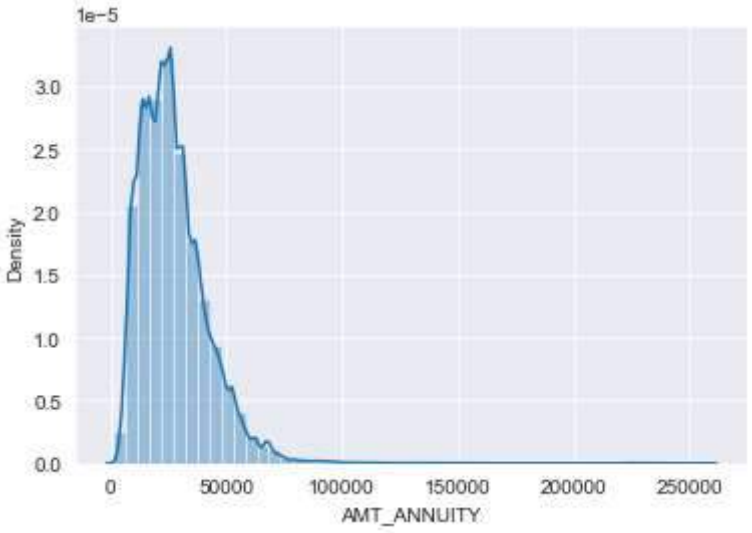
In [77]:
```
print("AMT_ANNUITY  :" ,application_data['AMT_ANNUITY'].isnull().sum())
```

```
AMT_ANNUITY  : 12
```

In [78]:
```
application_data['AMT_ANNUITY'].describe()
```

Out[78]:
```
count    307499.000000
mean      27108.573909
std       14493.737315
min        1615.500000
25%       16524.000000
50%       24903.000000
75%       34596.000000
max      258025.500000
Name: AMT_ANNUITY, dtype: float64
```

In [79]:
```
sns.set_style('darkgrid')
sns.distplot(application_data['AMT_ANNUITY'])
plt.show()
```



In [80]:
```
print("AMT_GOODS_PRICE   :" ,application_data['AMT_GOODS_PRICE'].isnull().sum())
```

```
AMT_GOODS_PRICE   : 278
```

In [81]:
```
application_data['AMT_GOODS_PRICE'].describe()
```

Out[81]:
```
count    3.072330e+05
mean     5.383962e+05
std      3.694465e+05
min      4.050000e+04
25%      2.385000e+05
50%      4.500000e+05
75%      6.795000e+05
max      4.050000e+06
Name: AMT_GOODS_PRICE, dtype: float64
```

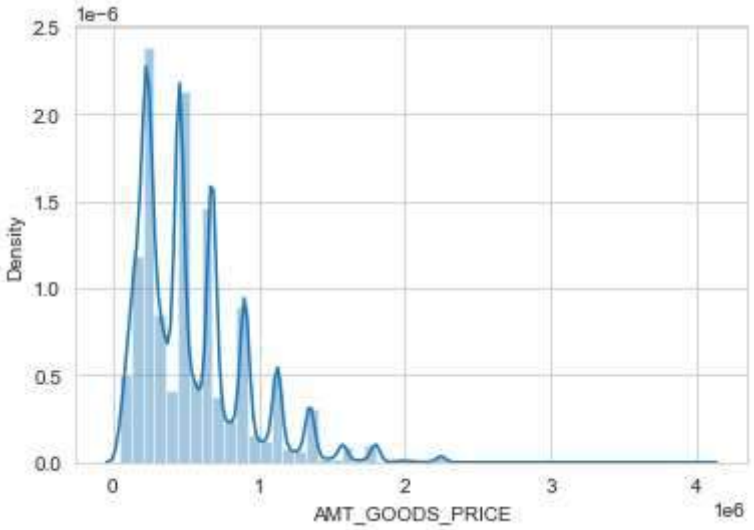
In [82]:
```
print("AMT_GOODS_PRICE   :" ,application_data['AMT_GOODS_PRICE'].isnull().sum())
```

```
AMT_GOODS_PRICE   : 278
```

In [83]:
```
application_data['AMT_GOODS_PRICE'].describe()
```

Out[83]:
```
count    3.072330e+05
mean     5.383962e+05
std      3.694465e+05
min      4.050000e+04
25%      2.385000e+05
50%      4.500000e+05
75%      6.795000e+05
max      4.050000e+06
Name: AMT_GOODS_PRICE, dtype: float64
```

In [84]:
```
sns.set_style('whitegrid')
sns.distplot(application_data['AMT_GOODS_PRICE'])
plt.show()
```



In [85]:
```
print("NAME_TYPE_SUITE :" ,application_data['NAME_TYPE_SUITE'].isnull().sum())
```

```
NAME_TYPE_SUITE : 1292
```

In [86]:
```
application_data['NAME_TYPE_SUITE'].value_counts()
```

Out[86]:
```
Unaccompanied      248526
Family              40149
Spouse, partner     11370
Children             3267
Other_B              1770
Other_A               866
Group of people       271
Name: NAME_TYPE_SUITE, dtype: int64
```

In [87]:
```
print("CNT_FAM_MEMBERS :" ,application_data['CNT_FAM_MEMBERS'].isnull().sum())
```

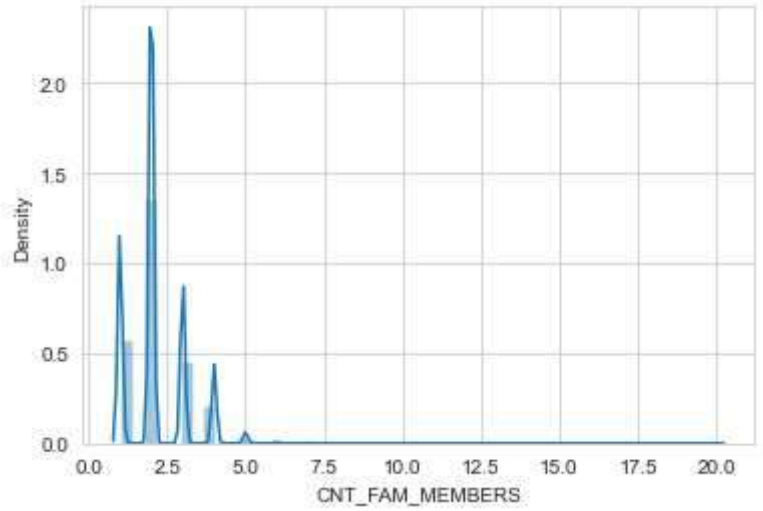
```
CNT_FAM_MEMBERS : 2
```

In [88]:
```
application_data['CNT_FAM_MEMBERS'].describe()
```

Out[88]:
```
count    307509.000000
mean          2.152665
std           0.910682
min           1.000000
```

```
25%           2.000000
50%           2.000000
75%           3.000000
max          20.000000
Name: CNT_FAM_MEMBERS, dtype: float64
```

In [89]:
```
sns.set_style('whitegrid')
sns.distplot(application_data['CNT_FAM_MEMBERS'])
plt.show()
```



In [90]:
```
print("DAYS_LAST_PHONE_CHANGE :" ,application_data['DAYS_LAST_PHONE_CHANGE'].isnull().sum())
```

```
DAYS_LAST_PHONE_CHANGE : 1
```

In [91]:
```
application_data['DAYS_LAST_PHONE_CHANGE'].describe()
```

Out[91]:
```
count    307510.000000
mean       -962.858788
std         826.808487
min       -4292.000000
25%       -1570.000000
50%        -757.000000
75%        -274.000000
max           0.000000
Name: DAYS_LAST_PHONE_CHANGE, dtype: float64
```

In [92]:
```
import statistics
statistics.mode(application_data['DAYS_LAST_PHONE_CHANGE'])
```

Out[92]:
```
0.0
```

In [93]:
```
print(type(application_data.info()))
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Data columns (total 70 columns):
 #   Column                       Non-Null Count   Dtype  
---  ------                       --------------   -----  
 0   SK_ID_CURR                   307511 non-null  int64  
 1   TARGET                       307511 non-null  int64  
 2   NAME_CONTRACT_TYPE           307511 non-null  object 
 3   CODE_GENDER                  307511 non-null  object 
 4   FLAG_OWN_CAR                 307511 non-null  object 
 5   FLAG_OWN_REALTY              307511 non-null  object 
 6   CNT_CHILDREN                 307511 non-null  int64  
 7   AMT_INCOME_TOTAL             307511 non-null  float64
 8   AMT_CREDIT                   307511 non-null  float64
 9   AMT_ANNUITY                  307499 non-null  float64
 10  AMT_GOODS_PRICE              307233 non-null  float64
 11  NAME_TYPE_SUITE              306219 non-null  object 
 12  NAME_INCOME_TYPE             307511 non-null  object 
 13  NAME_EDUCATION_TYPE          307511 non-null  object 
 14  NAME_FAMILY_STATUS           307511 non-null  object 
 15  NAME_HOUSING_TYPE            307511 non-null  object 
 16  REGION_POPULATION_RELATIVE   307511 non-null  float64
 17  DAYS_BIRTH                   307511 non-null  int64  
 18  DAYS_EMPLOYED                307511 non-null  int64  
 19  DAYS_REGISTRATION            307511 non-null  float64
 20  DAYS_ID_PUBLISH              307511 non-null  int64  
 21  FLAG_MOBIL                   307511 non-null  int64  
 22  FLAG_EMP_PHONE               307511 non-null  int64  
 23  FLAG_WORK_PHONE              307511 non-null  int64  
 24  FLAG_CONT_MOBILE             307511 non-null  int64  
 25  FLAG_PHONE                   307511 non-null  int64  
 26  FLAG_EMAIL                   307511 non-null  int64  
 27  CNT_FAM_MEMBERS              307509 non-null  float64
 28  REGION_RATING_CLIENT         307511 non-null  int64  
 29  REGION_RATING_CLIENT_W_CITY  307511 non-null  int64  
 30  WEEKDAY_APPR_PROCESS_START   307511 non-null  object 
```

```
 31  HOUR_APPR_PROCESS_START       307511 non-null  int64
 32  REG_REGION_NOT_LIVE_REGION    307511 non-null  int64
 33  REG_REGION_NOT_WORK_REGION    307511 non-null  int64
 34  LIVE_REGION_NOT_WORK_REGION   307511 non-null  int64
 35  REG_CITY_NOT_LIVE_CITY        307511 non-null  int64
 36  REG_CITY_NOT_WORK_CITY        307511 non-null  int64
 37  LIVE_CITY_NOT_WORK_CITY       307511 non-null  int64
 38  ORGANIZATION_TYPE             307511 non-null  object
 39  OBS_30_CNT_SOCIAL_CIRCLE      306490 non-null  float64
 40  DEF_30_CNT_SOCIAL_CIRCLE      306490 non-null  float64
 41  OBS_60_CNT_SOCIAL_CIRCLE      306490 non-null  float64
 42  DEF_60_CNT_SOCIAL_CIRCLE      306490 non-null  float64
 43  DAYS_LAST_PHONE_CHANGE        307510 non-null  float64
 44  FLAG_DOCUMENT_2               307511 non-null  int64
 45  FLAG_DOCUMENT_3               307511 non-null  int64
 46  FLAG_DOCUMENT_4               307511 non-null  int64
 47  FLAG_DOCUMENT_5               307511 non-null  int64
 48  FLAG_DOCUMENT_6               307511 non-null  int64
 49  FLAG_DOCUMENT_7               307511 non-null  int64
 50  FLAG_DOCUMENT_8               307511 non-null  int64
 51  FLAG_DOCUMENT_9               307511 non-null  int64
 52  FLAG_DOCUMENT_10              307511 non-null  int64
 53  FLAG_DOCUMENT_11              307511 non-null  int64
 54  FLAG_DOCUMENT_12              307511 non-null  int64
 55  FLAG_DOCUMENT_13              307511 non-null  int64
 56  FLAG_DOCUMENT_14              307511 non-null  int64
 57  FLAG_DOCUMENT_15              307511 non-null  int64
 58  FLAG_DOCUMENT_16              307511 non-null  int64
 59  FLAG_DOCUMENT_17              307511 non-null  int64
 60  FLAG_DOCUMENT_18              307511 non-null  int64
 61  FLAG_DOCUMENT_19              307511 non-null  int64
 62  FLAG_DOCUMENT_20              307511 non-null  int64
 63  FLAG_DOCUMENT_21              307511 non-null  int64
 64  AMT_REQ_CREDIT_BUREAU_HOUR    265992 non-null  float64
 65  AMT_REQ_CREDIT_BUREAU_DAY     265992 non-null  float64
 66  AMT_REQ_CREDIT_BUREAU_WEEK    265992 non-null  float64
 67  AMT_REQ_CREDIT_BUREAU_MON     265992 non-null  float64
 68  AMT_REQ_CREDIT_BUREAU_QRT     265992 non-null  float64
 69  AMT_REQ_CREDIT_BUREAU_YEAR    265992 non-null  float64
dtypes: float64(18), int64(41), object(11)
memory usage: 164.2+ MB
<class 'NoneType'>
```

In [94]:

```
application_data['DAYS_BIRTH'] = abs(application_data['DAYS_BIRTH'])
application_data['DAYS_ID_PUBLISH'] = abs(application_data['DAYS_ID_PUBLISH'])
```

```
application_data['DAYS_ID_PUBLISH'] = abs(application_data['DAYS_ID_PUBLISH'])
application_data['DAYS_LAST_PHONE_CHANGE'] = abs(application_data['DAYS_LAST_PHONE_CHANGE'])
```

In [95]:

```
display("application_data")
display(application_data.head())
```

'application_data'

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL
0	100002	1	Cash loans	M	N	Y	0	202500.0
1	100003	0	Cash loans	F	N	N	0	270000.0
2	100004	0	Revolving loans	M	Y	Y	0	67500.0
3	100006	0	Cash loans	F	N	Y	0	135000.0
4	100007	0	Cash loans	M	N	Y	0	121500.0

In [96]:

```
obj_dtypes = [i for i in application_data.select_dtypes(include=np.object).columns if i not in ["type"] ]
num_dtypes = [i for i in application_data.select_dtypes(include = np.number).columns if i not in ['SK_ID_CURR'] + [ 'TARG
```

In [97]:

```
print(color.BOLD + color.PURPLE + 'Categorical Columns' + color.END, "\n")
for x in range(len(obj_dtypes)):
    print(obj_dtypes[x])
```

Categorical Columns

```
NAME_CONTRACT_TYPE
CODE_GENDER
FLAG_OWN_CAR
FLAG_OWN_REALTY
NAME_TYPE_SUITE
NAME_INCOME_TYPE
NAME_EDUCATION_TYPE
NAME_FAMILY_STATUS
```

```
NAME_HOUSING_TYPE
WEEKDAY_APPR_PROCESS_START
ORGANIZATION_TYPE
```

In [98]:

```
print(color.BOLD + color.PURPLE +"Numerical Columns" + color.END, "\n")
for x in range(len(num_dtypes)):
    print(num_dtypes[x])
```

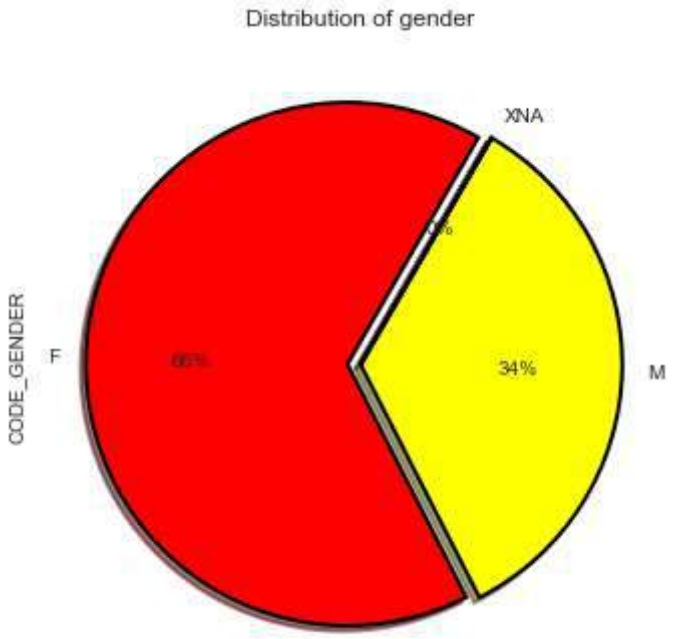
Numerical Columns

```
CNT_CHILDREN
AMT_INCOME_TOTAL
AMT_CREDIT
AMT_ANNUITY
AMT_GOODS_PRICE
REGION_POPULATION_RELATIVE
DAYS_BIRTH
DAYS_EMPLOYED
DAYS_REGISTRATION
DAYS_ID_PUBLISH
FLAG_MOBIL
FLAG_EMP_PHONE
FLAG_WORK_PHONE
FLAG_CONT_MOBILE
FLAG_PHONE
FLAG_EMAIL
CNT_FAM_MEMBERS
REGION_RATING_CLIENT
REGION_RATING_CLIENT_W_CITY
HOUR_APPR_PROCESS_START
REG_REGION_NOT_LIVE_REGION
REG_REGION_NOT_WORK_REGION
LIVE_REGION_NOT_WORK_REGION
REG_CITY_NOT_LIVE_CITY
REG_CITY_NOT_WORK_CITY
LIVE_CITY_NOT_WORK_CITY
OBS_30_CNT_SOCIAL_CIRCLE
DEF_30_CNT_SOCIAL_CIRCLE
OBS_60_CNT_SOCIAL_CIRCLE
DEF_60_CNT_SOCIAL_CIRCLE
DAYS_LAST_PHONE_CHANGE
FLAG_DOCUMENT_2
FLAG_DOCUMENT_3
FLAG_DOCUMENT_4
```

```
FLAG_DOCUMENT_5
FLAG_DOCUMENT_6
FLAG_DOCUMENT_7
FLAG_DOCUMENT_8
FLAG_DOCUMENT_9
FLAG_DOCUMENT_10
FLAG_DOCUMENT_11
FLAG_DOCUMENT_12
FLAG_DOCUMENT_13
FLAG_DOCUMENT_14
FLAG_DOCUMENT_15
FLAG_DOCUMENT_16
FLAG_DOCUMENT_17
FLAG_DOCUMENT_18
FLAG_DOCUMENT_19
FLAG_DOCUMENT_20
FLAG_DOCUMENT_21
AMT_REQ_CREDIT_BUREAU_HOUR
AMT_REQ_CREDIT_BUREAU_DAY
AMT_REQ_CREDIT_BUREAU_WEEK
AMT_REQ_CREDIT_BUREAU_MON
AMT_REQ_CREDIT_BUREAU_QRT
AMT_REQ_CREDIT_BUREAU_YEAR
```

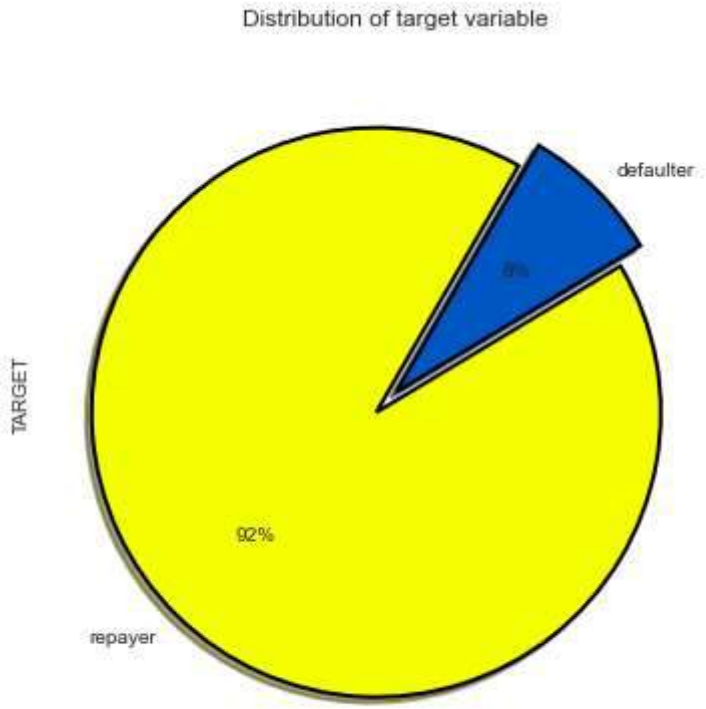
In [99]:

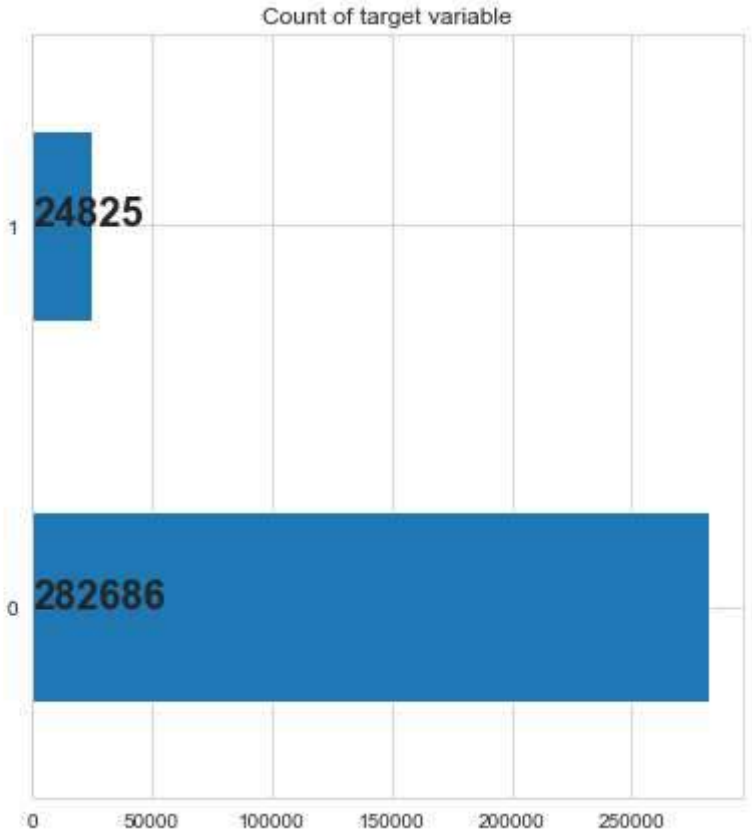
```
fig = plt.figure(figsize=(13,6))
plt.subplot(121)
application_data["CODE_GENDER"].value_counts().plot.pie(autopct = "%1.0f%%",colors = ["red","yellow"],startangle = 60,
                                                        wedgeprops={"linewidth":2,"edgecolor":"k"},explod
plt.title("Distribution of gender")
plt.show()
```



In [100...

```
plt.figure(figsize=(14,7))
plt.subplot(121)
application_data["TARGET"].value_counts().plot.pie(autopct = "%1.0f%%",colors = sns.color_palette("prism",7),startangle =
                                                   wedgeprops={"linewidth":2,"edgecolor":"k"},explod
plt.title("Distribution of target variable")

plt.subplot(122)
ax = application_data["TARGET"].value_counts().plot(kind="barh")

for i,j in enumerate(application_data["TARGET"].value_counts().values):
    ax.text(.7,i,j,weight = "bold",fontsize=20)

plt.title("Count of target variable")
plt.show()
```

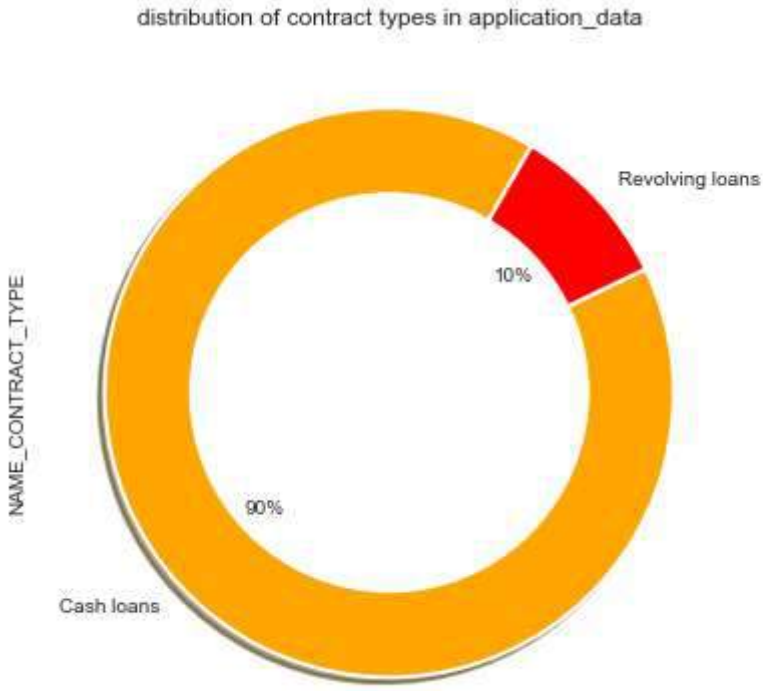


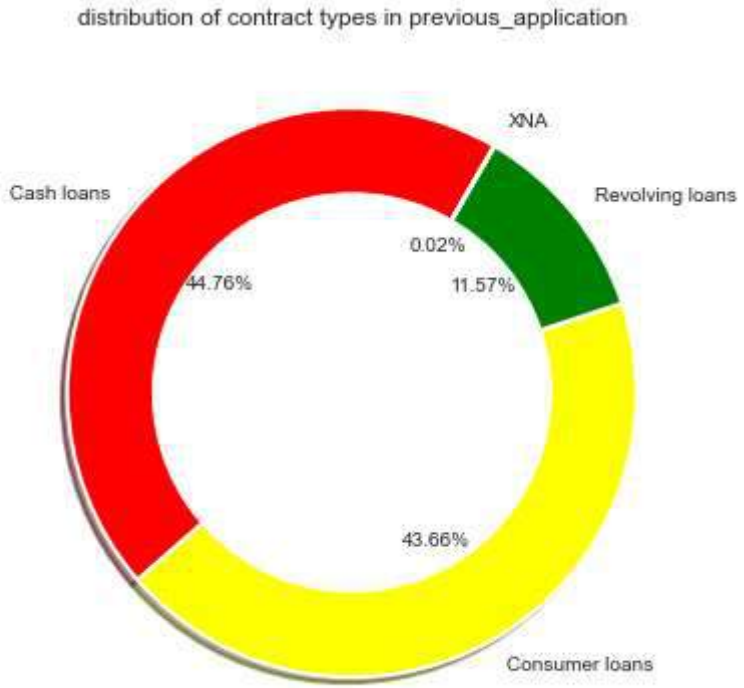


In [181...
```
application_data_x = application_data[[x for x in application_data.columns if x not in ["TARGET"]]]
previous_application_x = previous_application[[x for x in previous_application.columns if x not in ["TARGET"]]]
application_data_x["type"] = "application_data"
previous_application_x["type"] = "previous_application"
data = pd.concat([application_data_x,previous_application_x],axis=0)
```

In [182...
```
plt.figure(figsize=(14,7))
plt.subplot(121)
data[data["type"] == "application_data"]["NAME_CONTRACT_TYPE"].value_counts().plot.pie(autopct = "%1.0f%%",colors = ["ora
                                                                                    wedgeprops={"linewidth":2,"edgecolor":"white"},sh
circ = plt.Circle((0,0),.7,color="white")
plt.gca().add_artist(circ)
plt.title("distribution of contract types in application_data")

plt.subplot(122)
data[data["type"] == "previous_application"]["NAME_CONTRACT_TYPE"].value_counts().plot.pie(autopct = "%1.2f%%",colors = [
```

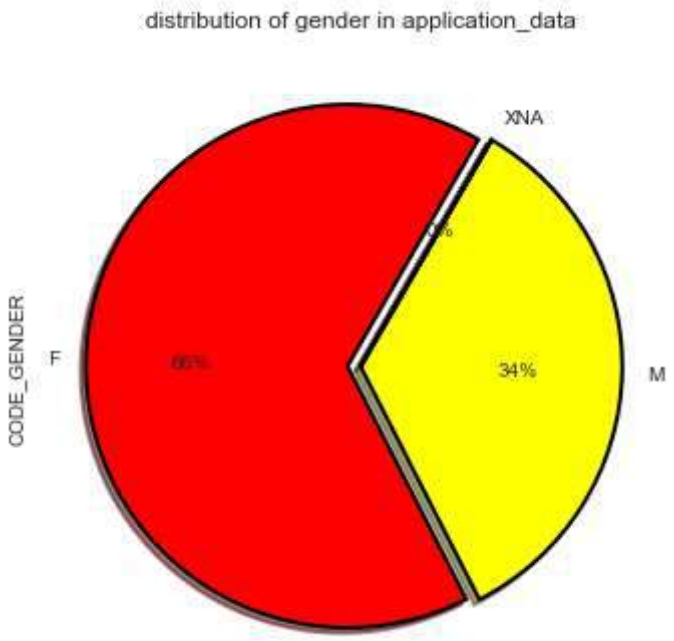
```
                                                                                wedgeprops={"linewidth":2,"edgecolor":"white"},sh
circ = plt.Circle((0,0),.7,color="white")
plt.gca().add_artist(circ)
plt.ylabel("")
plt.title("distribution of contract types in previous_application")
plt.show()

plt.show()
```





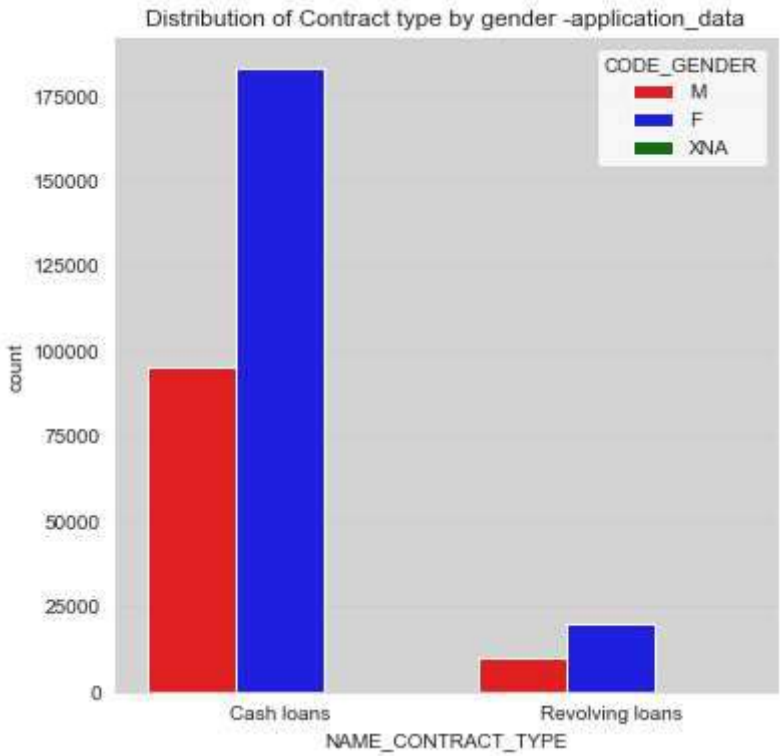
In [183...

```
fig = plt.figure(figsize=(13,6))
plt.subplot(121)
data[data["type"] == "application_data"]["CODE_GENDER"].value_counts().plot.pie(autopct = "%1.0f%%",colors = ["red","yell
                                                                                wedgeprops={"linewidth":2,"edgecolor":"k"},explod
plt.title("distribution of gender in application_data")
plt.show()
```



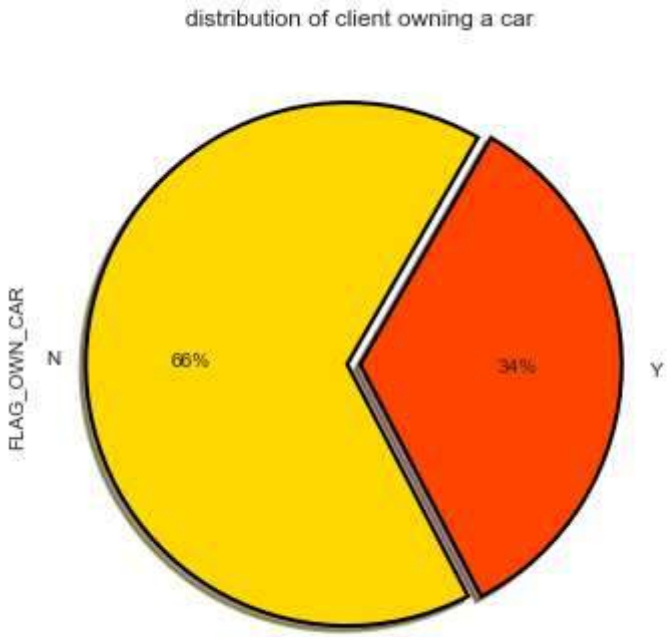
In [104...

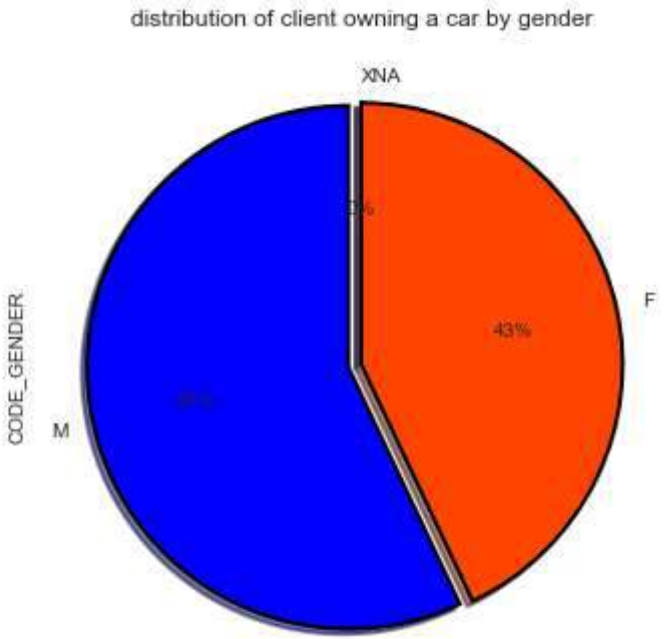
```
fig  = plt.figure(figsize=(13,6))
plt.subplot(121)
ax = sns.countplot("NAME_CONTRACT_TYPE",hue="CODE_GENDER",data=data[data["type"] == "application_data"],palette=["r","b",
ax.set_facecolor("lightgrey")
ax.set_title("Distribution of Contract type by gender -application_data")


plt.show()
```



In [106...

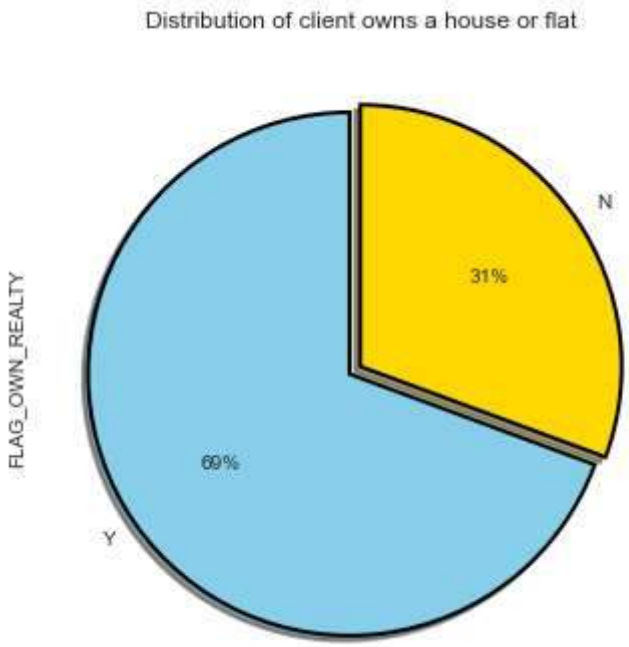
```
fig = plt.figure(figsize=(13,6))

plt.subplot(121)
data["FLAG_OWN_CAR"].value_counts().plot.pie(autopct = "%1.0f%%",colors = ["gold","orangered"],startangle = 60,
                                                                    wedgeprops={"linewidth":2,"edgecolor":"k"},explod
plt.title("distribution of client owning a car")

plt.subplot(122)
data[data["FLAG_OWN_CAR"] == "Y"]["CODE_GENDER"].value_counts().plot.pie(autopct = "%1.0f%%",colors = ["b","orangered"],s
                                                                    wedgeprops={"linewidth":2,"edgecolor":"k"},explod
plt.title("distribution of client owning a car by gender")

plt.show()
```

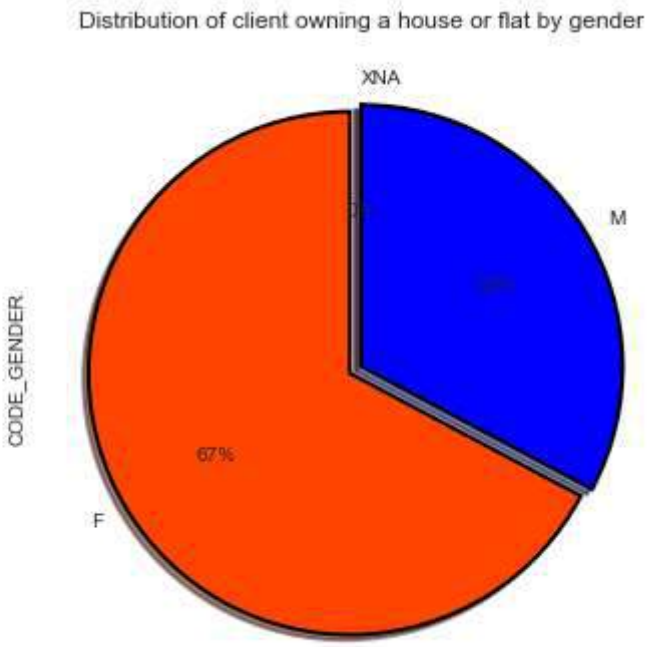




In [107...

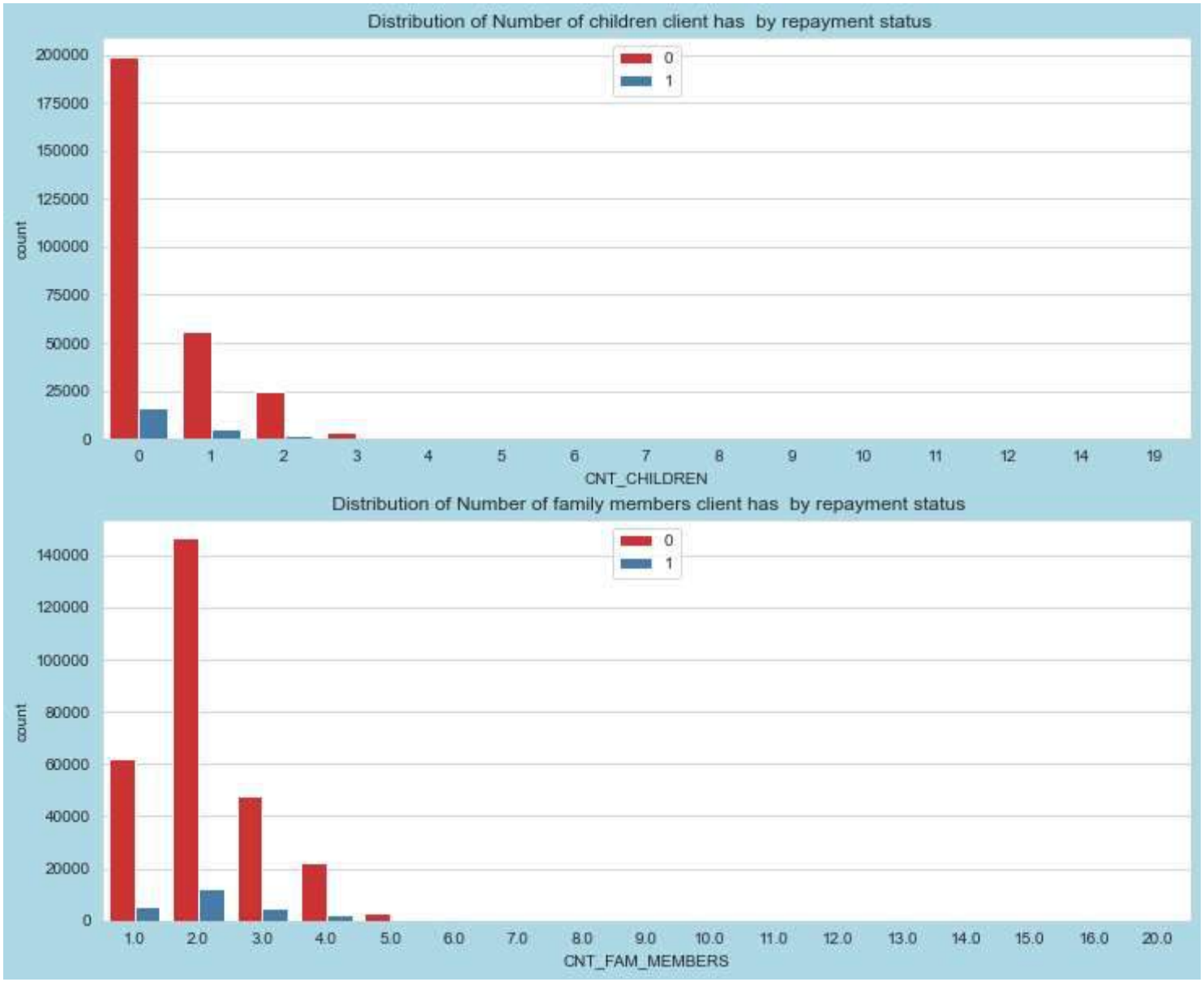
```
plt.figure(figsize=(13,6))
plt.subplot(121)
data["FLAG_OWN_REALTY"].value_counts().plot.pie(autopct = "%1.0f%%",colors = ["skyblue","gold"],startangle = 90,
                                                wedgeprops={"linewidth":2,"edgecolor":"k"},explode=[0.05,0],shadow =True)
plt.title("Distribution of client owns a house or flat")

plt.subplot(122)
data[data["FLAG_OWN_REALTY"] == "Y"]["CODE_GENDER"].value_counts().plot.pie(autopct = "%1.0f%%",colors = ["orangered","b"
                                                                            wedgeprops={"linewidth":2,"edgecolor":"k"},explod
plt.title("Distribution of client owning a house or flat by gender")
plt.show()
```





In [108...

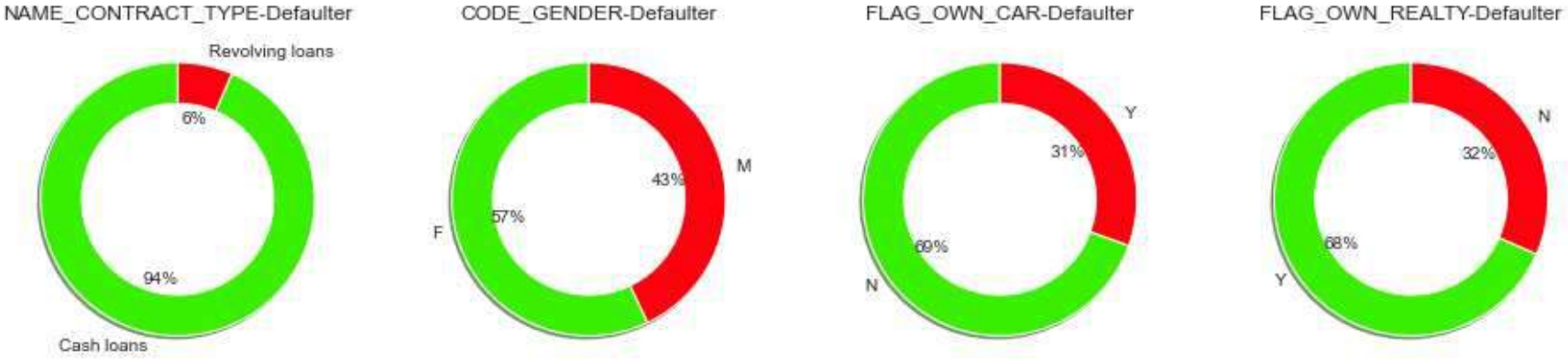
```
fig = plt.figure(figsize=(12,10))
plt.subplot(211)
sns.countplot(application_data["CNT_CHILDREN"],palette="Set1",hue=application_data["TARGET"])
plt.legend(loc="upper center")
plt.title(" Distribution of Number of children client has  by repayment status")
plt.subplot(212)
sns.countplot(application_data["CNT_FAM_MEMBERS"],palette="Set1",hue=application_data["TARGET"])
plt.legend(loc="upper center")
plt.title(" Distribution of Number of family members client has  by repayment status")
fig.set_facecolor("lightblue")
```

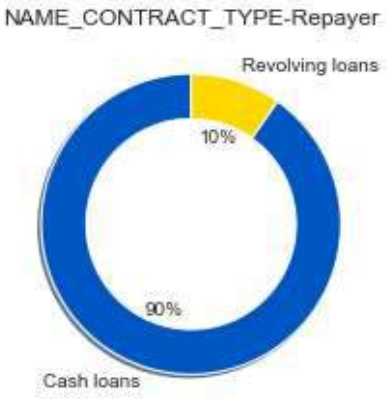


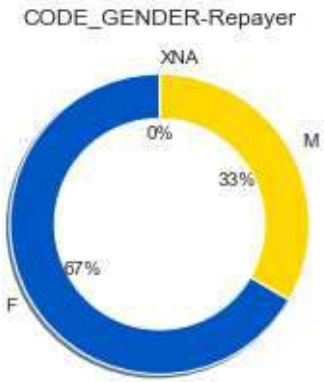
In [109_

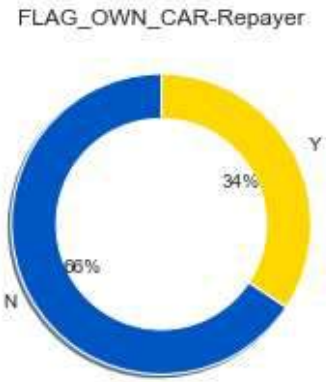
```
default = application_data[application_data["TARGET"]==1][[ 'NAME_CONTRACT_TYPE', 'CODE_GENDER','FLAG_OWN_CAR', 'FLAG_OWN
non_default = application_data[application_data["TARGET"]==0][[ 'NAME_CONTRACT_TYPE', 'CODE_GENDER','FLAG_OWN_CAR', 'FLAG

d_cols = ['NAME_CONTRACT_TYPE', 'CODE_GENDER','FLAG_OWN_CAR', 'FLAG_OWN_REALTY']
d_length = len(d_cols)
```

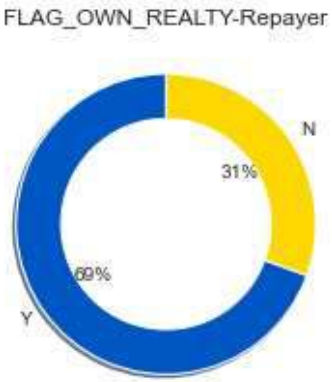
```
fig = plt.figure(figsize=(16,4))
for i,j in itertools.zip_longest(d_cols,range(d_length)):
    plt.subplot(1,4,j+1)
    default[i].value_counts().plot.pie(autopct = "%1.0f%%",colors = sns.color_palette("prism"),startangle = 90,
                                       wedgeprops={"linewidth":1,"edgecolor":"white"},shadow =True)
    circ = plt.Circle((0,0),.7,color="white")
    plt.gca().add_artist(circ)
    plt.ylabel("")
    plt.title(i+"-Defaulter")


fig = plt.figure(figsize=(16,4))
for i,j in itertools.zip_longest(d_cols,range(d_length)):
    plt.subplot(1,4,j+1)
    non_default[i].value_counts().plot.pie(autopct = "%1.0f%%",colors = sns.color_palette("prism",3),startangle = 90,
                                           wedgeprops={"linewidth":1,"edgecolor":"white"},shadow =True)
    circ = plt.Circle((0,0),.7,color="white")
    plt.gca().add_artist(circ)
    plt.ylabel("")
    plt.title(i+"-Repayer")
```





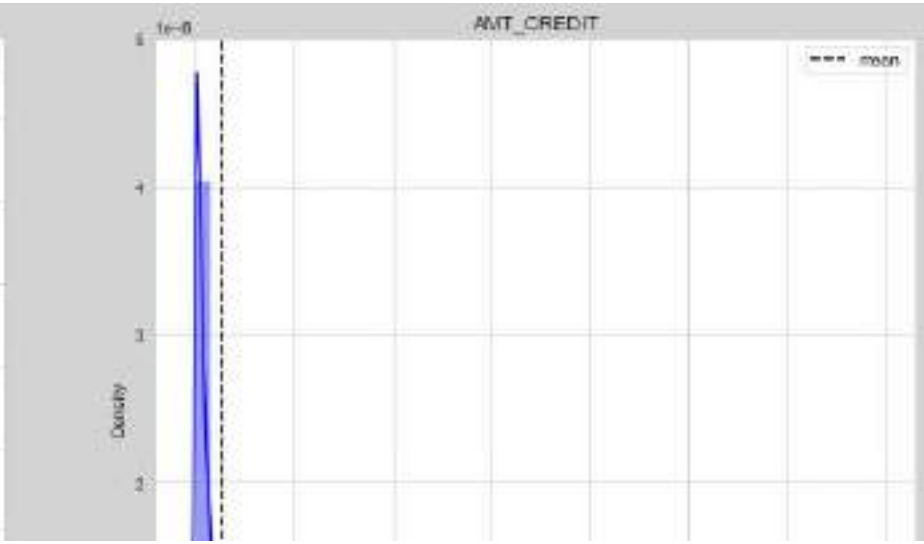


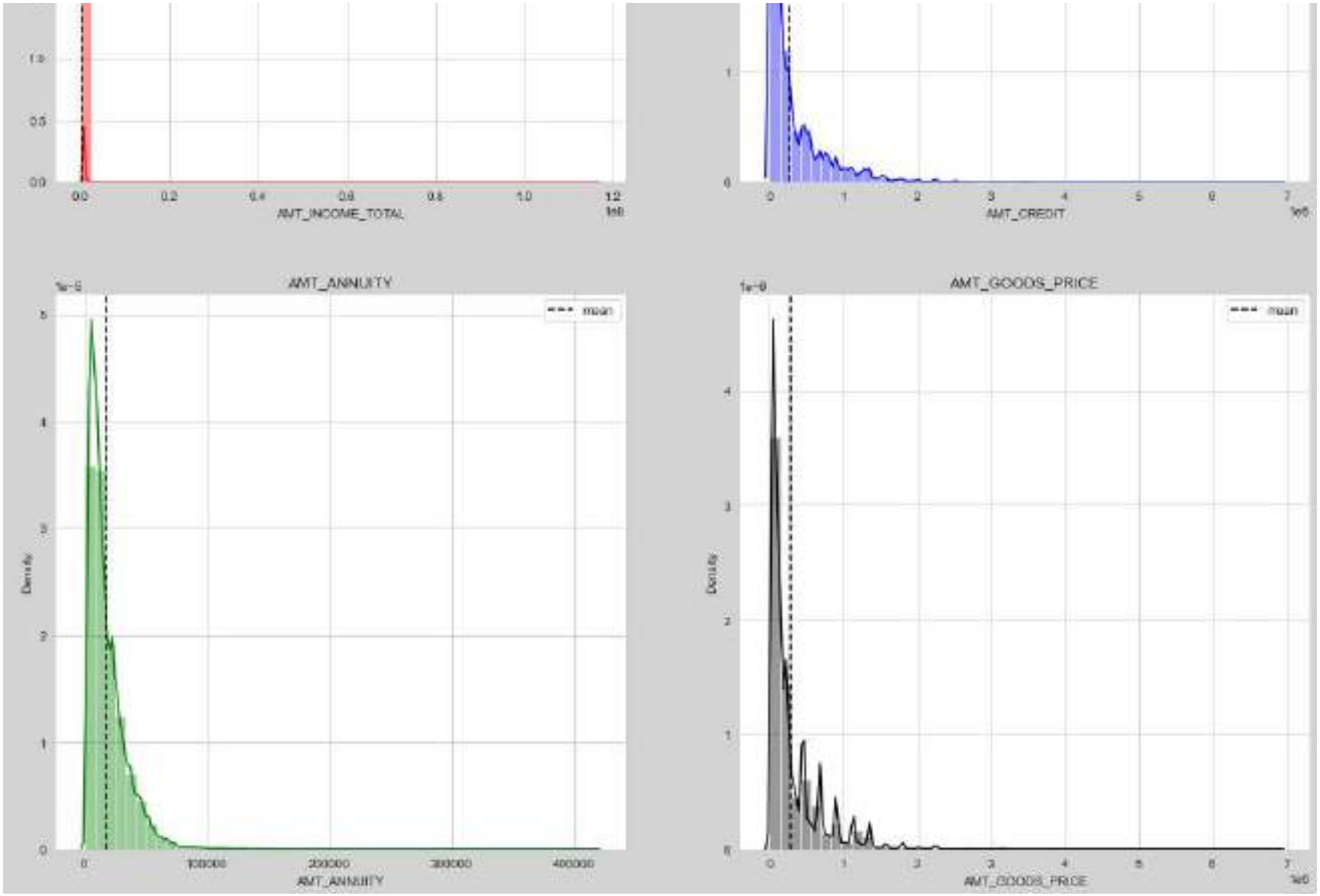




In [110...

```
cols = [ 'AMT_INCOME_TOTAL', 'AMT_CREDIT','AMT_ANNUITY', 'AMT_GOODS_PRICE']
length = len(cols)
cs = ["r","b","g","k"]

ax = plt.figure(figsize=(18,18))
ax.set_facecolor("lightgrey")
for i,j,k in itertools.zip_longest(cols,range(length),cs):
    plt.subplot(2,2,j+1)
    sns.distplot(data[data[i].notnull()][i],color=k)
    plt.axvline(data[i].mean(),label = "mean",linestyle="dashed",color="k")
    plt.legend(loc="best")
    plt.title(i)
    plt.subplots_adjust(hspace = .2)
```

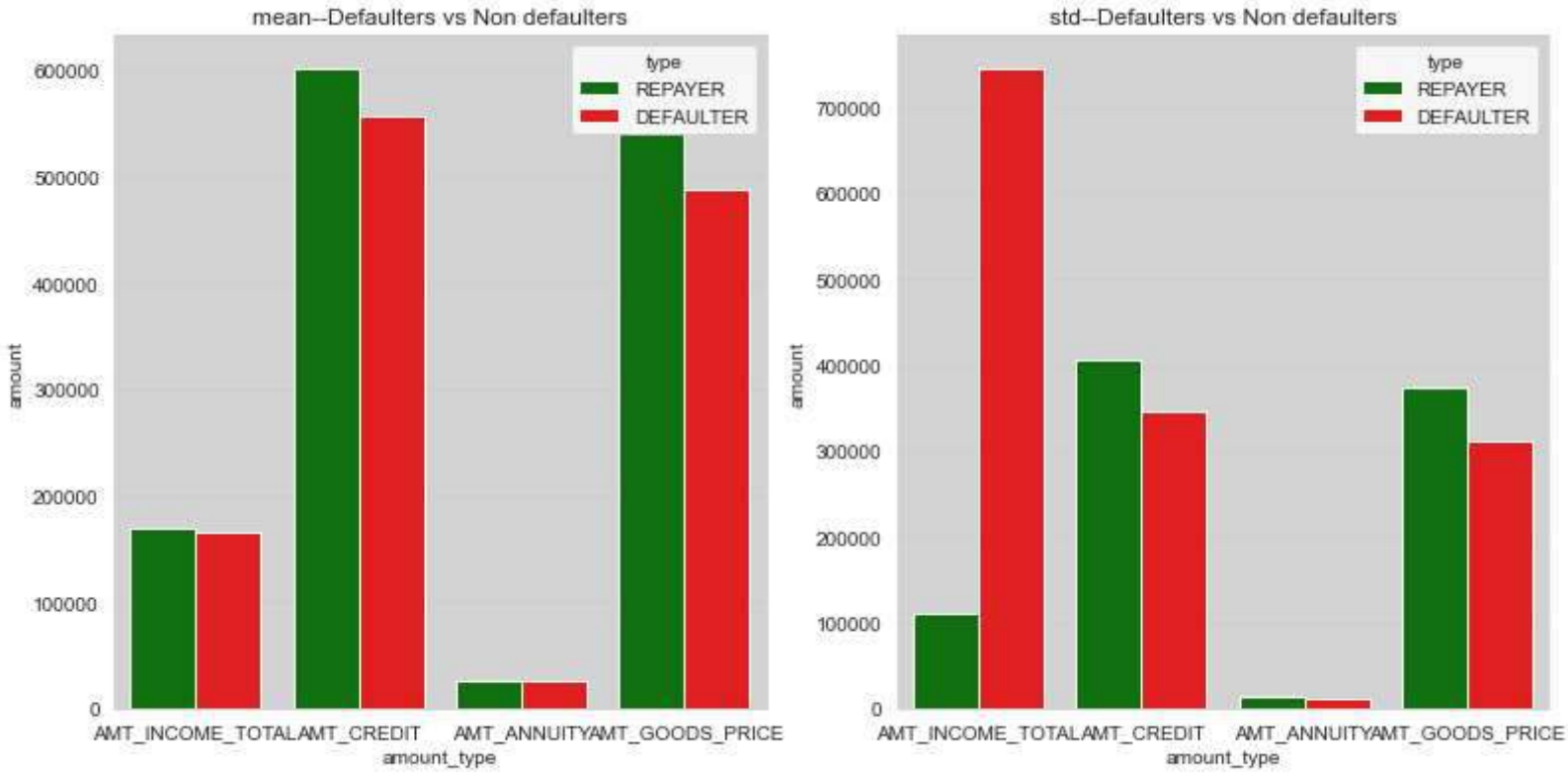


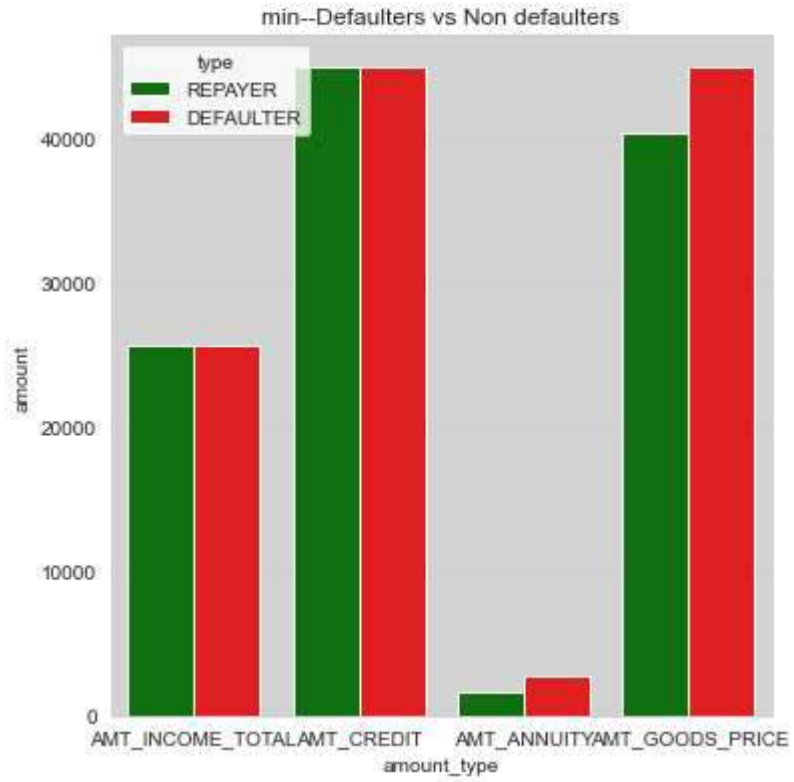


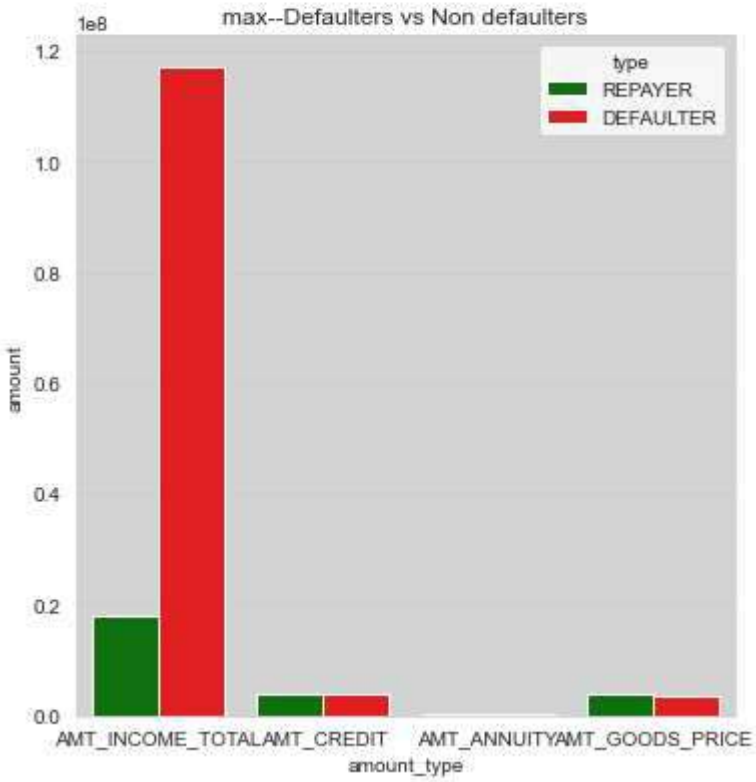
In [112...

```
df = application_data.groupby("TARGET")[cols].describe().transpose().reset_index()
df = df[df["level_1"].isin([ 'mean', 'std', 'min', 'max'])]
df_x = df[["level_0","level_1",0]]
df_y = df[["level_0","level_1",1]]
df_x = df_x.rename(columns={'level_0':"amount_type", 'level_1':"statistic", 0:"amount"})
df_x["type"] = "REPAYER"
df_y = df_y.rename(columns={'level_0':"amount_type", 'level_1':"statistic", 1:"amount"})
```

```
df_y["type"] = "DEFAULTER"
df_new = pd.concat([df_x,df_y],axis = 0)

stat = df_new["statistic"].unique().tolist()
length = len(stat)

plt.figure(figsize=(13,15))

for i,j in itertools.zip_longest(stat,range(length)):
    plt.subplot(2,2,j+1)
    fig = sns.barplot(df_new[df_new["statistic"] == i]["amount_type"],df_new[df_new["statistic"] == i]["amount"],
                hue=df_new[df_new["statistic"] == i]["type"],palette=["g","r"])
    plt.title(i + "--Defaulters vs Non defaulters")
    plt.subplots_adjust(hspace = .4)
    fig.set_facecolor("lightgrey")
```



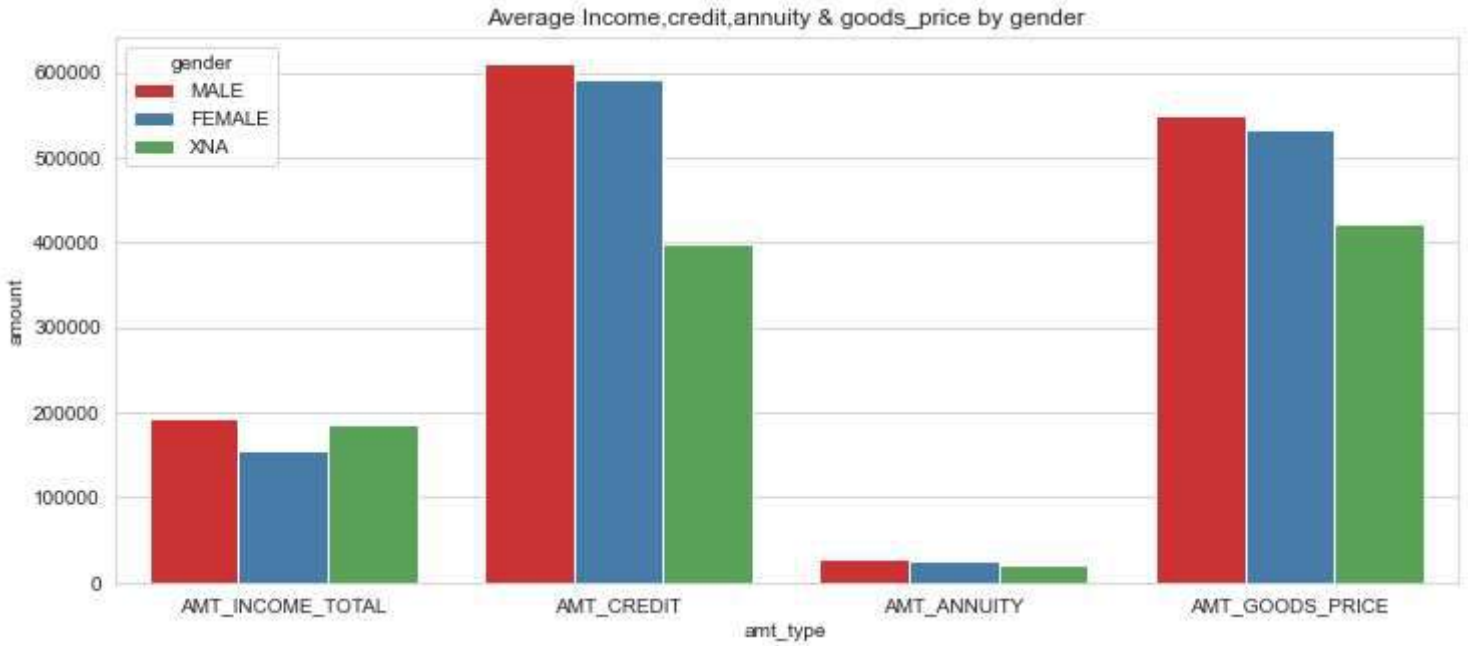




In [111...

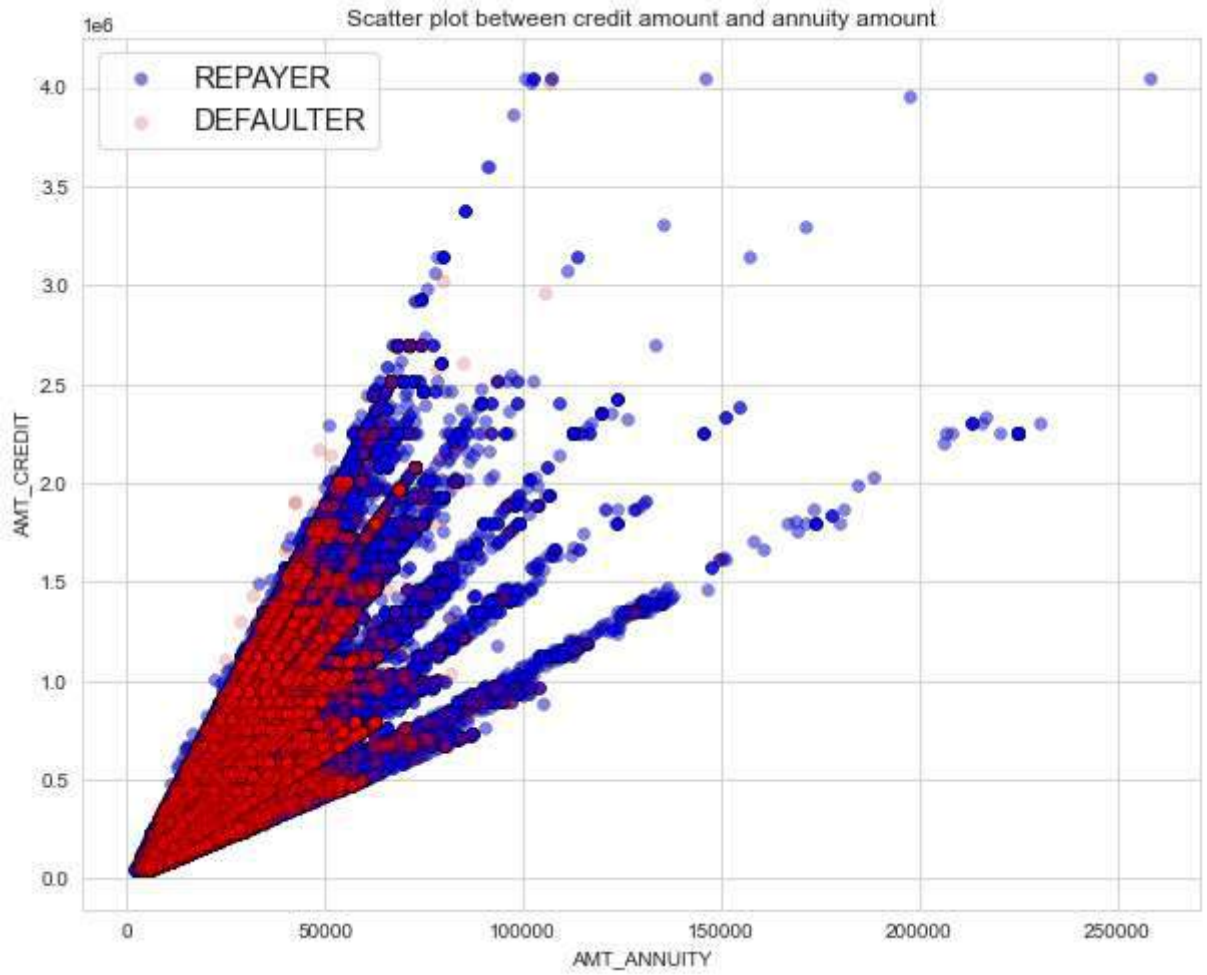
```
cols = [ 'AMT_INCOME_TOTAL', 'AMT_CREDIT','AMT_ANNUITY', 'AMT_GOODS_PRICE']

df1 = data.groupby("CODE_GENDER")[cols].mean().transpose().reset_index()

df_f   = df1[["index","F"]]
df_f   = df_f.rename(columns={'index':"amt_type", 'F':"amount"})
df_f["gender"] = "FEMALE"
df_m   = df1[["index","M"]]
df_m   = df_m.rename(columns={'index':"amt_type", 'M':"amount"})
df_m["gender"] = "MALE"
df_xna = df1[["index","XNA"]]
df_xna = df_xna.rename(columns={'index':"amt_type", 'XNA':"amount"})
df_xna["gender"] = "XNA"

df_gen = pd.concat([df_m,df_f,df_xna],axis=0)

plt.figure(figsize=(12,5))
ax = sns.barplot("amt_type","amount",data=df_gen,hue="gender",palette="Set1")
```

```
plt.title("Average Income,credit,annuity & goods_price by gender")
plt.show()
```



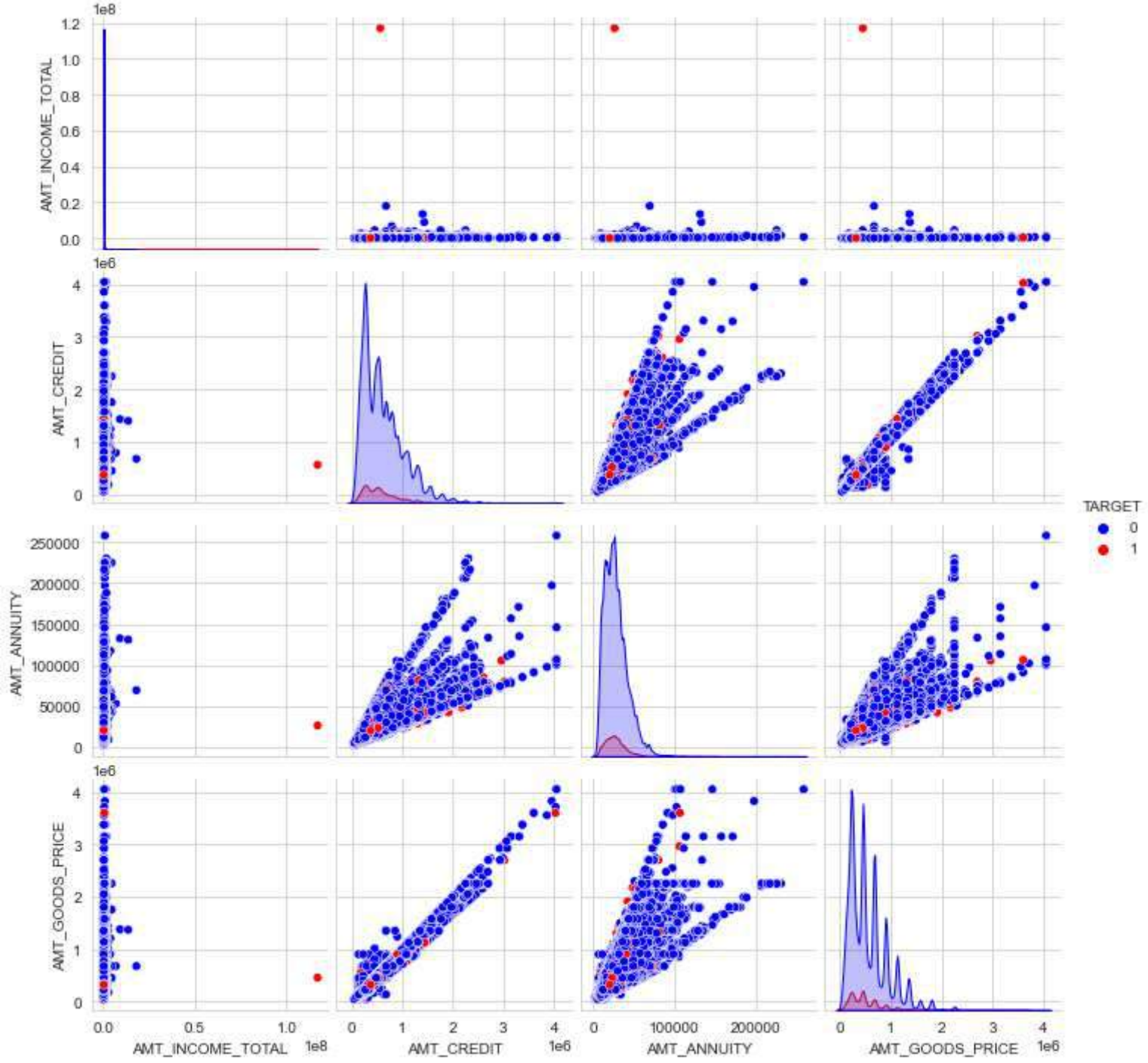
In [113...

```
fig = plt.figure(figsize=(10,8))
plt.scatter(application_data[application_data["TARGET"]==0]['AMT_ANNUITY'],application_data[application_data["TARGET"]==0
            color="b",alpha=.5,label="REPAYER",linewidth=.5,edgecolor="k")
plt.scatter(application_data[application_data["TARGET"]==1]['AMT_ANNUITY'],application_data[application_data["TARGET"]==1
            color="r",alpha=.2,label="DEFAULTER",linewidth=.5,edgecolor="k")
plt.legend(loc="best",prop={"size":15})
plt.xlabel("AMT_ANNUITY")
plt.ylabel("AMT_CREDIT")
plt.title("Scatter plot between credit amount and annuity amount")
plt.show()
```

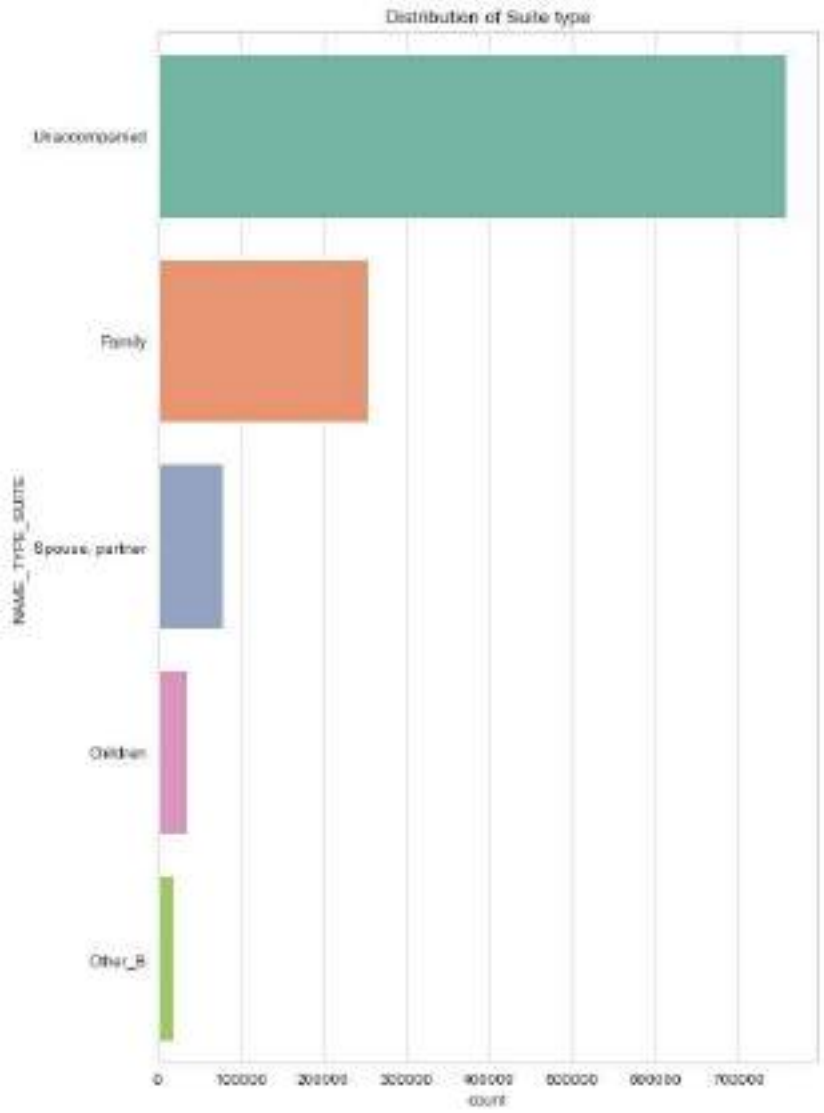


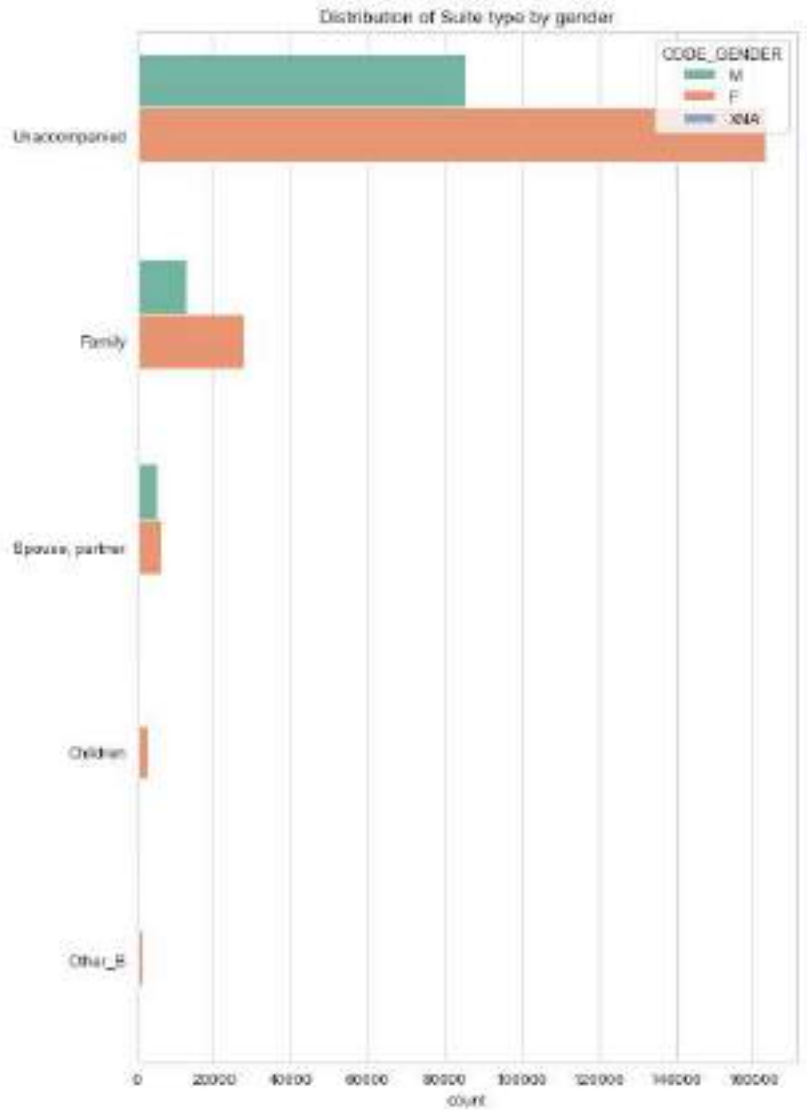
In [114...

```
amt = application_data[[ 'AMT_INCOME_TOTAL','AMT_CREDIT',
                         'AMT_ANNUITY', 'AMT_GOODS_PRICE',"TARGET"]]
amt = amt[(amt["AMT_GOODS_PRICE"].notnull()) & (amt["AMT_ANNUITY"].notnull())]
sns.pairplot(amt,hue="TARGET",palette=["b","r"])
plt.show()
```


1e8
AMT_INCOME_TOTAL
1e6
AMT_CREDIT
AMT_ANNUITY
1e6
AMT_GOODS_PRICE
AMT_INCOME_TOTAL
1e8
AMT_CREDIT
1e6
AMT_ANNUITY
AMT_GOODS_PRICE
1e6
TARGET
0
1

In [115]:
```
plt.figure(figsize=(18,12))
plt.subplot(121)
sns.countplot(y=data["NAME_TYPE_SUITE"],
              palette="Set2",
              order=data["NAME_TYPE_SUITE"].value_counts().index[:5])
plt.title("Distribution of Suite type")

plt.subplot(122)
sns.countplot(y=data["NAME_TYPE_SUITE"],
              hue=data["CODE_GENDER"],palette="Set2",
              order=data["NAME_TYPE_SUITE"].value_counts().index[:5])
plt.ylabel("")
plt.title("Distribution of Suite type by gender")
plt.subplots_adjust(wspace = .4)
```

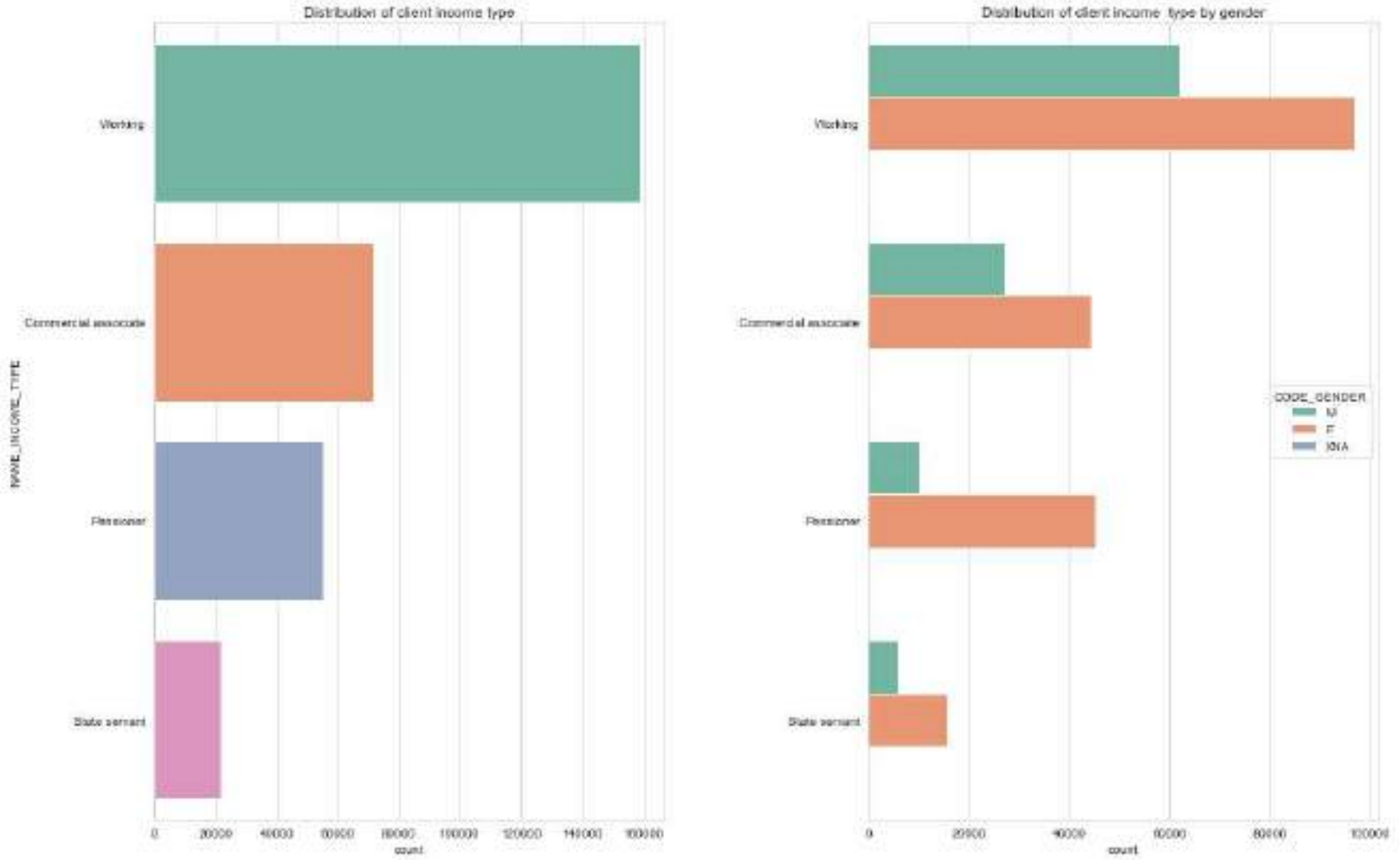




In [116]:

```
plt.figure(figsize=(18,12))
plt.subplot(121)
sns.countplot(y=data["NAME_INCOME_TYPE"],
              palette="Set2",
              order=data["NAME_INCOME_TYPE"].value_counts().index[:4])
plt.title("Distribution of client income type")

plt.subplot(122)
sns.countplot(y=data["NAME_INCOME_TYPE"],
```

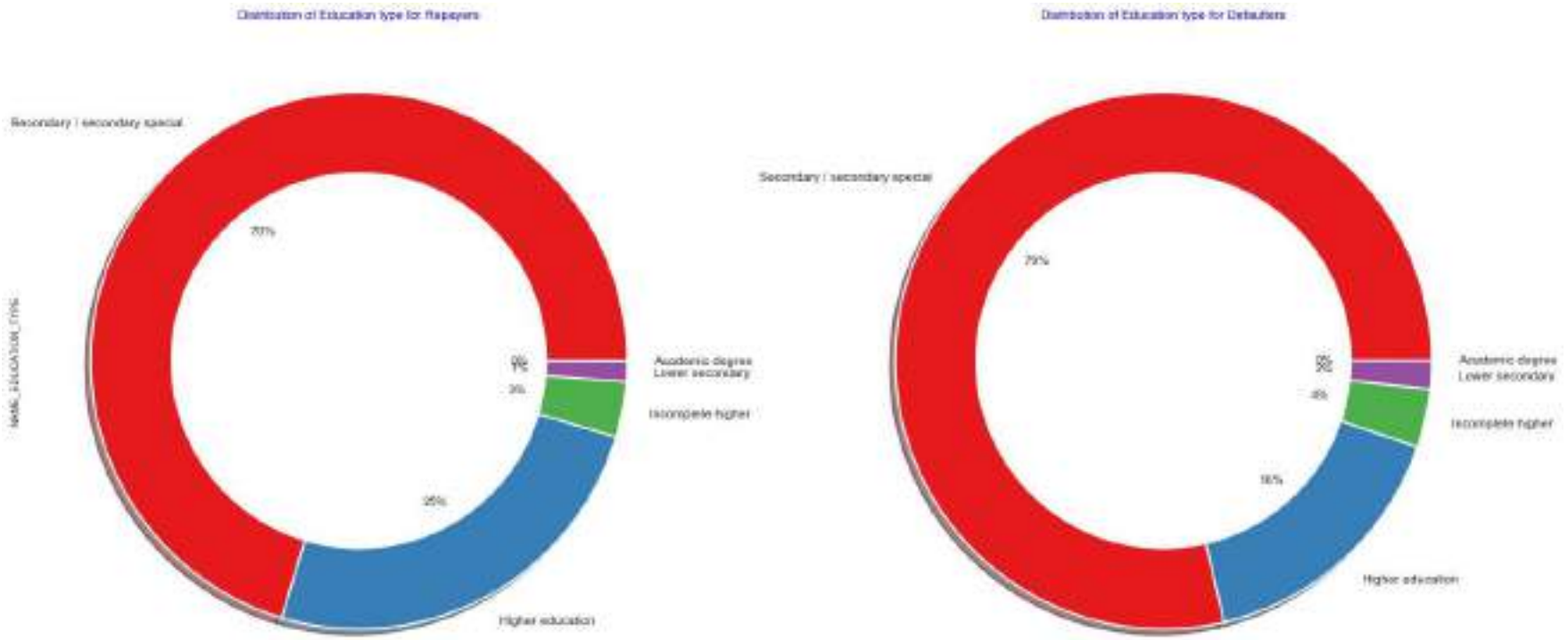
```
                hue=data["CODE_GENDER"],
                palette="Set2",
                order=data["NAME_INCOME_TYPE"].value_counts().index[:4])
plt.ylabel("")
plt.title("Distribution of client income  type by gender")
plt.subplots_adjust(wspace = .4)
```



In [117...

```
plt.figure(figsize=(25,25))
plt.subplot(121)
application_data[application_data["TARGET"]==0]["NAME_EDUCATION_TYPE"].value_counts().plot.pie(fontsize=12,autopct = "%1.
```

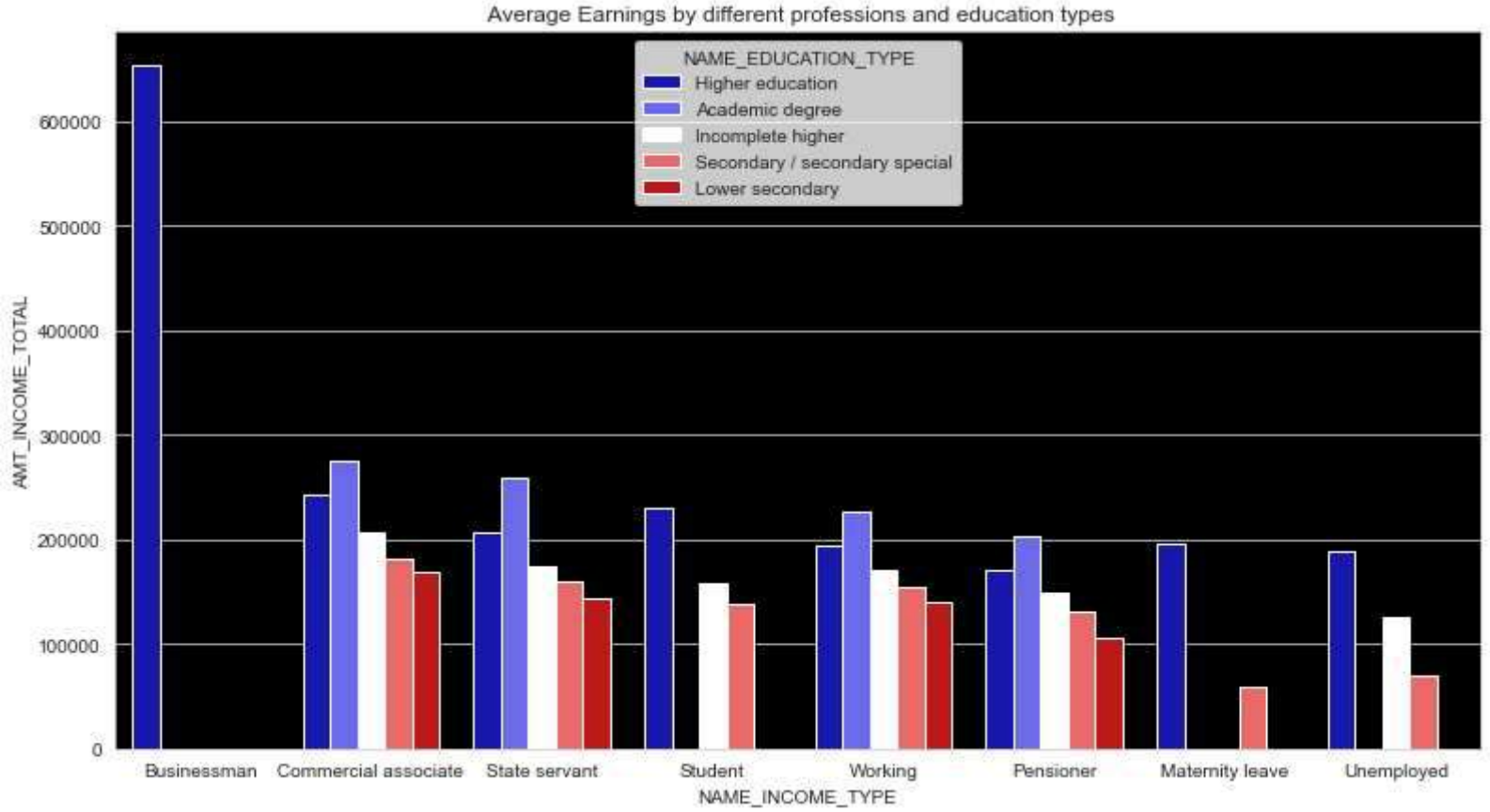
```
                                                                                    colors = sns.color_palet
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)
circ = plt.Circle((0,0),.7,color="white")
plt.gca().add_artist(circ)
plt.title("Distribution of Education type for Repayers",color="b")

plt.subplot(122)
application_data[application_data["TARGET"]==1]["NAME_EDUCATION_TYPE"].value_counts().plot.pie(fontsize=12,autopct = "%1.
                                                                                    colors = sns.color_palet
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)
circ = plt.Circle((0,0),.7,color="white")
plt.gca().add_artist(circ)
plt.title("Distribution of Education type for Defaulters",color="b")
plt.ylabel("")
plt.show()
```



In [118...

```
edu = data.groupby(['NAME_EDUCATION_TYPE','NAME_INCOME_TYPE'])['AMT_INCOME_TOTAL'].mean().reset_index().sort_values(by='A
fig = plt.figure(figsize=(13,7))
ax = sns.barplot('NAME_INCOME_TYPE','AMT_INCOME_TOTAL',data=edu,hue='NAME_EDUCATION_TYPE',palette="seismic")
```

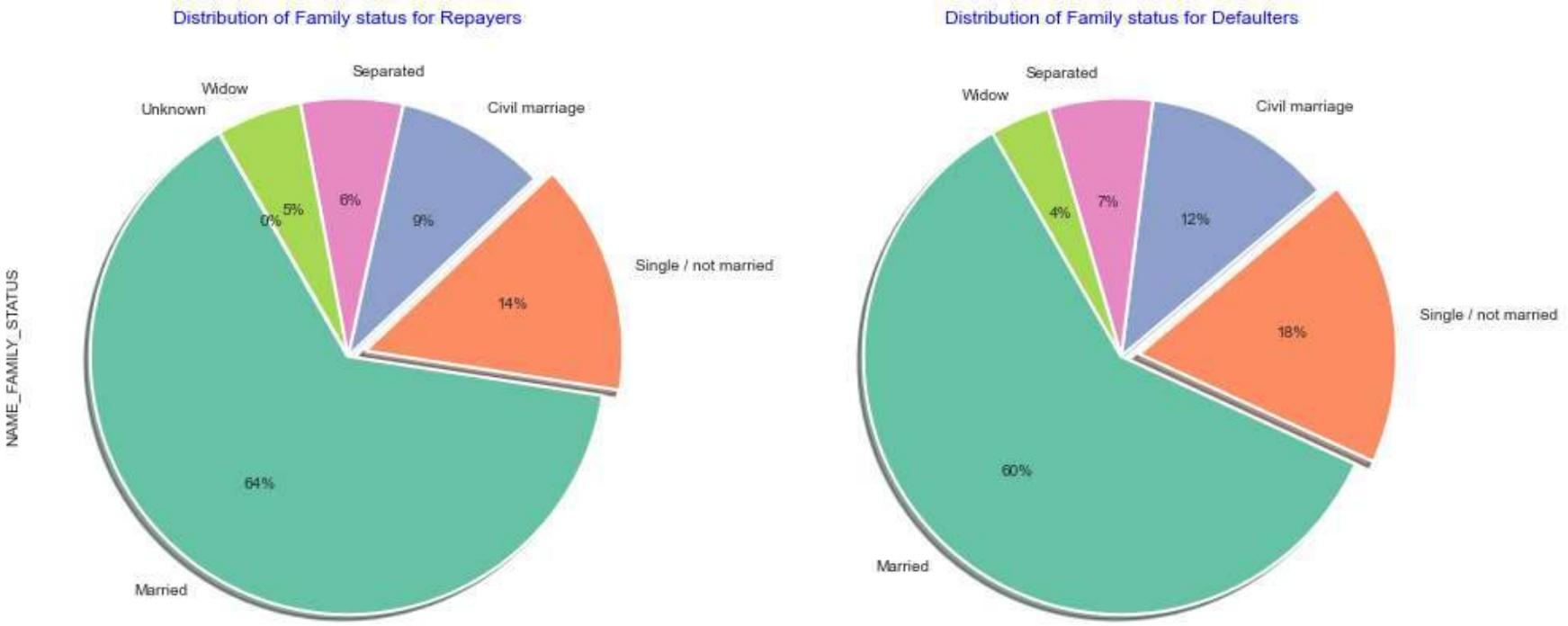
```
ax.set_facecolor("k")
plt.title(" Average Earnings by different professions and education types")
plt.show()
```



In [119...

```
plt.figure(figsize=(16,8))
plt.subplot(121)
application_data[application_data["TARGET"]==0]["NAME_FAMILY_STATUS"].value_counts().plot.pie(autopct = "%1.0f%%",
                                                    startangle=120,colors = sns.color_palette("Set2",7),
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True,explode=[0,.07,

plt.title("Distribution of Family status for Repayers",color="b")

plt.subplot(122)
application_data[application_data["TARGET"]==1]["NAME_FAMILY_STATUS"].value_counts().plot.pie(autopct = "%1.0f%%",
                                                startangle=120,colors = sns.color_palette("Set2",7),
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True,explode=[0,.07,
```

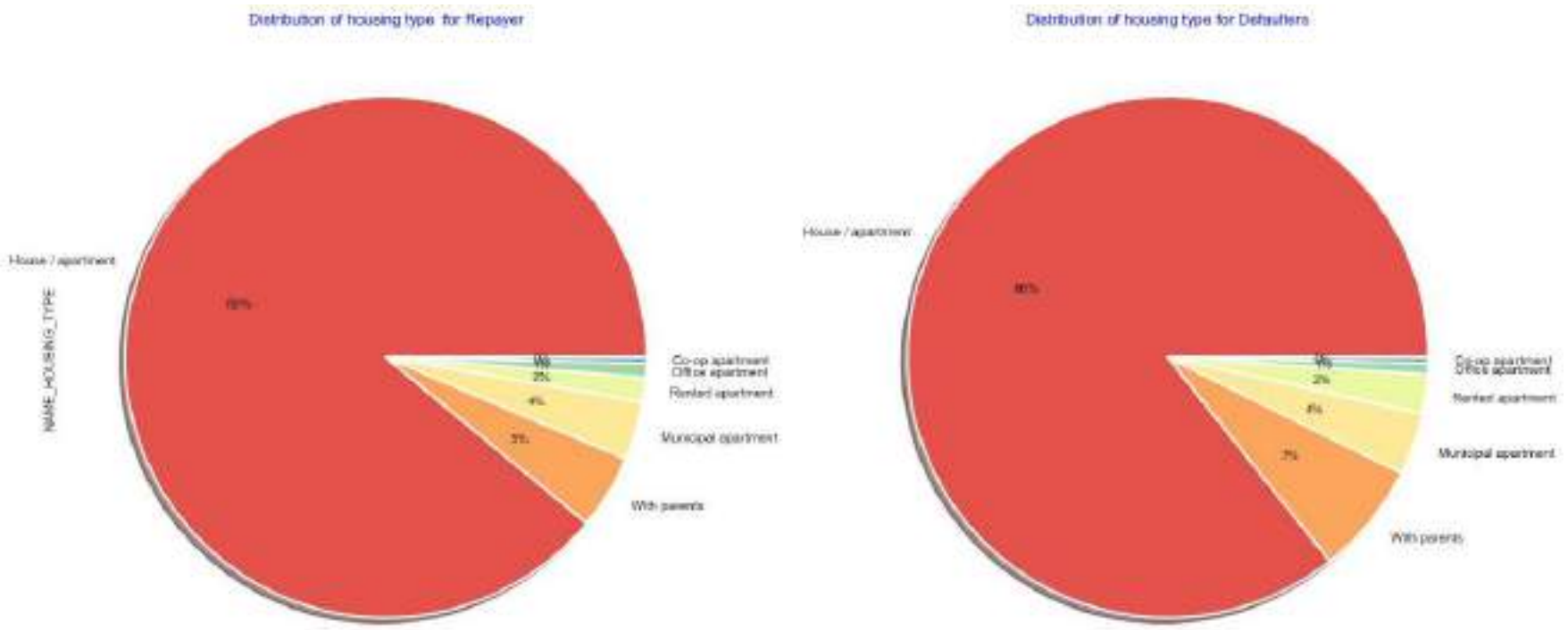
```
plt.title("Distribution of Family status for Defaulters",color="b")
plt.ylabel("")
plt.show()
```



In [120...

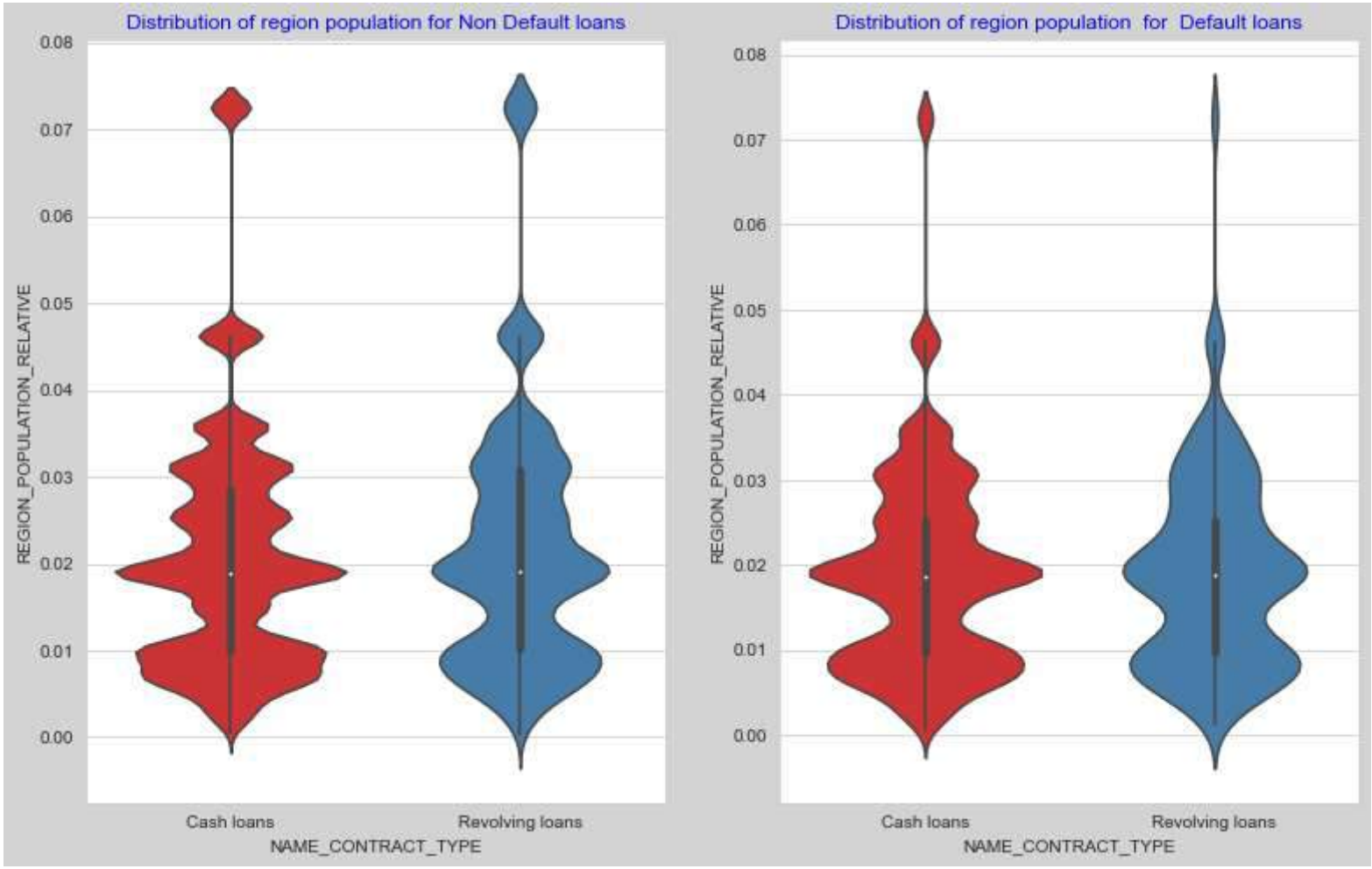
```
plt.figure(figsize=(20,20))
plt.subplot(121)
application_data[application_data["TARGET"]==0]["NAME_HOUSING_TYPE"].value_counts().plot.pie(autopct = "%1.0f%%",fontsize
                                                    colors = sns.color_palette("Spectral"),
                                            wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)

plt.title("Distribution of housing type  for Repayer",color="b")

plt.subplot(122)
application_data[application_data["TARGET"]==1]["NAME_HOUSING_TYPE"].value_counts().plot.pie(autopct = "%1.0f%%",fontsize
                                                colors = sns.color_palette("Spectral"),
                                            wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)
```

```
plt.title("Distribution of housing type for Defaulters",color="b")
plt.ylabel("")
plt.show()
```



In [121...

```
fig = plt.figure(figsize=(13,8))

plt.subplot(121)
sns.violinplot(y=application_data[application_data["TARGET"]==0]["REGION_POPULATION_RELATIVE"]
               ,x=application_data[application_data["TARGET"]==0]["NAME_CONTRACT_TYPE"],
               palette="Set1")
plt.title("Distribution of region population for Non Default loans",color="b")
plt.subplot(122)
sns.violinplot(y = application_data[application_data["TARGET"]==1]["REGION_POPULATION_RELATIVE"]
               ,x=application_data[application_data["TARGET"]==1]["NAME_CONTRACT_TYPE"]
               ,palette="Set1")
plt.title("Distribution of region population  for  Default loans",color="b")

plt.subplots_adjust(wspace = .2)
fig.set_facecolor("lightgrey")
```

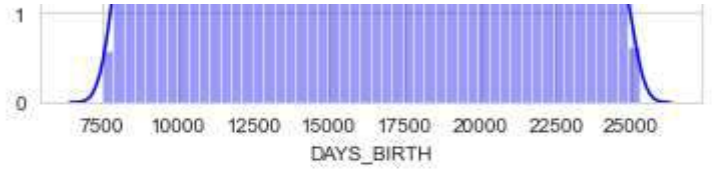


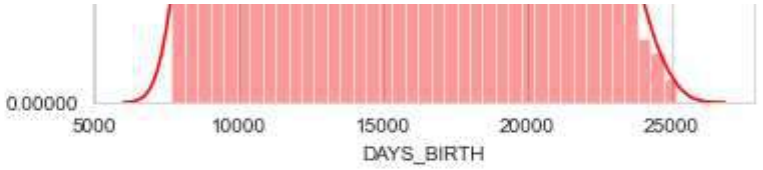
In [123]:

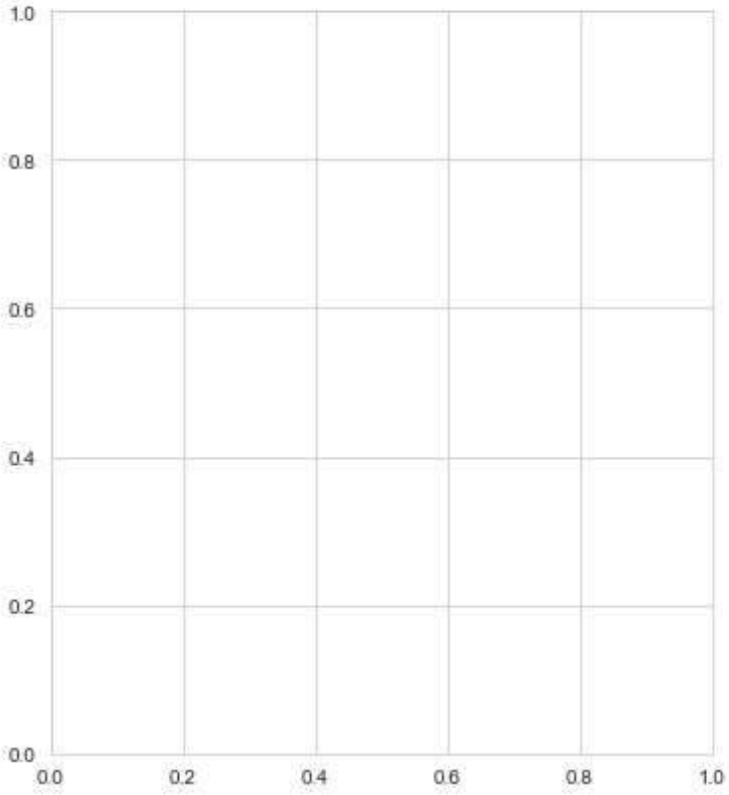
```
fig = plt.figure(figsize=(13,15))

plt.subplot(221)
sns.distplot(application_data[application_data["TARGET"]==0]["DAYS_BIRTH"],color="b")
plt.title("Age Distribution of repayers")

plt.subplot(222)
sns.distplot(application_data[application_data["TARGET"]==1]["DAYS_BIRTH"],color="r")
plt.title("Age Distribution of defaulters")

plt.subplot(223)
sns.lvplot(application_data["TARGET"],application_data["DAYS_BIRTH"],hue=application_data["CODE_GENDER"],palette=["b","gr
plt.axhline(application_data["DAYS_BIRTH"].mean(),linestyle="dashed",color="k",label ="average age of client")
```

```
plt.legend(loc="lower right")
plt.title("Client age vs Loan repayment status(hue=gender)")

plt.subplot(224)
sns.lvplot(application_data["TARGET"],application_data["DAYS_BIRTH"],hue=application_data["NAME_CONTRACT_TYPE"],palette=[
plt.axhline(application_data["DAYS_BIRTH"].mean(),linestyle="dashed",color="k",label ="average age of client")
plt.legend(loc="lower right")
plt.title("Client age vs Loan repayment status(hue=contract type)")

plt.subplots_adjust(wspace = .2,hspace = .3)

fig.set_facecolor("lightgrey")
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_17028/2495597695.py in <module>
     10
     11 plt.subplot(223)
---> 12 sns.lvplot(application_data["TARGET"],application_data["DAYS_BIRTH"],hue=application_data["CODE_GENDER"],palette=
["b","grey","m"])
     13 plt.axhline(application_data["DAYS_BIRTH"].mean(),linestyle="dashed",color="k",label ="average age of client")
     14 plt.legend(loc="lower right")

AttributeError: module 'seaborn' has no attribute 'lvplot'
```

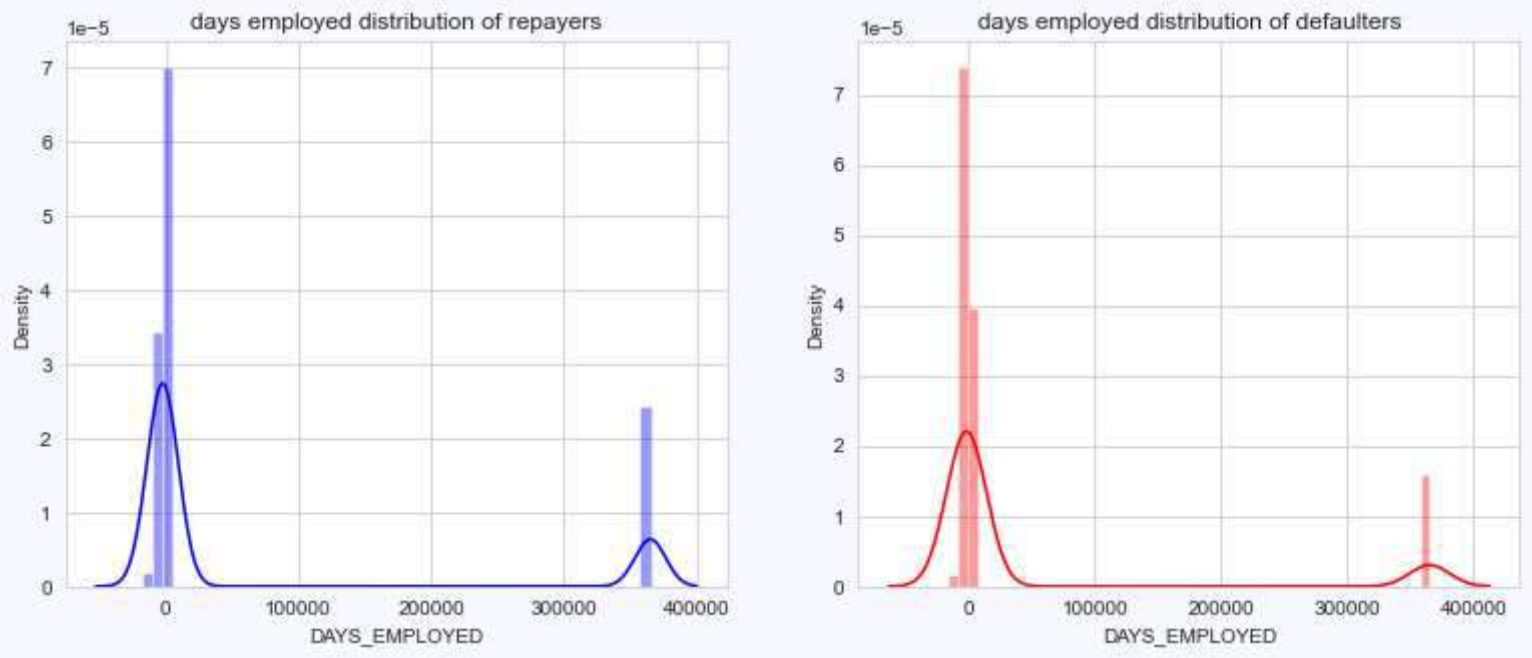






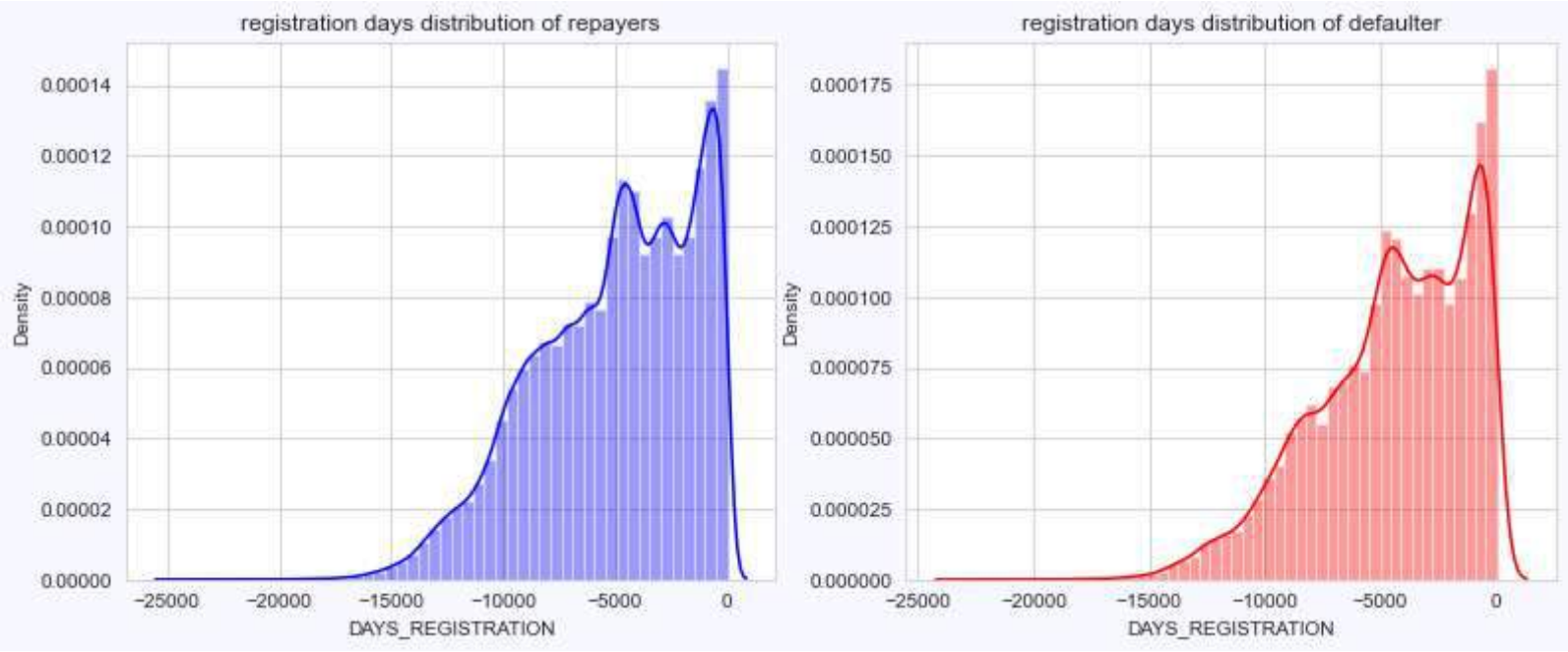
In [124...

```
fig = plt.figure(figsize=(13,5))

plt.subplot(121)
sns.distplot(application_data[application_data["TARGET"]==0]["DAYS_EMPLOYED"],color="b")
plt.title("days employed distribution of repayers")

plt.subplot(122)
sns.distplot(application_data[application_data["TARGET"]==1]["DAYS_EMPLOYED"],color="r")
plt.title("days employed distribution of defaulters")

fig.set_facecolor("ghostwhite")
```



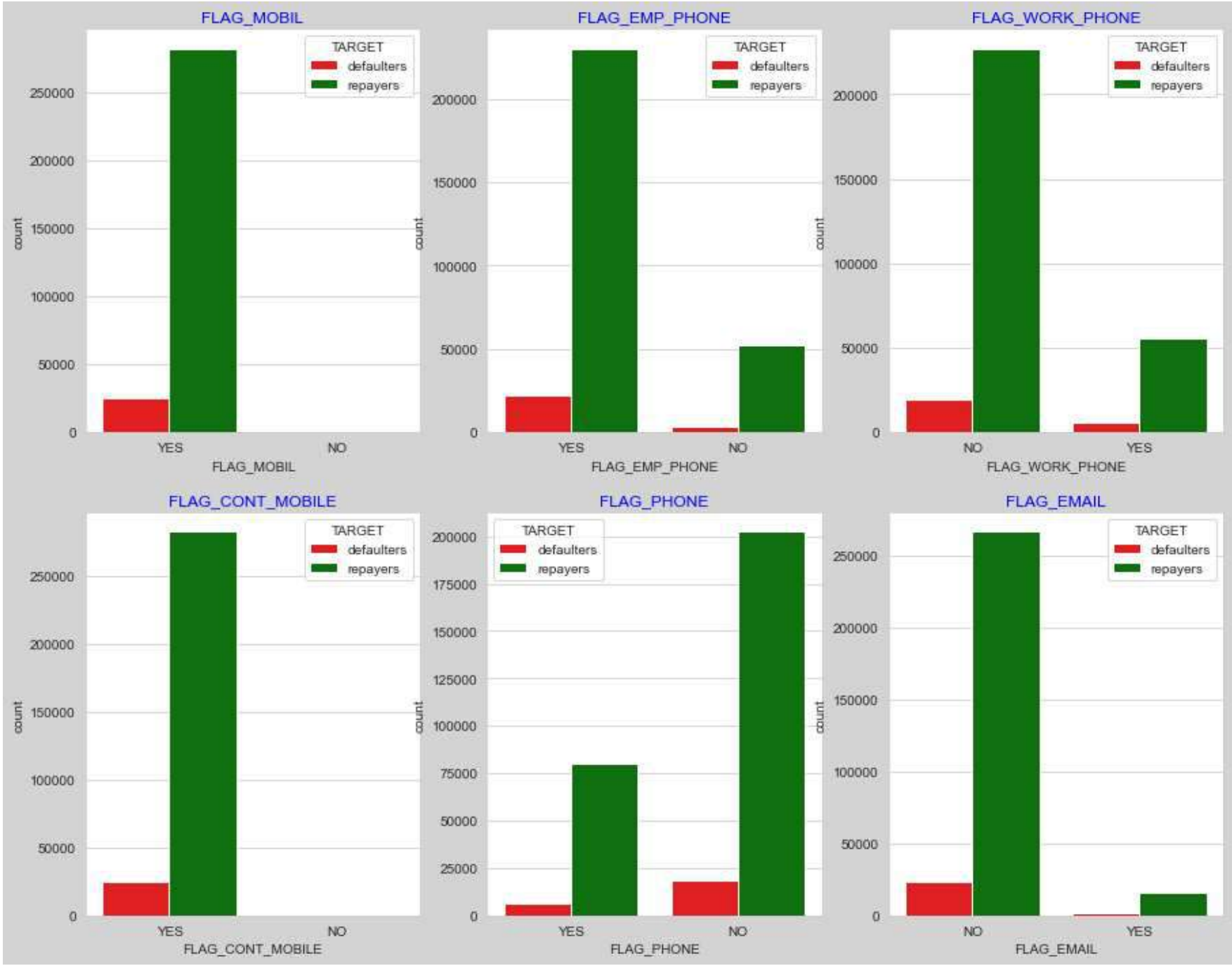
In [125]:

```
fig = plt.figure(figsize=(13,5))

plt.subplot(121)
sns.distplot(application_data[application_data["TARGET"]==0]["DAYS_REGISTRATION"],color="b")
plt.title("registration days distribution of repayers")

plt.subplot(122)
sns.distplot(application_data[application_data["TARGET"]==1]["DAYS_REGISTRATION"],color="r")
plt.title("registration days distribution of defaulter")

fig.set_facecolor("ghostwhite")
```



In [126]:

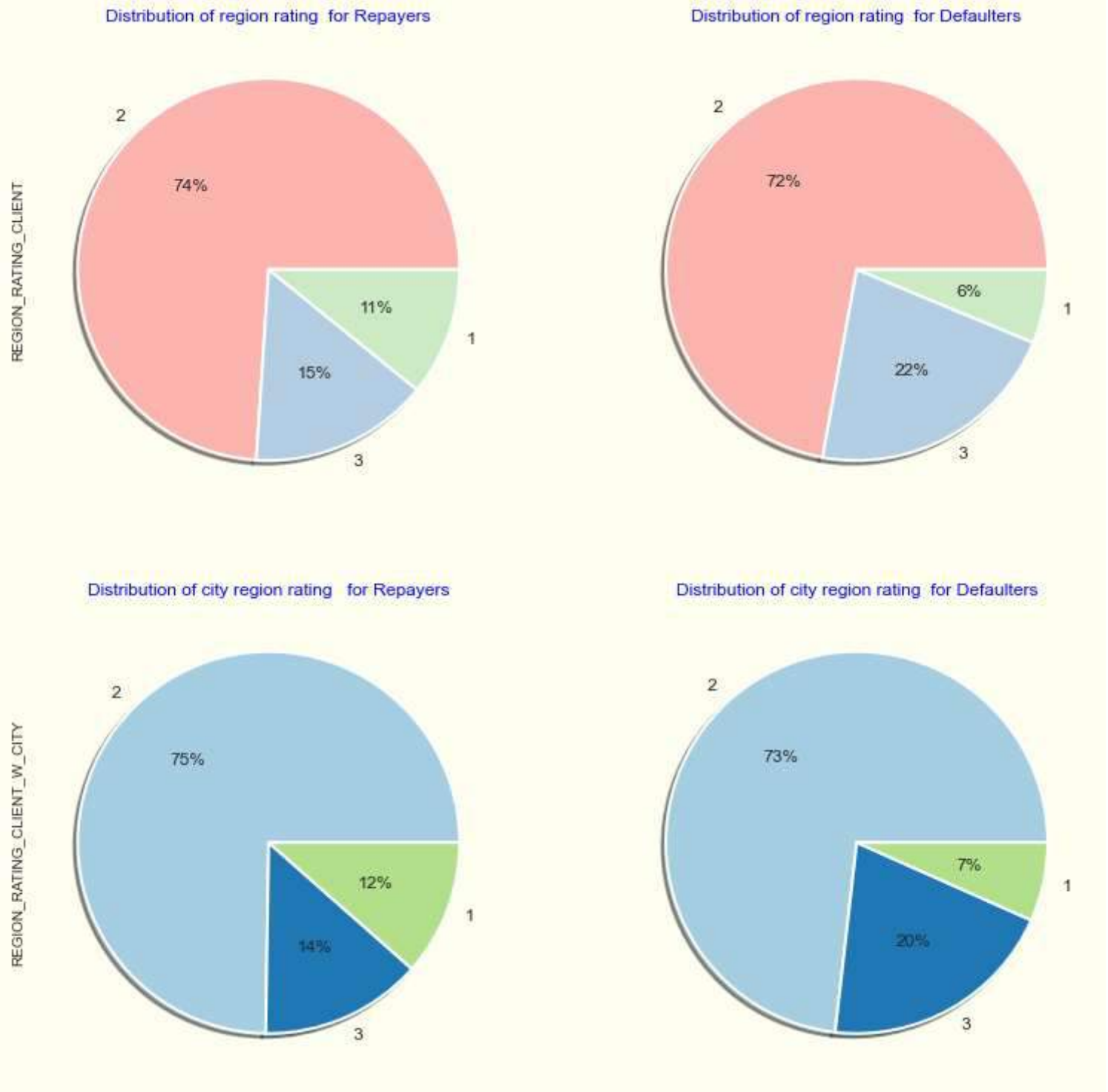
```
x   = application_data[['FLAG_MOBIL', 'FLAG_EMP_PHONE', 'FLAG_WORK_PHONE', 'FLAG_CONT_MOBILE',
       'FLAG_PHONE', 'FLAG_EMAIL',"TARGET"]]
x["TARGET"] = x["TARGET"].replace({0:"repayers",1:"defaulters"})
x  = x.replace({1:"YES",0:"NO"})

cols = ['FLAG_MOBIL', 'FLAG_EMP_PHONE', 'FLAG_WORK_PHONE', 'FLAG_CONT_MOBILE',
       'FLAG_PHONE', 'FLAG_EMAIL']
length = len(cols)

fig = plt.figure(figsize=(15,12))
fig.set_facecolor("lightgrey")

for i,j in itertools.zip_longest(cols,range(length)):
    plt.subplot(2,3,j+1)
    sns.countplot(x[i],hue=x["TARGET"],palette=["r","g"])
    plt.title(i,color="b")
```



In [127...

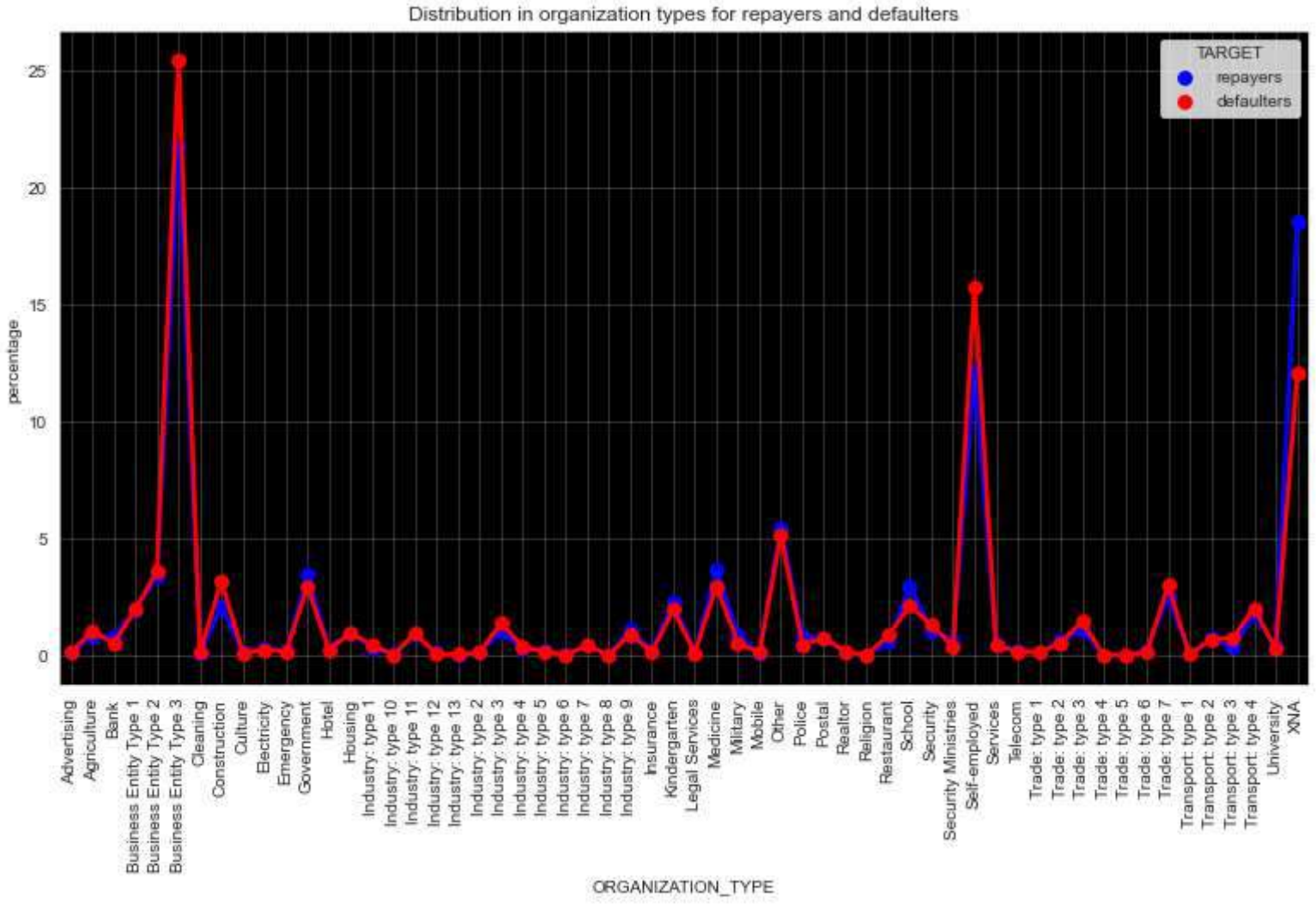
```
fig = plt.figure(figsize=(13,13))
```

```
plt.subplot(221)
application_data[application_data["TARGET"]==0]["REGION_RATING_CLIENT"].value_counts().plot.pie(autopct = "%1.0f%%",fonts
                                                        colors = sns.color_palette("Pastel1"),
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)

plt.title("Distribution of region rating  for Repayers",color="b")

plt.subplot(222)
application_data[application_data["TARGET"]==1]["REGION_RATING_CLIENT"].value_counts().plot.pie(autopct = "%1.0f%%",fonts
                                                    colors = sns.color_palette("Pastel1"),
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)


plt.title("Distribution of region rating  for Defaulters",color="b")
plt.ylabel("")

plt.subplot(223)
application_data[application_data["TARGET"]==0]["REGION_RATING_CLIENT_W_CITY"].value_counts().plot.pie(autopct = "%1.0f%%
                                                        colors = sns.color_palette("Paired"),
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)

plt.title("Distribution of city region rating   for Repayers",color="b")

plt.subplot(224)
application_data[application_data["TARGET"]==1]["REGION_RATING_CLIENT_W_CITY"].value_counts().plot.pie(autopct = "%1.0f%%
                                                    colors = sns.color_palette("Paired"),
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)


plt.title("Distribution of city region rating  for Defaulters",color="b")
plt.ylabel("")
fig.set_facecolor("ivory")
```


Distribution of region rating for Repayers
REGION_RATING_CLIENT
2
74%
11%
1
15%
3
Distribution of region rating for Defaulters
2
72%
6%
1
22%
3
Distribution of city region rating for Repayers
REGION_RATING_CLIENT_W_CITY
2
75%
12%
1
14%
3
Distribution of city region rating for Defaulters
2
73%
7%
1
20%
3

In [130...

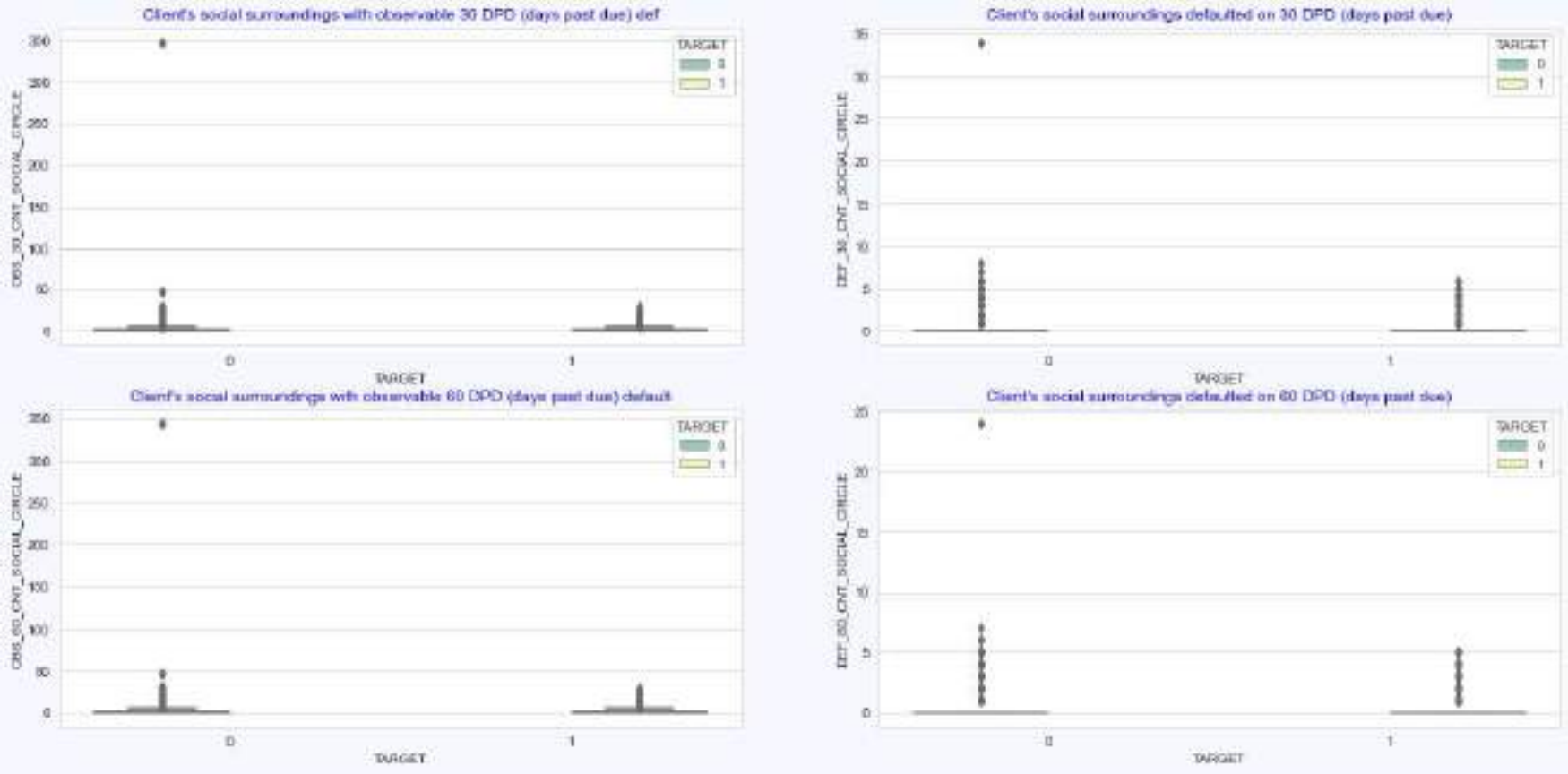
```
org = application_data.groupby("TARGET").agg({"ORGANIZATION_TYPE":"value_counts"})
org = org.rename(columns = {"ORGANIZATION_TYPE":"value_counts"}).reset_index()
org_0 = org[org["TARGET"] == 0]
org_1 = org[org["TARGET"] == 1]
org_0["percentage"] = org_0["value_counts"]*100/org_0["value_counts"].sum()
org_1["percentage"] = org_1["value_counts"]*100/org_1["value_counts"].sum()

organization = pd.concat([org_0,org_1],axis=0)
organization = organization.sort_values(by="ORGANIZATION_TYPE",ascending=True)

organization["TARGET"] = organization["TARGET"].replace({0:"repayers",1:"defaulters"})

organization
plt.figure(figsize=(13,7))
ax = sns.pointplot("ORGANIZATION_TYPE","percentage",
                   data=organization,hue="TARGET",palette=["b","r"])
plt.xticks(rotation=90)
plt.grid(True,alpha=.3)
ax.set_facecolor("k")
ax.set_title("Distribution in organization types for repayers and defaulters")
plt.show()
```



In [131...

```
fig = plt.figure(figsize=(20,20))
plt.subplot(421)
sns.boxplot(data=application_data,x='TARGET',y='OBS_30_CNT_SOCIAL_CIRCLE',
            hue="TARGET", palette="Set3")
plt.title("Client's social surroundings with observable 30 DPD (days past due) def",color="b")
plt.subplot(422)
sns.boxplot(data=application_data,x='TARGET',y='DEF_30_CNT_SOCIAL_CIRCLE',
            hue="TARGET", palette="Set3")
plt.title("Client's social surroundings defaulted on 30 DPD (days past due)",color="b")
plt.subplot(423)
sns.boxplot(data=application_data,x='TARGET',y='OBS_60_CNT_SOCIAL_CIRCLE',
```

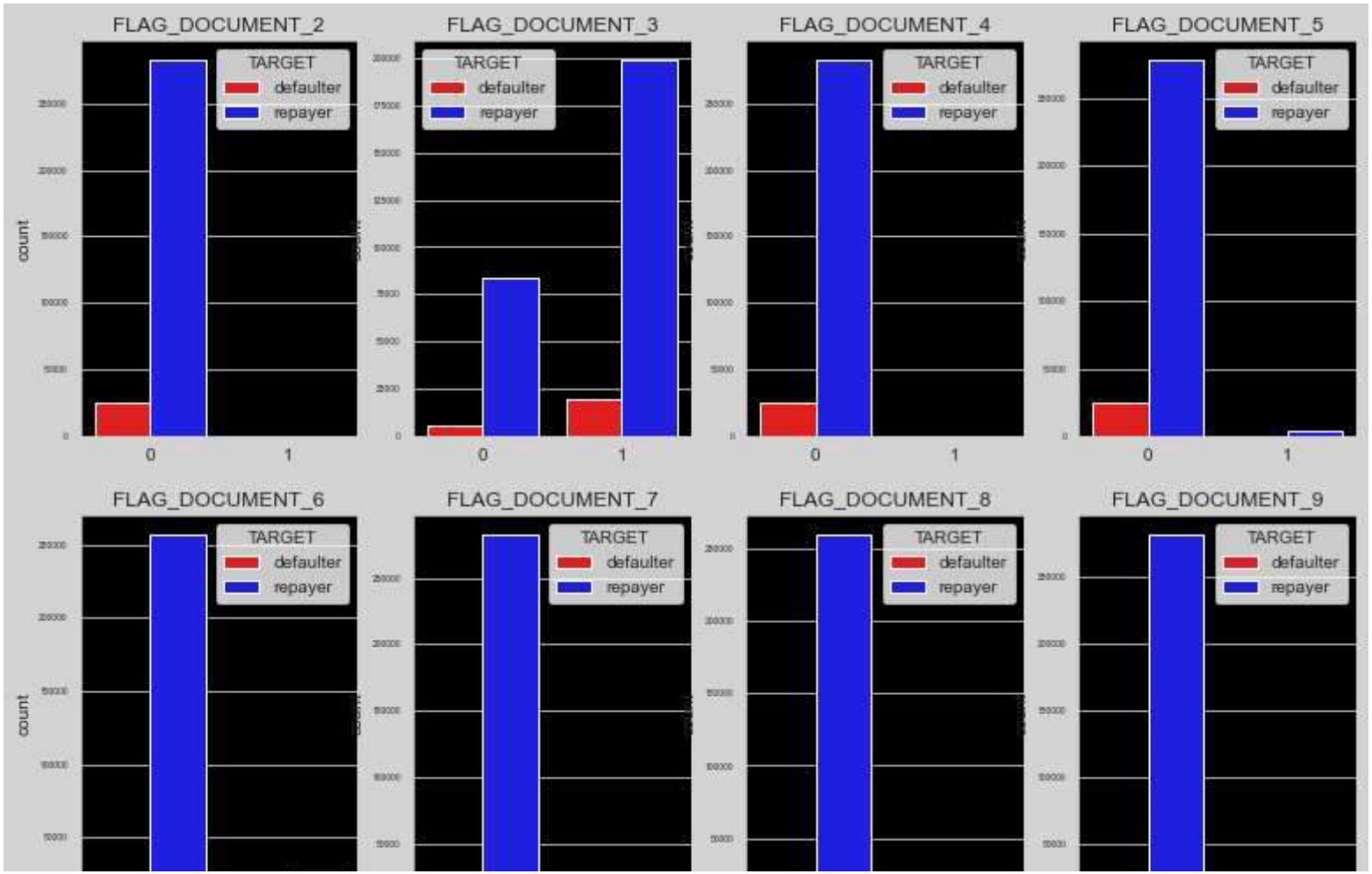
```
            hue="TARGET", palette="Set3")
plt.title("Client's social surroundings with observable 60 DPD (days past due) default",color="b")
plt.subplot(424)
sns.boxplot(data=application_data,x='TARGET',y='DEF_60_CNT_SOCIAL_CIRCLE',
            hue="TARGET", palette="Set3")
plt.title("Client's social surroundings defaulted on 60 DPD (days past due)",color="b")
fig.set_facecolor("ghostwhite")
```

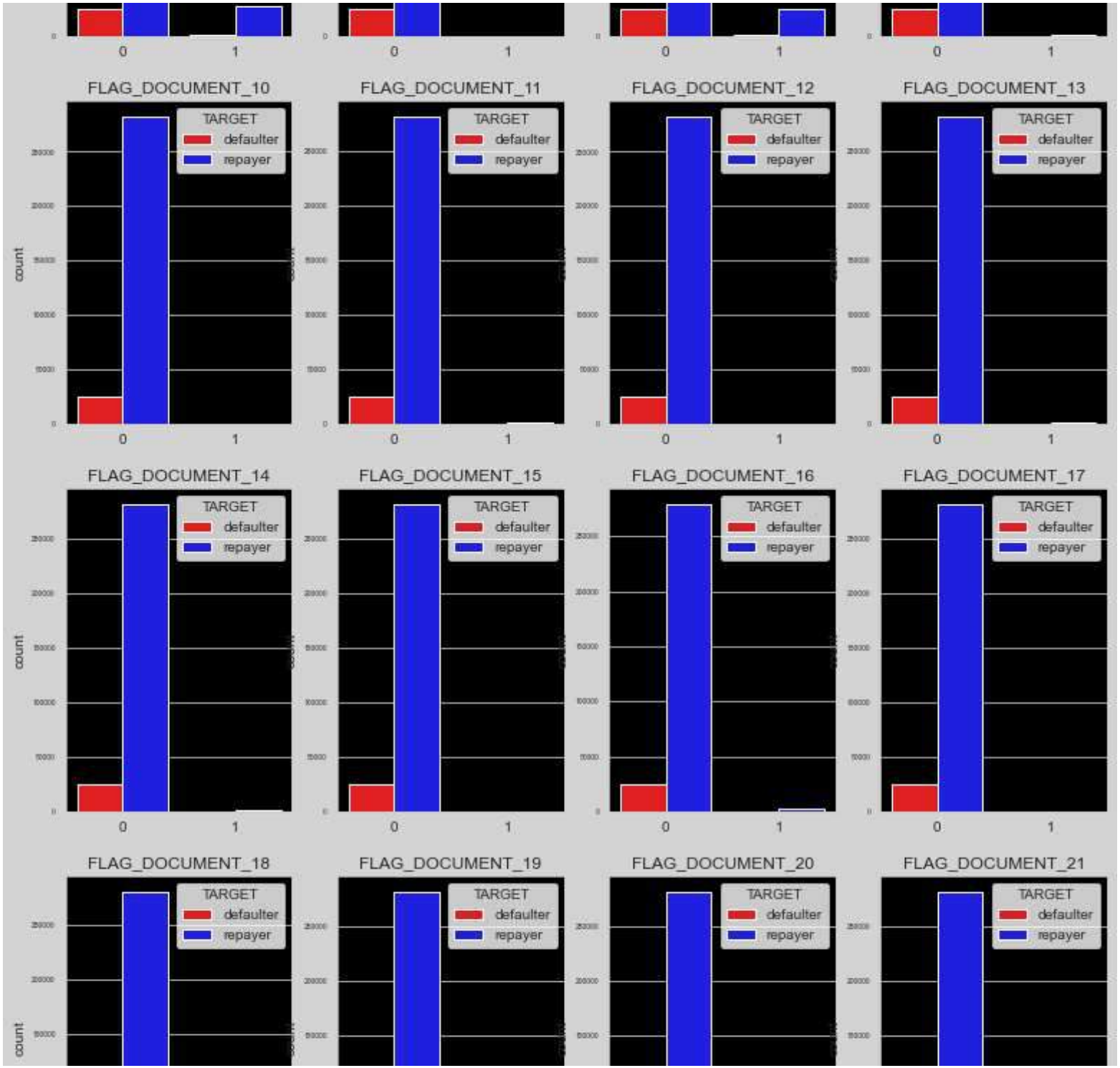


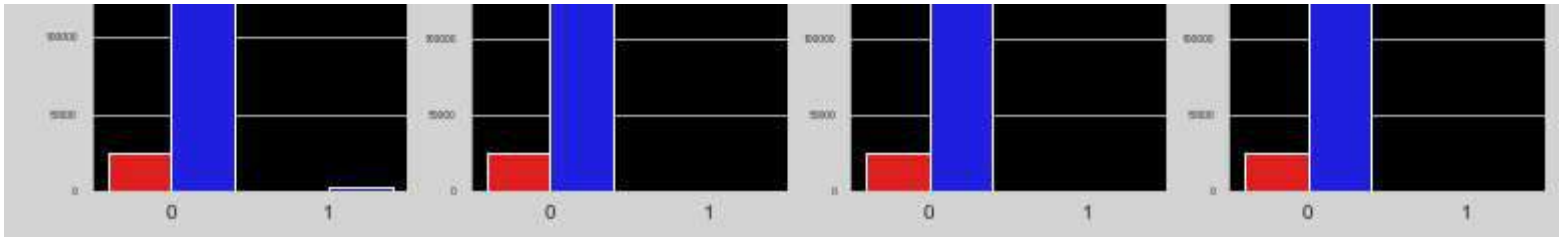
In [133...

```
cols = [ 'FLAG_DOCUMENT_2', 'FLAG_DOCUMENT_3',
       'FLAG_DOCUMENT_4', 'FLAG_DOCUMENT_5', 'FLAG_DOCUMENT_6',
       'FLAG_DOCUMENT_7', 'FLAG_DOCUMENT_8', 'FLAG_DOCUMENT_9',
       'FLAG_DOCUMENT_10', 'FLAG_DOCUMENT_11', 'FLAG_DOCUMENT_12',
       'FLAG_DOCUMENT_13', 'FLAG_DOCUMENT_14', 'FLAG_DOCUMENT_15',
       'FLAG_DOCUMENT_16', 'FLAG_DOCUMENT_17', 'FLAG_DOCUMENT_18',
       'FLAG_DOCUMENT_19', 'FLAG_DOCUMENT_20', 'FLAG_DOCUMENT_21']

df_flag = application_data[cols+["TARGET"]]
```

```
length = len(cols)

df_flag["TARGET"] = df_flag["TARGET"].replace({1:"defaulter",0:"repayer"})

fig = plt.figure(figsize=(13,24))
fig.set_facecolor("lightgrey")
for i,j in itertools.zip_longest(cols,range(length)):
    plt.subplot(5,4,j+1)
    ax = sns.countplot(df_flag[i],hue=df_flag["TARGET"],palette=["r","b"])
    plt.yticks(fontsize=5)
    plt.xlabel("")
    plt.title(i)
    ax.set_facecolor("k")
```

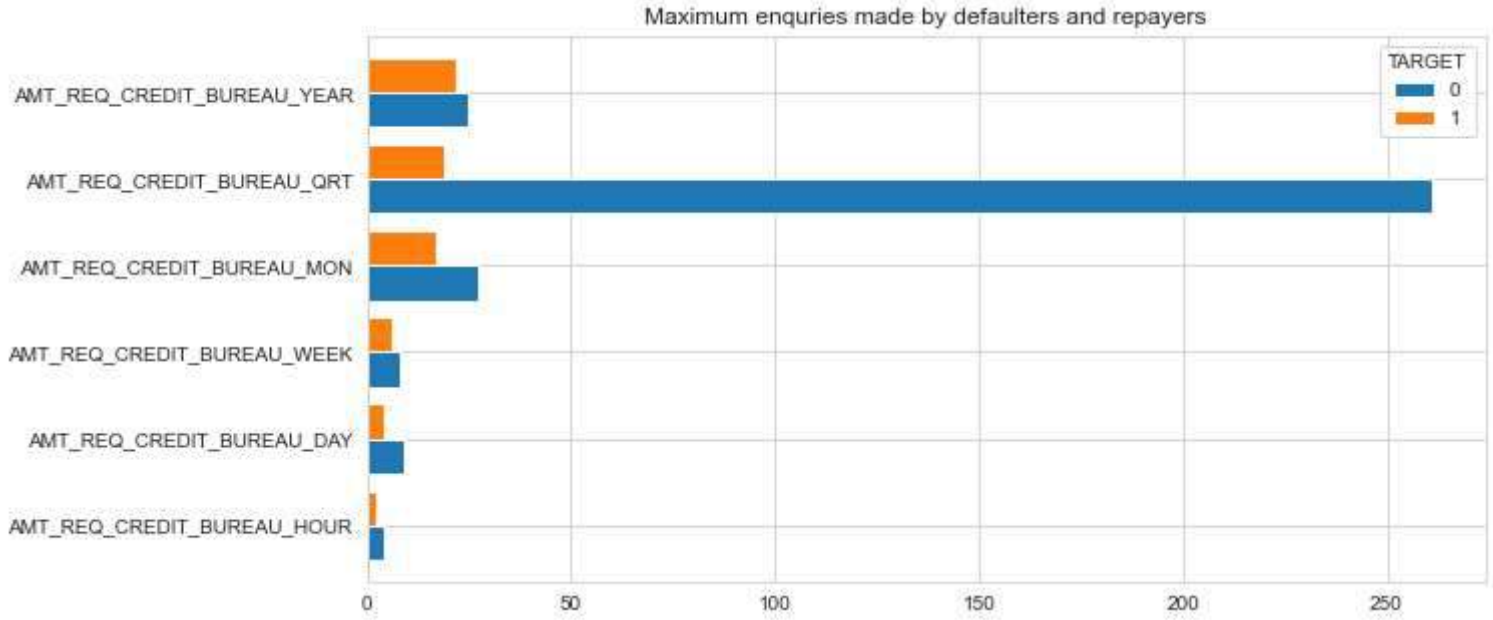


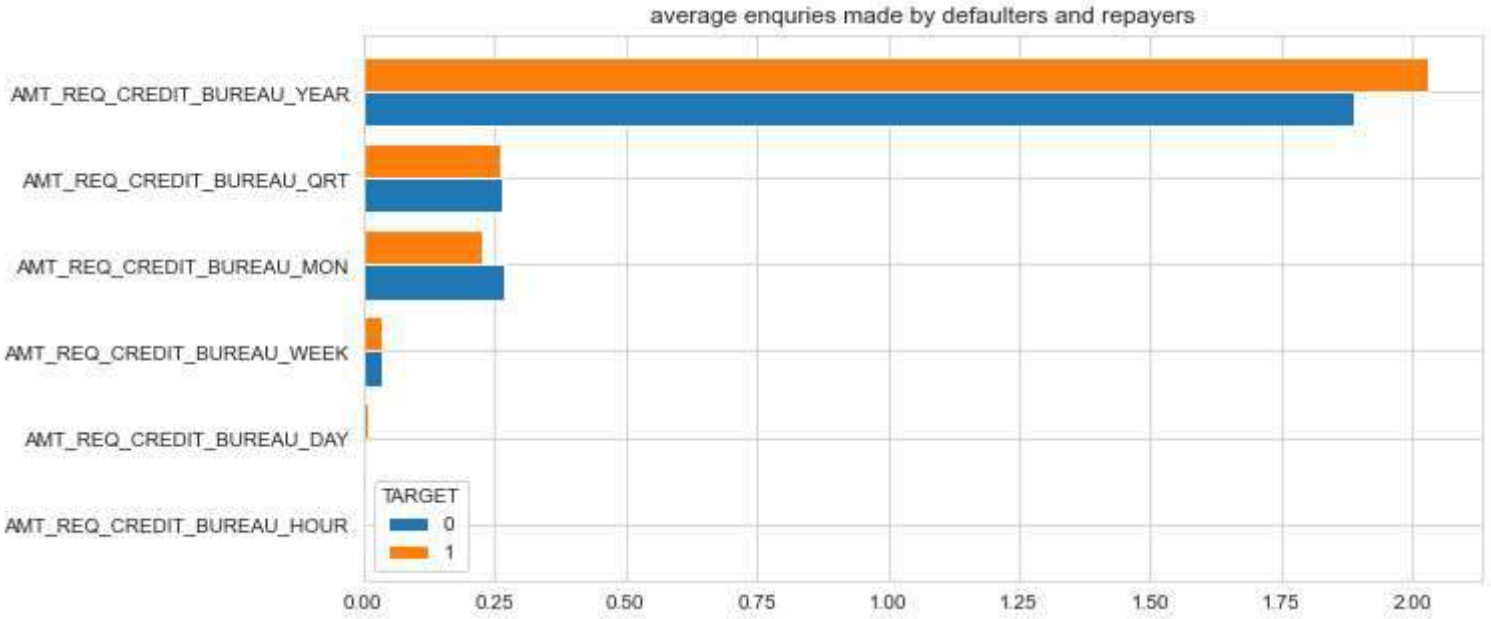

FLAG_DOCUMENT_10
FLAG_DOCUMENT_11
FLAG_DOCUMENT_12
FLAG_DOCUMENT_13
FLAG_DOCUMENT_14
FLAG_DOCUMENT_15
FLAG_DOCUMENT_16
FLAG_DOCUMENT_17
FLAG_DOCUMENT_18
FLAG_DOCUMENT_19
FLAG_DOCUMENT_20
FLAG_DOCUMENT_21
TARGET
defaulter
repayer
count
0
1

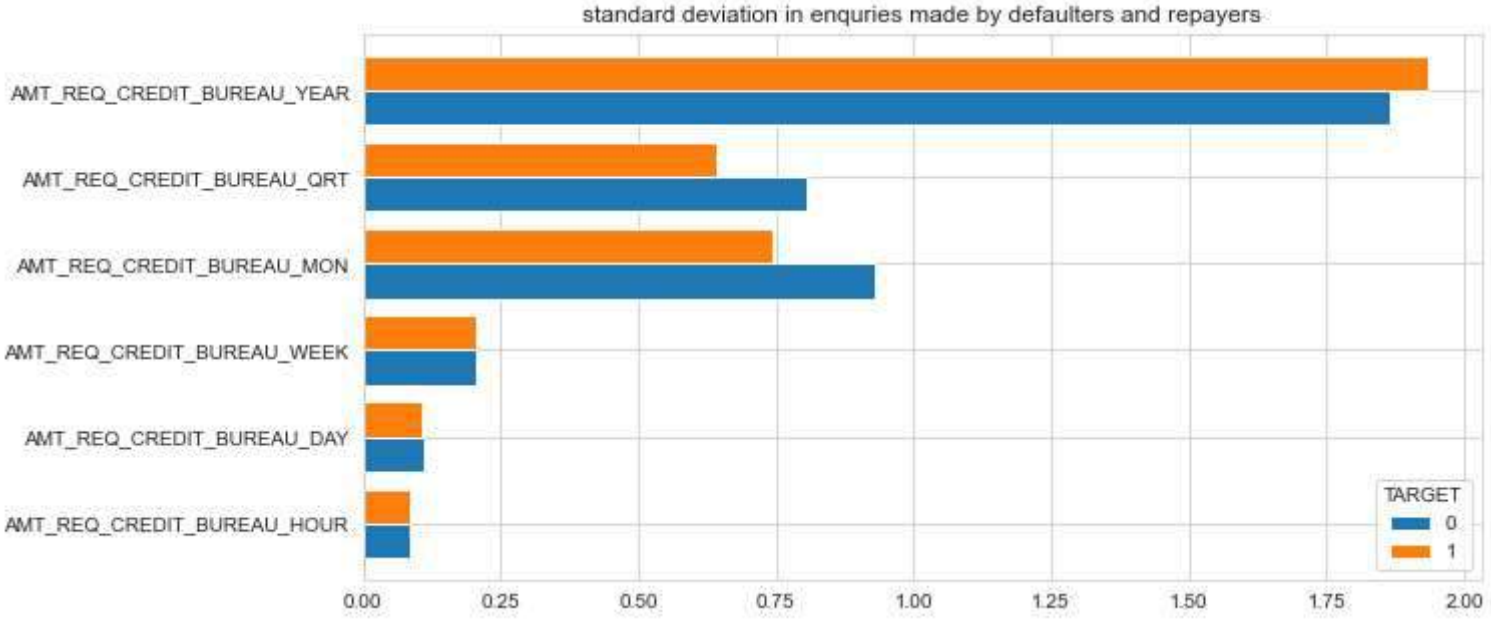


In [134...

```
cols = ['AMT_REQ_CREDIT_BUREAU_HOUR', 'AMT_REQ_CREDIT_BUREAU_DAY',
       'AMT_REQ_CREDIT_BUREAU_WEEK', 'AMT_REQ_CREDIT_BUREAU_MON',
       'AMT_REQ_CREDIT_BUREAU_QRT', 'AMT_REQ_CREDIT_BUREAU_YEAR']
application_data.groupby("TARGET")[cols].max().transpose().plot(kind="barh",
                                                                figsize=(10,5),width=.8)
plt.title("Maximum enquries made by defaulters and repayers")
application_data.groupby("TARGET")[cols].mean().transpose().plot(kind="barh",
                                                                 figsize=(10,5),width=.8)
plt.title("average enquries made by defaulters and repayers")
application_data.groupby("TARGET")[cols].std().transpose().plot(kind="barh",
                                                                figsize=(10,5),width=.8)
plt.title("standard deviation in enquries made by defaulters and repayers")
plt.show()
```



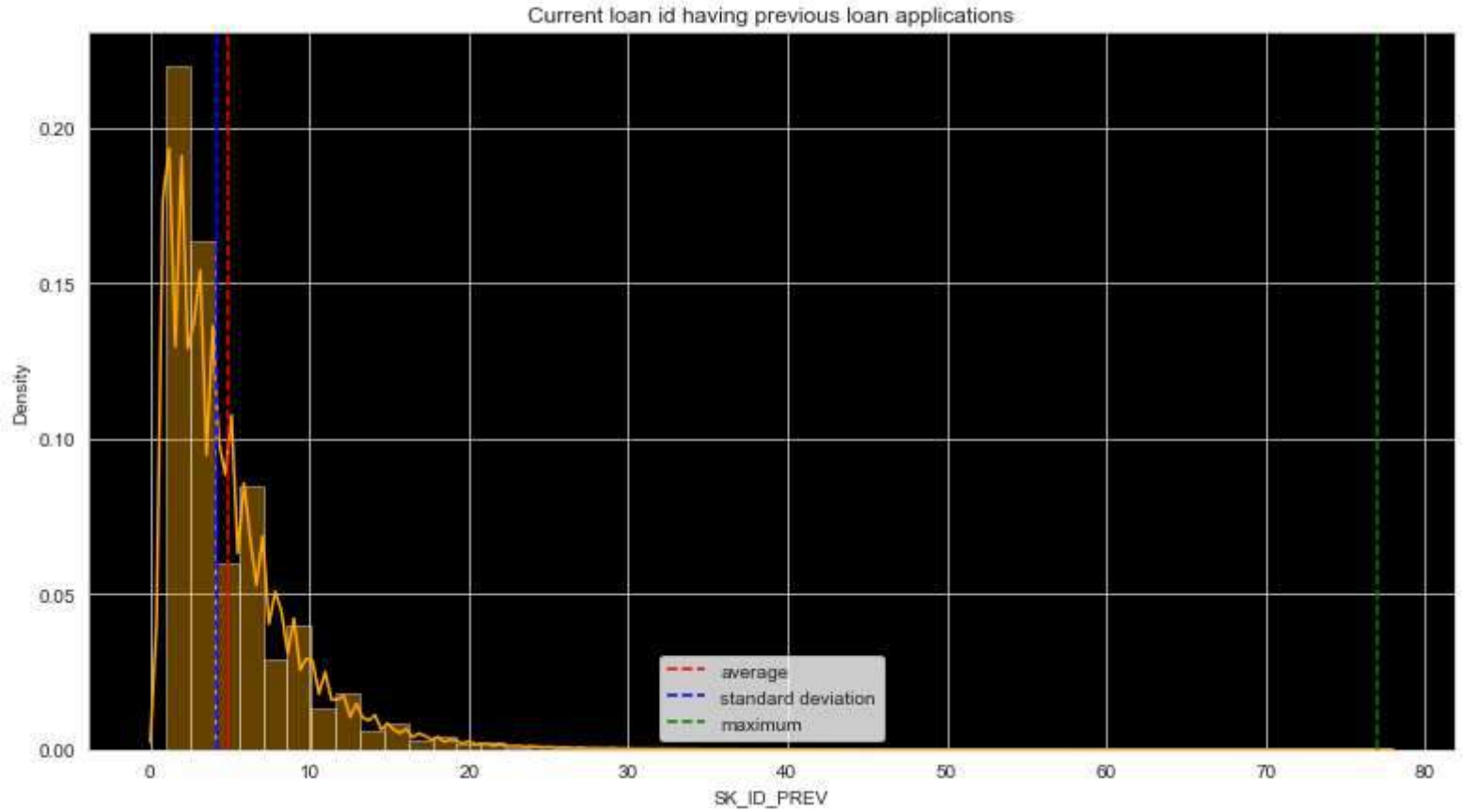




In [135...

```
x = previous_application.groupby("SK_ID_CURR")["SK_ID_PREV"].count().reset_index()
plt.figure(figsize=(13,7))
ax = sns.distplot(x["SK_ID_PREV"],color="orange")
plt.axvline(x["SK_ID_PREV"].mean(),linestyle="dashed",color="r",label="average")
```

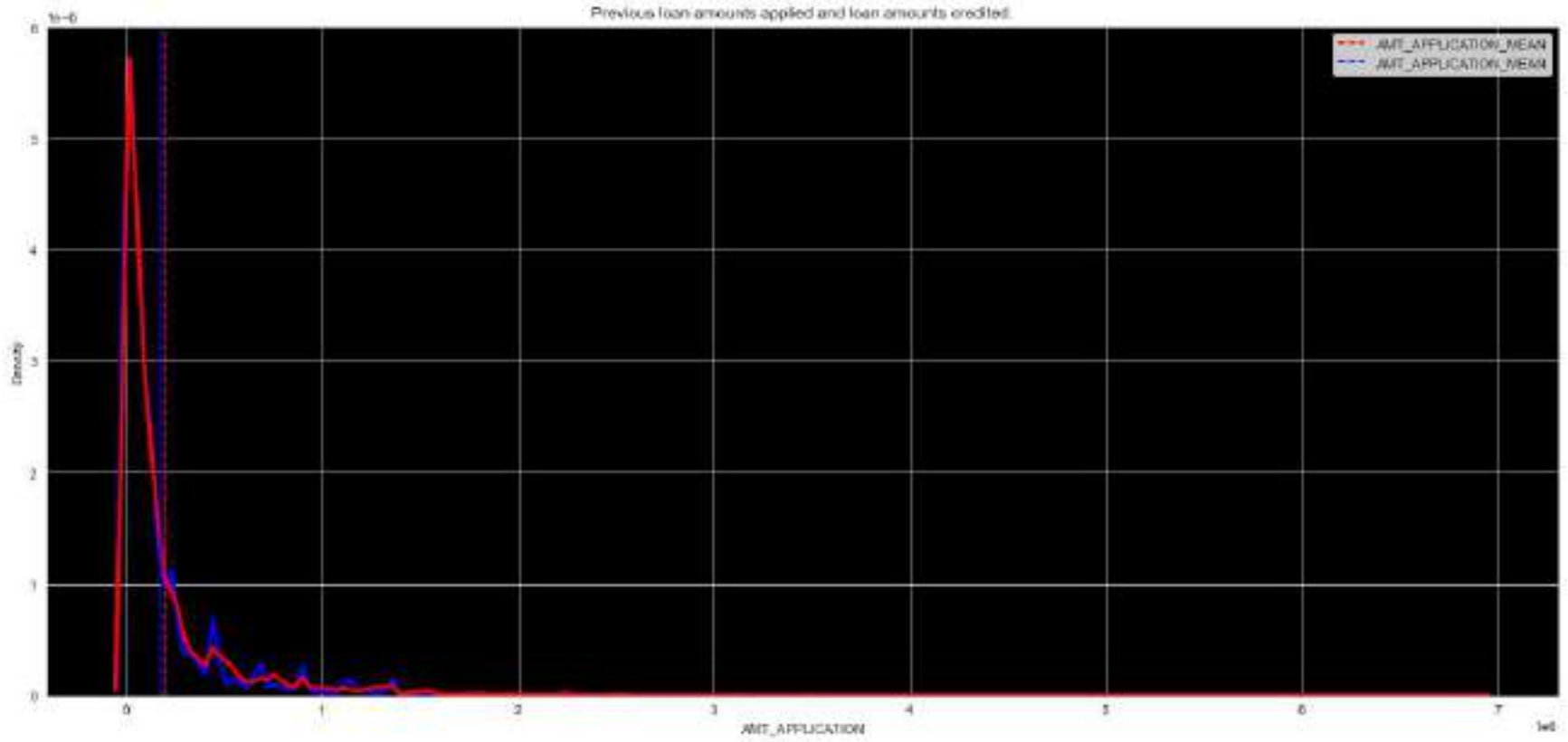
```
plt.axvline(x["SK_ID_PREV"].std(),linestyle="dashed",color="b",label="standard deviation")
plt.axvline(x["SK_ID_PREV"].max(),linestyle="dashed",color="g",label="maximum")
plt.legend(loc="best")
plt.title("Current loan id having previous loan applications")
ax.set_facecolor("k")
```

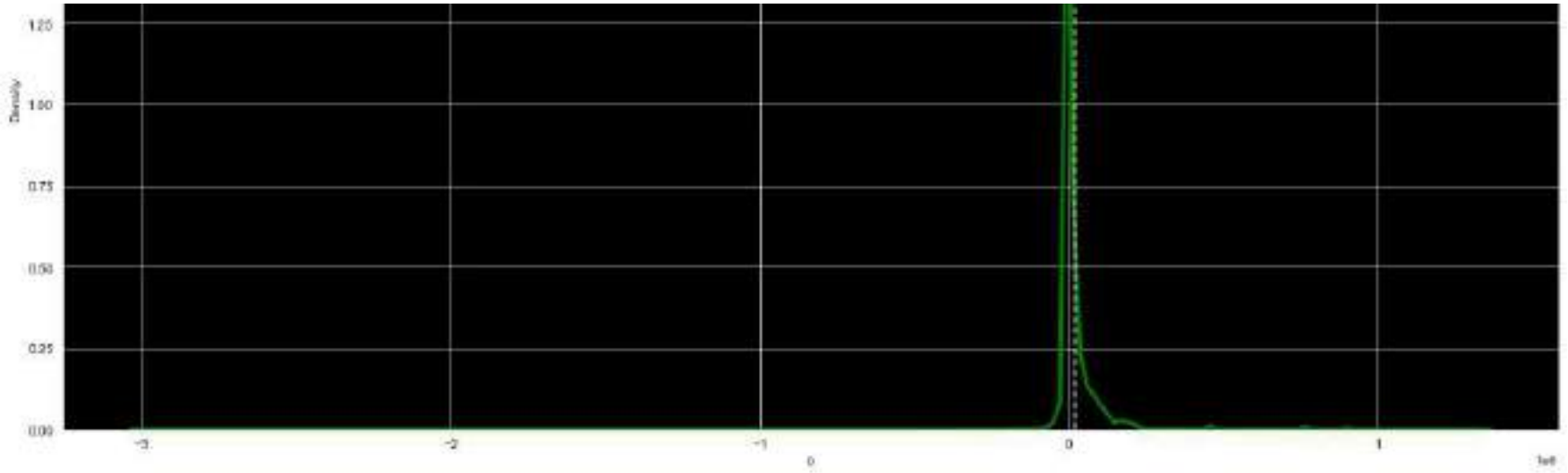


In [138...

```
plt.figure(figsize=(20,20))
plt.subplot(211)
ax = sns.kdeplot(previous_application["AMT_APPLICATION"],color="b",linewidth=3)
ax = sns.kdeplot(previous_application[previous_application["AMT_CREDIT"].notnull()]["AMT_CREDIT"],color="r",linewidth=3)
plt.axvline(previous_application[previous_application["AMT_CREDIT"].notnull()]["AMT_CREDIT"].mean(),color="r",linestyle="
plt.axvline(previous_application["AMT_APPLICATION"].mean(),color="b",linestyle="dashed",label="AMT_APPLICATION_MEAN")
plt.legend(loc="best")
plt.title("Previous loan amounts applied and loan amounts credited.")
ax.set_facecolor("k")
```

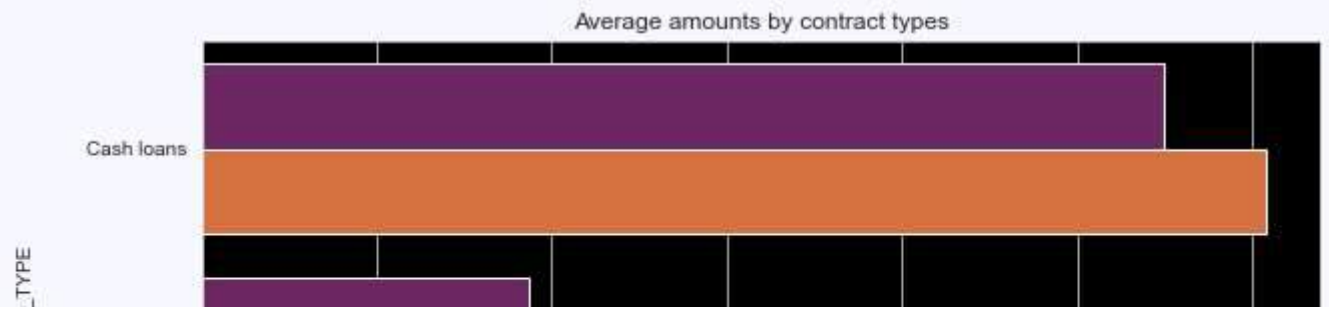
```
plt.subplot(212)
diff = (previous_application["AMT_CREDIT"] - previous_application["AMT_APPLICATION"]).reset_index()
diff = diff[diff[0].notnull()]
ax1 = sns.kdeplot(diff[0],color="g",linewidth=3,label = "difference in amount requested by client and amount credited")
plt.axvline(diff[0].mean(),color="white",linestyle="dashed",label = "mean")
plt.title("difference in amount requested by client and amount credited")
ax1.legend(loc="best")
ax1.set_facecolor("k")
```

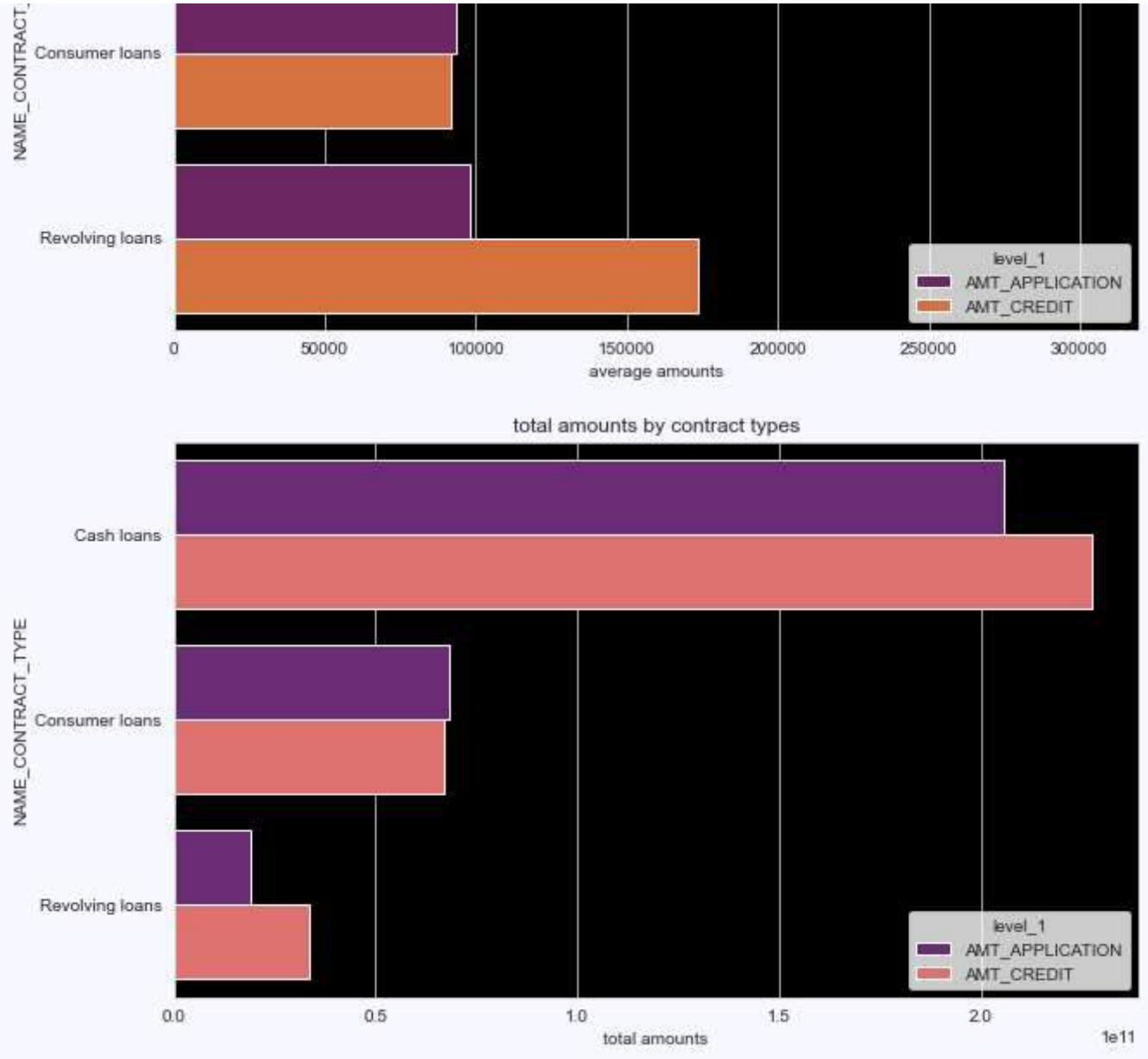




In [139...

```
mn = previous_application.groupby("NAME_CONTRACT_TYPE")[["AMT_APPLICATION","AMT_CREDIT"]].mean().stack().reset_index()
tt = previous_application.groupby("NAME_CONTRACT_TYPE")[["AMT_APPLICATION","AMT_CREDIT"]].sum().stack().reset_index()
fig = plt.figure(figsize=(10,13))
fig.set_facecolor("ghostwhite")
plt.subplot(211)
ax = sns.barplot(0,"NAME_CONTRACT_TYPE",data=mn[:6],hue="level_1",palette="inferno")
ax.set_facecolor("k")
ax.set_xlabel("average amounts")
ax.set_title("Average amounts by contract types")

plt.subplot(212)
ax1 = sns.barplot(0,"NAME_CONTRACT_TYPE",data=tt[:6],hue="level_1",palette="magma")
ax1.set_facecolor("k")
ax1.set_xlabel("total amounts")
ax1.set_title("total amounts by contract types")
plt.subplots_adjust(hspace = .2)
plt.show()
```

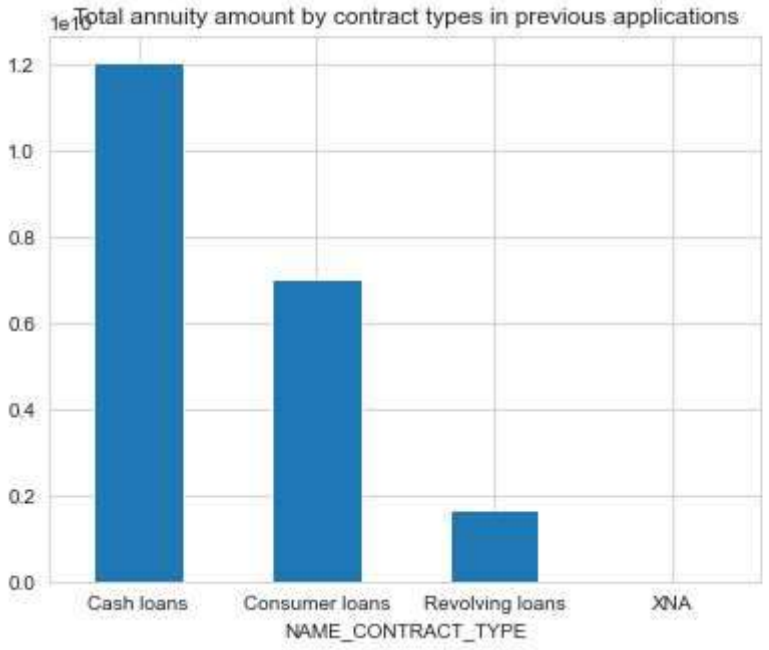


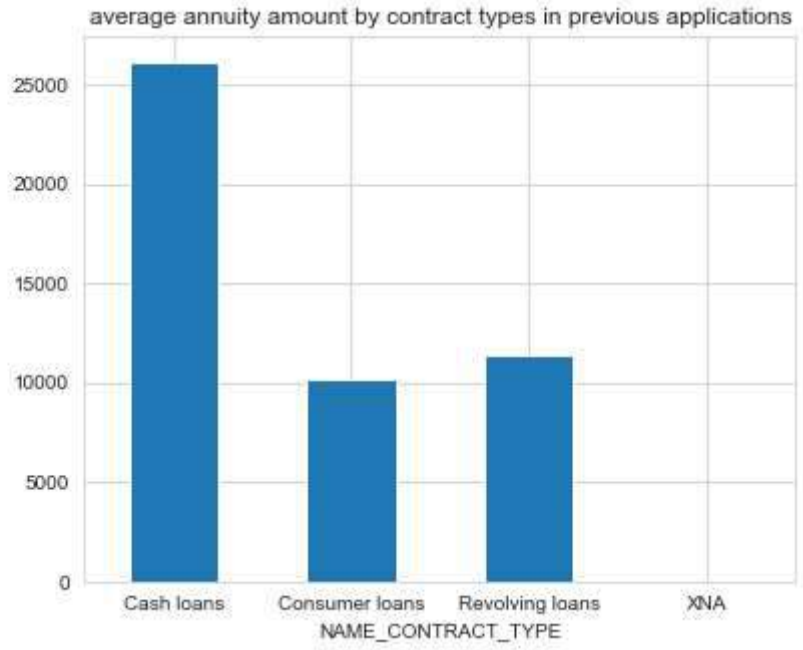


In [140...

```
plt.figure(figsize=(14,5))
plt.subplot(121)
previous_application.groupby("NAME_CONTRACT_TYPE")["AMT_ANNUITY"].sum().plot(kind="bar")
plt.xticks(rotation=0)
plt.title("Total annuity amount by contract types in previous applications")
plt.subplot(122)
```

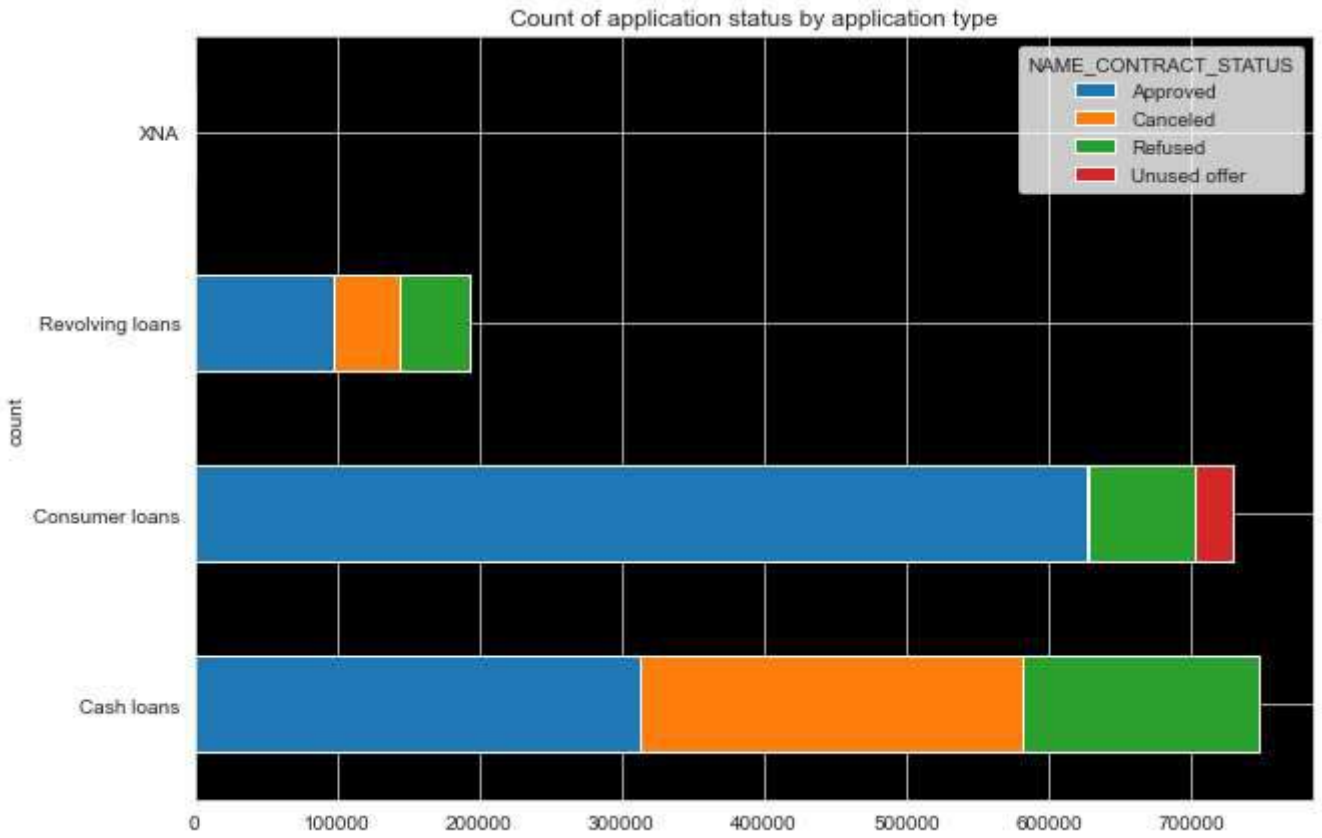
```
previous_application.groupby("NAME_CONTRACT_TYPE")["AMT_ANNUITY"].mean().plot(kind="bar")
plt.title("average annuity amount by contract types in previous applications")
plt.xticks(rotation=0)
plt.show()
```





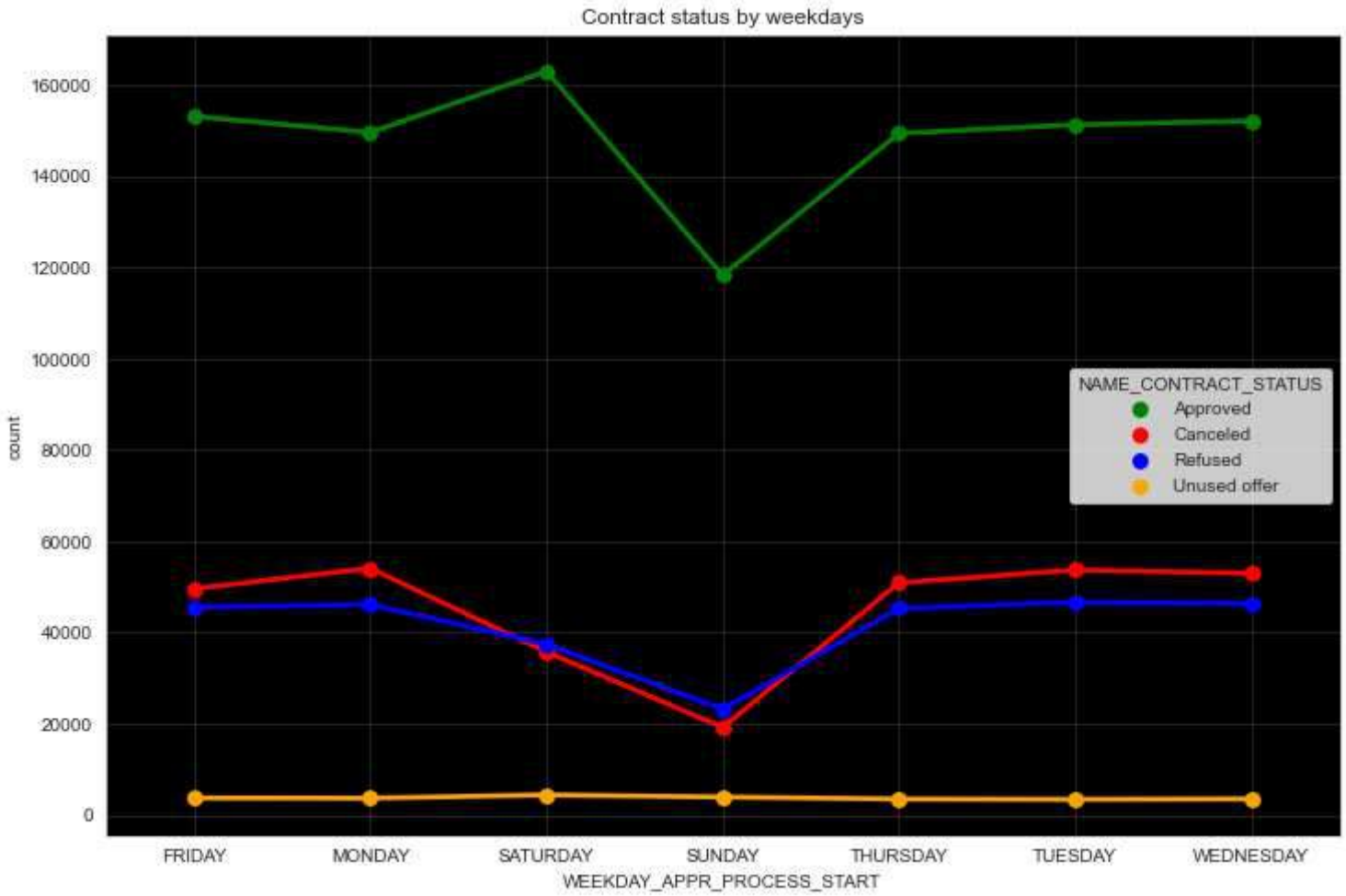
In [141]:

```
ax = pd.crosstab(previous_application["NAME_CONTRACT_TYPE"],previous_application["NAME_CONTRACT_STATUS"]).plot(kind="barh
plt.xticks(rotation =0)
plt.ylabel("count")
plt.title("Count of application status by application type")
ax.set_facecolor("k")
```



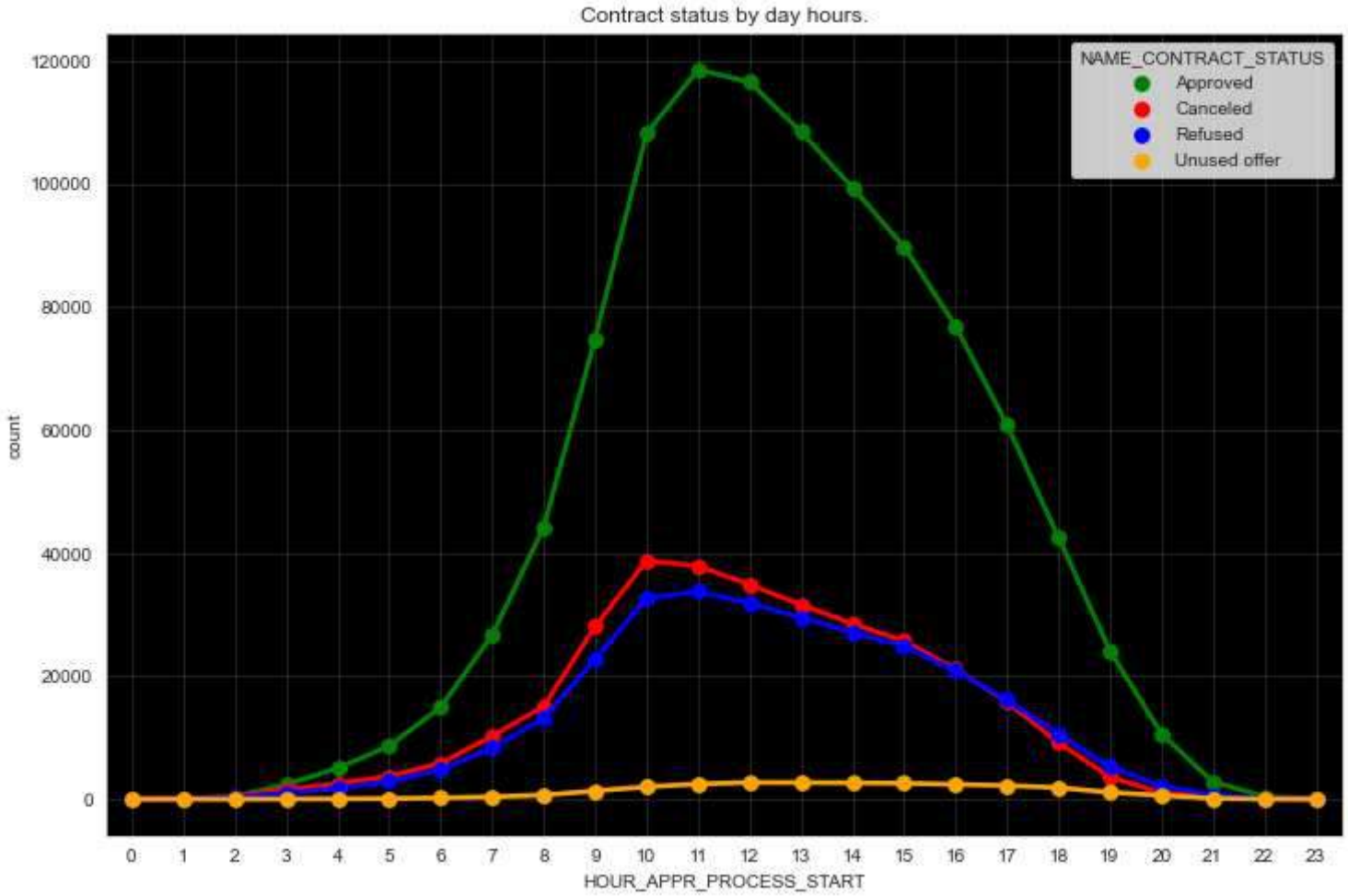
In [142...

```
hr = pd.crosstab(previous_application["WEEKDAY_APPR_PROCESS_START"],previous_application["NAME_CONTRACT_STATUS"]).stack()
plt.figure(figsize=(12,8))
ax = sns.pointplot(hr["WEEKDAY_APPR_PROCESS_START"],hr[0],hue=hr["NAME_CONTRACT_STATUS"],palette=["g","r","b","orange"],s
ax.set_facecolor("k")
ax.set_ylabel("count")
ax.set_title("Contract status by weekdays")
plt.grid(True,alpha=.2)
```



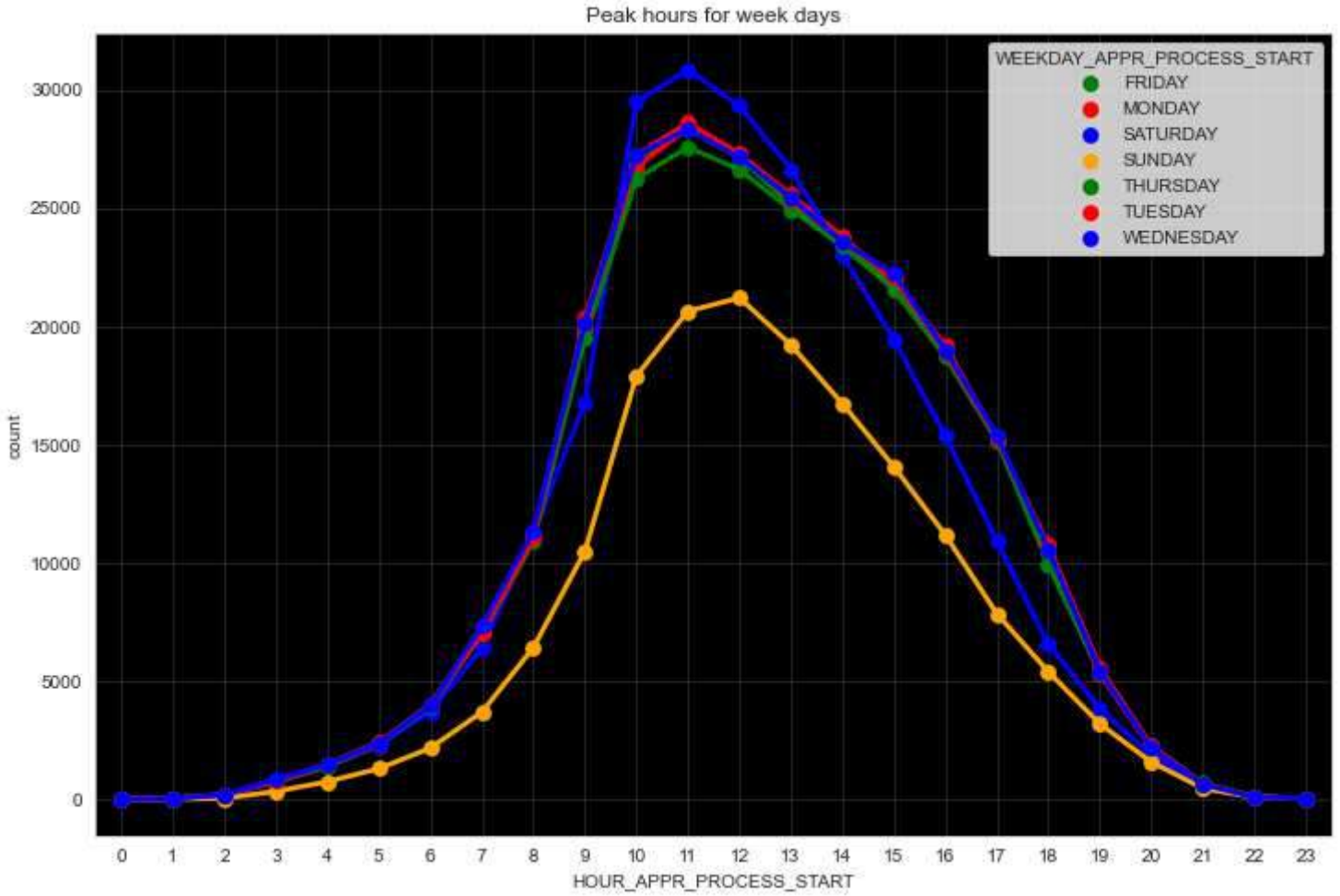
In [143]:

```
hr = pd.crosstab(previous_application["HOUR_APPR_PROCESS_START"],previous_application["NAME_CONTRACT_STATUS"]).stack().re
plt.figure(figsize=(12,8))
ax = sns.pointplot(hr["HOUR_APPR_PROCESS_START"],hr[0],hue=hr["NAME_CONTRACT_STATUS"],palette=["g","r","b","orange"],scal
ax.set_facecolor("k")
ax.set_ylabel("count")
ax.set_title("Contract status by day hours.")
plt.grid(True,alpha=.2)
```



In [144_

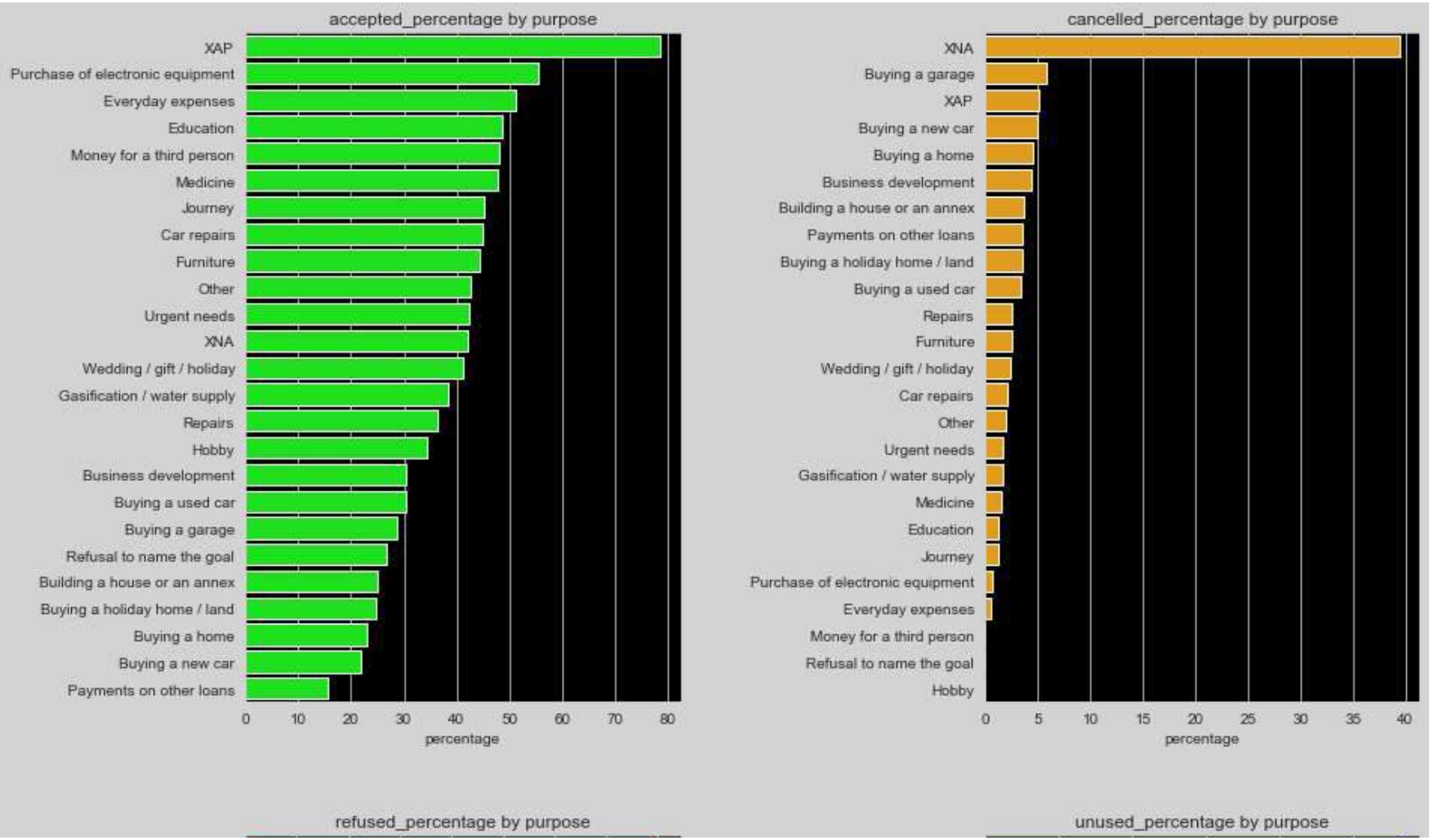
```
hr = pd.crosstab(previous_application["HOUR_APPR_PROCESS_START"],previous_application["WEEKDAY_APPR_PROCESS_START"]).stac
plt.figure(figsize=(12,8))
ax = sns.pointplot(hr["HOUR_APPR_PROCESS_START"],hr[0],hue=hr["WEEKDAY_APPR_PROCESS_START"],palette=["g","r","b","orange"
ax.set_facecolor("k")
ax.set_ylabel("count")
ax.set_title("Peak hours for week days")
plt.grid(True,alpha=.2)
```

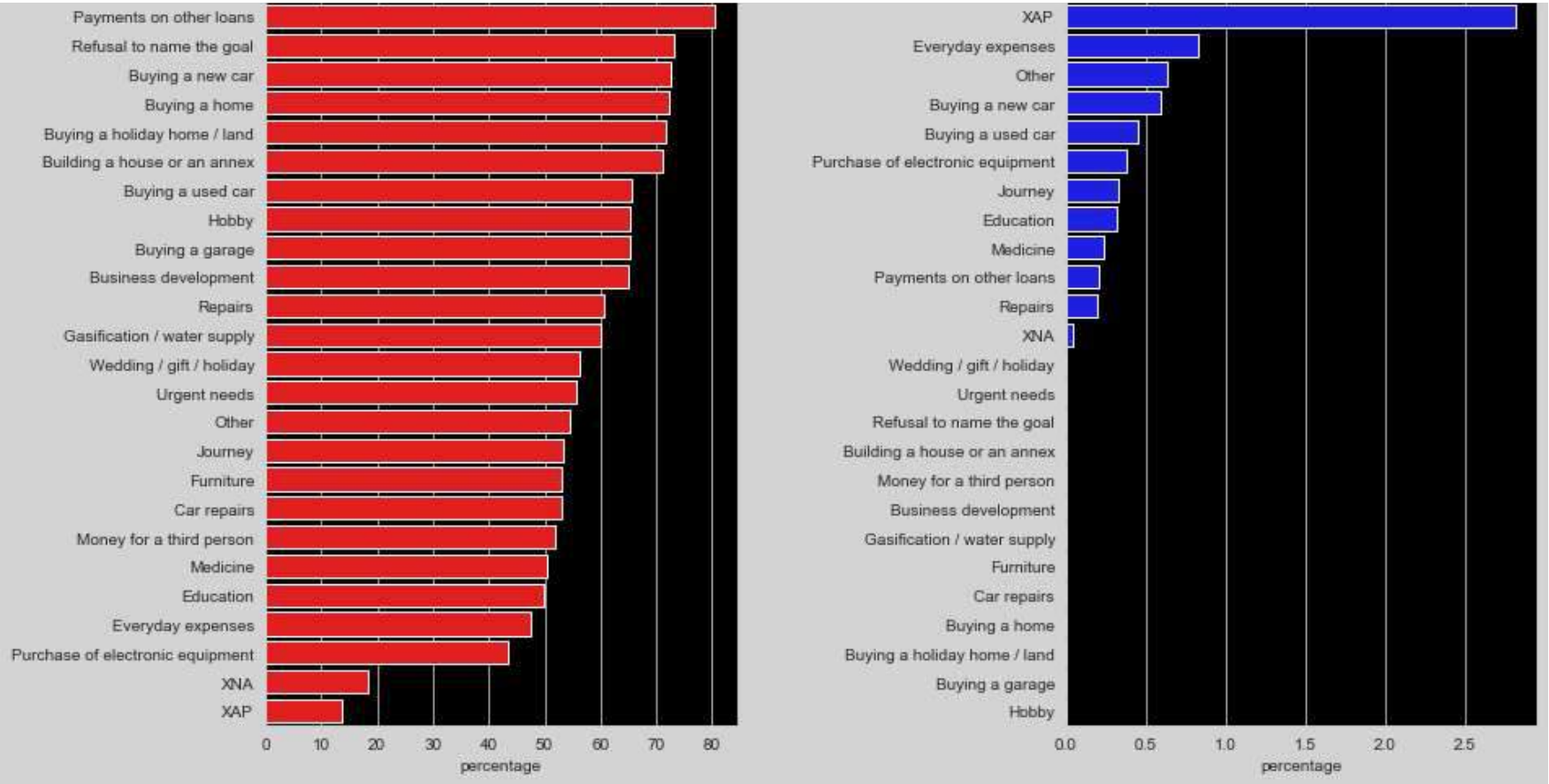


In [145...

```
previous_application[["NAME_CASH_LOAN_PURPOSE","NAME_CONTRACT_STATUS"]]
purpose = pd.crosstab(previous_application["NAME_CASH_LOAN_PURPOSE"],previous_application["NAME_CONTRACT_STATUS"])
purpose["a"] = (purpose["Approved"]*100)/(purpose["Approved"]+purpose["Canceled"]+purpose["Refused"]+purpose["Unused offe
purpose["c"] = (purpose["Canceled"]*100)/(purpose["Approved"]+purpose["Canceled"]+purpose["Refused"]+purpose["Unused offe
purpose["r"] = (purpose["Refused"]*100)/(purpose["Approved"]+purpose["Canceled"]+purpose["Refused"]+purpose["Unused offer
purpose["u"] = (purpose["Unused offer"]*100)/(purpose["Approved"]+purpose["Canceled"]+purpose["Refused"]+purpose["Unused
purpose_new = purpose[["a","c","r","u"]]
purpose_new = purpose_new.stack().reset_index()
purpose_new["NAME_CONTRACT_STATUS"] = purpose_new["NAME_CONTRACT_STATUS"].replace({"a":"accepted_percentage","c":"cancell
                                                                                   "r":"refused_percentage","u":"unused_percentage"})

lst = purpose_new["NAME_CONTRACT_STATUS"].unique().tolist()
length = len(lst)
```

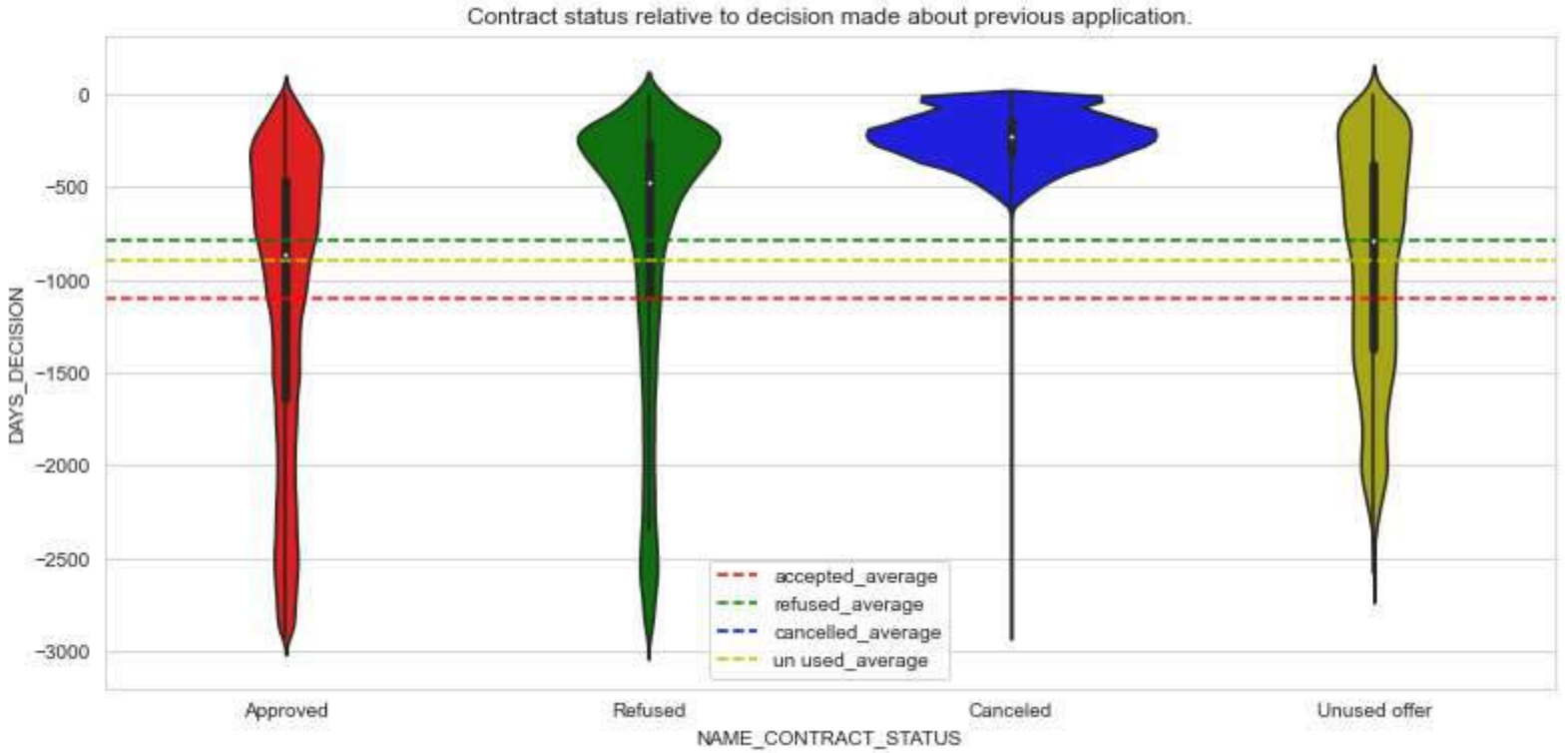
```
cs = ["lime","orange","r","b"]

fig = plt.figure(figsize=(14,18))
fig.set_facecolor("lightgrey")
for i,j,k in itertools.zip_longest(lst,range(length),cs):
    plt.subplot(2,2,j+1)
    dat = purpose_new[purpose_new["NAME_CONTRACT_STATUS"] == i]
    ax = sns.barplot(0,"NAME_CASH_LOAN_PURPOSE",data=dat.sort_values(by=0,ascending=False),color=k)
    plt.ylabel("")
    plt.xlabel("percentage")
    plt.title(i+" by purpose")
    plt.subplots_adjust(wspace = .7)
    ax.set_facecolor("k")
```





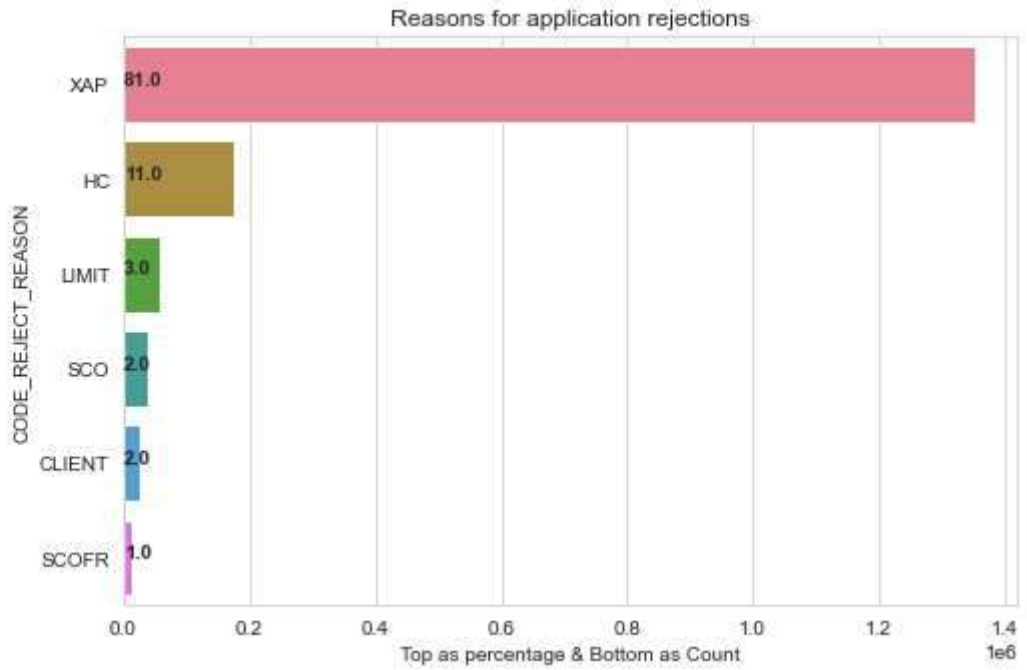
In [146]:

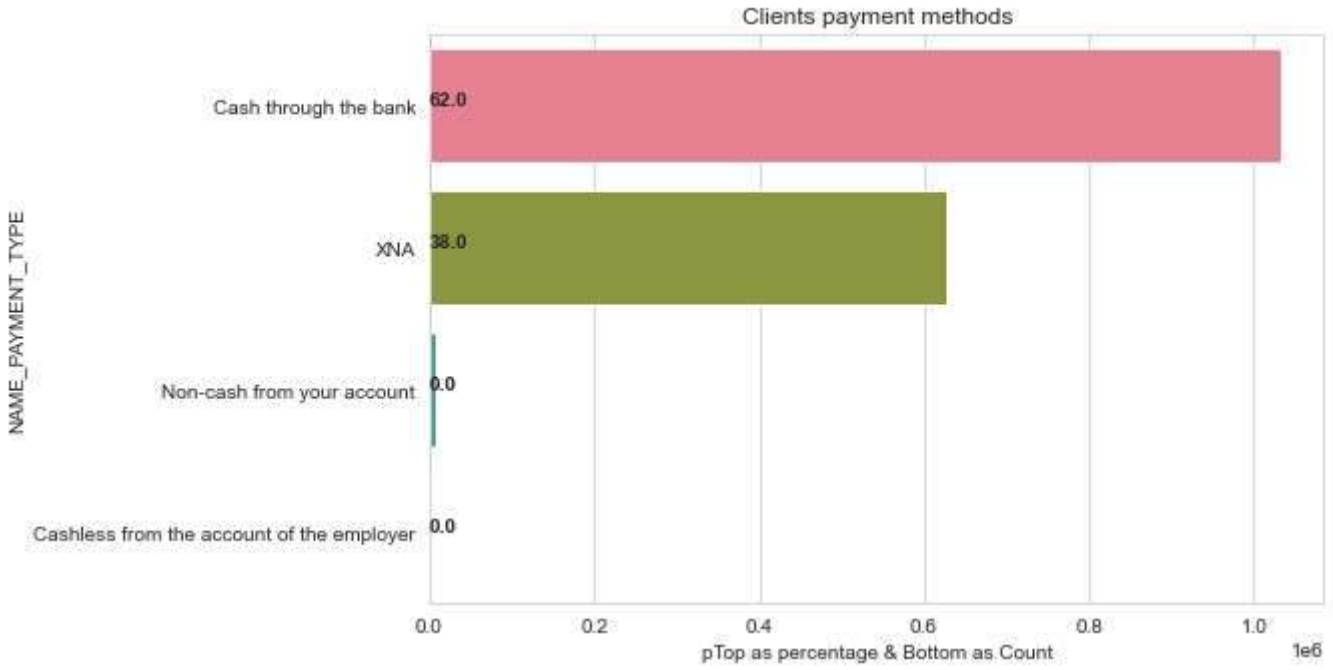
```
plt.figure(figsize=(13,6))
sns.violinplot(y= previous_application["DAYS_DECISION"],
               x = previous_application["NAME_CONTRACT_STATUS"],palette=["r","g","b","y"])
plt.axhline(previous_application[previous_application["NAME_CONTRACT_STATUS"] == "Approved"]["DAYS_DECISION"].mean(),
            color="r",linestyle="dashed",label="accepted_average")
plt.axhline(previous_application[previous_application["NAME_CONTRACT_STATUS"] == "Refused"]["DAYS_DECISION"].mean(),
            color="g",linestyle="dashed",label="refused_average")
plt.axhline(previous_application[previous_application["NAME_CONTRACT_STATUS"] == "Cancelled"]["DAYS_DECISION"].mean(),col
            linestyle="dashed",label="cancelled_average")
plt.axhline(previous_application[previous_application["NAME_CONTRACT_STATUS"] == "Unused offer"]["DAYS_DECISION"].mean(),
            linestyle="dashed",label="un used_average")
plt.legend(loc="best")

plt.title("Contract status relative to decision made about previous application.")
plt.show()
```



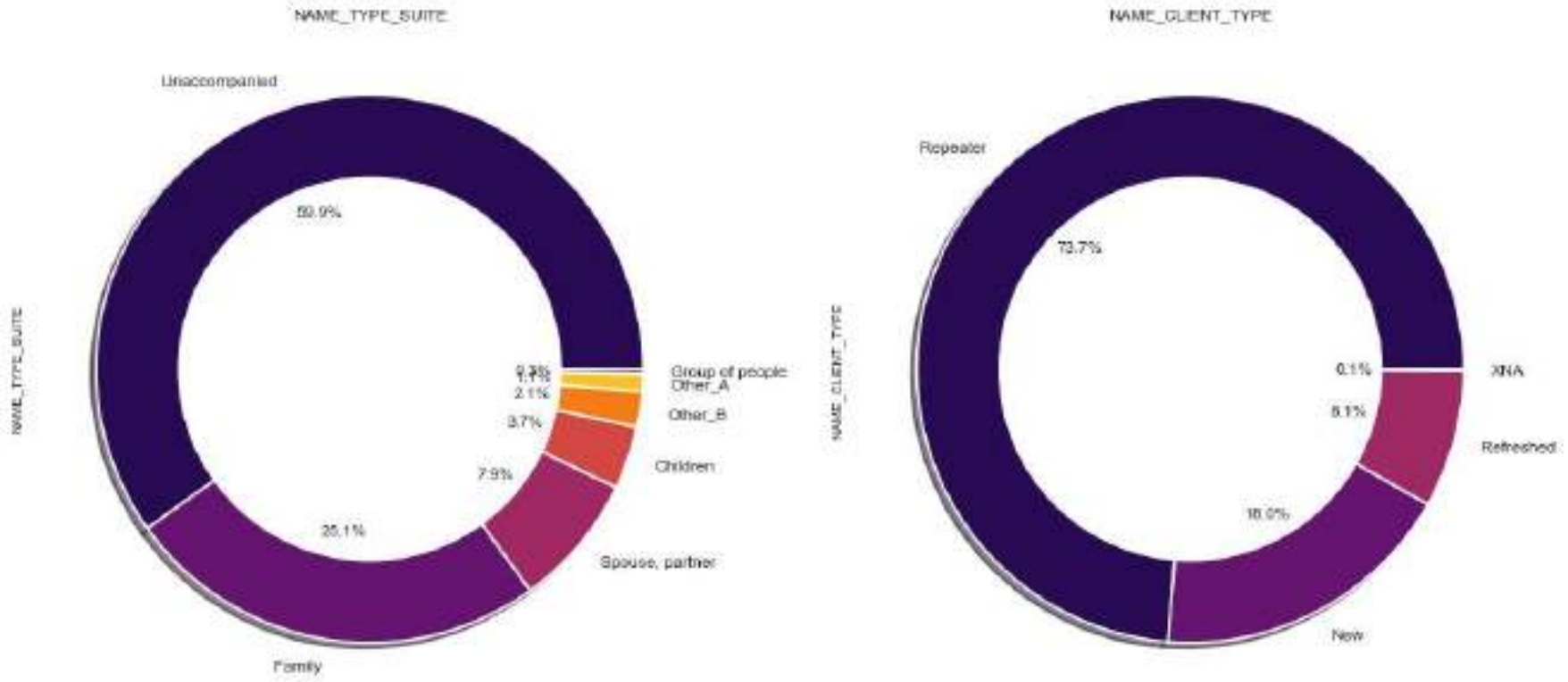
In [147]:

```
plt.figure(figsize=(8,12))
plt.subplot(211)
rej = previous_application["CODE_REJECT_REASON"].value_counts().reset_index()
ax = sns.barplot("CODE_REJECT_REASON","index",data=rej[:6],palette="husl")
for i,j in enumerate(np.around((rej["CODE_REJECT_REASON"][:6].values*100/(rej["CODE_REJECT_REASON"][:6].sum())))):
    ax.text(.7,i,j,weight="bold")
plt.xlabel("Top as percentage & Bottom as Count")
plt.ylabel("CODE_REJECT_REASON")
plt.title("Reasons for application rejections")

plt.subplot(212)
pay = previous_application["NAME_PAYMENT_TYPE"].value_counts().reset_index()
ax1 = sns.barplot("NAME_PAYMENT_TYPE","index",data=pay,palette="husl")
for i,j in enumerate(np.around((pay["NAME_PAYMENT_TYPE"].values*100/(pay["NAME_PAYMENT_TYPE"].sum())))):
    ax1.text(.7,i,j,weight="bold")
plt.xlabel("pTop as percentage & Bottom as Count")
plt.ylabel("NAME_PAYMENT_TYPE")
plt.title("Clients payment methods")
plt.subplots_adjust(hspace = .3)
```


Reasons for application rejections
CODE_REJECT_REASON
XAP
HC
LIMIT
SCO
CLIENT
SCOFR
81.0
11.0
3.0
2.0
2.0
1.0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1e6
Top as percentage & Bottom as Count

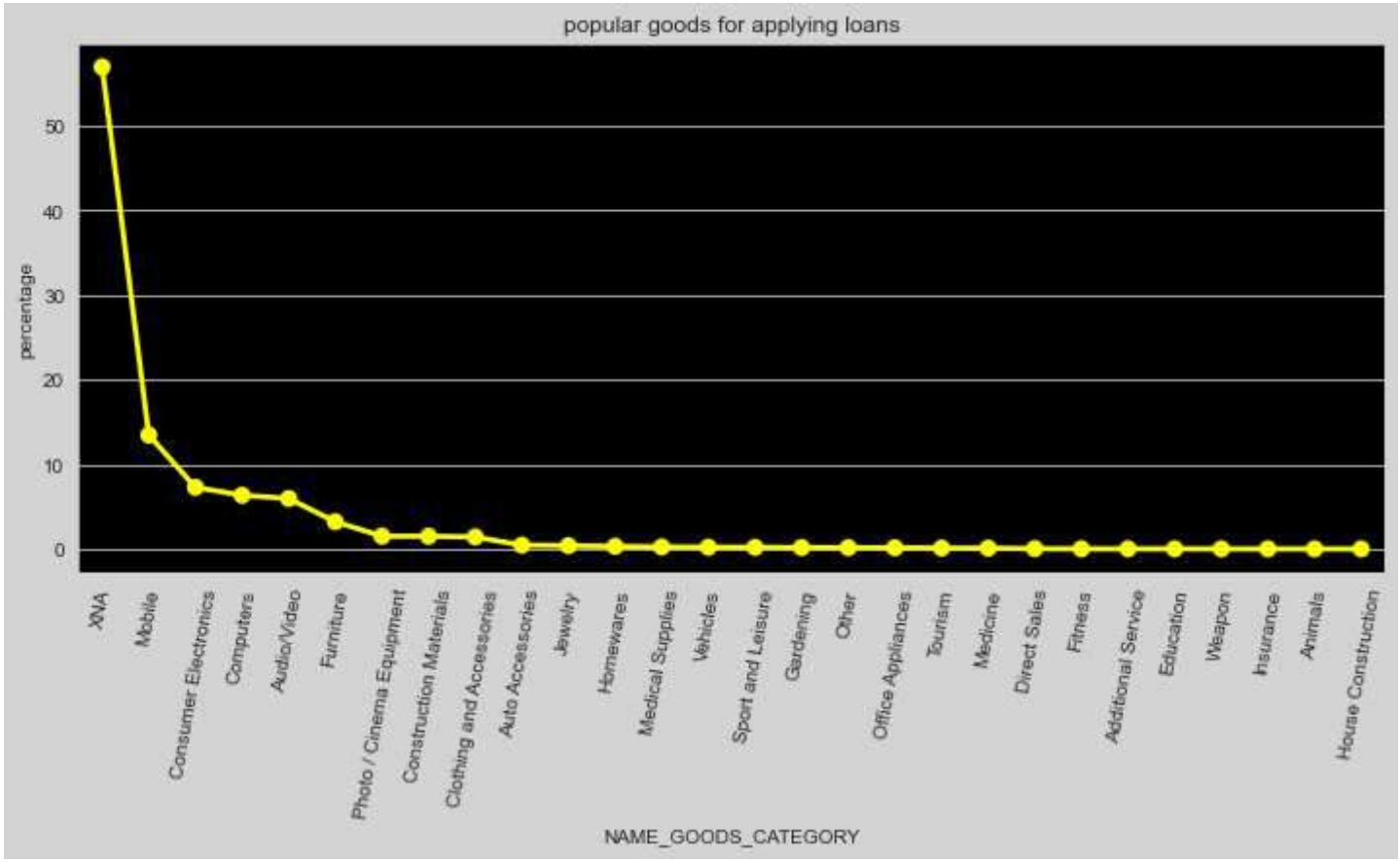

Clients payment methods
NAME_PAYMENT_TYPE
Cash through the bank
XNA
Non-cash from your account
Cashless from the account of the employer
62.0
38.0
0.0
0.0
0.0
0.2
0.4
0.6
0.8
1.0
1e6
pTop as percentage & Bottom as Count

In [148]:
```
plt.figure(figsize=(20,20))
plt.subplot(121)
previous_application["NAME_TYPE_SUITE"].value_counts().plot.pie(autopct = "%1.1f%%",fontsize=12,
                                                               colors = sns.color_palette("inferno"),
                                                    wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)
circ = plt.Circle((0,0),.7,color="white")
plt.gca().add_artist(circ)
plt.title("NAME_TYPE_SUITE")

plt.subplot(122)
previous_application["NAME_CLIENT_TYPE"].value_counts().plot.pie(autopct = "%1.1f%%",fontsize=12,
                                                               colors = sns.color_palette("inferno"),
                                                    wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)
circ = plt.Circle((0,0),.7,color="white")
plt.gca().add_artist(circ)
plt.title("NAME_CLIENT_TYPE")
plt.show()
```



In [149]:

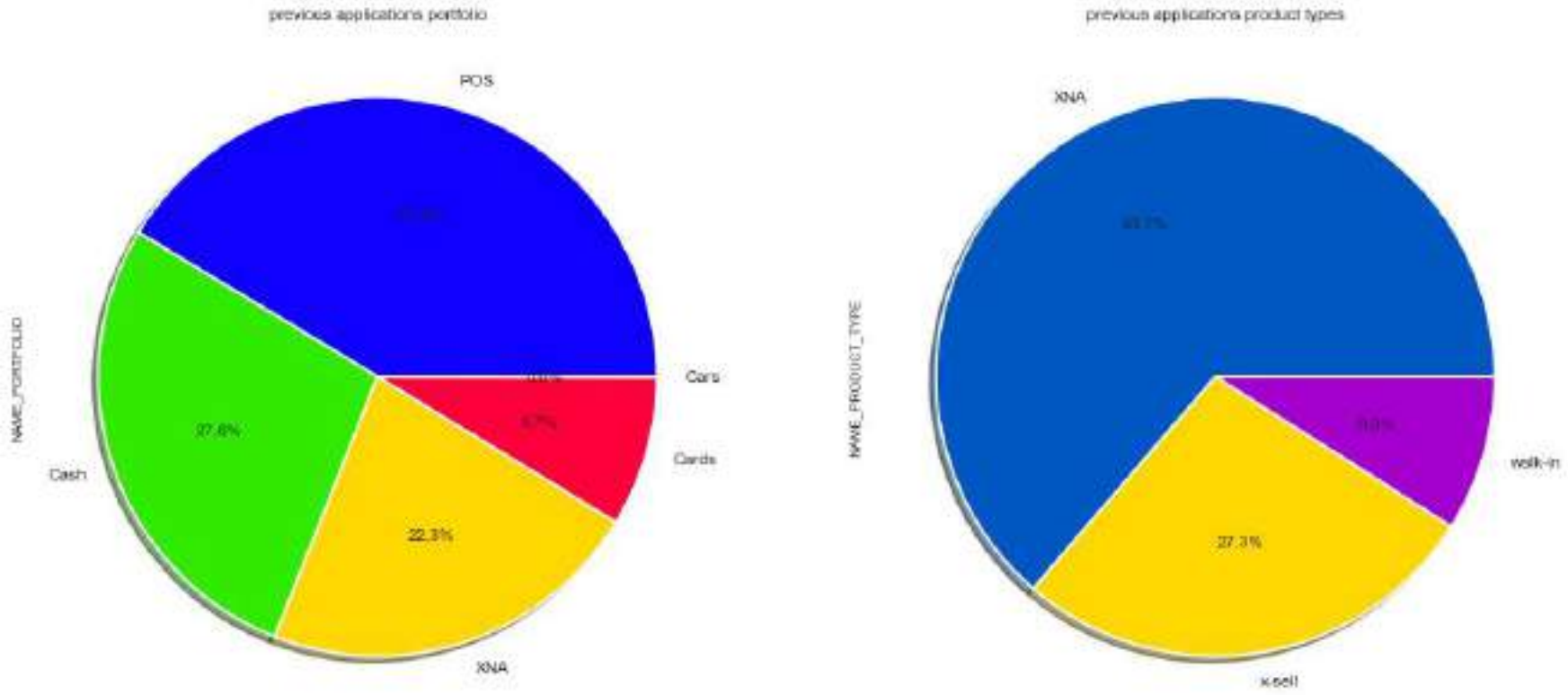
```
goods = previous_application["NAME_GOODS_CATEGORY"].value_counts().reset_index()
goods["percentage"] = round(goods["NAME_GOODS_CATEGORY"]*100/goods["NAME_GOODS_CATEGORY"].sum(),2)
fig = plt.figure(figsize=(12,5))
ax = sns.pointplot("index","percentage",data=goods,color="yellow")
plt.xticks(rotation = 80)
plt.xlabel("NAME_GOODS_CATEGORY")
plt.ylabel("percentage")
plt.title("popular goods for applying loans")
ax.set_facecolor("k")
fig.set_facecolor('lightgrey')
```



In [158...

```
plt.figure(figsize=(20,20))
plt.subplot(121)
previous_application["NAME_PORTFOLIO"].value_counts().plot.pie(autopct = "%1.1f%%",fontsize=12,
                                                               colors = sns.color_palette("prism",5),
                                                    wedgeprops={"linewidth":2,"edgecolor":"white"},
```

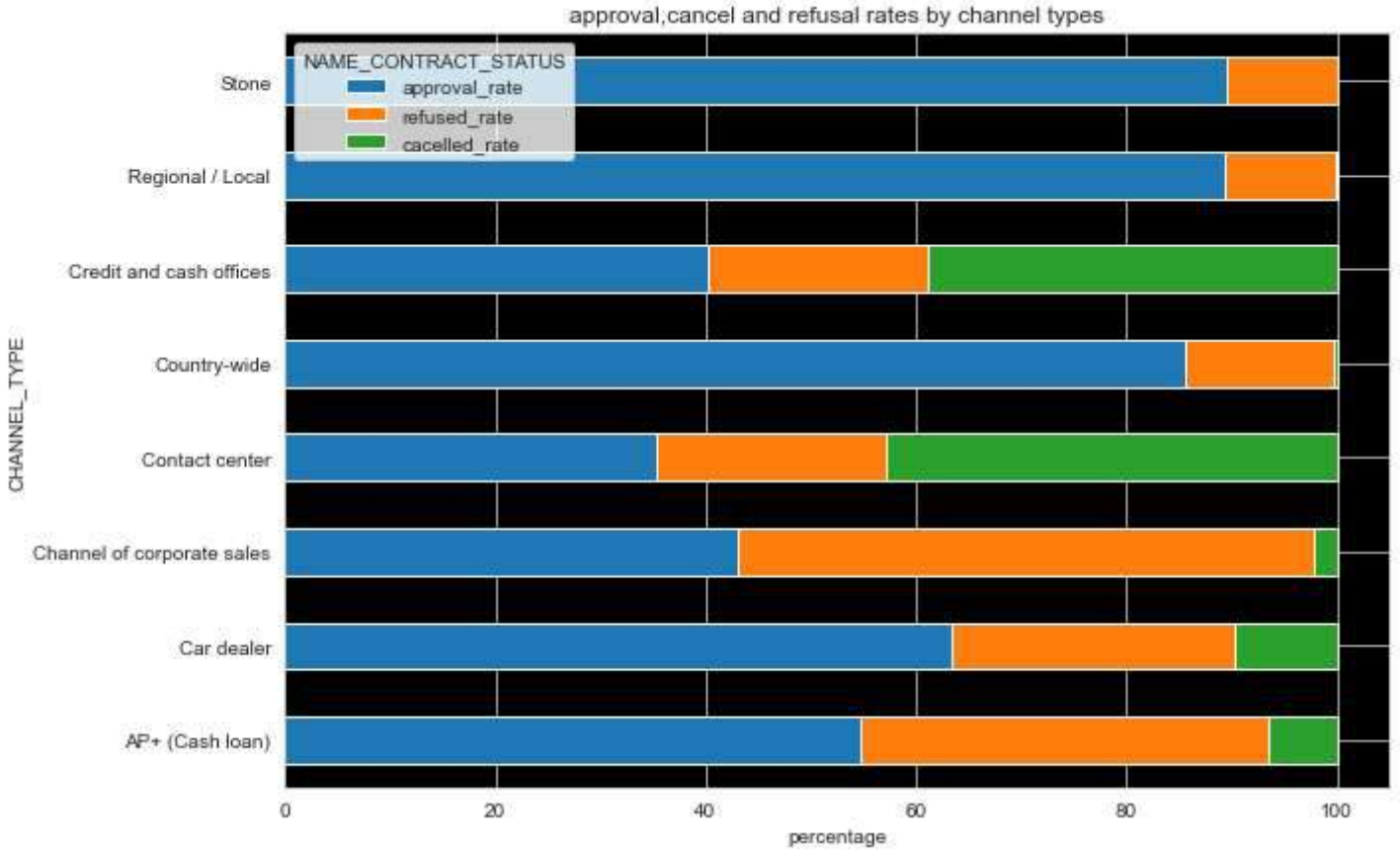
```
                                                        shadow =True)
plt.title("previous applications portfolio")
plt.subplot(122)
previous_application["NAME_PRODUCT_TYPE"].value_counts().plot.pie(autopct = "%1.1f%%",fontsize=12,
                                                                  colors = sns.color_palette("prism",3),
                                                wedgeprops={"linewidth":2,"edgecolor":"white"},
                                                                  shadow =True)
plt.title("previous applications product types")
plt.show()
```



In [151...

```
app = pd.crosstab(previous_application["CHANNEL_TYPE"],previous_application["NAME_CONTRACT_STATUS"])
app1 = app
app1["approval_rate"] = app1["Approved"]*100/(app1["Approved"]+app1["Refused"]+app1["Canceled"])
app1["refused_rate"]  = app1["Refused"]*100/(app1["Approved"]+app1["Refused"]+app1["Canceled"])
app1["cacelled_rate"] = app1["Canceled"]*100/(app1["Approved"]+app1["Refused"]+app1["Canceled"])
app2 = app[["approval_rate","refused_rate","cacelled_rate"]]
ax = app2.plot(kind="barh",stacked=True,figsize=(10,7))
ax.set_facecolor("k")
```

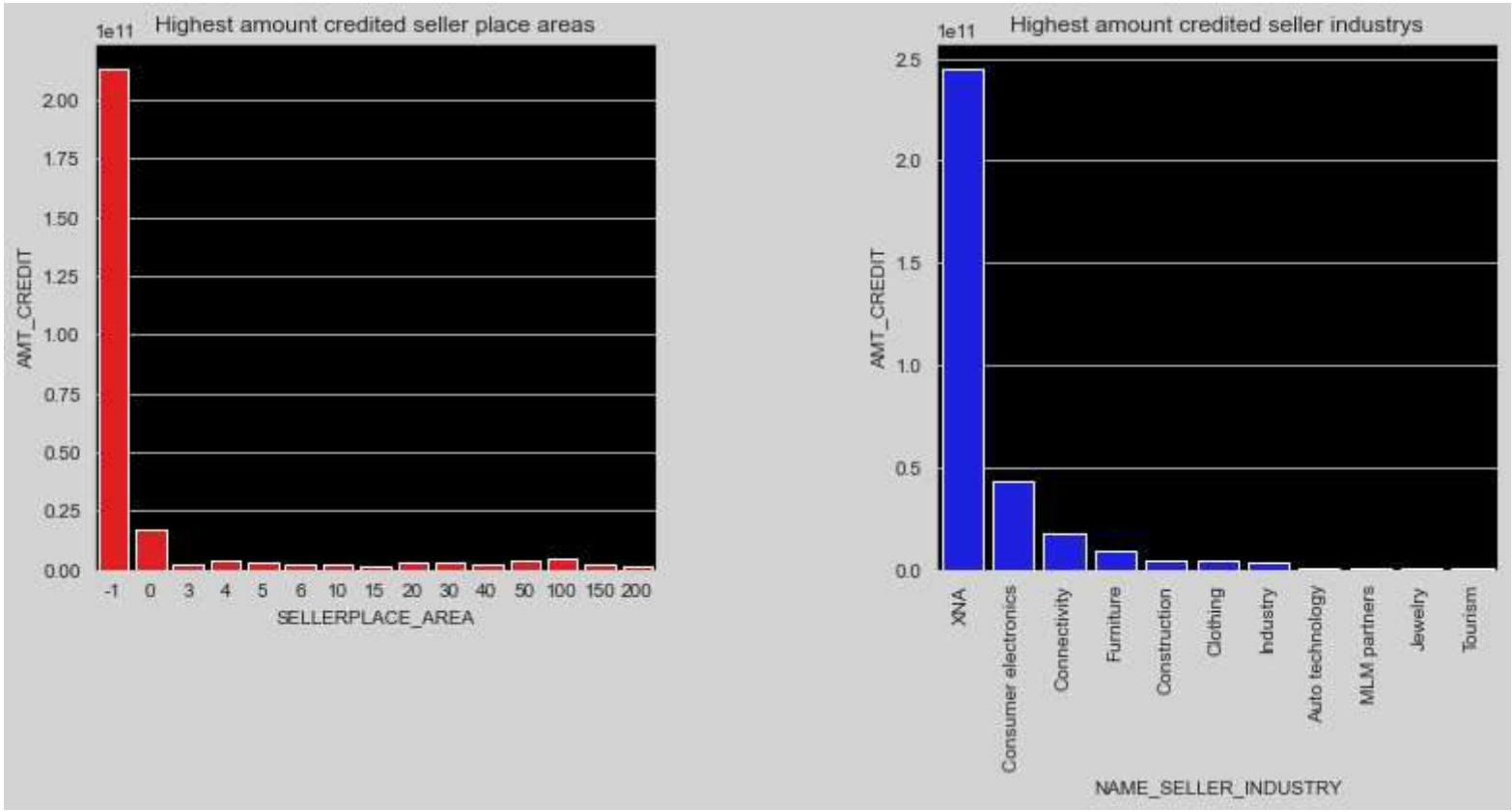
```
ax.set_xlabel("percentage")
ax.set_title("approval,cancel and refusal rates by channel types")
plt.show()
```



In [152..

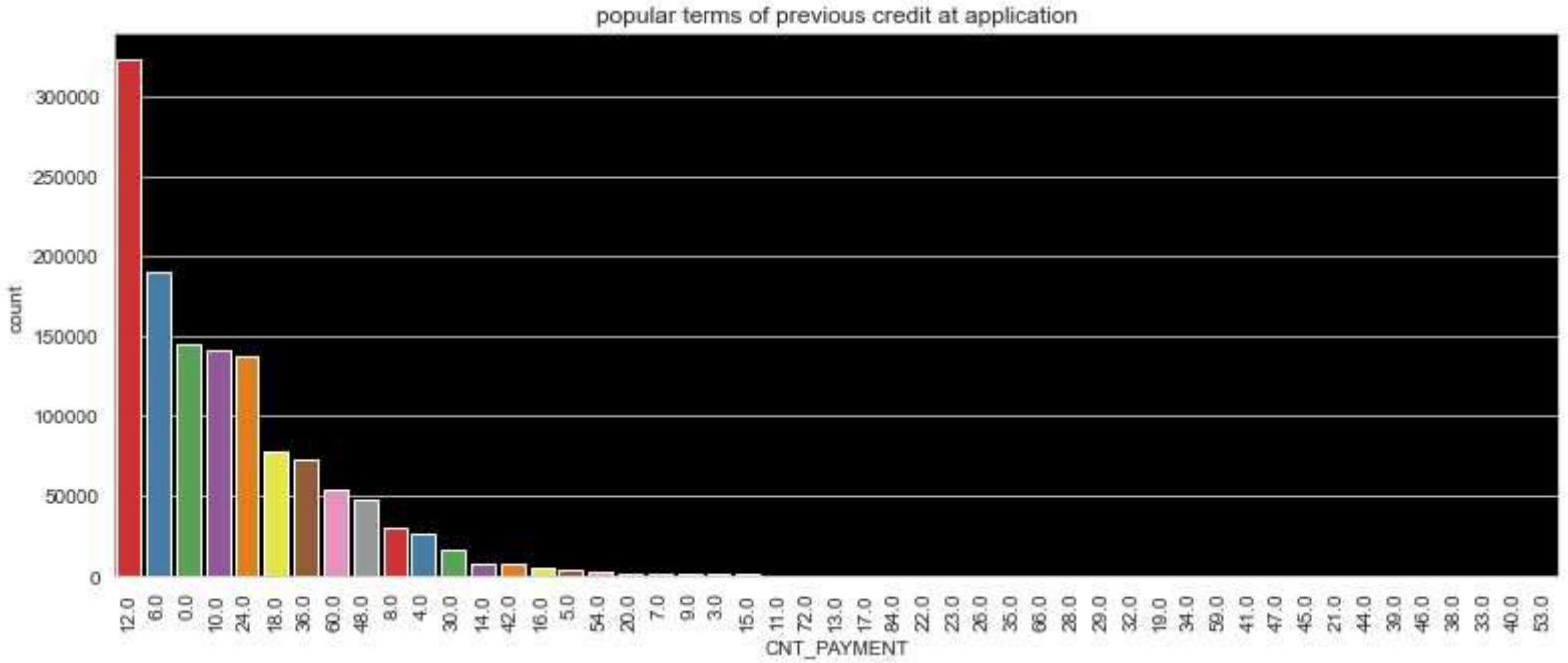
```
fig = plt.figure(figsize=(13,5))
plt.subplot(121)
are = previous_application.groupby("SELLERPLACE_AREA")["AMT_CREDIT"].sum().reset_index()
are = are.sort_values(by ="AMT_CREDIT",ascending = False)
ax = sns.barplot(y= "AMT_CREDIT",x ="SELLERPLACE_AREA",data=are[:15],color="r")
ax.set_facecolor("k")
ax.set_title("Highest amount credited seller place areas")

plt.subplot(122)
sell = previous_application.groupby("NAME_SELLER_INDUSTRY")["AMT_CREDIT"].sum().reset_index().sort_values(by = "AMT_CREDI
ax1=sns.barplot(y = "AMT_CREDIT",x = "NAME_SELLER_INDUSTRY",data=sell,color="b")
ax1.set_facecolor("k")
```

```
ax1.set_title("Highest amount credited seller industrys")
plt.xticks(rotation=90)
plt.subplots_adjust(wspace = .5)
fig.set_facecolor("lightgrey")
```



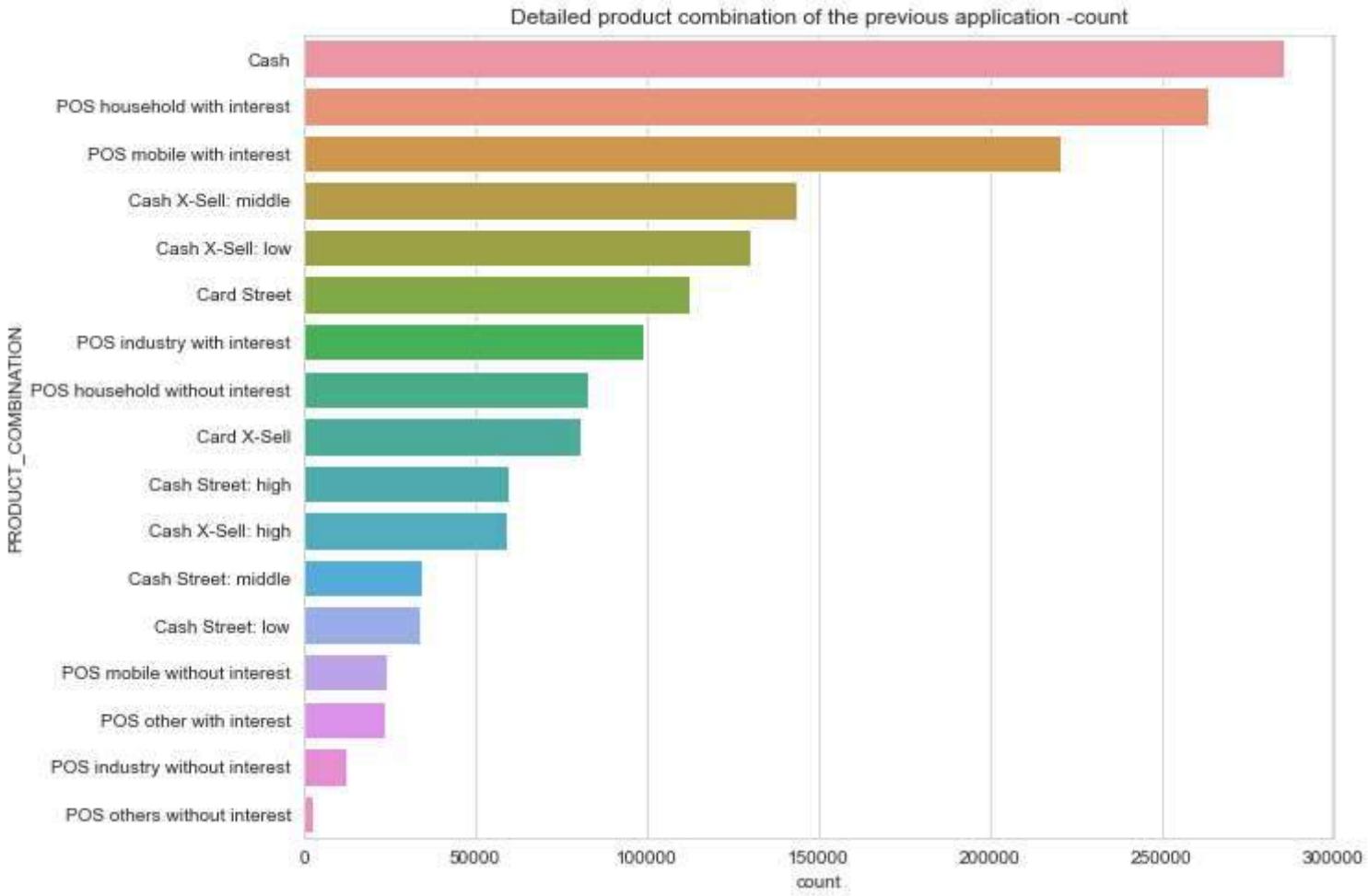
In [153...

```
plt.figure(figsize=(13,5))
ax = sns.countplot(previous_application["CNT_PAYMENT"],palette="Set1",order=previous_application["CNT_PAYMENT"].value_cou
ax.set_facecolor("k")
plt.xticks(rotation = 90)
plt.title("popular terms of previous credit at application")
plt.show()
```



In [154_

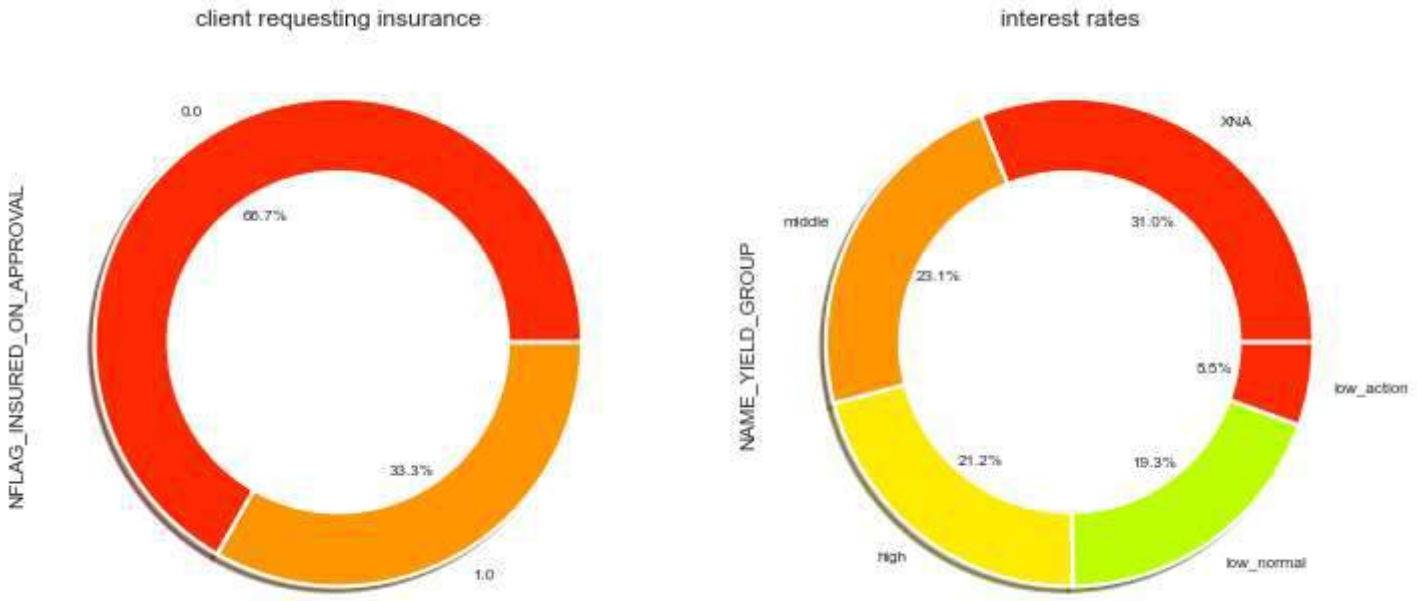
```
plt.figure(figsize=(10,8))
sns.countplot(y = previous_application["PRODUCT_COMBINATION"],order=previous_application["PRODUCT_COMBINATION"].value_cou
plt.title("Detailed product combination of the previous application -count")
plt.show()
```



In [155...

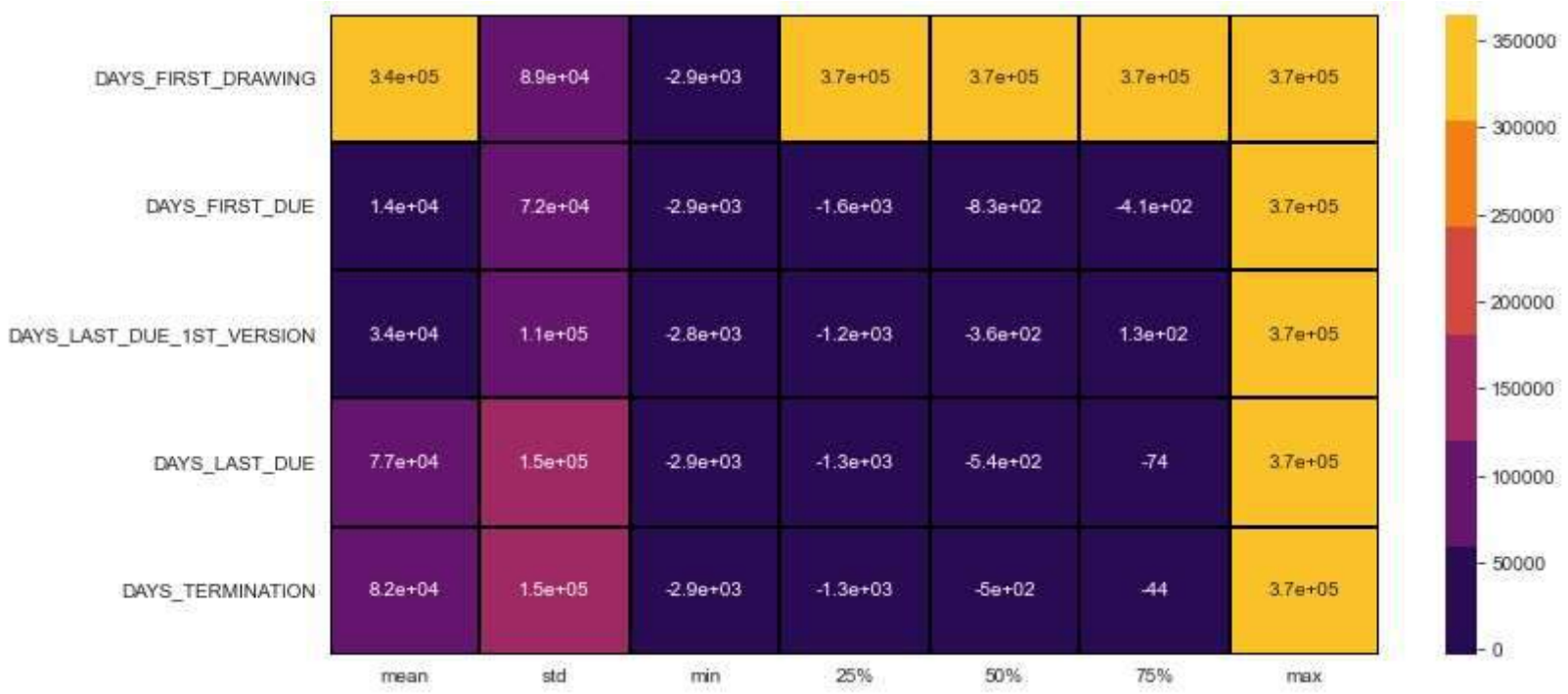
```
plt.figure(figsize=(12,6))
plt.subplot(121)
previous_application["NFLAG_INSURED_ON_APPROVAL"].value_counts().plot.pie(autopct = "%1.1f%%",fontsize=8,
                                                                          colors = sns.color_palette("prism",4),
                                                                  wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)
circ = plt.Circle((0,0),.7,color="white")
plt.gca().add_artist(circ)
plt.title("client requesting insurance")

plt.subplot(122)
previous_application["NAME_YIELD_GROUP"].value_counts().plot.pie(autopct = "%1.1f%%",fontsize=8,
                                                                          colors = sns.color_palette("prism",4),
                                                                  wedgeprops={"linewidth":2,"edgecolor":"white"},shadow =True)
```

```
circ = plt.Circle((0,0),.7,color="white")
plt.gca().add_artist(circ)
plt.title("interest rates")
plt.show()
```



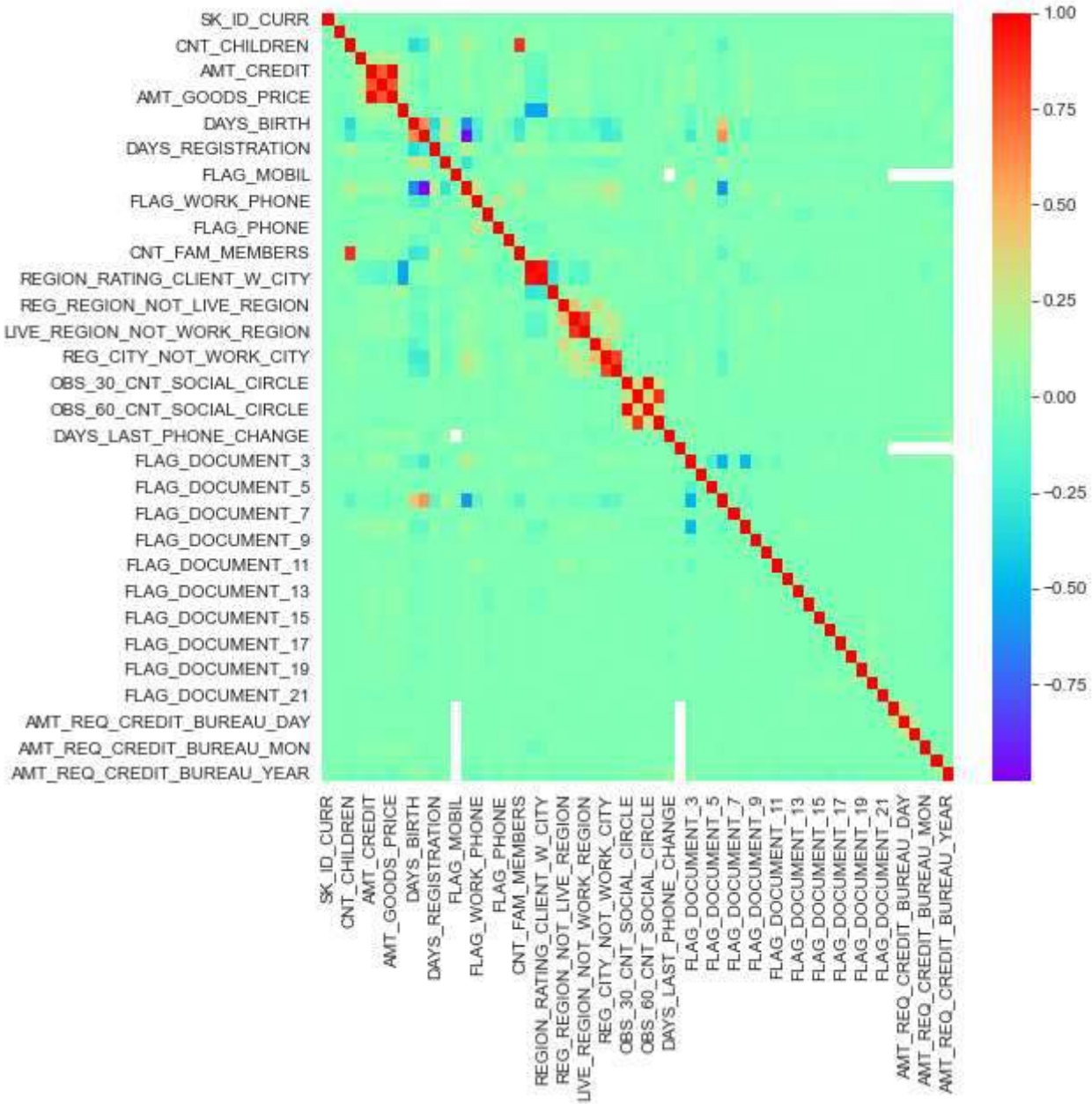
In [156...

```
cols = ['DAYS_FIRST_DRAWING', 'DAYS_FIRST_DUE', 'DAYS_LAST_DUE_1ST_VERSION','DAYS_LAST_DUE', 'DAYS_TERMINATION']
plt.figure(figsize=(12,6))
sns.heatmap(previous_application[cols].describe()[1:].transpose(),
            annot=True,linewidth=2,linecolor="k",cmap=sns.color_palette("inferno"))
plt.show()
```



In [157]:

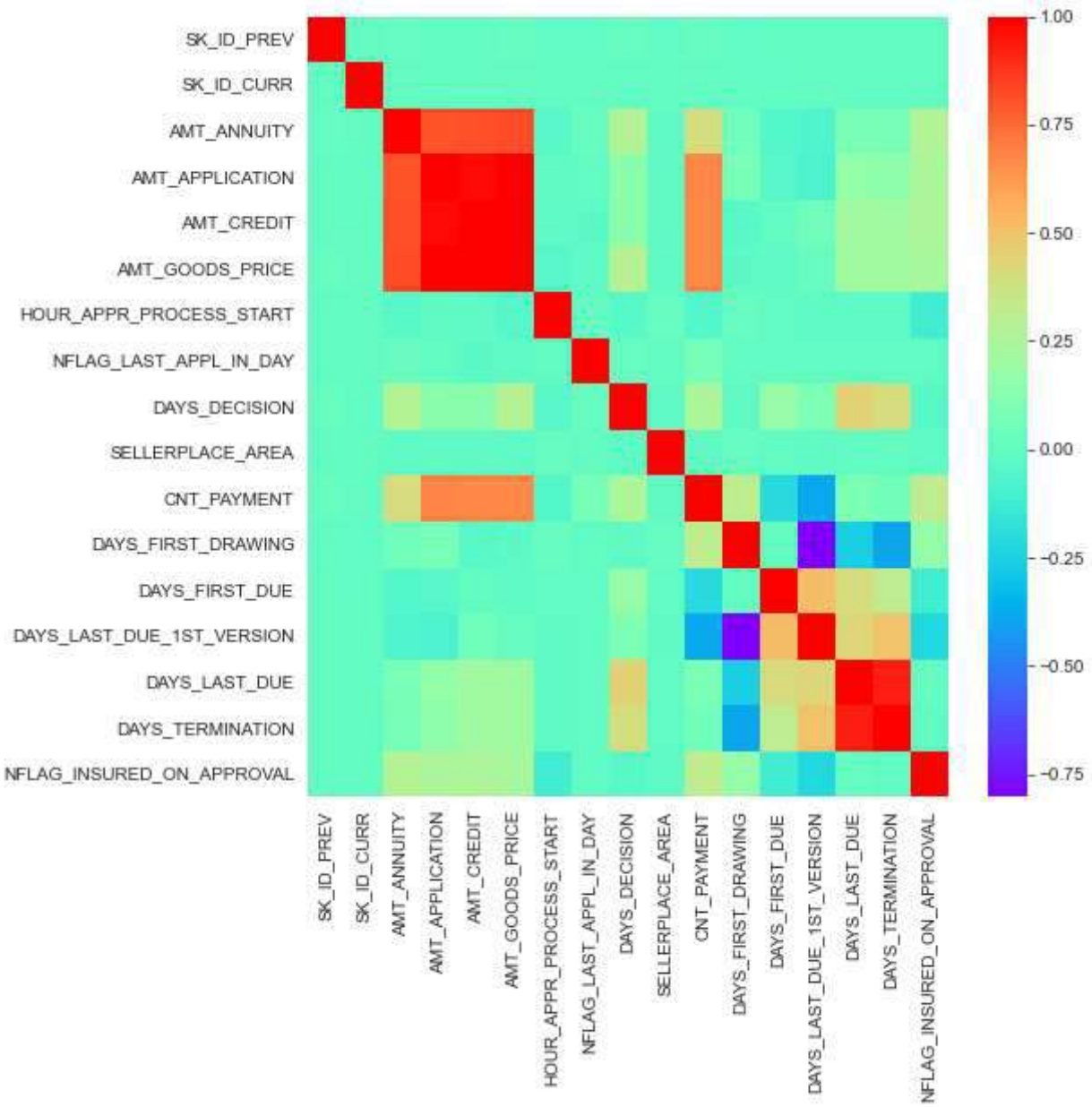
```
corrmat = application_data.corr()

f, ax = plt.subplots(figsize =(8, 8))
sns.heatmap(corrmat, ax = ax, cmap ="rainbow")
plt.show()
```



In [158...

```
corrmat = previous_application.corr()

f, ax = plt.subplots(figsize =(8, 8))
```

```
sns.heatmap(corrmat, ax = ax, cmap ="rainbow")
plt.show()
```



In [159...

```
corrmat = previous_application.corr()
corrdf = corrmat.where(np.triu(np.ones(corrmat.shape), k=1).astype(np.bool))
```

```
corrdf = corrdf.unstack().reset_index()
corrdf.columns = ['Var1', 'Var2', 'Correlation']
corrdf.dropna(subset = ['Correlation'], inplace = True)
corrdf['Correlation'] = round(corrdf['Correlation'], 2)
corrdf['Correlation'] = abs(corrdf['Correlation'])
corrdf.sort_values(by = 'Correlation', ascending = False).head(10)
```

Out[159...

	Var1	Var2	Correlation
88	AMT_GOODS_PRICE	AMT_APPLICATION	1.00
89	AMT_GOODS_PRICE	AMT_CREDIT	0.99
71	AMT_CREDIT	AMT_APPLICATION	0.98
269	DAYS_TERMINATION	DAYS_LAST_DUE	0.93
87	AMT_GOODS_PRICE	AMT_ANNUITY	0.82
70	AMT_CREDIT	AMT_ANNUITY	0.82
53	AMT_APPLICATION	AMT_ANNUITY	0.81
232	DAYS_LAST_DUE_1ST_VERSION	DAYS_FIRST_DRAWING	0.80
173	CNT_PAYMENT	AMT_APPLICATION	0.68
174	CNT_PAYMENT	AMT_CREDIT	0.67

In [160...

```
df_repayer = application_data[application_data['TARGET'] == 0]
df_defaulter = application_data[application_data['TARGET'] == 1]
```

In [161...

```
corrmat = df_repayer.corr()
corrdf = corrmat.where(np.triu(np.ones(corrmat.shape), k=1).astype(np.bool))
corrdf = corrdf.unstack().reset_index()
corrdf.columns = ['Var1', 'Var2', 'Correlation']
corrdf.dropna(subset = ['Correlation'], inplace = True)
corrdf['Correlation'] = round(corrdf['Correlation'], 2)
corrdf['Correlation'] = abs(corrdf['Correlation'])
corrdf.sort_values(by = 'Correlation', ascending = False).head(10)
```

Out[161...

	Var1	Var2	Correlation
776	FLAG_EMP_PHONE	DAYS_EMPLOYED	1.00

	Var1	Var2	Correlation
1798	OBS_60_CNT_SOCIAL_CIRCLE	OBS_30_CNT_SOCIAL_CIRCLE	1.00
358	AMT_GOODS_PRICE	AMT_CREDIT	0.99
1199	REGION_RATING_CLIENT_W_CITY	REGION_RATING_CLIENT	0.95
1064	CNT_FAM_MEMBERS	CNT_CHILDREN	0.88
1858	DEF_60_CNT_SOCIAL_CIRCLE	DEF_30_CNT_SOCIAL_CIRCLE	0.86
1439	LIVE_REGION_NOT_WORK_REGION	REG_REGION_NOT_WORK_REGION	0.86
1619	LIVE_CITY_NOT_WORK_CITY	REG_CITY_NOT_WORK_CITY	0.83
359	AMT_GOODS_PRICE	AMT_ANNUITY	0.78
299	AMT_ANNUITY	AMT_CREDIT	0.77

In [162...

```
corrmat = df_defaulter.corr()
corrdf = corrmat.where(np.triu(np.ones(corrmat.shape), k=1).astype(np.bool))
corrdf = corrdf.unstack().reset_index()
corrdf.columns = ['Var1', 'Var2', 'Correlation']
corrdf.dropna(subset = ['Correlation'], inplace = True)
corrdf['Correlation'] = round(corrdf['Correlation'], 2)
corrdf['Correlation'] = abs(corrdf['Correlation'])
corrdf.sort_values(by = 'Correlation', ascending = False).head(10)
```

Out[162...

	Var1	Var2	Correlation
1798	OBS_60_CNT_SOCIAL_CIRCLE	OBS_30_CNT_SOCIAL_CIRCLE	1.00
776	FLAG_EMP_PHONE	DAYS_EMPLOYED	1.00
358	AMT_GOODS_PRICE	AMT_CREDIT	0.98
1199	REGION_RATING_CLIENT_W_CITY	REGION_RATING_CLIENT	0.96
1064	CNT_FAM_MEMBERS	CNT_CHILDREN	0.89
1858	DEF_60_CNT_SOCIAL_CIRCLE	DEF_30_CNT_SOCIAL_CIRCLE	0.87
1439	LIVE_REGION_NOT_WORK_REGION	REG_REGION_NOT_WORK_REGION	0.85
1619	LIVE_CITY_NOT_WORK_CITY	REG_CITY_NOT_WORK_CITY	0.78

	Var1	Var2	Correlation
299	AMT_ANNUITY	AMT_CREDIT	0.75
359	AMT_GOODS_PRICE	AMT_ANNUITY	0.75

In [163..

```
mergeddf =  pd.merge(application_data,previous_application,on='SK_ID_CURR')
mergeddf.head()
```

Out[163..

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE_x	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL
0	100002	1	Cash loans	M	N	Y	0	202500.0
1	100003	0	Cash loans	F	N	N	0	270000.0
2	100003	0	Cash loans	F	N	N	0	270000.0
3	100003	0	Cash loans	F	N	N	0	270000.0
4	100004	0	Revolving loans	M	Y	Y	0	67500.0

In [164..

```
y = mergeddf.groupby('SK_ID_CURR').size()
dfA = mergeddf.groupby('SK_ID_CURR').agg({'TARGET': np.sum})
dfA['count'] = y
display(dfA.head(10))
```

SK_ID_CURR	TARGET	count
100002	1	1
100003	0	3
100004	0	1
100006	0	9

SK_ID_CURR	TARGET	count
100007	0	6
100008	0	5
100009	0	7
100010	0	1
100011	0	4
100012	0	4

In [165...

```
dfA.sort_values(by = 'count',ascending=False).head(10)
```

Out[165...

SK_ID_CURR	TARGET	count
265681	0	73
173680	0	72
242412	0	68
206783	0	67
389950	0	64
382179	0	64
198355	0	63
345161	0	62
446486	0	62
238250	0	61

In [166...

```
df_repayer = dfA[dfA['TARGET'] == 0]
df_defaulter = dfA[dfA['TARGET'] == 1]
```

In [167]:
```
df_repayer.sort_values(by = 'count',ascending=False).head(10)
```

Out[167]:

SK_ID_CURR	TARGET	count
265681	0	73
173680	0	72
242412	0	68
206783	0	67
382179	0	64
389950	0	64
198355	0	63
446486	0	62
345161	0	62
280586	0	61

In [168]:
```
df_defaulter.sort_values(by = 'count',ascending=False).head(10)
```

Out[168]:

SK_ID_CURR	TARGET	count
100002	1	1
333349	1	1
333587	1	1
333582	1	1
333534	1	1
333506	1	1
333419	1	1
333355	1	1

SK_ID_CURR	TARGET	count
333337	1	1
334761	1	1

In [ ]: