Credit Analysis

June 15, 2023

Done by Elie Diwambuena

Language: Python

The AB bank would like to examine the credit profile of its customers pools who have applied for a loan. They ask their credit risk analysis (you/me) to analyze the default risk involve with each customer and come up with a credit score based on which they can decide whether to give a loan or decline it. The IT department has provided the required dataset and the management expects the results in two weeks.

Part I. Exploratory Data Analysis

As often necessary and recommend, we start with univarite and bivariate analysis of variables observed for this analysis. Hence, in this first part we will explore the characteristics of and relationships between different variables. There many tools that can be used such as Excel, R and other online platform like Flourish to analyze and visualize data. However, in this analysis, we will use the Python programming language. The benefit with Python is that it is open source, meaning that we do not need a license or a pro suscription to use it and that it is often involving with many people contributing to its existing libraries.

For this first part, we need Pandas, matplotlib and seaborn libraries and use a friendly programming interface, the Jupyter Notebook, where we can code right inside the document and make it ready for sharing. Thus, we proceed with loading up the library. We postpone the construction of the model until the second part of this analysis.

1. Pre-analysis

- Importing the libraries

```
[1]: import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as sns
```

- **Loading the data**

```
[2]: dataset = pd.read_excel("loan_data.xlsx")
```

```
[3]: dataset.head()
```

```
[3]:   CustomerNo Gender  Age CivilState Dependents     Education Self-employed  \
     0      AB001   Male   63      Widow          0      Graduate            No
     1      AB002   Male   63    Married          1      Graduate            No
     2      AB003   Male   63    Married          0      Graduate           Yes
     3      AB004   Male   63    Married          0  Not Graduate            No
     4      AB005   Male   63  Unmarried          0      Graduate            No

        IncomePerMonth  CoAppIncome  LoanAmount  LoanTerm DefaultHistory  \
     0            5849          0.0       12800       3.0            Yes
     1           81000       1508.0       12800       3.0            Yes
     2           63337          0.0        6600       3.0            Yes
     3           51763       2358.0       12000       3.0            Yes
     4           39999          0.0       14100       3.0            Yes

        Property_Value OtherDebts
     0           90662        Yes
     1               0         No
     2               0         No
     3           84831        Yes
     4           89686         No
```

- **Checking for inconsistency in data type of each vairable**

```
[4]: # Info table
     dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1595 entries, 0 to 1594
Data columns (total 14 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   CustomerNo      1595 non-null   object
 1   Gender          1584 non-null   object
 2   Age             1595 non-null   int64
 3   CivilState      1595 non-null   object
 4   Dependents      1585 non-null   object
 5   Education       1595 non-null   object
 6   Self-employed   1575 non-null   object
 7   IncomePerMonth  1595 non-null   int64
 8   CoAppIncome     1595 non-null   float64
 9   LoanAmount      1595 non-null   int64
 10  LoanTerm        1595 non-null   float64
 11  DefaultHistory  1595 non-null   object
 12  Property_Value  1595 non-null   int64
 13  OtherDebts      1595 non-null   object
dtypes: float64(2), int64(4), object(8)
memory usage: 174.6+ KB
```

It is necessary to check for the data type at this stage of the analysis to avoid getting errors afterwards. There are some operations that are specific to certain data types but that cannot be perform on other data types. And there are no operation - not that I am awared of - that can be performed between two variables of different data types (i.e., 100 + "100" , this will simply raise an error because the first 100 is numerical and the second is a string or character).

The dataset contains 13 variables which include among others the customer number (CustomerNo), gender, age, civil status, how many people are dependent on a customer (Dependents), education, income and others. We expect variables such as age, income per month, loan amount, loan term and property values to be numerical. That is also what we see in Dtype columns (float64 and int64 are numerical data types). The dependents and other debts variables are of object type, meaning a combination of either numerical & character or simply characters. This is not what we expected but a quick check of the values in those columns reveals that they are in fact categorical variables. Hence, there seems to be no problem related to the data types.

- **Missing values**

[5]:
```
# Are there any missing values?
dataset.isna().sum()
```

[5]:
```
CustomerNo         0
Gender            11
Age                0
CivilState         0
Dependents        10
Education          0
Self-employed     20
IncomePerMonth     0
CoAppIncome        0
LoanAmount         0
LoanTerm           0
DefaultHistory     0
Property_Value     0
OtherDebts         0
dtype: int64
```

[6]:
```
# Correct for missing observations
dataset.dropna(inplace=True)
```

[7]:
```
dataset.isna().sum()
```

[7]:
```
CustomerNo        0
Gender            0
Age               0
CivilState        0
Dependents        0
Education         0
Self-employed     0
```

```
IncomePerMonth    0
CoAppIncome       0
LoanAmount        0
LoanTerm          0
DefaultHistory    0
Property_Value    0
OtherDebts        0
dtype: int64
```

The second setp is to checking for missing values within the dataset. As shown in the previous info table, most variables have 1595 observations and the Gende, Dependents and Self-employed variables have missing values. The total missing values are 41 or 2.5% of the dataset. Since this is not very significant, we decide to simply drop them and proceed with the analysis.

2. Descriptive statistics

[8]:
```python
# Descriptive for numerical variables
colnames =dataset.columns
numerical_var = dataset.loc[:,['Age', 'LoanAmount','Property_Value',␣
 ↪'IncomePerMonth', 'CoAppIncome','LoanTerm']]

print("Table 1. Descriptive statistics","\n")
numerical_var.describe().round()
```

```
Table 1. Descriptive statistics
```

[8]:
```
          Age  LoanAmount  Property_Value  IncomePerMonth  CoAppIncome  \
count  1555.0      1555.0          1555.0          1555.0       1555.0
mean     42.0     13479.0         34425.0          5143.0       1561.0
std      11.0      7755.0         35172.0          5705.0       2423.0
min      19.0         0.0             0.0             0.0          0.0
25%      33.0      8900.0             0.0          2876.0          0.0
50%      42.0     12700.0         23476.0          3812.0       1213.0
75%      51.0     17300.0         68299.0          5417.0       2264.0
max      63.0     70000.0         99942.0         81000.0      41667.0

       LoanTerm
count    1555.0
mean        3.0
std         1.0
min         0.0
25%         3.0
50%         3.0
75%         3.0
```

```
max          4.0
```

The average applicant is **42** years old, asked for about **13.5** thousand, earn about **5143** as income and his/her co-applicant about **1561** and asked for a **3 month** loan. Based on the average and median values, we can assert that the distribution of loan amount and co-applicant income seems to be normal but the distribution for the remaining variables seem to be skewed. We can verify this later on with some histograms.

2.1 Gender and civil status factors

```
[9]: ### How much do applicants differ by gender and civil status?
     gr = dataset.groupby(by="Gender").mean().round()
     gr
```

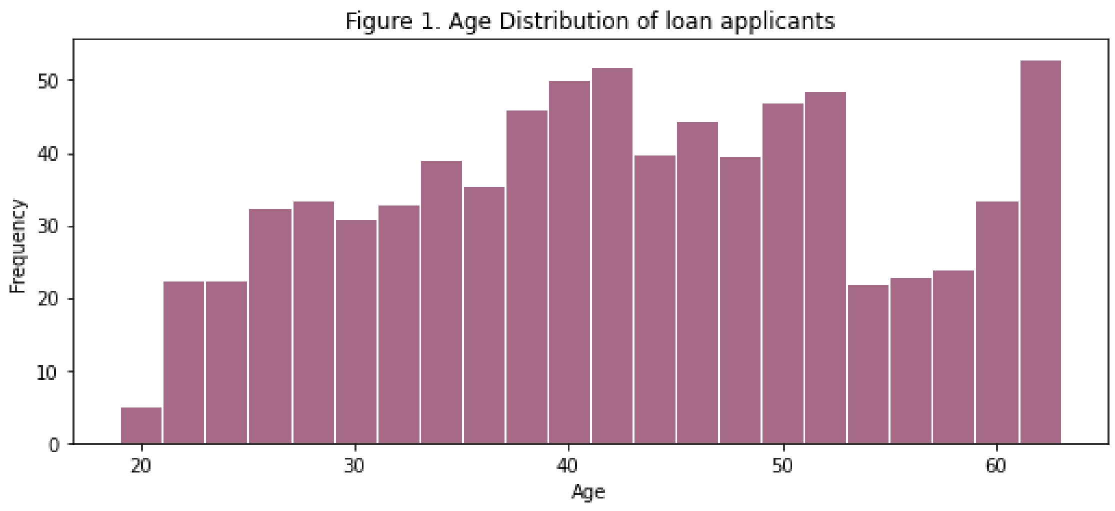
```
[9]:          Age  IncomePerMonth  CoAppIncome  LoanAmount  LoanTerm  \
     Gender
     Female  42.0          4879.0       1245.0     12553.0       3.0
     Male    42.0          5201.0       1630.0     13680.0       3.0

             Property_Value
     Gender
     Female         35113.0
     Male           34275.0
```

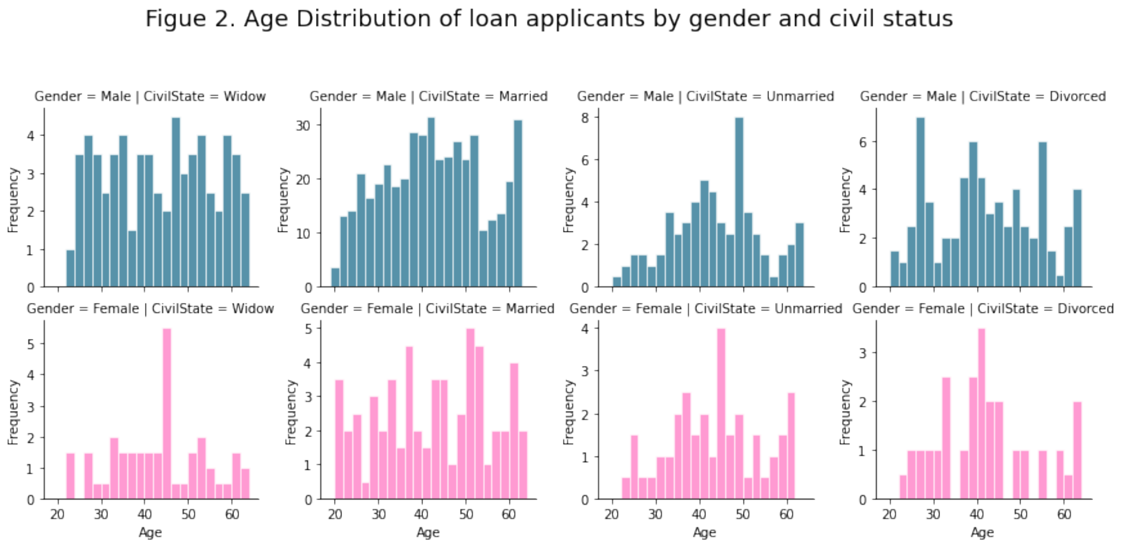
Let start with some basics. We look at the descriptive statistics of numerical variables as shown above. We find that **Male** customers are **1 year younger** than female, **earn more** than than female, **ask for higher loan** but with the same term compared to female and have **less valuable collateral** than female **on average**.

Next, we examine the distribtuion of age alone and then break it into gender and civil status subplots. This helps in identifying trends that are particular to certain attributes. The distribution of age alone as shown as shon in figue 1 does not tell us much except the fact that the average age is about 40 and that there is a high concentration of individuals aged somewhere between 30 and 50 and the distribution seems to raise again around the age of 60.

```
[10]: plt.figure(figsize=(10,4))
      sns.histplot(data=dataset, x ="Age",
                   stat="frequency",
                   binwidth=2,
                   edgecolor = "white",
                   color = "#89375F",
                  )
      plt.title("Figure 1. Age Distribution of loan applicants")
      plt.show()
```



```
[11]: facet = sns.FacetGrid(dataset,row="Gender", col="CivilState", hue␣
       ↪="Gender",palette = ["#1F6E8C","#FF78C4"], sharey=False)
      facet.map_dataframe(sns.histplot,x ="Age",
                    stat="frequency",
                    binwidth=2,
                   edgecolor = "white")
      facet.fig.subplots_adjust(top=0.8)
      facet.fig.suptitle("Figue 2. Age Distribution of loan applicants by gender and␣
       ↪civil status", fontsize=18)
      plt.show()
```



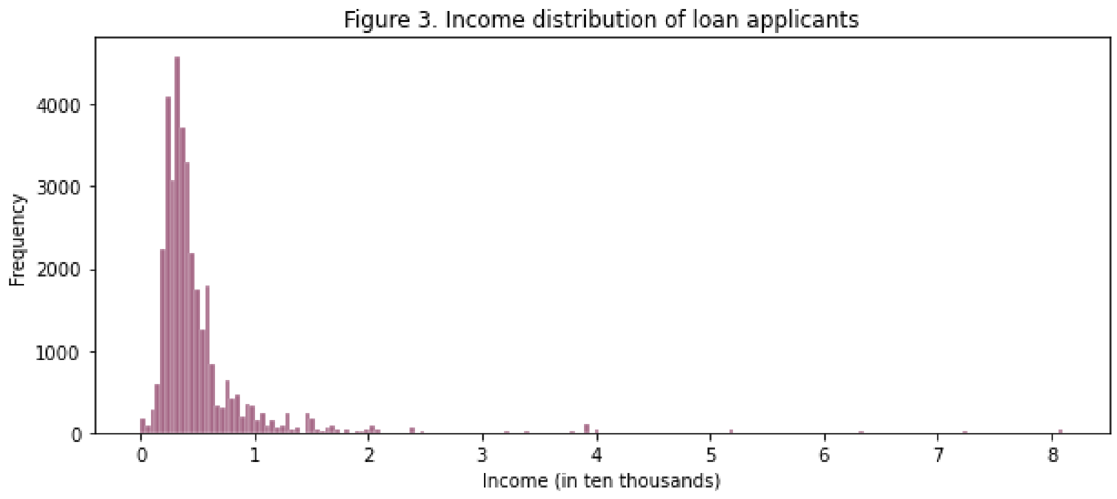
From figure 2, it is obvious that by breaking it across gender and civil status, we can see that the applicants pool is concentrated with married people of male gender. Also, the male gender seems to be the predominant category in generral across all civil status. This is not surpising since men are assumingly more likely to engage in entrepreneurial activity than their female counterpart.

The take-away from this is that we now know that a typical loan applicant is very likely to be a man who is married and aged around 40 years old. This information can be very useful for targetting marketing or sale campaign and in many other ways.

2.2 Income, gender and civil status

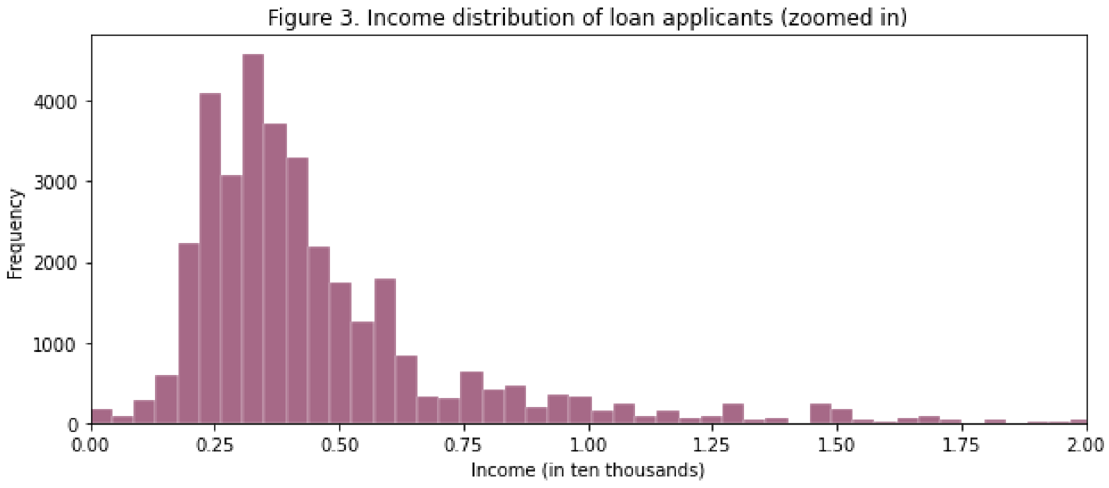
[12]:
```
# Modify the scale of salary
dt = dataset
dt.IncomePerMonth = dt.IncomePerMonth/10000
```

[13]:
```
plt.figure(figsize=(10,4))
sns.histplot(data=dt, x ="IncomePerMonth",
             stat="frequency",
             edgecolor = "white",
             color = "#89375F",
            )
plt.title("Figure 3. Income distribution of loan applicants")
plt.xlabel("Income (in ten thousands)")
plt.show()
```



[14]:
```
plt.figure(figsize=(10,4))
sns.histplot(data=dt, x ="IncomePerMonth",
             stat="frequency",
             edgecolor = "white",
             color = "#89375F",
            )
plt.title("Figure 3. Income distribution of loan applicants (zoomed in)")
```

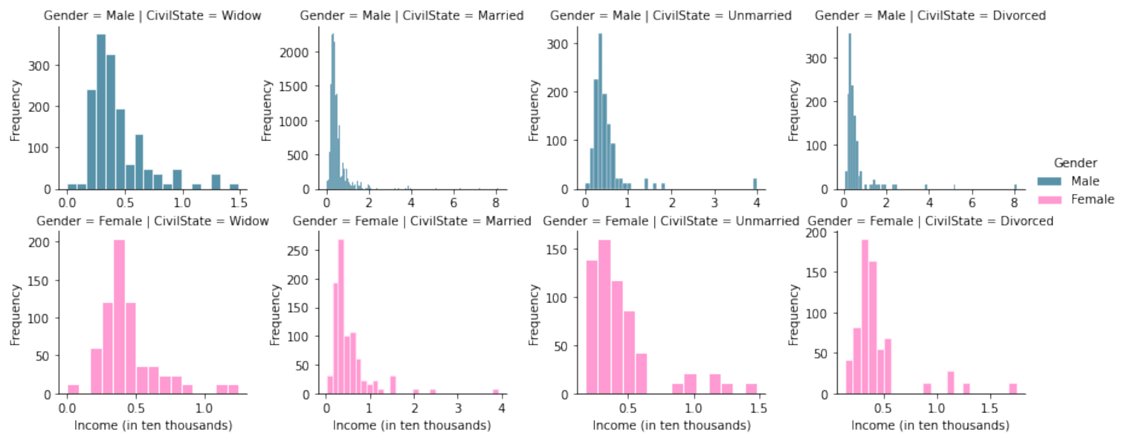
```python
plt.xlabel("Income (in ten thousands)")
plt.xlim(0,2)
plt.show()
```



As shown in figure 3, the distribution of income seems to be skewed. However, when we zoom in we can see that the distribution is still a bit skewed but not as much as we thought. There are of course serval outliers that strech it more. Figure 4 shows how income is distributed across gender and civil status. We see that most outliers are presented in the married and divorced male category. When we zoom in the distribution seems to be symmetrical within civil status. For instance, the distribution of widow, married and unmarried male is similar to that of widow, married and unmarried female, although with different frequencies.

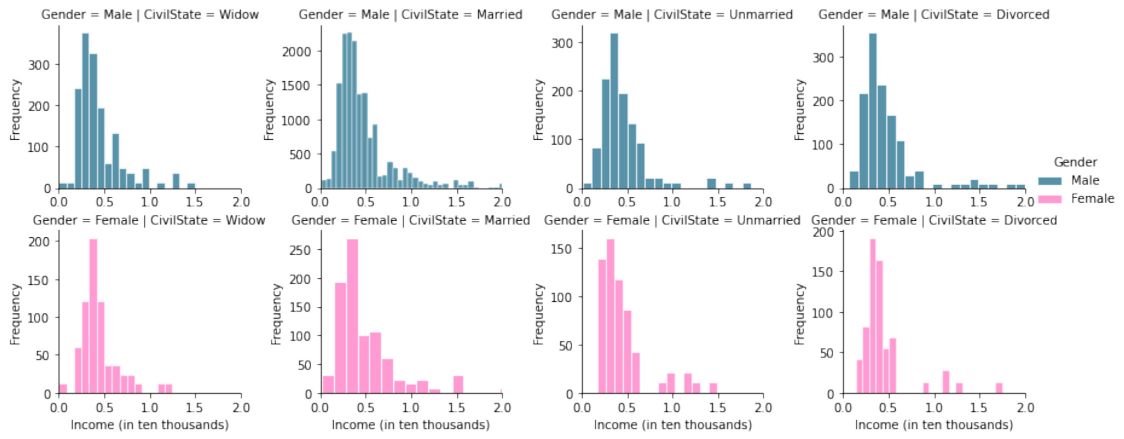
[15]:
```python
facet = sns.FacetGrid(dt,row="Gender", col="CivilState", hue ="Gender",palette␣
↪= ["#1F6E8C","#FF78C4"], sharey=False, sharex=False)
facet.map_dataframe(sns.histplot,x ="IncomePerMonth",
            stat="frequency",
            edgecolor = "white")
facet.set_xlabels("Income (in ten thousands)"),
facet.fig.subplots_adjust(top=0.8)
facet.fig.suptitle("Figure 4. Income Distribution", fontsize=14)
facet.add_legend()
plt.show()
```

Figure 4. Income Distribution



```
[16]: facet = sns.FacetGrid(dt,row="Gender", col="CivilState", hue ="Gender",palette␣
        ↪= ["#1F6E8C","#FF78C4"], sharey=False, sharex=False, xlim=(0,2))
      facet.map_dataframe(sns.histplot,x ="IncomePerMonth",
                  stat="frequency",
                  edgecolor = "white")
      facet.set_xlabels("Income (in ten thousands)"),
      facet.fig.subplots_adjust(top=0.8)
      facet.fig.suptitle("Figure 4. Income Distribution (zoomed in)", fontsize=14)
      facet.add_legend()
      plt.show()
```

Figure 4. Income Distribution (zoomed in)



What we can take from this is that most applicants seem earn somewhere around 50 000 irrestpective of gender and civil status. However, married male & female applicants are more likely to earn more.

The outliers are very common in real life. However, we should take this with a grain of salt since this could simply be a matter of representativity of other categories. The sample size of female applicants and other civil state categories might be underrepresented.

2.3 Loan amount, gender and civil status

[17]:
```
# What about Loan Amount distribution by gender?
plt.figure(figsize=(10,4))
plt.title("Figure 5. Distribution of loan application amount by gender and␣
 ↪civil status")
sns.boxplot(data = dataset,
            y="CivilState",
            x="LoanAmount",
            hue ="Gender",
            palette = ["#1F6E8C","#FF78C4"],
            ) #sns.color_palette("Paired"))
plt.xlabel("Loan amount")
plt.show()
```

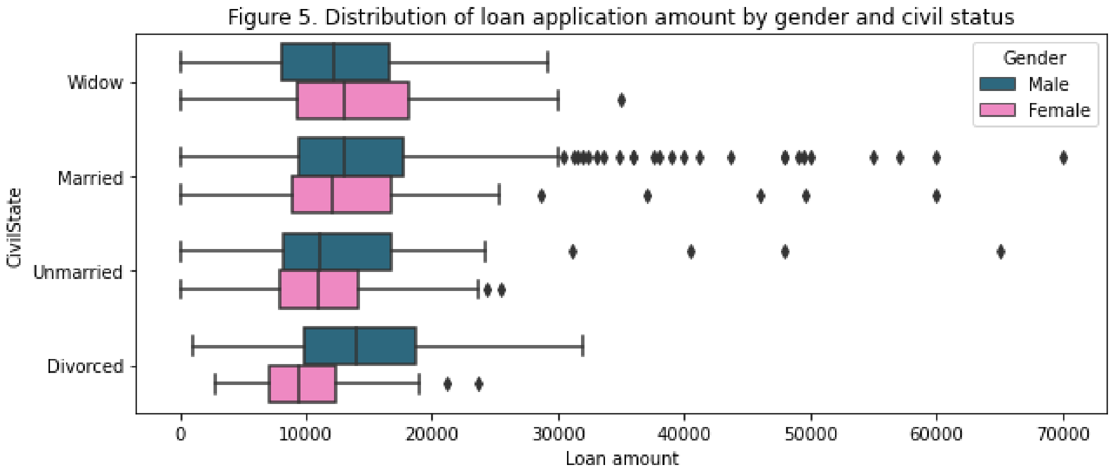


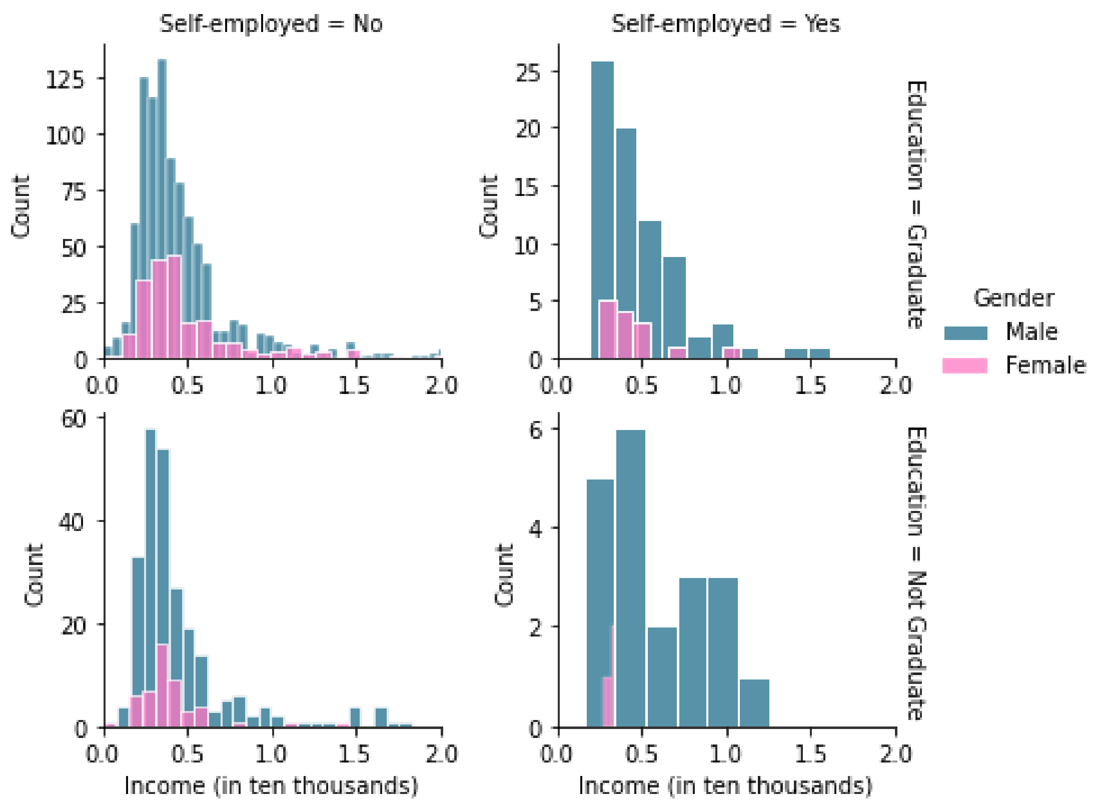
Figure 5. shows that the distribution of loan amount across civil status and by gender is somehow normally distributed. Similarly, the married category have more outliers than others. What is also interesting is that female widow ask for on averga far higher loan amount than their counterparts and divorced male also ask for higher loan amount on average compared to other male categories. But in general, there is no a lot of difference.

2.4 Income, loan amount, education and job type

[18]:
```
facet = sns.FacetGrid(dataset,row="Education", col="Self-employed", hue␣
 ↪="Gender",palette = ["#1F6E8C","#FF78C4"], sharey=False,␣
 ↪sharex=False,margin_titles=True, legend_out=True,xlim=(0,2))
facet.map_dataframe(sns.histplot,x ="IncomePerMonth",
                    edgecolor = "white")
```

```
facet.set_xlabels("Income (in ten thousands)")
facet.fig.subplots_adjust(top=0.8)
facet.fig.suptitle("Figure 6. Income distribution by education and job type␣
  ↪(zoomed in)", fontsize=14)
facet.add_legend()
plt.show()
```

Figure 6. Income distribution by education and job type (zoomed in)

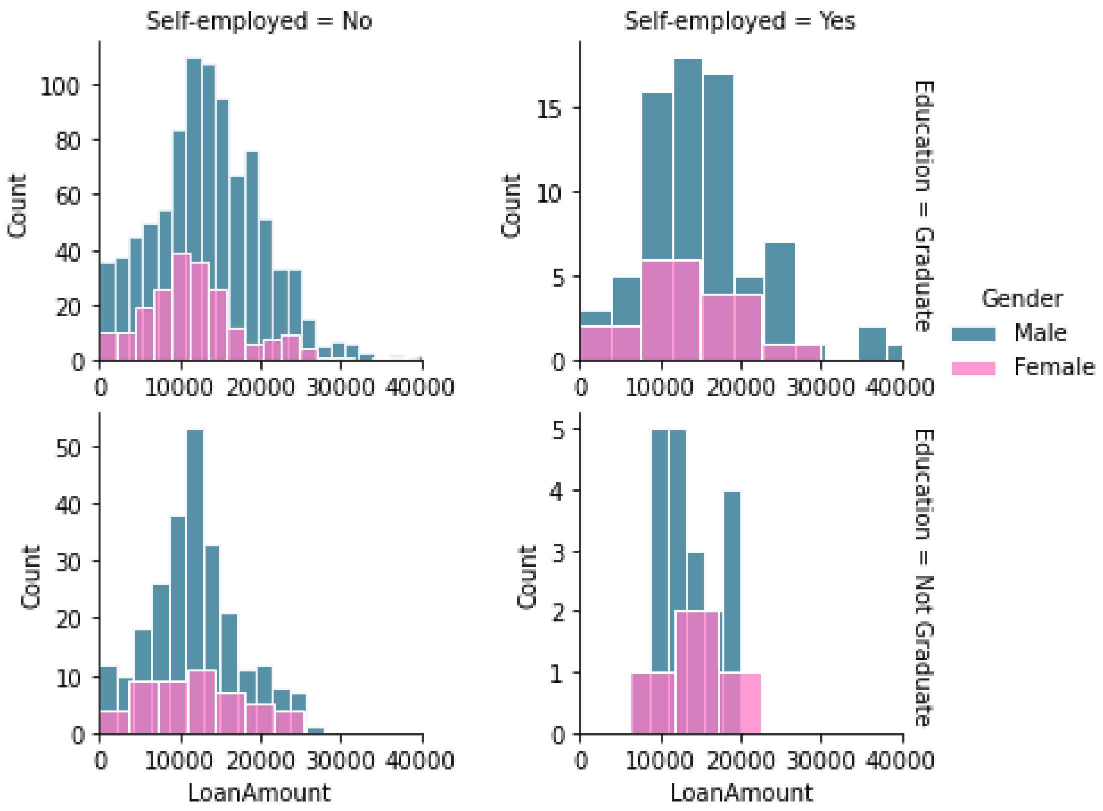


There appears to be some differences in the distribution of income across employment type but symmetry across education level as shown in figure 6. For instance, the distrubtions of applicants who are self-employed and those are not are different but the distrubtions of applicants who have graduated and those who have not are similar. We also notice a similar trend in figure 7 where laon amounts is plotted across employment type and education level.

Thus, the difference in income distribution seems to come from employment types rather than education level. However, it should noted that **until now we have not yet find any evidence of the differences but simply have observed some apparent differences**. We must do a difference test to confirm our observations but we will leave it for now.

```
[19]: facet = sns.FacetGrid(dt,row="Education", col="Self-employed", hue␣
        ↪="Gender",palette = ["#1F6E8C","#FF78C4"], sharey=False, sharex=False,␣
        ↪margin_titles=True, xlim=(0,40000))
      facet.map_dataframe(sns.histplot,x ="LoanAmount",
                          edgecolor = "white")
      facet.fig.subplots_adjust(top=0.8)
      facet.fig.suptitle("Figure 7. Distribution of loan amount by education and job␣
        ↪type (Zoomed in)")
      facet.add_legend()
      plt.show()
```

Figure 7. Distribution of loan amount by education and job type (Zoomed in)



Similarly, there appears to be some differences in the distribution of loan amount across education and employment type. Applicants who are not self-employed seems to ask for higher loan amount regardless of education level. However, the distribution of applicants who are self-employed seems to differ across education level. Graduated self-employed ask for Nonetheless, we think this difference, although expected, might simply be due to the small representativity of not-graduated self-employed customers.

2.5 Income and Age

[20]:
```
color = ["#40128B","#DD58D6"]
plt.figure(figsize=(16,10))
sns.scatterplot(data=dataset,
                x='IncomePerMonth',
                y="Age", hue="Gender",
                style="CivilState",
                palette = color,
                style_order=["Married","Unmarried","Divorced","Widow"]
               )
plt.xlim(-0.1,2)
plt.title("Figure 8. Age and Income of loan applicants (zoomed in)",␣
 ↪fontsize=14)
plt.xlabel("Income (in ten thousands)")
plt.show()
```

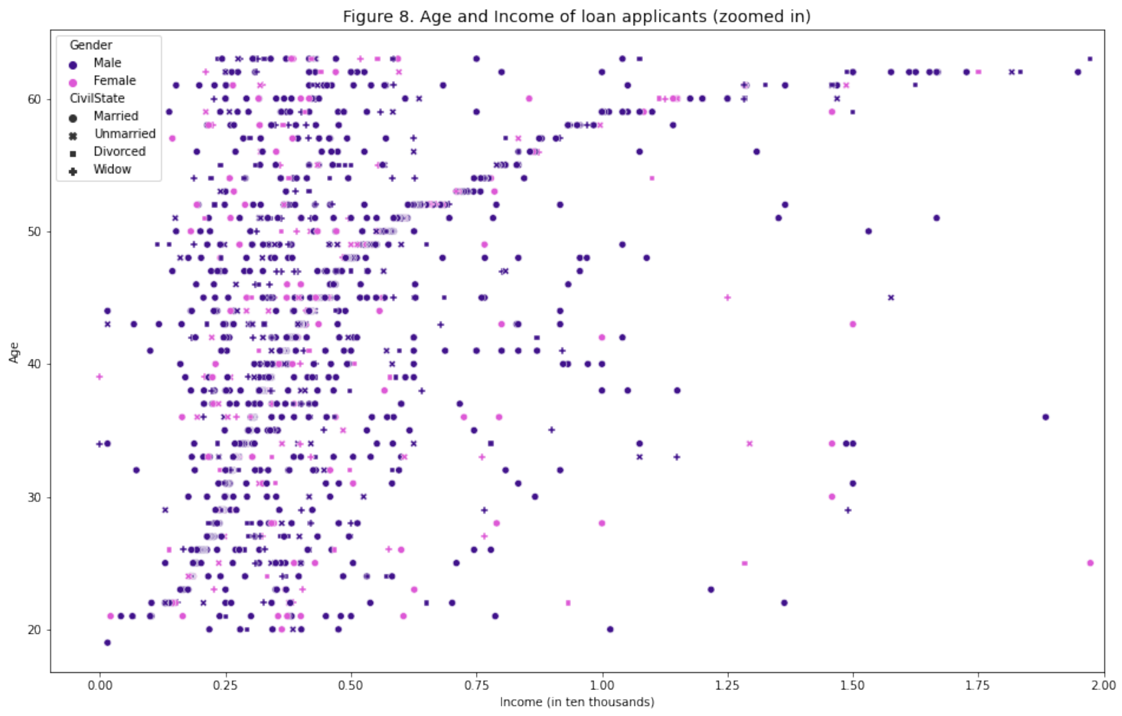
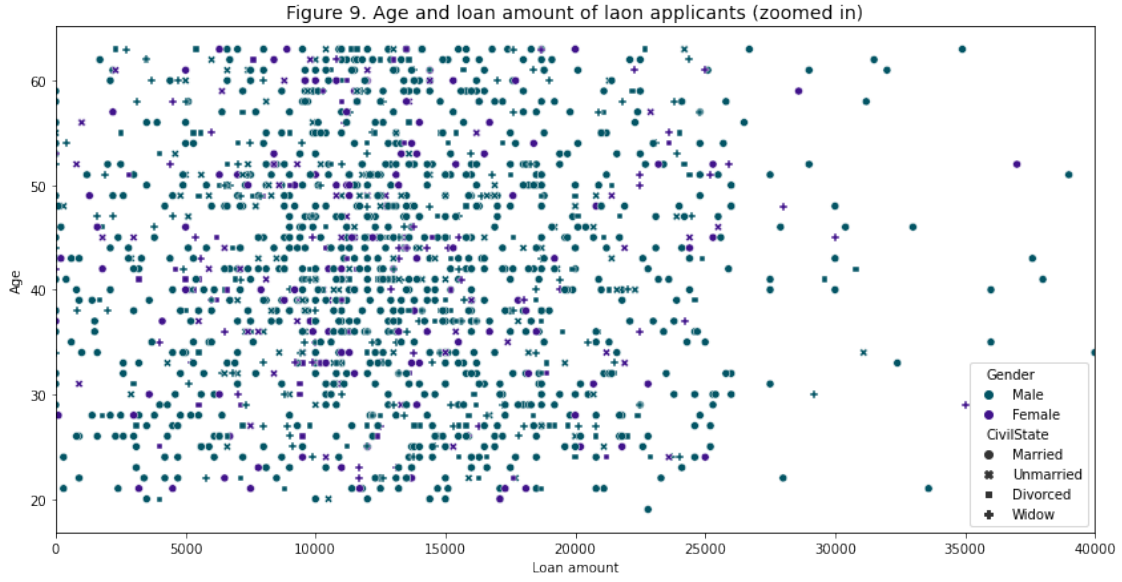


Figure 8 shows the relationship between age and income. In general, there appears to be not clear link. However, we observe a small increasing line trend that seems to form through the data. This line trend is predominantly composed of married male category, suggesting that married male tends to earn more with age. For the remainig, no clear trends was observed. We also observed no relationship between age and loan amount as shown in figure 9.

[21]:
```
color = ["#025464","#40128B"]
plt.figure(figsize=(14,7))
```

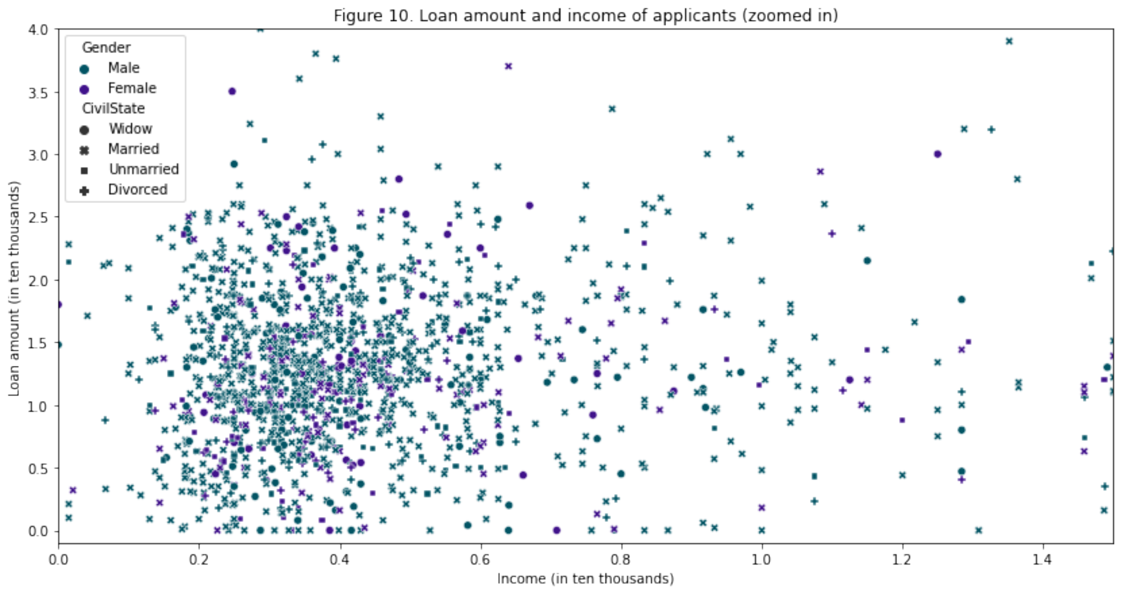
```
sns.scatterplot(data=dataset,
                x='LoanAmount',
                y="Age", hue="Gender",
                style="CivilState",
                palette = color,
                style_order=["Married","Unmarried","Divorced","Widow"]
               )
plt.xlim(-1,40000)
plt.title("Figure 9. Age and loan amount of laon applicants (zoomed in)",␣
 ↪fontsize=14)
plt.xlabel("Loan amount")
plt.show()
```



2.6 Loan and Income

[22]:
```
dt.LoanAmount = dt.LoanAmount/10000
```

[23]:
```
color = ["#025464","#40128B"]
plt.figure(figsize=(14,7))
sns.scatterplot(data=dt,
                x='IncomePerMonth',
                y="LoanAmount", hue="Gender",
                style="CivilState",
                palette = color
               )
plt.ylim(-0.1,4)
plt.xlim(0,1.5)
plt.title("Figure 10. Loan amount and income of applicants (zoomed in)")
plt.xlabel("Income (in ten thousands)")
plt.ylabel("Loan amount (in ten thousands)")
plt.show()
```



We observe an increasing trend between 2000 and 6000 in income and 5000 and 25000 in loan amounts. However, this trend ceases when income is greater than 6000. What we can say is that for individual with income between 2000 and 6000 appears, the loan amount seems to be very restrictive. We see an increasing trend meaning the higher income individuals get higher loan. This is expected as the bank seek to minimize risk by giving small amount to highly risky profile and long amount to safe profile. Nonetheless, this tends to matter less as income grows over 6 000.

2.7 Correlation

We conclude this first part with a correlation matrix of all the variables provided in the dataset. While there are 13 variables, we think it is possible but not neceassry to analyze every single variable. That is why we only selected some variables based on our knowledge and intuition on how they can influence credit policy at the bank level. However, if later on a variable appears to be relevant, we shall do the necessary univariate and monovarite analysis of that variable in case it has not been done.

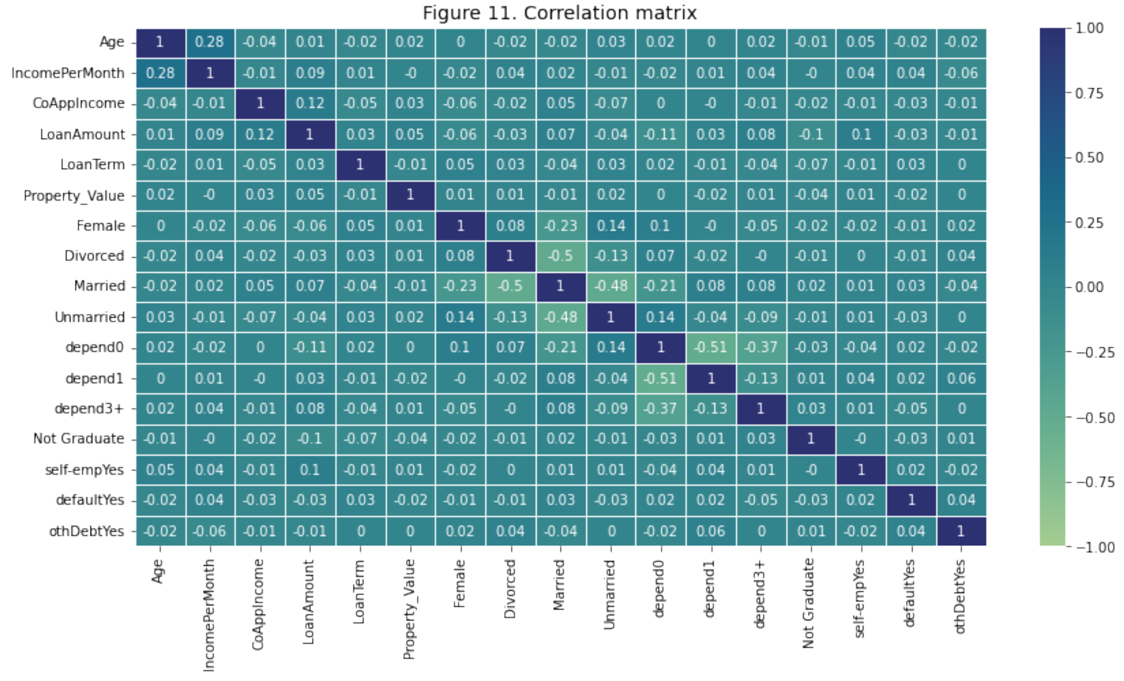
[24]:
```
dta = pd.get_dummies(dataset,
               columns=['Gender','CivilState', 'Dependents', 'Education',
                              'Self-employed', 'DefaultHistory',␣
 ↪'OtherDebts'],
               prefix = ["","","depend","","self-emp","default","othDebt"],
               prefix_sep=""
              )
```

For every categorical variable with n categories, we need n - 1 dummy variables. For example, gender has both Female and Male category. However, if an individual observation is not a male, it is obviously a female. Hence, there is not additional benefit for having two gender dummy variables. The same can be said for other variables as well.

```
[25]: # Getting rid of extra dummies
      dta = dta.iloc[:,[1,2,3,4,5,6,7,9,10,11,13,14,16,18,20,22,24]]
```

```
[26]: plt.figure(figsize=(14,7))
      sns.heatmap(dta.corr().round(2),vmin=-1, vmax=1,cmap="crest",␣
       ↪linecolor="white", linewidths=1, annot=True)
      plt.title("Figure 11. Correlation matrix", fontsize=14)
      plt.show()
```

Figure 11. Correlation matrix



	Age	IncomePerMonth	CoAppIncome	LoanAmount	LoanTerm	Property_Value	Female	Divorced	Married	Unmarried	depend0	depend1	depend3+	Not Graduate	self-empYes	defaultYes	othDebtYes
Age	1	0.28	-0.04	0.01	-0.02	0.02	0	-0.02	-0.02	0.03	0.02	0	0.02	-0.01	0.05	-0.02	-0.02
IncomePerMonth	0.28	1	-0.01	0.09	0.01	-0	-0.02	0.04	0.02	-0.01	-0.02	0.01	0.04	-0	0.04	0.04	-0.06
CoAppIncome	-0.04	-0.01	1	0.12	-0.05	0.03	-0.06	-0.02	0.05	-0.07	0	-0	-0.01	-0.02	-0.01	-0.03	-0.01
LoanAmount	0.01	0.09	0.12	1	0.03	0.05	-0.06	-0.03	0.07	-0.04	-0.11	0.03	0.08	-0.1	0.1	-0.03	-0.01
LoanTerm	-0.02	0.01	-0.05	0.03	1	-0.01	0.05	0.03	-0.04	0.03	0.02	-0.01	-0.04	-0.07	-0.01	0.03	0
Property_Value	0.02	-0	0.03	0.05	-0.01	1	0.01	0.01	-0.01	0.02	0	-0.02	0.01	-0.04	0.01	-0.02	0
Female	0	-0.02	-0.06	-0.06	0.05	0.01	1	0.08	-0.23	0.14	0.1	-0	-0.05	-0.02	-0.02	-0.01	0.02
Divorced	-0.02	0.04	-0.02	-0.03	0.03	0.01	0.08	1	-0.5	-0.13	0.07	-0.02	-0	-0.01	0	-0.01	0.04
Married	-0.02	0.02	0.05	0.07	-0.04	-0.01	-0.23	-0.5	1	-0.48	-0.21	0.08	0.08	0.02	0.01	0.03	-0.04
Unmarried	0.03	-0.01	-0.07	-0.04	0.03	0.02	0.14	-0.13	-0.48	1	0.14	-0.04	-0.09	-0.01	0.01	-0.03	0
depend0	0.02	-0.02	0	-0.11	0.02	0	0.1	0.07	-0.21	0.14	1	-0.51	-0.37	-0.03	-0.04	0.02	-0.02
depend1	0	0.01	-0	0.03	-0.01	-0.02	-0	-0.02	0.08	-0.04	-0.51	1	-0.13	0.01	0.04	0.02	0.06
depend3+	0.02	0.04	-0.01	0.08	-0.04	0.01	-0.05	-0	0.08	-0.09	-0.37	-0.13	1	0.03	0.01	-0.05	0
Not Graduate	-0.01	-0	-0.02	-0.1	-0.07	-0.04	-0.02	-0.01	0.02	-0.01	-0.03	0.01	0.03	1	-0	-0.03	0.01
self-empYes	0.05	0.04	-0.01	0.1	-0.01	0.01	-0.02	0	0.01	0.01	-0.04	0.04	0.01	-0	1	0.02	-0.02
defaultYes	-0.02	0.04	-0.03	-0.03	0.03	-0.02	-0.01	-0.01	0.03	-0.03	0.02	0.02	-0.05	-0.03	0.02	1	0.04
othDebtYes	-0.02	-0.06	-0.01	-0.01	0	0	0.02	0.04	-0.04	0	-0.02	0.06	0	0.01	-0.02	0.04	1

Figure 11 shows no strong correlation between two independent variables. However, maried and divorced, married and unmarried, and depend0 and depend1 have quite higher negative correlation. This might suggest, for instance, combining some of those cateegories into a single one. We will also postpone this task until part 2 of the analysis.

If you enjoyed reading this analysis, don't hesitate to contact me for any further inquiry @ eliediwa9@gmail.com.

```
[ ]:
```