PROJECT A: Credit Card Defaults

In [1]:

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:

```
filepath = 'C:/Users/User/Downloads/default_credit.xls'
```

In [3]:

```
df = pd.read_excel(filepath)
```

In [4]:

```
df.head()
```

Out[4]:

	ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	...	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1	PAY_AMT2	PA
0	1	20000	2	2	1	24	2	2	-1	-1	...	0	0	0	0	689	
1	2	120000	2	2	2	26	-1	2	0	0	...	3272	3455	3261	0	1000	
2	3	90000	2	2	2	34	0	0	0	0	...	14331	14948	15549	1518	1500	
3	4	50000	2	2	1	37	0	0	0	0	...	28314	28959	29547	2000	2019	
4	5	50000	1	2	1	57	-1	0	-1	0	...	20940	19146	19131	2000	36681	

5 rows × 25 columns

In [5]:

```
![image.png](attachment:image.png)
```

```
'[image.png]' is not recognized as an internal or external command,
operable program or batch file.
```

In [6]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30000 entries, 0 to 29999
Data columns (total 25 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   ID                          30000 non-null  int64
 1   LIMIT_BAL                   30000 non-null  int64
 2   SEX                         30000 non-null  int64
 3   EDUCATION                   30000 non-null  int64
 4   MARRIAGE                    30000 non-null  int64
 5   AGE                         30000 non-null  int64
 6   PAY_0                       30000 non-null  int64
 7   PAY_2                       30000 non-null  int64
 8   PAY_3                       30000 non-null  int64
 9   PAY_4                       30000 non-null  int64
 10  PAY_5                       30000 non-null  int64
 11  PAY_6                       30000 non-null  int64
 12  BILL_AMT1                   30000 non-null  int64
 13  BILL_AMT2                   30000 non-null  int64
 14  BILL_AMT3                   30000 non-null  int64
 15  BILL_AMT4                   30000 non-null  int64
 16  BILL_AMT5                   30000 non-null  int64
 17  BILL_AMT6                   30000 non-null  int64
 18  PAY_AMT1                    30000 non-null  int64
 19  PAY_AMT2                    30000 non-null  int64
 20  PAY_AMT3                    30000 non-null  int64
 21  PAY_AMT4                    30000 non-null  int64
 22  PAY_AMT5                    30000 non-null  int64
 23  PAY_AMT6                    30000 non-null  int64
 24  default payment next month  30000 non-null  int64
dtypes: int64(25)
memory usage: 5.7 MB
```

In [7]:

```
df.describe().T
```

Out[7]:

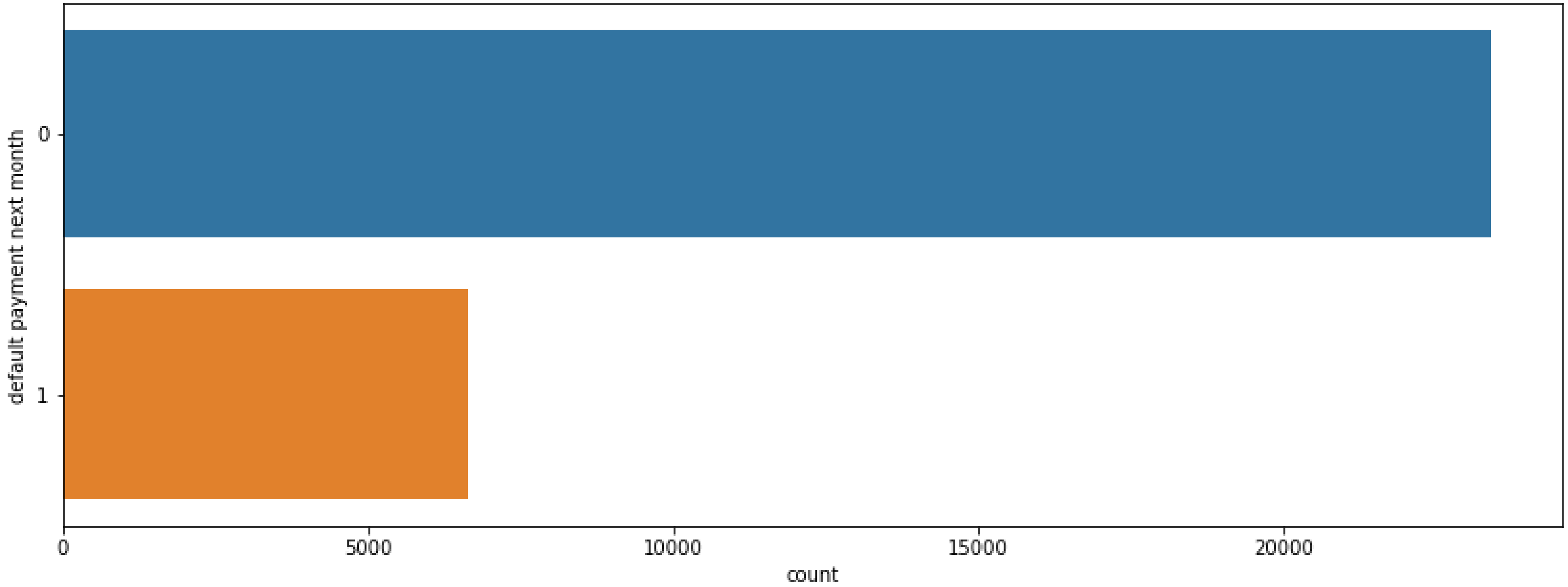
	count	mean	std	min	25%	50%	75%	max
ID	30000.0	15000.500000	8660.398374	1.0	7500.75	15000.5	22500.25	30000.0
LIMIT_BAL	30000.0	167484.322667	129747.661567	10000.0	50000.00	140000.0	240000.00	1000000.0
SEX	30000.0	1.603733	0.489129	1.0	1.00	2.0	2.00	2.0
EDUCATION	30000.0	1.853133	0.790349	0.0	1.00	2.0	2.00	6.0

	count	mean	std	min	25%	50%	75%	max
MARRIAGE	30000.0	1.551867	0.521970	0.0	1.00	2.0	2.00	3.0
AGE	30000.0	35.485500	9.217904	21.0	28.00	34.0	41.00	79.0
PAY_0	30000.0	-0.016700	1.123802	-2.0	-1.00	0.0	0.00	8.0
PAY_2	30000.0	-0.133767	1.197186	-2.0	-1.00	0.0	0.00	8.0
PAY_3	30000.0	-0.166200	1.196868	-2.0	-1.00	0.0	0.00	8.0
PAY_4	30000.0	-0.220667	1.169139	-2.0	-1.00	0.0	0.00	8.0
PAY_5	30000.0	-0.266200	1.133187	-2.0	-1.00	0.0	0.00	8.0
PAY_6	30000.0	-0.291100	1.149988	-2.0	-1.00	0.0	0.00	8.0
BILL_AMT1	30000.0	51223.330900	73635.860576	-165580.0	3558.75	22381.5	67091.00	964511.0
BILL_AMT2	30000.0	49179.075167	71173.768783	-69777.0	2984.75	21200.0	64006.25	983931.0
BILL_AMT3	30000.0	47013.154800	69349.387427	-157264.0	2666.25	20088.5	60164.75	1664089.0
BILL_AMT4	30000.0	43262.948967	64332.856134	-170000.0	2326.75	19052.0	54506.00	891586.0
BILL_AMT5	30000.0	40311.400967	60797.155770	-81334.0	1763.00	18104.5	50190.50	927171.0
BILL_AMT6	30000.0	38871.760400	59554.107537	-339603.0	1256.00	17071.0	49198.25	961664.0
PAY_AMT1	30000.0	5663.580500	16563.280354	0.0	1000.00	2100.0	5006.00	873552.0
PAY_AMT2	30000.0	5921.163500	23040.870402	0.0	833.00	2009.0	5000.00	1684259.0
PAY_AMT3	30000.0	5225.681500	17606.961470	0.0	390.00	1800.0	4505.00	896040.0
PAY_AMT4	30000.0	4826.076867	15666.159744	0.0	296.00	1500.0	4013.25	621000.0
PAY_AMT5	30000.0	4799.387633	15278.305679	0.0	252.50	1500.0	4031.50	426529.0
PAY_AMT6	30000.0	5215.502567	17777.465775	0.0	117.75	1500.0	4000.00	528666.0
default payment next month	30000.0	0.221200	0.415062	0.0	0.00	0.0	0.00	1.0

In [8]:

```
# check for amount of defaults in the data using countplot
plt.figure(figsize=(14,5))
```

```
sns.countplot(y="default payment next month", data=df)
plt.show()
```

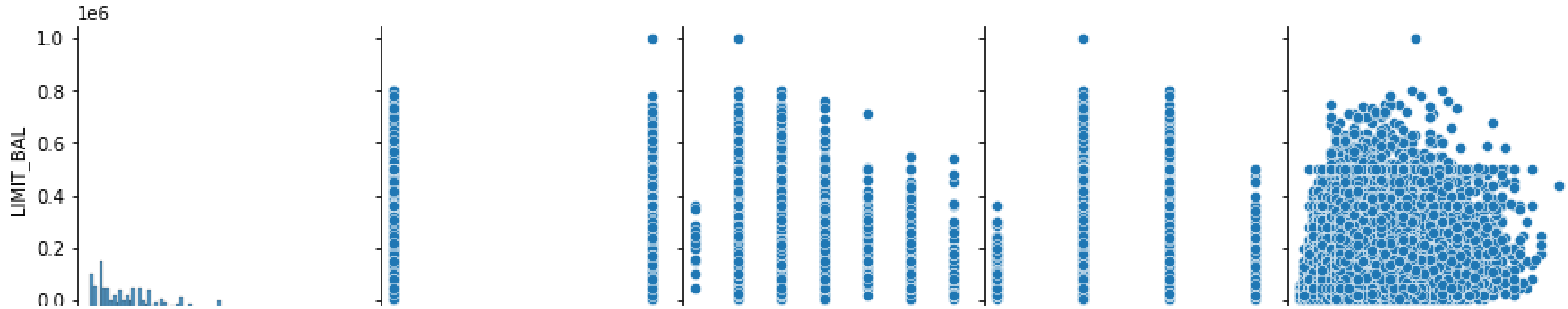


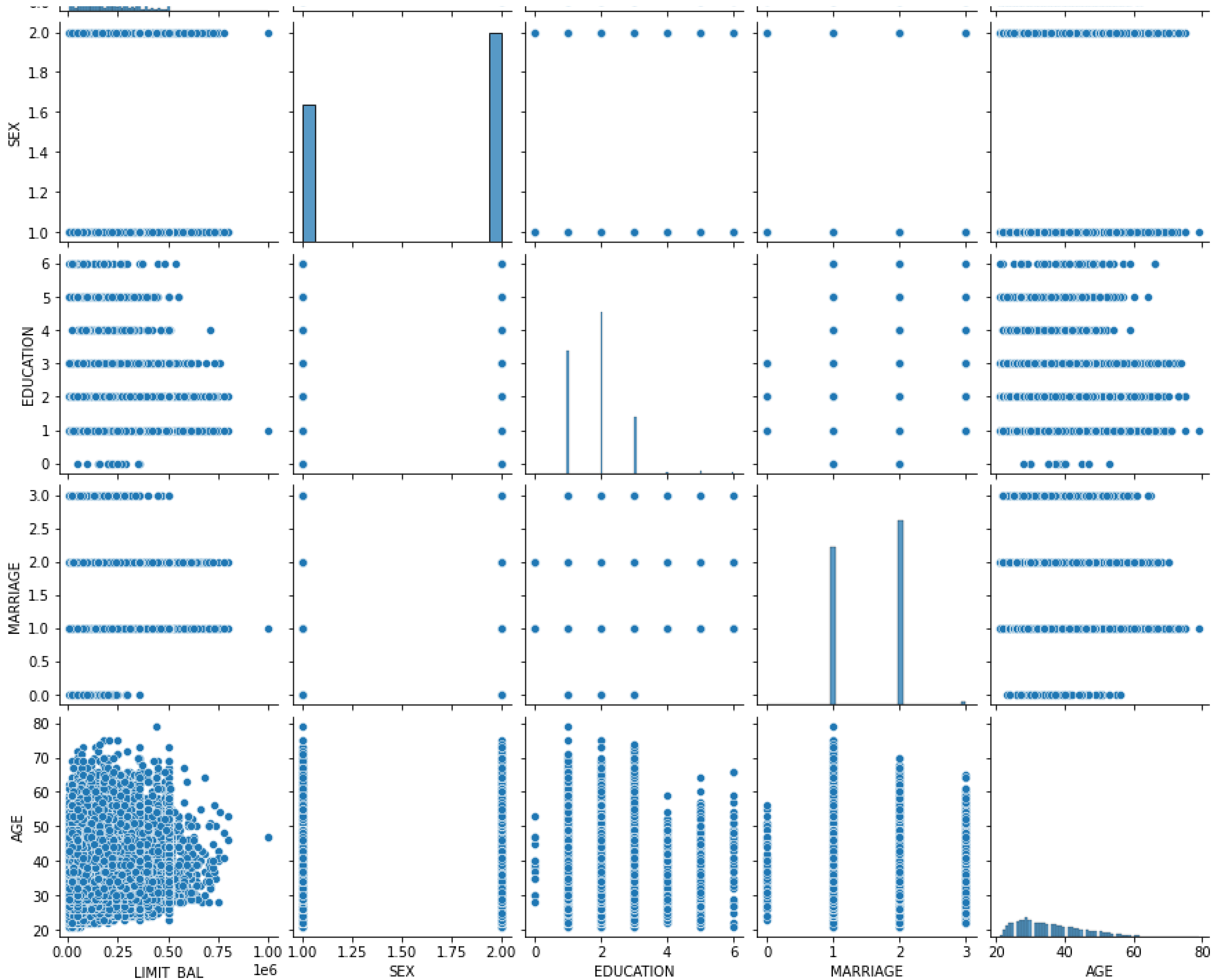
From above plot we can see that around 21.7% i.e. 6500 people are defaulters in total of 30000 records.

Univariate Analysis: Univariate analysis is the most basic form of the data analysis technique. When we want to understand the data contained by only one variable and don't want to deal with the causes or effect relationships then a Univariate analysis technique is used.

In [9]:
```
sns.pairplot(df[['LIMIT_BAL','SEX','EDUCATION','MARRIAGE','AGE']])
```

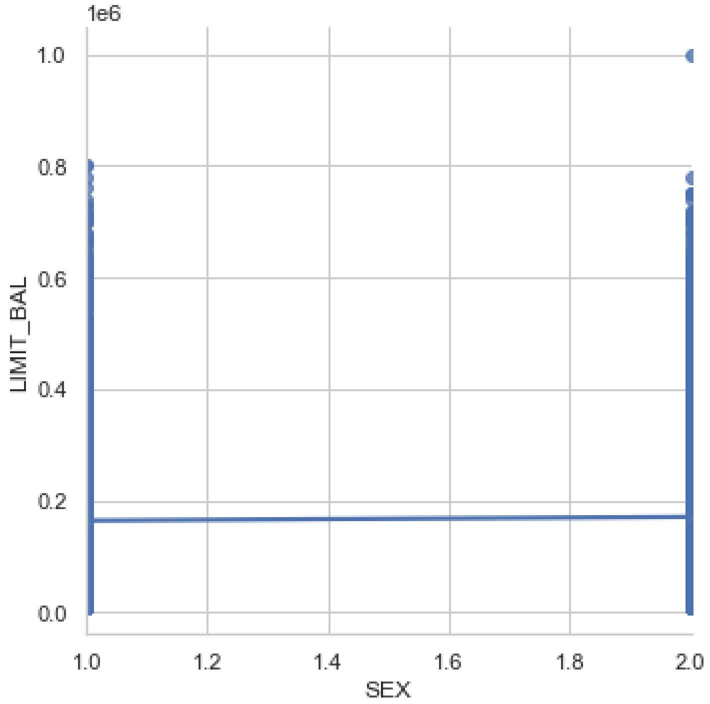
Out[9]: <seaborn.axisgrid.PairGrid at 0x210a8d29ee0>




SEX
EDUCATION
MARRIAGE
AGE
LIMIT_BAL
1e6
SEX
EDUCATION
MARRIAGE
AGE

In [10]:

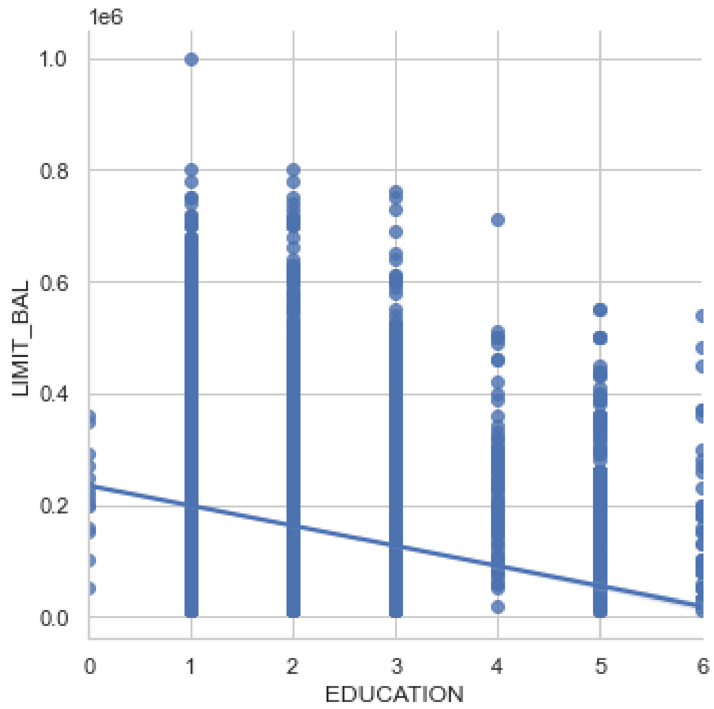
```
import seaborn as sns
sns.set(style="whitegrid")
ax = sns.lmplot(x="SEX", y="LIMIT_BAL", data=df)
```



Limit_Bal is the same across sex.

In [11]:

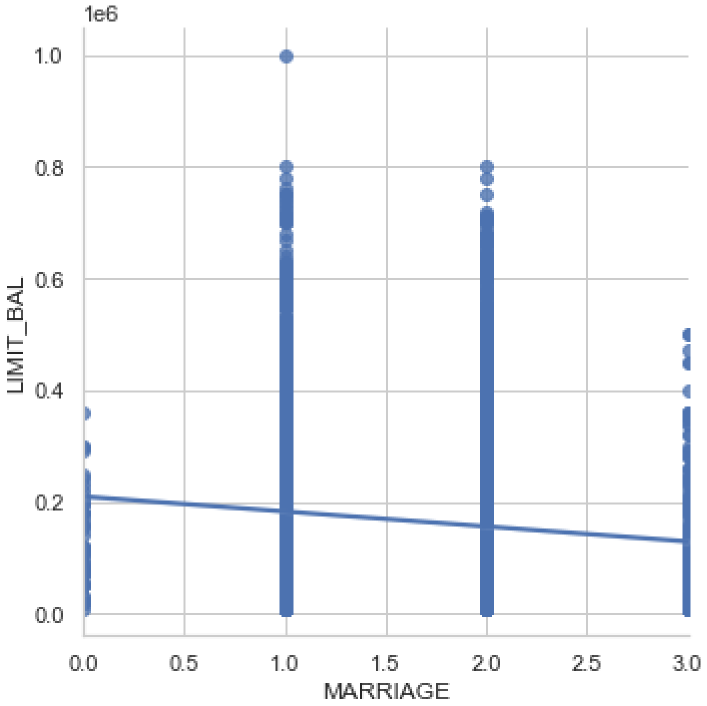
```
sns.set(style="whitegrid")
ax = sns.lmplot(x="EDUCATION", y="LIMIT_BAL", data=df)
```



EDUCATION: (1=graduate school, 2=university, 3=high school, 4=others, 5=unknown, 6=unknown).

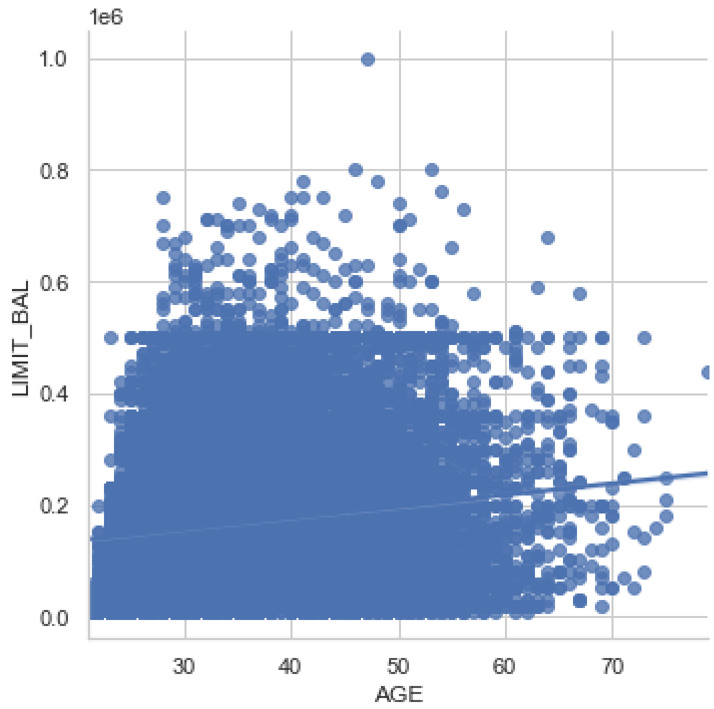
Limit_Bal decrease from graduate school, University, high school down to others.

In [12]:
```
sns.set(style="whitegrid")
ax = sns.lmplot(x="MARRIAGE", y="LIMIT_BAL", data=df)
```



MARRIAGE: Marital status (1=married, 2=single, 3=others) : No much of difference

In [13]:
```
sns.set(style="whitegrid")
ax = sns.lmplot(x="AGE", y="LIMIT_BAL", data=df)
```



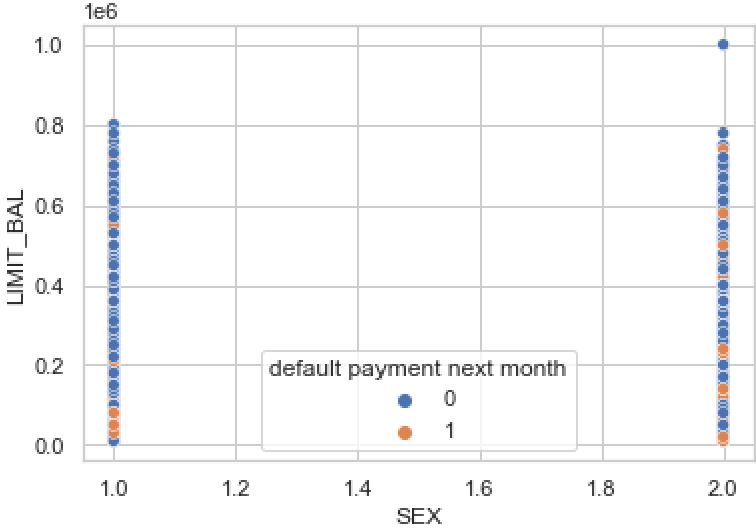
From above plot, i can infer that age above 60 received higher limit_Bal. Meaning those age group could be credit card worthy and attention should be giving to them.

In [14]:

```
sns.scatterplot(x='SEX', y ='LIMIT_BAL' ,
data = df , hue = 'default payment next month')
```

Out[14]:

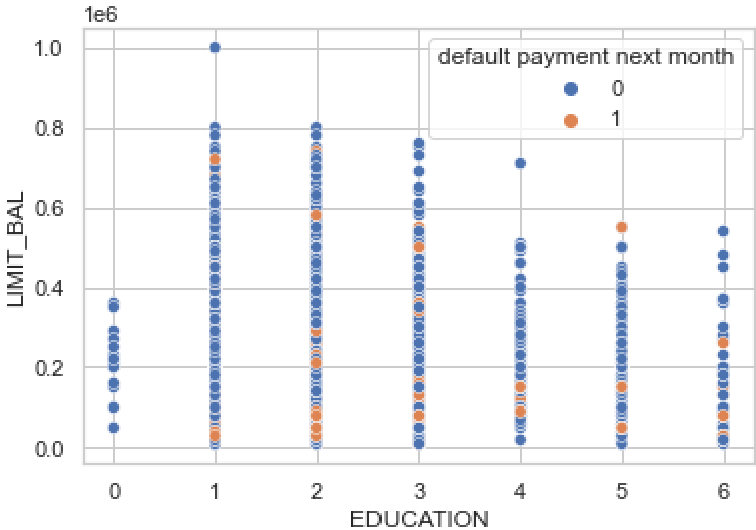
```
<AxesSubplot:xlabel='SEX', ylabel='LIMIT_BAL'>
```



In the plot above we can observe that Limit_Bal does not clearly determine default payment across sex.

In [15]:
```
sns.scatterplot(x='EDUCATION', y ='LIMIT_BAL' ,
data = df , hue = 'default payment next month')
```

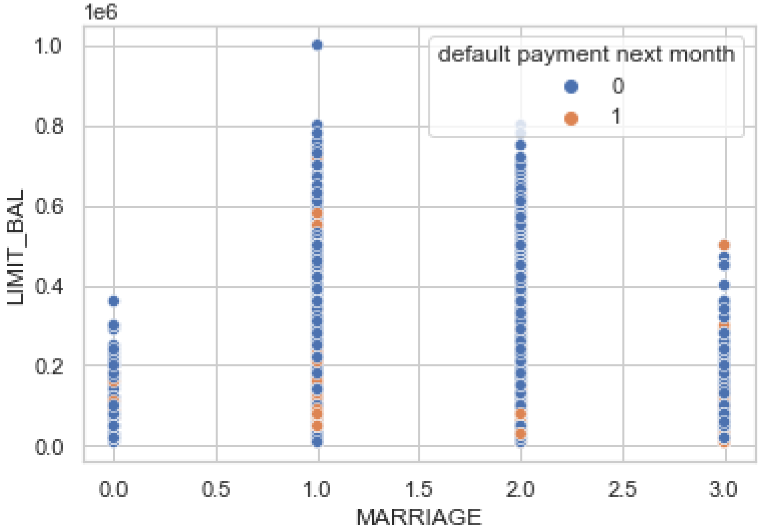
Out[15]: <AxesSubplot:xlabel='EDUCATION', ylabel='LIMIT_BAL'>



Above plot shows unevenly distributed default payment across education levels university taking the lead.

In [16]:
```
sns.scatterplot(x='MARRIAGE', y ='LIMIT_BAL' ,
data = df , hue = 'default payment next month')
```

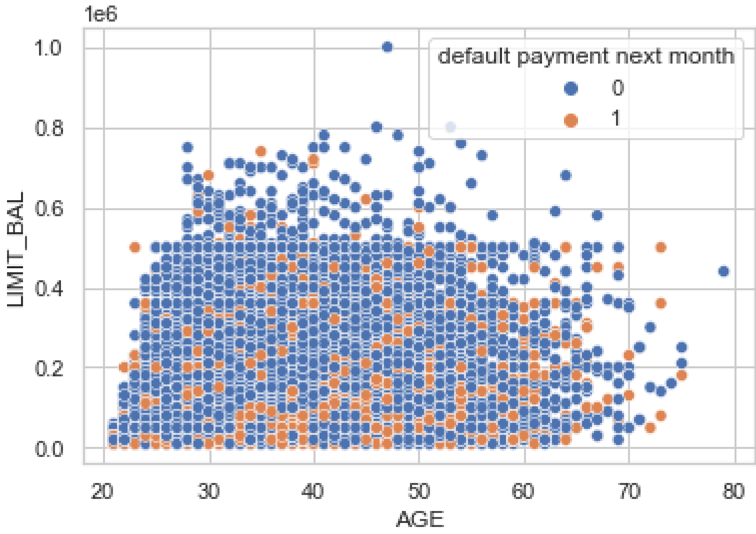
Out[16]: <AxesSubplot:xlabel='MARRIAGE', ylabel='LIMIT_BAL'>



From above plot, married customers were giving higher limit_Bal. Implies that they are more reliable in terms of default.

In [17]:
```
sns.scatterplot(x='AGE', y ='LIMIT_BAL' ,
data = df , hue = 'default payment next month')
```

Out[17]: <AxesSubplot:xlabel='AGE', ylabel='LIMIT_BAL'>



The above plot indicates high clusters of default payment amongst the age group between 20 and 60 within limit_Bal range of 0.0 to 0.5

In [ ]:

```

```

From this graph, we could roughly see those non-default creditors and their families tend to have higher given credits, and non-default creditors tend to be older. However, the effect is not very obvious because of the scale.

Significant features members/labels

In [18]:

```
educLevels = sorted(df.EDUCATION.unique())
```

In [19]:

```
educLevels
```

Out[19]: [0, 1, 2, 3, 4, 5, 6]

The EDUCATION column has 7 unique values, but as per our data description, we have only 4 unique values.

In [20]:

```
df
```

Out[20]:

	ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	...	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1	PAY_AI
0	1	20000	2	2	1	24	2	2	-1	-1	...	0	0	0	0	
1	2	120000	2	2	2	26	-1	2	0	0	...	3272	3455	3261	0	1
2	3	90000	2	2	2	34	0	0	0	0	...	14331	14948	15549	1518	1
3	4	50000	2	2	1	37	0	0	0	0	...	28314	28959	29547	2000	2
4	5	50000	1	2	1	57	-1	0	-1	0	...	20940	19146	19131	2000	36
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
29995	29996	220000	1	3	1	39	0	0	0	0	...	88004	31237	15980	8500	20
29996	29997	150000	1	3	2	43	-1	-1	-1	-1	...	8979	5190	0	1837	3
29997	29998	30000	1	2	2	37	4	3	2	-1	...	20878	20582	19357	0	
29998	29999	80000	1	3	1	41	1	-1	0	0	...	52774	11855	48944	85900	3
29999	30000	50000	1	2	1	46	0	0	0	0	...	36535	32428	15313	2078	1

30000 rows × 25 columns

In [21]:

```
sorted(df.EDUCATION.unique())
```

Out[21]: `[0, 1, 2, 3, 4, 5, 6]`

In [22]:

```
df.groupby(['EDUCATION']).count()
```

Out[22]:

	ID	LIMIT_BAL	SEX	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	PAY_5	...	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1	PAY
EDUCATION																

	ID	LIMIT_BAL	SEX	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	PAY_5	...	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1	PAY
EDUCATION																
0	14	14	14	14	14	14	14	14	14	14	...	14	14	14	14	
1	10585	10585	10585	10585	10585	10585	10585	10585	10585	10585	...	10585	10585	10585	10585	
2	14030	14030	14030	14030	14030	14030	14030	14030	14030	14030	...	14030	14030	14030	14030	
3	4917	4917	4917	4917	4917	4917	4917	4917	4917	4917	...	4917	4917	4917	4917	
4	123	123	123	123	123	123	123	123	123	123	...	123	123	123	123	
5	280	280	280	280	280	280	280	280	280	280	...	280	280	280	280	
6	51	51	51	51	51	51	51	51	51	51	...	51	51	51	51	

7 rows × 24 columns

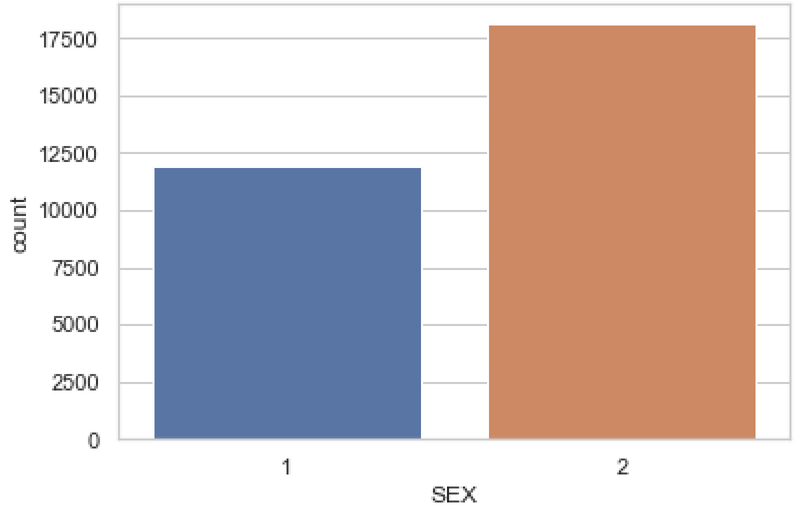
EDUCATION: (1=graduate school, 2=university, 3=high school, 4=others, 5=unknown, 6=unknown)

In [23]:

```
# plot count plot for the sex column
sns.countplot(df.SEX)
```

```
C:\Users\User\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg:
x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword w
ill result in an error or misinterpretation.
  warnings.warn(
```

Out[23]: `<AxesSubplot:xlabel='SEX', ylabel='count'>`



SEX: Gender (1=male, 2=female)

In [24]:
```
df['SEX'].unique()
```

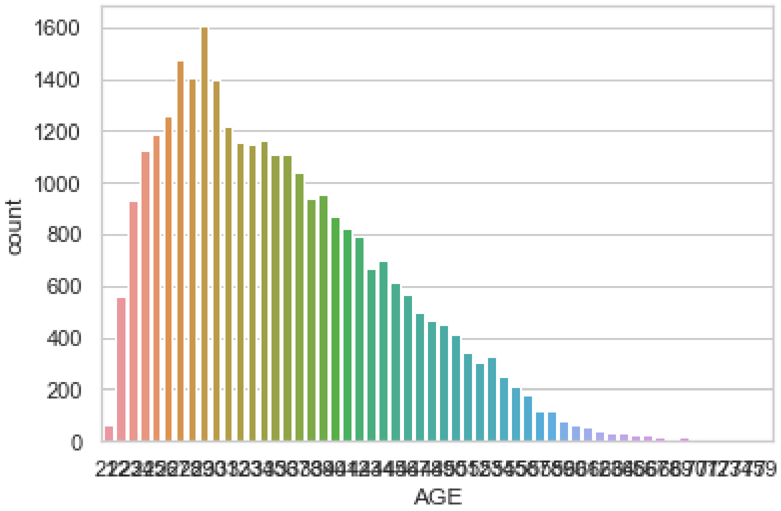
Out[24]:
```
array([2, 1], dtype=int64)
```

Women (Gender: 2) are likely to default more than Male (Gender: 1).

In [25]:
```
sns.countplot(df.AGE)
```

```
C:\Users\User\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg:
x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword w
ill result in an error or misinterpretation.
  warnings.warn(
```

Out[25]:
```
<AxesSubplot:xlabel='AGE', ylabel='count'>
```



By analyzing the above plot, we find that very few older people are likely to default credit cards after turning 50. Also, people between the ages of 20– and 40 likely to be a defaulters group. This provides us an insight that youth showed tendency to default. Hence, credit card issuing firms could review the amount of limit_bal for the youth and target people above 50.

Default payment (1=yes, 0=no)

In [26]:
```
df['default payment next month'].unique()
```

Out[26]:
```
array([1, 0], dtype=int64)
```

In [27]:
```
df_educaLevel = df['EDUCATION'].unique()
```

In [28]:
```
df_educaLevel
```

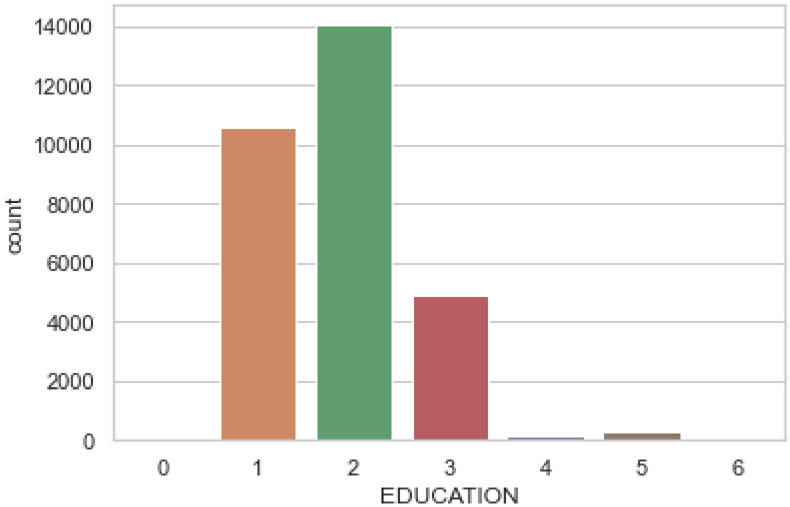
Out[28]:
```
array([2, 1, 3, 5, 4, 6, 0], dtype=int64)
```

In [29]:
```
# plot count plot for the Education column
sns.countplot(df.EDUCATION)
```

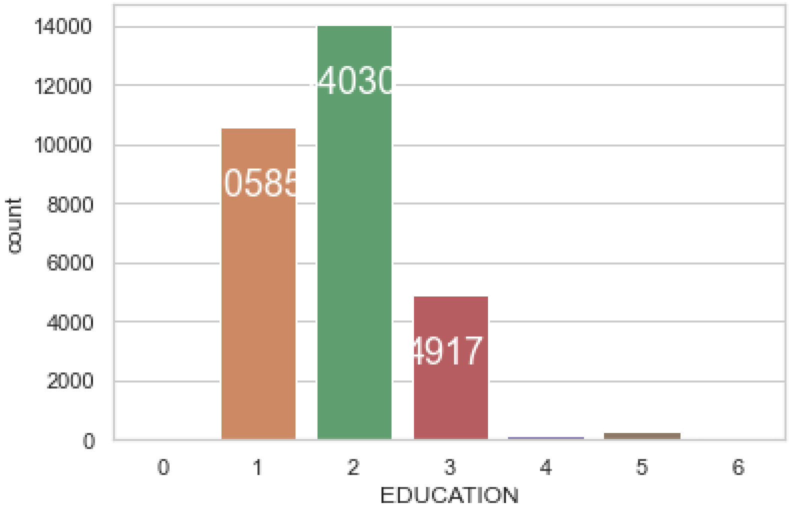
```
C:\Users\User\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg:
x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword w
```

```
ill result in an error or misinterpretation.
  warnings.warn(
```

Out[29]: <AxesSubplot:xlabel='EDUCATION', ylabel='count'>



In [30]:
```
ax = sns.countplot(x='EDUCATION', data = df)
for p in ax.patches:
    ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.35, p.get_height()), ha='center', va='top', color='white', size=18)
plt.show()
```



From above plot for 'Education Levels' we can infer that the defaulters rate is increasing amongs the University customers with hence their limit_bal should be reviewed.

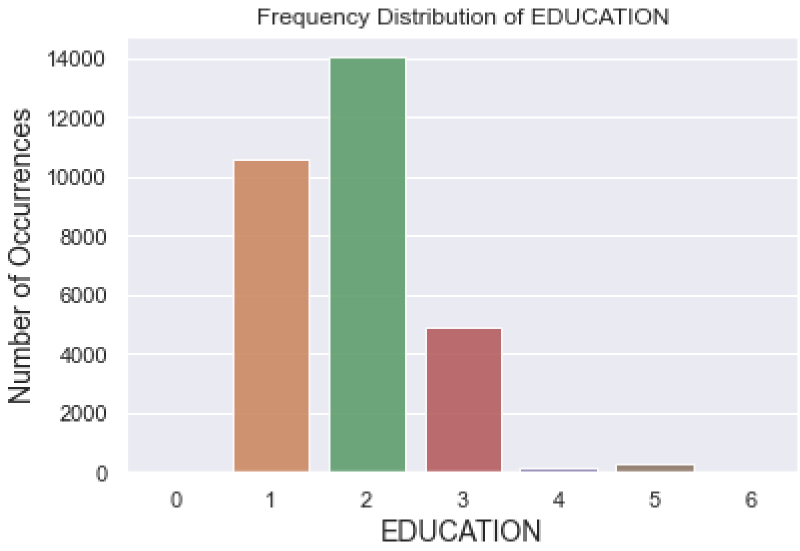
In [31]:
```
cat_df_educaLevel = df['EDUCATION'].unique()
```

In [32]:
```
cat_df_educaLevel
```

Out[32]:
```
array([2, 1, 3, 5, 4, 6, 0], dtype=int64)
```

In [33]:
```
%matplotlib inline
import seaborn as sns
import matplotlib.pyplot as plt
educLevels_count = df['EDUCATION'].value_counts()
sns.set(style="darkgrid")
sns.barplot(educLevels_count.index, educLevels_count.values, alpha=0.9)
plt.title('Frequency Distribution of EDUCATION')
plt.ylabel('Number of Occurrences', fontsize=14)
plt.xlabel('EDUCATION', fontsize=14)
plt.show()
```

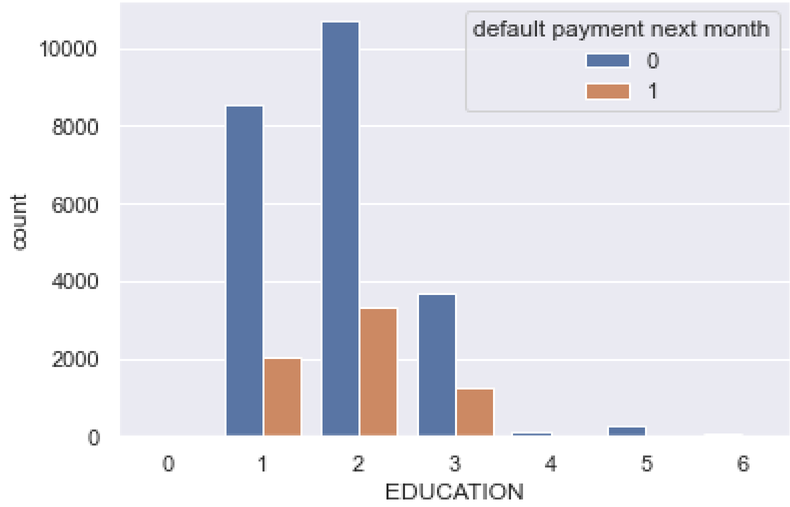
```
C:\Users\User\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword args:
x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keywor
d will result in an error or misinterpretation.
  warnings.warn(
```



Bivariate Analysis

In [34]:
```
sns.countplot(x='EDUCATION',hue='default payment next month',data = df)
```

Out[34]: <AxesSubplot:xlabel='EDUCATION', ylabel='count'>



In [35]:
```
import pandas as pd
```

```
data = pd.read_excel('C:/Users/User/Downloads/default_credit.xls')
freq_dis_educLevels = df['EDUCATION'].value_counts()
freq_dis_educLevels
```

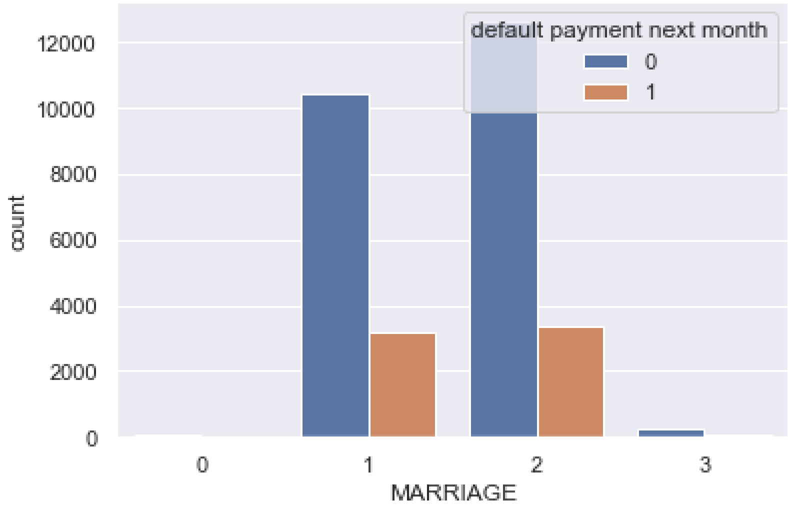
Out[35]:
```
2    14030
1    10585
3     4917
5      280
4      123
6       51
0       14
Name: EDUCATION, dtype: int64
```

The table above shows that category 2 which is University constitutes high rate of defaulters.

In [36]:
```
_educaLevel = df['MARRIAGE'].unique()
```

In [37]:
```
sns.countplot(x='MARRIAGE',hue='default payment next month',data = df)
```

Out[37]:
```
<AxesSubplot:xlabel='MARRIAGE', ylabel='count'>
```



In [38]:
```
cat_df_mar_Status = df['MARRIAGE'].unique()
```

In [39]:
```
cat_df_mar_Status
```

Out[39]:
```
array([1, 2, 3, 0], dtype=int64)
```

From above plot for 'MARRIAGE' we can infer that the defaulters rate is nearly constant for feature 'MARRIAGE', hence rate of default will not depend on 'MARRIAGE' feature.

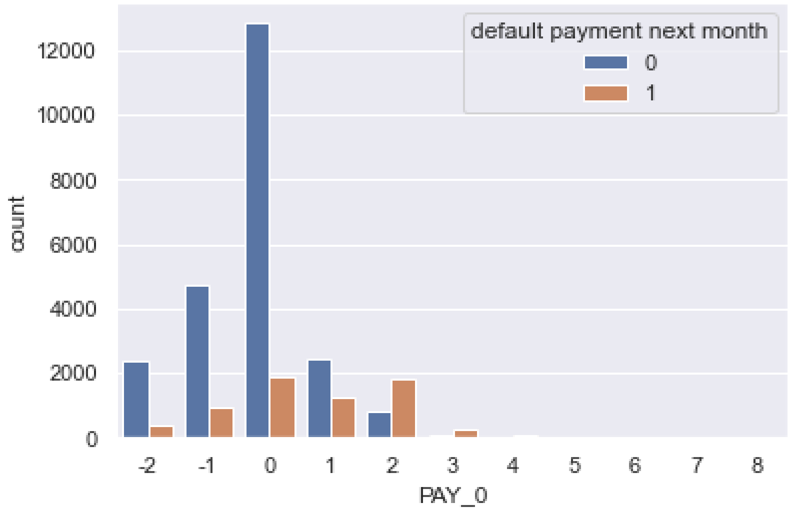
In [40]:
```
import pandas as pd
data = pd.read_excel('C:/Users/User/Downloads/default_credit.xls')
freq_dis_mar_Status = df['MARRIAGE'].value_counts()
freq_dis_mar_Status
```

Out[40]:
```
2    15964
1    13659
3      323
0       54
Name: MARRIAGE, dtype: int64
```

From above table and countplot no significant difference amongs marital status.

In [41]:
```
sns.countplot(x='PAY_0',hue='default payment next month',data = df)
```

Out[41]:
```
<AxesSubplot:xlabel='PAY_0', ylabel='count'>
```

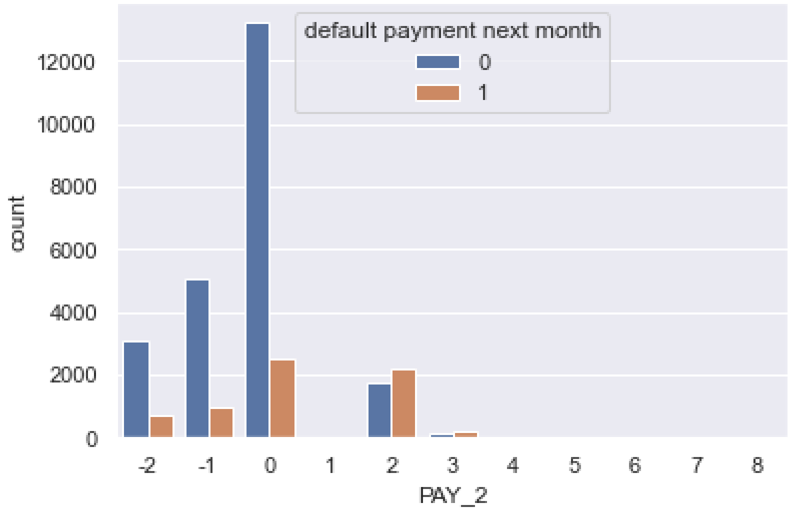


PAY_(0- 6): Repayment status in (September — April), 2005 (-1=pay duly, 1=payment delay for one month, 2=payment delay for two months, … 8=payment delay for eight months, 9=payment delay for nine months and above)

From above plot, it shows that default payment is higher amongst the customers that delay payment for one month and two months.

In [42]:
```
sns.countplot(x='PAY_2',hue='default payment next month',data = df)
```

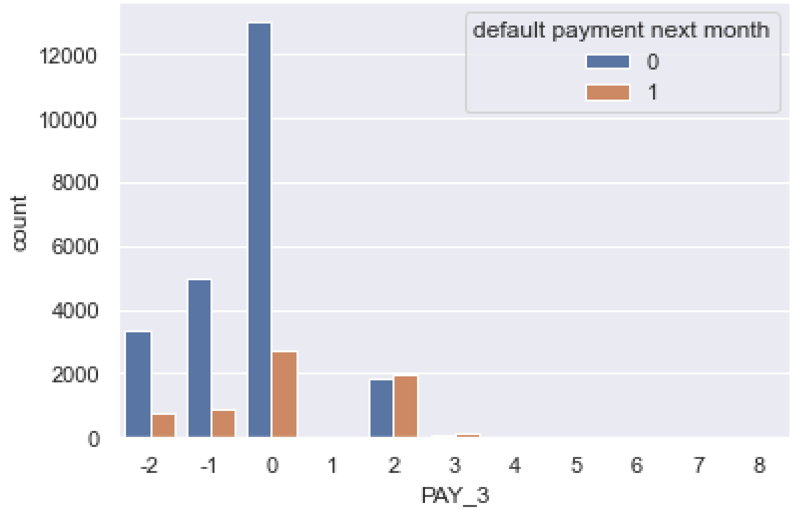
Out[42]: <AxesSubplot:xlabel='PAY_2', ylabel='count'>



From above plot, it shows that default payment is higher amongst the customers that delay payment two months.

In [43]:
```
sns.countplot(x='PAY_3',hue='default payment next month',data = df)
```

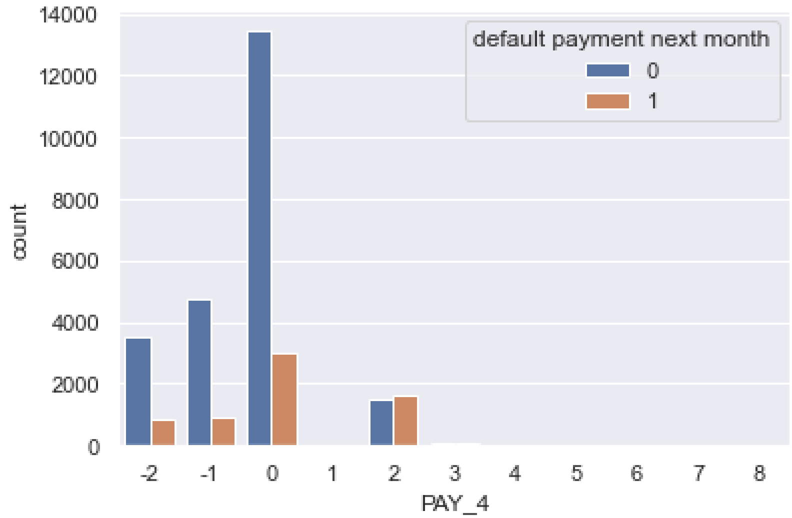
Out[43]: <AxesSubplot:xlabel='PAY_3', ylabel='count'>



From PAY_3 plot, the customers that delay payment two months have the same numbers of default payment and non- default payment.

In [44]:
```
sns.countplot(x='PAY_4',hue='default payment next month',data = df)
```

Out[44]: <AxesSubplot:xlabel='PAY_4', ylabel='count'>

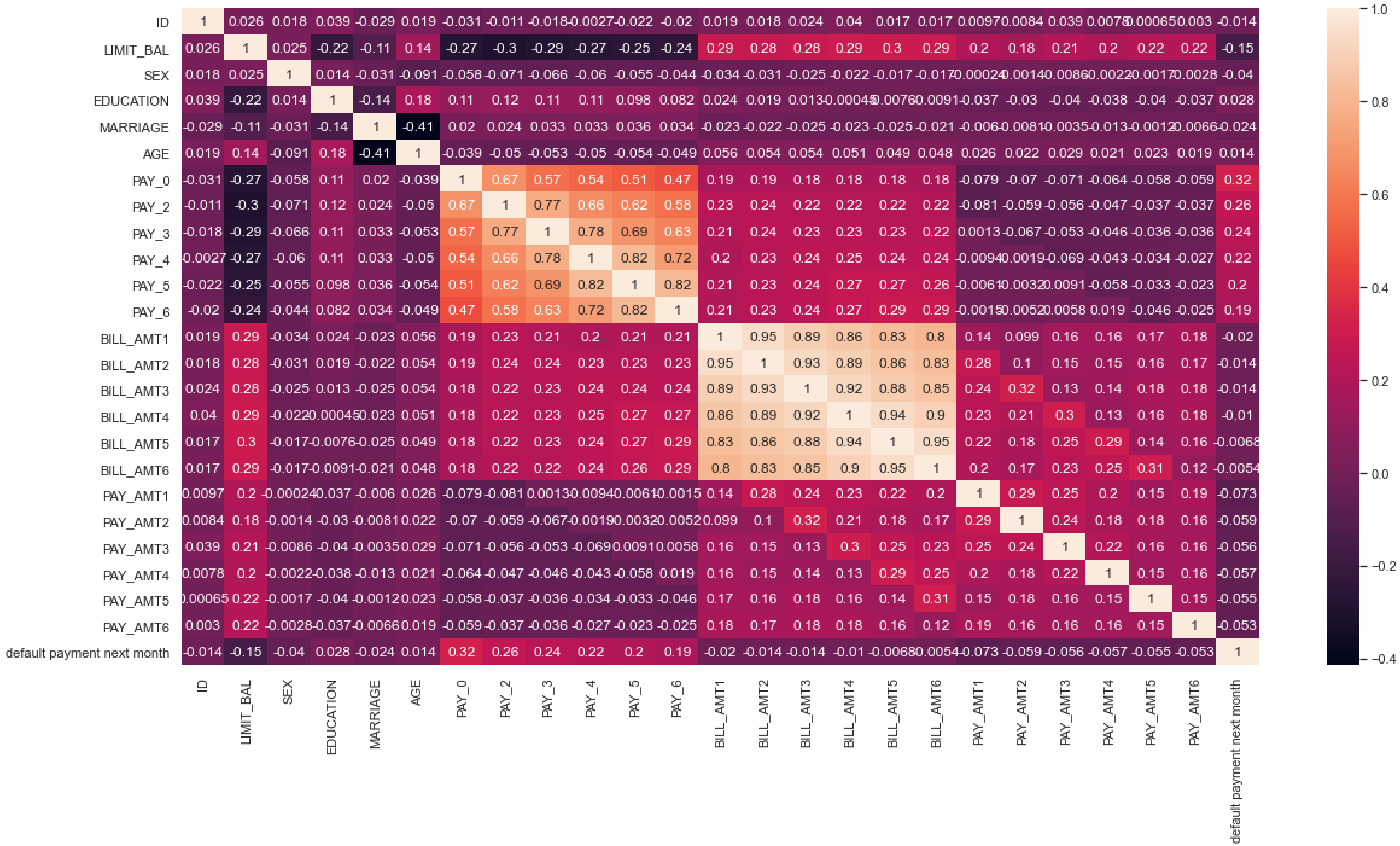


No much difference from PAY_3 plot.

In [45]:
```python
Data  = df
plt.figure(figsize=(20,10))
#sns.heatmap(data.corr())

sns.heatmap(data.corr(), annot = True)
```

Out[45]: <AxesSubplot:>



The figure above is the result of a correlation matrix with all the columns in the dataset. There are two parts to be considerd: 1. features correlation with the target variable, 2.highly correlated BILL*ATM(1–6)s, and PAY*(0–6)s.

SUMMARY

The following Features: 'LIMIT_BAL' 'SEX' 'EDUCATION' 'MARRIAGE' 'AGE' played significant roles towards building a profile of the customers most likely to default using techniques such as univariate and bivariate analysis.

In [ ]:

In [ ]: