* * * * * * * * * * * * * * * * *Assignment no 1* * * * * * * * * * * * * * *

Name - Kiran Shete

Roll no - 23577

In [1]:

```
#!pip install -U featuretools
```

Importing all Libaries

In [2]:

```
import pandas as pd
import numpy as np
import featuretools as ft
```

Ignore warnings from pandas

In [3]:

```
import warnings
warnings.filterwarnings('ignore')
```

Read in the data

In [4]:

```
clients = pd.read_csv('data/clients.csv', parse_dates = ['joined'])
loans = pd.read_csv('data/loans.csv', parse_dates = ['loan_start', 'loan_en
payments = pd.read_csv('data/payments.csv', parse_dates = ['payment_date'])
```

In [5]:

```
clients.head()
```

Out[5]:

	client_id	joined	income	credit_score
0	46109	2002-04-16	172677	527
1	49545	2007-11-14	104564	770
2	41480	2013-03-11	122607	585
3	46180	2001-11-06	43851	562
4	25707	2006-10-06	211422	621

Sample of loans Table

In [6]:
```
loans.sample(10)
```

Out[6]:

	client_id	loan_type	loan_amount	repaid	loan_id	loan_start	loan_end	rate
329	44387	cash	765	0	10080	2000-07-05	2003-03-28	4.59
353	26695	home	3915	1	10947	2008-12-08	2011-04-17	0.32
87	25707	credit	8267	0	10104	2010-08-02	2012-05-21	3.18
143	35089	cash	10017	0	11561	2000-05-24	2002-06-15	1.89
105	39505	home	12596	0	10861	2006-04-26	2008-01-05	0.60
36	49545	other	14074	1	11215	2013-04-23	2014-11-16	1.22
230	32961	cash	2630	0	10906	2012-04-21	2014-06-14	3.98
420	49624	credit	9296	1	10714	2003-04-28	2005-05-04	3.44
264	44601	cash	6636	1	11246	2009-12-07	2011-12-31	2.83
101	39505	cash	9600	0	10966	2014-06-27	2017-02-15	0.25

In [7]:
```
payments.sample(10)
```

Out[7]:

	loan_id	payment_amount	payment_date	missed
563	10363	368	2014-08-21	0
1350	10271	1459	2009-10-04	0
3413	10330	1992	2002-04-27	1
1792	11669	337	2014-07-15	1
1352	10271	1500	2009-12-14	1
580	10205	924	2010-09-25	1
597	11211	1690	2012-05-07	0
2382	11876	452	2003-09-25	1
1104	11539	2874	2006-06-16	0
1622	11443	1789	2010-07-04	0

Create a month column

In [8]:
```
clients['join_month'] = clients['joined'].dt.month
```

Create a log of income column

In [9]:
```
clients['log_income'] = np.log(clients['income'])

clients.head()
```

Out[9]:

	client_id	joined	income	credit_score	join_month	log_income
0	46109	2002-04-16	172677	527	4	12.059178
1	49545	2007-11-14	104564	770	11	11.557555
2	41480	2013-03-11	122607	585	3	11.716739
3	46180	2001-11-06	43851	562	11	10.688553
4	25707	2006-10-06	211422	621	10	12.261611

Groupby client id and calculate max, min previous loan size

In [10]:
```
stats = loans.groupby('client_id')['loan_amount'].agg([ 'max', 'min'])
stats.columns = [ 'min_loan_amount', 'max_loan_amount']
stats.head()
```

Out[10]:

	min_loan_amount	max_loan_amount
client_id		
25707	13913	1212
26326	13464	1164
26695	14865	2389
26945	14593	653
29841	14837	2778

Merge with the clients dataframe

In [11]:
```
clients.merge(stats, left_on = 'client_id', right_index=True, how = 'right'
```

Out[11]:

	client_id	joined	income	credit_score	join_month	log_income	min_loan_amount	n
4	25707	2006-10-06	211422	621	10	12.261611	13913	
10	26326	2004-05-06	227920	633	5	12.336750	13464	
19	26695	2004-08-27	174532	680	8	12.069863	14865	
24	26945	2000-11-26	214516	806	11	12.276140	14593	
13	29841	2002-08-17	38354	523	8	10.554614	14837	
6	32726	2006-05-01	235705	730	5	12.370336	14802	
15	32885	2002-05-13	58955	642	5	10.984530	14162	
12	32961	2009-04-07	230341	714	4	12.347316	14784	
7	35089	2010-03-01	131176	771	3	11.784295	13194	
8	35214	2003-08-08	95849	696	8	11.470529	14767	

In [ ]: