Project description Project is to prepare a report for a bank's loan division. You'll need to find out if a customer's marital status and number of children have an impact on whether they will default on a loan. The bank already has some data on customers' credit worthiness. Your report will be considered when building a credit score for a potential customer. A credit score is used to evaluate the ability of a potential borrower to repay their loan.

1 Analyzing borrowers' risk of defaulting

to prepare a report for a bank's loan division. We need to find out if a customer's marital status and number of children has an impact on whether they will default on a loan. The bank already has some data on customers' credit worthiness.

Our report will be considered when building a **credit scoring** of a potential customer. A ** credit scoring ** is used to evaluate the ability of a potential borrower to repay their loan.

1.1 Step 1. Open the data file and have a look at the general information.

In [1]:
```
import numpy as np
import pandas as pd
from nltk.stem import SnowballStemmer
english_stemmer = SnowballStemmer('english')
df=pd.read_csv('/datasets/credit_scoring_eng.csv')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21525 entries, 0 to 21524
Data columns (total 12 columns):
children            21525 non-null int64
days_employed       19351 non-null float64
dob_years           21525 non-null int64
education           21525 non-null object
education_id        21525 non-null int64
family_status       21525 non-null object
family_status_id    21525 non-null int64
gender              21525 non-null object
income_type         21525 non-null object
debt                21525 non-null int64
total_income        19351 non-null float64
purpose             21525 non-null object
dtypes: float64(2), int64(5), object(5)
memory usage: 2.0+ MB
```

Success: Thank you for collecting all imports in the first cell!

In [2]:
```
df.head(13)
```

Out[2]:

	children	days_employed	dob_years	education	education_id	family_status	family_status_i
0	1	-8437.673028	42	bachelor's degree	0	married	
1	1	-4024.803754	36	secondary education	1	married	
2	0	-5623.422610	33	Secondary Education	1	married	
3	3	-4124.747207	32	secondary education	1	married	
4	0	340266.072047	53	secondary education	1	civil partnership	
5	0	-926.185831	27	bachelor's degree	0	civil partnership	
6	0	-2879.202052	43	bachelor's degree	0	married	
7	0	-152.779569	50	SECONDARY EDUCATION	1	married	
8	2	-6929.865299	35	BACHELOR'S DEGREE	0	civil partnership	
9	0	-2188.756445	41	secondary education	1	married	
10	2	-4171.483647	36	bachelor's degree	0	married	
11	0	-792.701887	40	secondary education	1	married	
12	0	NaN	65	secondary education	1	civil partnership	

1.2 Conclusion

The data has 12 columns and 21525. In data two columns ['days_employed','total_income'] have missing values. This can be known by seeing the number of non null entries in these column. Also by seeing, head and tails it can be further ensured. Likewise, there are two columns having float64 datatype, five columns with int64, and five columns with 'object' datatype.

Success: Data loading and initial analysis are well done.

1.3 Step 2. Data preprocessing

1.4 Processing missing values

In [3]:
```
df1=df[df['days_employed'].isnull()]
print(df['education'].value_counts())
print('\n')
print(df1['education'].value_counts())
```

```
secondary education    13750
bachelor's degree       4718
SECONDARY EDUCATION      772
Secondary Education      711
some college             668
BACHELOR'S DEGREE        274
Bachelor's Degree        268
primary education        250
Some College              47
SOME COLLEGE              29
PRIMARY EDUCATION         17
Primary Education         15
graduate degree            4
Graduate Degree            1
GRADUATE DEGREE            1
Name: education, dtype: int64


secondary education    1408
bachelor's degree       496
SECONDARY EDUCATION      67
Secondary Education      65
some college             55
Bachelor's Degree        25
BACHELOR'S DEGREE        23
primary education        19
Some College              7
SOME COLLEGE              7
PRIMARY EDUCATION         1
Primary Education         1
Name: education, dtype: int64
```

In [4]:
```
print(df[df['days_employed'].isnull()].head(15))
```

```
     children  days_employed  dob_years            education  education_id  \
12          0            NaN         65  secondary education             1   
26          0            NaN         41  secondary education             1   
29          0            NaN         63  secondary education             1   
41          0            NaN         50  secondary education             1   
55          0            NaN         54  secondary education             1   
65          0            NaN         21  secondary education             1   
67          0            NaN         52    bachelor's degree             0   
72          1            NaN         32    bachelor's degree             0   
82          2            NaN         50    bachelor's degree             0   
83          0            NaN         52  secondary education             1   
90          2            NaN         35    bachelor's degree             0   
94          1            NaN         34    bachelor's degree             0   
96          0            NaN         44  SECONDARY EDUCATION             1   
97          0            NaN         47    bachelor's degree             0   
120         0            NaN         46  secondary education             1   

         family_status  family_status_id gender    income_type  debt  \
12   civil partnership                 1      M        retiree     0   
26             married                 0      M  civil servant     0   
29           unmarried                 4      F        retiree     0   
41             married                 0      F  civil servant     0   
55   civil partnership                 1      F        retiree     1   
65           unmarried                 4      M       business     0   
67             married                 0      F        retiree     0   
72             married                 0      M  civil servant     0   
82             married                 0      F       employee     0   
83             married                 0      M       employee     0   
90             married                 0      F       employee     0   
94   civil partnership                 1      F       business     0   
96             married                 0      F       employee     0   
97             married                 0      F       employee     0   
120            married                 0      F       employee     0   

     total_income                                  purpose  
12            NaN                        to have a wedding  
26            NaN                                education  
29            NaN                    building a real estate  
41            NaN                  second-hand car purchase  
55            NaN                        to have a wedding  
65            NaN  transactions with commercial real estate  
67            NaN       purchase of the house for my family  
72            NaN  transactions with commercial real estate  
82            NaN                                  housing  
83            NaN                                  housing  
90            NaN                     housing transactions  
94            NaN                         having a wedding  
96            NaN                buy residential real estate  
97            NaN                        profile education  
120           NaN                     university education  
```

In [5]:
```
print(df['income_type'].value_counts())
print('\n')
print(df1['income_type'].value_counts())
```

```
employee                       11119
business                        5085
retiree                         3856
civil servant                   1459
entrepreneur                       2
unemployed                         2
student                            1
paternity / maternity leave        1
Name: income_type, dtype: int64


employee         1105
business          508
retiree           413
civil servant     147
entrepreneur        1
Name: income_type, dtype: int64
```

In [6]:
```
df['total_income'].mean()
```

Out[6]:
```
26787.56835465867
```

In [7]:
```
df.groupby('income_type')['total_income'].mean()
```

Out[7]:
```
income_type
business                       32386.793835
civil servant                  27343.729582
employee                       25820.841683
entrepreneur                   79866.103000
paternity / maternity leave     8612.661000
retiree                        21940.394503
student                        15712.260000
unemployed                     21014.360500
Name: total_income, dtype: float64
```

In [8]:
```
df.groupby('family_status')['total_income'].mean()
```

Out[8]:
```
family_status
civil partnership    26694.428597
divorced             27189.354550
married              27041.784689
unmarried            26934.069805
widow / widower      22984.208556
Name: total_income, dtype: float64
```

In [9]:
```
df.groupby('education')['total_income'].mean()
```

Out[9]:
```
education
BACHELOR'S DEGREE      31986.861044
Bachelor's Degree      34431.954823
GRADUATE DEGREE        31771.321000
Graduate Degree        15800.399000
PRIMARY EDUCATION      24571.721938
Primary Education      17721.156643
SECONDARY EDUCATION    24212.151756
SOME COLLEGE           28239.453545
Secondary Education    24590.943454
Some College           25470.138250
bachelor's degree      33137.325707
graduate degree        30047.107000
primary education      21115.023866
secondary education    24616.530030
some college           29307.668763
Name: total_income, dtype: float64
```

In [10]:
```
df=df.dropna()
```

1.5 Conclusion

on analysisng data, missing values appeared in 'days_employed' and'total_income' columns. On analysing missing values, it can be seen that they are random. The propertion of missing values (days_employed and total_income) belonging to all subcategories of each column and total number of costomer belonging to that subcategory are roughly about 10 percents. Looking at overall mean of total_income and average total_income on different subcategories are different. In this condition, rows containing missing values can be deleted as the overall characteristic can expected to remain similar.

Success: Step was done not bad.. But in my opinion it would be better to fill missing values. But your decision is also okay. There is no single solution, this is partly creative work .

1.6 Data type replacement

In [11]:
```
df['total_income']=df['total_income'].astype(int)
df['days_employed']=df['days_employed'].astype(int)
df['days_employed'] = df['days_employed'].abs()
df['children'] = df['children'].abs()
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 19351 entries, 0 to 21524
Data columns (total 12 columns):
children            19351 non-null int64
days_employed       19351 non-null int64
dob_years           19351 non-null int64
education           19351 non-null object
education_id        19351 non-null int64
family_status       19351 non-null object
family_status_id    19351 non-null int64
gender              19351 non-null object
income_type         19351 non-null object
debt                19351 non-null int64
total_income        19351 non-null int64
purpose             19351 non-null object
dtypes: int64(7), object(5)
memory usage: 1.9+ MB
```

1.7 Conclusion

'total_income' and 'days_employed' are converted to integer type from floating type as it reduces data size and ease for eye. Also, number of children and days employed are converted to absolute value as there were some negetive values.

1.8 Processing duplicates

In [12]:
```
duplicateDFRow = df[df.duplicated()]
print(duplicateDFRow)
```

```
Empty DataFrame
Columns: [children, days_employed, dob_years, education, education_id, family_s
tatus, family_status_id, gender, income_type, debt, total_income, purpose]
Index: []
```

In [13]:
```
df['education']= df['education'].str.lower()
df.head(10)
```

Out[13]:

	children	days_employed	dob_years	education	education_id	family_status	family_status_id	g
0	1	8437	42	bachelor's degree	0	married	0	
1	1	4024	36	secondary education	1	married	0	
2	0	5623	33	secondary education	1	married	0	
3	3	4124	32	secondary education	1	married	0	
4	0	340266	53	secondary education	1	civil partnership	1	
5	0	926	27	bachelor's degree	0	civil partnership	1	
6	0	2879	43	bachelor's degree	0	married	0	
7	0	152	50	secondary education	1	married	0	
8	2	6929	35	bachelor's degree	0	civil partnership	1	
9	0	2188	41	secondary education	1	married	0	

In [14]:
```
print(df['education'].value_counts())
```

```
secondary education    13693
bachelor's degree       4716
some college             675
primary education        261
graduate degree            6
Name: education, dtype: int64
```

1.9 Conclusion

There was not any duplicated row but the string on 'education' column was written in different cases making different category for same level of study. This problem is solved by converting all string to lowercase in that column.

1.10 Categorizing Data

In [15]:
```
print(df.groupby('children')['debt'].sum().sort_values())
```

```
children
5      0
4      3
20     8
3     22
2    177
1    409
0    952
Name: debt, dtype: int64
```

In [16]:
```
matrix=pd.pivot_table(df , index=['children'], values=['dob_years', 'debt','educa
                    aggfunc={'dob_years': np.median,'debt':[np.median,np.mean],'e
print(matrix)
```

```
         days_employed      debt        dob_years education_id        \
                  mean      mean median    median         mean median
children
0         92518.577498  0.074902    0.0      48.0     0.832415    1.0
1         23454.310691  0.093230    0.0      38.0     0.790745    1.0
2          5493.903296  0.095624    0.0      35.0     0.791464    1.0
3          9337.850340  0.074830    0.0      35.5     0.819728    1.0
4         13862.558824  0.088235    0.0      34.5     0.794118    1.0
5          1432.000000  0.000000    0.0      36.0     1.250000    1.0
20        39623.134328  0.119403    0.0      41.0     0.865672    1.0

         family_status_id        total_income
                     mean median         mean
children
0                1.121558    0.0  26421.911487
1                0.811033    0.0  27368.122179
2                0.445705    0.0  27495.850891
3                0.391156    0.0  29322.153061
4                0.470588    0.0  27289.323529
5                0.125000    0.0  27268.250000
20               0.656716    0.0  26994.820896
```

In [17]:
```
df['children']=df["children"].replace(20 , 2)
#""" '20' could be resulted from typos, so it should be either '2' or '0'"""
## for 20, more of parameters means  are close to '2' compared to '0'.
```

In [18]:
```
debt_children=df.groupby('children')['debt'].sum().sort_values()
print(debt_children)
```

```
children
5      0
4      3
3     22
2    185
1    409
0    952
Name: debt, dtype: int64
```

In [19]:
```
total_debt_children=df['children'].value_counts().sort_values()
print(total_debt_children)
```

```
5        8
4       34
3      294
2     1918
1     4387
0    12710
Name: children, dtype: int64
```

In [20]:
```
repayment_kids= (total_debt_children-debt_children)/total_debt_children*100
print(repayment_kids)
```

```
5    100.000000
4     91.176471
3     92.517007
2     90.354536
1     90.677000
0     92.509835
dtype: float64
```

1.11 Conclusion

Type *Markdown* and LaTeX: α^2

1.12 Step 3. Answer these questions

- Is there a relation between having kids and repaying a loan on time?

From the above result, it can be clearly seen that having more kids increase the probability of paying loan on time.

Success: Correct.

In [21]:
```
pivot_family_status=pd.pivot_table(df , index=['family_status'], values=['debt'],
                        aggfunc={'debt':np.mean})
print(pivot_family_status.sort_values('debt'))
```

```
                       debt
family_status
widow / widower    0.064740
divorced           0.070175
married            0.075922
civil partnership  0.090763
unmarried          0.100594
```

1.13 Conclusion

Type *Markdown* and LaTeX: α^2

- Is there a relation between marital status and repaying a loan on time?

From the above result, the co-relation between 'family status' and the average defaulted debt on that category. So, it can be cocluded that the widow/widower have higher chance of repaying loan on time and unmarried have lower repaying rate.

In [22]:
```
print(df['total_income'].min())
print(df['total_income'].max())
print(df['total_income'].mean())
```

```
3306
362496
26787.071262467056
```

In [23]:
```
low_income=df[df['total_income']< 20000]
low_income.shape
```

Out[23]:
```
(7369, 12)
```

In [24]:
```
def income_group(row):

    total_income = row['total_income']

    if total_income <= 40000:
        return 'low'


    if (total_income >= 40000 and total_income <= 60000):
        return 'medium'
    if (total_income >= 60000):
        return 'high'

df['income_group'] = df.apply(income_group, axis=1)
df.head()
```

Out[24]:

	children	days_employed	dob_years	education	education_id	family_status	family_status_id	g
0	1	8437	42	bachelor's degree	0	married	0	
1	1	4024	36	secondary education	1	married	0	
2	0	5623	33	secondary education	1	married	0	
3	3	4124	32	secondary education	1	married	0	
4	0	340266	53	secondary education	1	civil partnership	1	

In [25]:
```
pivot_income_group=pd.pivot_table(df , index=['income_group'], values=['debt'],
                        aggfunc={'debt':np.mean})
print(pivot_income_group.sort_values('debt'))
```

```
                  debt
income_group
high          0.056548
medium        0.072897
low           0.083258
```

1.14 Conclusion

Type *Markdown* and LaTeX: α^2

- Is there a relation between income level and repaying a loan on time?

From the result, it can be seen that lower income group has higher default debt. So, there is co-relation between income and repaying loan on time.

In [29]:

```
#Split the sentences to lists of words.
df['category'] = df['purpose'].str.split()

# Make sure we see the full column.
pd.set_option('display.max_colwidth', -1)
df['stemmed']=df['category'].apply(lambda x: [english_stemmer.stem(y) for y in x]

# Stem every word.
df = df.drop(columns=['category']) # Get rid of the unstemmed column.
def debt_purpose(row):
    purpose = row['stemmed']


    for query in purpose:
        for word in query.split(" "):
            stemmed_word = english_stemmer.stem(word)
            if 'real' in stemmed_word:
                return 'Housing'

            if 'hous' in stemmed_word:
                return  'Housing'

            if 'properti' in stemmed_word:
                return  'Housing'

            if 'car' in stemmed_word:
                return  'car'

            if'wed' in stemmed_word:
                return  'Wedding'

            if 'educ' in stemmed_word:
                return  'Education'


df['debt_purpose'] = df.apply(debt_purpose, axis=1)
df.tail(15)
```

Out[29]:

	children	days_employed	dob_years	education	education_id	family_status	family_status_i
21509	0	362161	59	bachelor's degree	0	married	
21511	0	612	29	bachelor's degree	0	civil partnership	
21512	0	165	26	bachelor's degree	0	unmarried	
21513	0	1166	35	secondary education	1	married	
21514	0	280	27	some college	2	unmarried	
21515	1	467	28	secondary education	1	married	

	children	days_employed	dob_years	education	education_id	family_status	family_status_i
21516	0	914	42	bachelor's degree	0	married	
21517	0	404	42	bachelor's degree	0	civil partnership	
21518	0	373995	59	secondary education	1	married	
21519	1	2351	37	graduate degree	4	divorced	
21520	1	4529	43	secondary education	1	civil partnership	
21521	0	343937	67	secondary education	1	married	
21522	1	2113	38	secondary education	1	civil partnership	
21523	3	3112	38	secondary education	1	married	
21524	2	1984	40	secondary education	1	married	

In [30]:
```
pivot_purpose=pd.pivot_table(df , index=['debt_purpose'], values=['debt'],
                    aggfunc={'debt':np.mean})
print(pivot_purpose.sort_values('debt'))
```

```
                  debt
debt_purpose
Housing       0.073273
Wedding       0.075274
Education     0.092493
car           0.094175
```

Conclusion

Type *Markdown* and LaTeX: α^2

2 How do different loan purposes affect on-time repayment of the loan?

From this result more chances of repayment is on housing, property and real state related loan. Then, the repayment probability on time is on 'wedding' related loans followed by 'education'and 'car' related debt respectively.

Conclusion

overall, the repayment probability on time depends on various factor, namely: having kids, family_status, income level and debt purpose.

Step 4. General conclusion

In general, the data has some missing value, which were evenly distributed in all categories in each row. Missing value are random in nature. Since, missing value were quantitavive values so one possibility would be replace by average value. But, the average value do not concide with average value obtained by grouping subcategories in different columns. So, average value could change the average value of some categories. on the other hand, deleting missing rows does not have severe impact on any categories. Same educational level were written in different format making confusion. so, all values in 'education column' are converted to lowercase. Likewise, there was typos on number of children. '20' was written in some rows which is not realistic. so, proper replacement for that was found by comparing other values related to every value of 'childern'. Lastly, the repayment probability are determined by pivot table for different indices and 'debt' value.

2.1 Project Readiness Checklist

- [x] file open;
- [x] file examined;
- [x] missing values defined;
- [x] missing values are filled;
- [x] an explanation of which missing value types were detected;
- [x] explanation for the possible causes of missing values;
- [x] an explanation of how the blanks are filled;
- [x] replaced the real data type with an integer;
- [x] an explanation of which method is used to change the data type and why;
- [x] duplicates deleted;
- [x] an explanation of which method is used to find and remove duplicates;
- [x] description of the possible reasons for the appearance of duplicates in the data;
- [x] data is categorized;
- [x] an explanation of the principle of data categorization;
- [x] an answer to the question "Is there a relation between having kids and repaying a loan on time?";
- [x] an answer to the question " Is there a relation between marital status and repaying a loan on time?";
- [x] an answer to the question " Is there a relation between income level and repaying a loan on time?";
- [x] an answer to the question " How do different loan purposes affect on-time repayment of the loan?"
- [x] conclusions are present on each stage;
- [x] a general conclusion is made.

2.2 Thank yo very much!!