Case Study: Statistical Test ¶

Warren Fernandes 8940

Abhi Gupta 8944

Vinyas Kulal 8949

Liny Mathew 8950

In [ ]:

```
import pandas as pd
import numpy as np
```

In [3]:

```
application_record = pd.read_csv('application_record.csv')
credit_record = pd.read_csv('credit_record.csv')
```

In [4]:

```
application_record
```

Out[4]:

	ID	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	NAME_INCOME_TYPE	NA
0	5008804	M	Y	Y	0	427500.0	Working	
1	5008805	M	Y	Y	0	427500.0	Working	
2	5008806	M	Y	Y	0	112500.0	Working	
3	5008808	F	N	Y	0	270000.0	Commercial associate	
4	5008809	F	N	Y	0	270000.0	Commercial associate	
...	...	...	...	...	...	...	...	
438552	6840104	M	N	Y	0	135000.0	Pensioner	
438553	6840222	F	N	N	0	103500.0	Working	
438554	6841878	F	N	N	0	54000.0	Commercial associate	
438555	6842765	F	N	Y	0	72000.0	Pensioner	
438556	6842885	F	N	Y	0	121500.0	Working	

438557 rows × 18 columns

In [5]:

```
credit_record
```

Out[5]:

	ID	MONTHS_BALANCE	STATUS
0	5001711	0	X
1	5001711	-1	0
2	5001711	-2	0
3	5001711	-3	0
4	5001712	0	C
...	...	...	...
1048570	5150487	-25	C
1048571	5150487	-26	C
1048572	5150487	-27	C
1048573	5150487	-28	C
1048574	5150487	-29	C

1048575 rows × 3 columns

In [6]:

```
# Replace X,C values with 0 as they are identified as Good clients
credit_record.replace(['X','C'], 0,inplace=True)
```

In [7]:

```
credit_record.STATUS = pd.to_numeric(credit_record.STATUS)
```

In [8]:

```
# Searching for customers who have at least one late month
drop_ls = []
for i in range(len(credit_record)):
    if credit_record.STATUS[i] != 0:
        drop_ls.append(credit_record.ID[i])
```

In [9]:

```
len(drop_ls)
```

Out[9]:

```
14194
```

In [10]:

```
# Changing the STATUS of any client with at least one late month to 1
for i in range(len(credit_record)):
        if credit_record.ID[i] in drop_ls:
            credit_record.STATUS[i] = 1
```

In [11]:

```
credit_record.STATUS.value_counts()
```

Out[11]:

```
0    904764
1    143811
Name: STATUS, dtype: int64
```

In [12]:

```
credit_record.drop_duplicates(inplace=True)
credit_record
```

Out[12]:

	ID	MONTHS_BALANCE	STATUS
0	5001711	0	0
1	5001711	-1	0
2	5001711	-2	0
3	5001711	-3	0
4	5001712	0	0
...	...	...	...
1048570	5150487	-25	0
1048571	5150487	-26	0
1048572	5150487	-27	0
1048573	5150487	-28	0
1048574	5150487	-29	0

1048575 rows × 3 columns

In [13]:

```
print(f'No. of IDs in application_record = {len(application_record.ID)} No. of IDs in credit_record = {len(credit_record.ID)}'
```

```
No. of IDs in application_record = 438557 No. of IDs in credit_record = 1048575
```

In [14]:

```
dataset = application_record.merge(credit_record, on=['ID'], how='inner')
 # on to choose which column to merger on
 # How to get merge only the intersection between them
```

In [15]:

```
dataset.drop(['ID'],inplace=True,axis=1)
```

In [16]:

```
dataset.duplicated().sum()
```

Out[16]:

```
412393
```

In [17]:

```
dataset.drop_duplicates(inplace=True)
```

In [18]:

```
dataset
```

Out[18]:

	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	NAME_INCOME_TYPE	NAME_EDUC
0	M	Y	Y	0	427500.0	Working	Hig
1	M	Y	Y	0	427500.0	Working	Hig
2	M	Y	Y	0	427500.0	Working	Hig
3	M	Y	Y	0	427500.0	Working	Hig
4	M	Y	Y	0	427500.0	Working	Hig
...	...	...	...	...	...	...	
777710	M	N	Y	0	112500.0	Working	Seconda
777711	M	N	Y	0	112500.0	Working	Seconda
777712	M	N	Y	0	112500.0	Working	Seconda
777713	M	N	Y	0	112500.0	Working	Seconda
777714	M	N	Y	0	112500.0	Working	Seconda

365322 rows × 19 columns

In [19]:

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 365322 entries, 0 to 777714
Data columns (total 19 columns):
 #   Column               Non-Null Count   Dtype  
---  ------               --------------   -----  
 0   CODE_GENDER          365322 non-null  object 
 1   FLAG_OWN_CAR         365322 non-null  object 
 2   FLAG_OWN_REALTY      365322 non-null  object 
 3   CNT_CHILDREN         365322 non-null  int64  
 4   AMT_INCOME_TOTAL     365322 non-null  float64
 5   NAME_INCOME_TYPE     365322 non-null  object 
 6   NAME_EDUCATION_TYPE  365322 non-null  object 
 7   NAME_FAMILY_STATUS   365322 non-null  object 
 8   NAME_HOUSING_TYPE    365322 non-null  object 
 9   DAYS_BIRTH           365322 non-null  int64  
 10  DAYS_EMPLOYED        365322 non-null  int64  
 11  FLAG_MOBIL           365322 non-null  int64  
 12  FLAG_WORK_PHONE      365322 non-null  int64  
 13  FLAG_PHONE           365322 non-null  int64  
 14  FLAG_EMAIL           365322 non-null  int64  
 15  OCCUPATION_TYPE      252192 non-null  object 
 16  CNT_FAM_MEMBERS      365322 non-null  float64
 17  MONTHS_BALANCE       365322 non-null  int64  
 18  STATUS               365322 non-null  int64  
dtypes: float64(2), int64(9), object(8)
memory usage: 55.7+ MB
```

In [20]:

```
dataset.describe()
```

Out[20]:

	CNT_CHILDREN	AMT_INCOME_TOTAL	DAYS_BIRTH	DAYS_EMPLOYED	FLAG_MOBIL	FLAG_WORK_PHONE	FLAG_PHONE	FLAG
count	365322.000000	3.653220e+05	365322.000000	365322.000000	365322.0	365322.000000	365322.000000	365322
mean	0.425742	1.848982e+05	-16161.482656	60776.306365	1.0	0.221878	0.294214	0
std	0.768540	1.017316e+05	4144.182785	139028.719425	0.0	0.415510	0.455689	0
min	0.000000	2.700000e+04	-25152.000000	-15713.000000	1.0	0.000000	0.000000	0
25%	0.000000	1.170000e+05	-19614.000000	-3208.000000	1.0	0.000000	0.000000	0
50%	0.000000	1.575000e+05	-15849.000000	-1566.000000	1.0	0.000000	0.000000	0
75%	1.000000	2.250000e+05	-12676.000000	-378.000000	1.0	0.000000	1.000000	0
max	19.000000	1.575000e+06	-7489.000000	365243.000000	1.0	1.000000	1.000000	1

In [21]:

```
dataset.isna().sum()
```

Out[21]:

```
CODE_GENDER                0
FLAG_OWN_CAR               0
FLAG_OWN_REALTY            0
CNT_CHILDREN               0
AMT_INCOME_TOTAL           0
NAME_INCOME_TYPE           0
NAME_EDUCATION_TYPE        0
NAME_FAMILY_STATUS         0
NAME_HOUSING_TYPE          0
DAYS_BIRTH                 0
DAYS_EMPLOYED              0
FLAG_MOBIL                 0
FLAG_WORK_PHONE            0
FLAG_PHONE                 0
FLAG_EMAIL                 0
OCCUPATION_TYPE       113130
CNT_FAM_MEMBERS            0
MONTHS_BALANCE             0
STATUS                     0
dtype: int64
```

In [22]:

```
dataset.isna().sum().sum()
```

Out[22]:

```
113130
```

In [23]:

```
dataset.OCCUPATION_TYPE
```

Out[23]:

```
0               NaN
1               NaN
2               NaN
3               NaN
4               NaN
            ...
777710     Laborers
777711     Laborers
777712     Laborers
777713     Laborers
777714     Laborers
Name: OCCUPATION_TYPE, Length: 365322, dtype: object
```

In [24]:

```
dataset.OCCUPATION_TYPE.value_counts()
```

Out[24]:

```
Laborers                 62839
Core staff               34175
Sales staff              33786
Managers                 31066
Drivers                  23349
High skill tech staff    14459
Medicine staff           11937
Accountants              11926
Security staff            6851
Cooking staff             6663
Cleaning staff            5201
Private service staff     2989
Low-skill Laborers        2000
Secretaries               1523
Waiters/barmen staff      1272
HR staff                   973
IT staff                   617
Realty agents              566
Name: OCCUPATION_TYPE, dtype: int64
```

In [25]:

```
dataset.OCCUPATION_TYPE.replace(np.nan, 'Other', inplace = True)
```

In [26]:

```
dataset.OCCUPATION_TYPE.value_counts()
```

Out[26]:

```
Other                    113130
Laborers                  62839
Core staff                34175
Sales staff               33786
Managers                  31066
Drivers                   23349
High skill tech staff     14459
Medicine staff            11937
Accountants               11926
Security staff             6851
Cooking staff              6663
Cleaning staff             5201
Private service staff      2989
Low-skill Laborers         2000
Secretaries                1523
Waiters/barmen staff       1272
HR staff                    973
IT staff                    617
Realty agents               566
Name: OCCUPATION_TYPE, dtype: int64
```

In [28]:

```
dataset
```

Out[28]:

	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	NAME_INCOME_TYPE	NAME_EDUC
0	M	Y	Y	0	427500.0	Working	Hig
1	M	Y	Y	0	427500.0	Working	Hig
2	M	Y	Y	0	427500.0	Working	Hig
3	M	Y	Y	0	427500.0	Working	Hig
4	M	Y	Y	0	427500.0	Working	Hig
...	...	...	...	...	...	...	
777710	M	N	Y	0	112500.0	Working	Seconda
777711	M	N	Y	0	112500.0	Working	Seconda
777712	M	N	Y	0	112500.0	Working	Seconda
777713	M	N	Y	0	112500.0	Working	Seconda
777714	M	N	Y	0	112500.0	Working	Seconda

365322 rows × 19 columns

In [ ]:

```
dataset.to_csv('credit.csv')
```

One Sample

Test for Proportion

In [89]:

```
sample = dataset.sample(frac=0.10)
```

In [91]:

```
sample.describe()
```

Out[91]:

	CNT_CHILDREN	AMT_INCOME_TOTAL	DAYS_BIRTH	DAYS_EMPLOYED	FLAG_MOBIL	FLAG_WORK_PHONE	FLAG_PHONE	FLAG_I
count	36532.000000	3.653200e+04	36532.000000	36532.000000	36532.0	36532.000000	36532.000000	36532.0
mean	0.429131	1.844514e+05	-16186.627724	61935.026443	1.0	0.221422	0.294810	0.0
std	0.769898	1.019580e+05	4154.943376	140002.017208	0.0	0.415210	0.455964	0.2
min	0.000000	2.700000e+04	-25152.000000	-15713.000000	1.0	0.000000	0.000000	0.0
25%	0.000000	1.170000e+05	-19661.000000	-3174.000000	1.0	0.000000	0.000000	0.0
50%	0.000000	1.575000e+05	-15849.000000	-1555.000000	1.0	0.000000	0.000000	0.0
75%	1.000000	2.250000e+05	-12705.000000	-356.000000	1.0	0.000000	1.000000	0.0
max	19.000000	1.575000e+06	-7757.000000	365243.000000	1.0	1.000000	1.000000	1.0

In [93]:

```
sample.CODE_GENDER.value_counts()
```

Out[93]:

```
F    23937
M    12595
Name: CODE_GENDER, dtype: int64
```

In [101]:

```
proportions_ztest(count=23937, nobs=36532, value=0.66)
```

Out[101]:

```
(-1.9166880588315112, 0.05527757319676388)
```

In [102]:

```
dataset['CODE_GENDER'].value_counts(normalize=True)
```

Out[102]:

```
F    0.655737
M    0.344263
Name: CODE_GENDER, dtype: float64
```

Test for mean

In [103]:

```
from statsmodels.stats.weightstats import ztest
```

In [124]:

```
ztest(x1=sample['AMT_INCOME_TOTAL'], value=190000)
```

Out[124]:

```
(-10.401568432043778, 2.4388536106918665e-25)
```

In [125]:

```
dataset['AMT_INCOME_TOTAL'].mean()
```

Out[125]:

```
184898.23890157178
```

Two Sample

In [156]:

```
sample1 = dataset.sample(frac=0.10)
sample2 = dataset.sample(frac=0.10)
```

In [157]:

```
sample1
```

Out[157]:

	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	NAME_INCOME_TYPE	NAME_EDUC
183834	M	N	Y	0	135000.0	Working	Secondar
98641	M	Y	N	1	157500.0	Commercial associate	Secondar
573812	M	Y	Y	0	157500.0	Working	Secondar
577461	F	Y	Y	2	90000.0	Commercial associate	Secondar
383968	F	Y	Y	0	337500.0	Working	Secondar
...	...	...	...	...	...	...	
381732	F	N	Y	0	90000.0	Working	Inco
34201	F	Y	Y	0	112500.0	Pensioner	Hig
515880	F	N	Y	0	72000.0	Pensioner	Secondar
211624	M	N	Y	0	112500.0	Commercial associate	Secondar
294057	M	N	N	3	58500.0	Working	Secondar

36532 rows × 19 columns

In [158]:

```
sample2
```

Out[158]:

	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	NAME_INCOME_TYPE	NAME_EDUCATIO
281	M	Y	Y	2	450000.0	Working	Higher e
724	M	N	N	0	135000.0	Commercial associate	Secondary / se
776	M	Y	N	1	450000.0	Working	Higher e
221	F	N	Y	0	135000.0	Working	Secondary / se
349	M	Y	Y	0	270000.0	Working	Secondary / se
...	...	...	...	...	...	...	
323	F	N	N	0	180000.0	Working	Higher e
185	F	N	Y	1	117000.0	Working	Secondary / se
643	F	N	Y	3	67500.0	Commercial associate	Secondary / se
972	M	N	Y	2	207000.0	Commercial associate	Secondary / se
662	M	Y	Y	0	540000.0	Working	Secondary / se

32 rows × 19 columns

In [159]:

```
sample1['CODE_GENDER'].value_counts()
```

Out[159]:

```
F    23976
M    12556
Name: CODE_GENDER, dtype: int64
```

In [160]:

```
sample2['CODE_GENDER'].value_counts()
```

Out[160]:

```
F    23955
M    12577
Name: CODE_GENDER, dtype: int64
```

Difference of Two Proportions

In [161]:

```
count = np.array([23976,23955])
nobs = np.array([36532,36532])
proportions_ztest(count, nobs)
```

Out[161]:

```
(0.16354625601354528, 0.8700883563502826)
```

In [162]:

```
sample1['CODE_GENDER'].value_counts(normalize=True)
```

Out[162]:

```
F    0.656301
M    0.343699
Name: CODE_GENDER, dtype: float64
```

In [163]:

```
sample2['CODE_GENDER'].value_counts(normalize=True)
```

Out[163]:

```
F    0.655726
M    0.344274
Name: CODE_GENDER, dtype: float64
```

In [165]:

```
sample1['FLAG_OWN_CAR'].value_counts()
```

Out[165]:

```
N    22717
Y    13815
Name: FLAG_OWN_CAR, dtype: int64
```

In [166]:

```
sample2['FLAG_OWN_CAR'].value_counts()
```

Out[166]:

```
N    22526
Y    14006
Name: FLAG_OWN_CAR, dtype: int64
```

In [168]:

```
count = np.array([22717,22526])
nobs = np.array([36532,36532])
proportions_ztest(count, nobs, value=0.01)
```

Out[168]:

```
(-1.3281186463712324, 0.18413891442018981)
```

Difference of Two means

In [169]:

```
ztest(x1=sample1['AMT_INCOME_TOTAL'], x2=sample2['AMT_INCOME_TOTAL'], value=0)
```

Out[169]:

```
(-1.2696213576394317, 0.2042195378968562)
```

In [170]:

```
sample1['AMT_INCOME_TOTAL'].mean()
```

Out[170]:

```
184719.69859301436
```

In [171]:

```
sample2['AMT_INCOME_TOTAL'].mean()
```

Out[171]:

```
185672.07594711485
```

One Sample, Two Measures

Chisquare Test for independence

In [172]:

```
from statsmodels.stats.proportion import proportions_chisquare
```

In [176]:

```
np.array([23976,23955])
nobs = np.array([36532,36532])
proportions_chisquare(count, nobs)
```

Out[176]:

```
(2.117610336688899,
 0.1456135874098347,
 (array([[22717., 13815.],
         [22526., 14006.]]),
  array([[22621.5, 13910.5],
         [22621.5, 13910.5]])))
```

In [179]:

```
count = np.array([22717,22526])
nobs = np.array([36532,36532])
proportions_chisquare(count, nobs)
```

Out[179]:

```
(2.117610336688899,
 0.1456135874098347,
 (array([[22717., 13815.],
         [22526., 14006.]]),
  array([[22621.5, 13910.5],
         [22621.5, 13910.5]])))
```

Regression Analysis

In [180]:

```
from statsmodels.api import OLS
```

In [181]:

```
mod = OLS(dataset['DAYS_BIRTH'], dataset['AMT_INCOME_TOTAL'])
```

In [182]:

```
res = mod.fit()
```

In [183]:

```
print(res.summary())
```

```
                            OLS Regression Results
=======================================================================================
Dep. Variable:             DAYS_BIRTH   R-squared (uncentered):                   0.703
Model:                            OLS   Adj. R-squared (uncentered):              0.703
Method:                 Least Squares   F-statistic:                          8.668e+05
Date:                Tue, 07 Mar 2023   Prob (F-statistic):                        0.00
Time:                        18:02:42   Log-Likelihood:                     -3.8481e+06
No. Observations:              365322   AIC:                                  7.696e+06
Df Residuals:                  365321   BIC:                                  7.696e+06
Df Model:                           1
Covariance Type:            nonrobust
====================================================================================
                       coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------
AMT_INCOME_TOTAL    -0.0663   7.12e-05   -931.013      0.000      -0.066      -0.066
==============================================================================
Omnibus:                   132324.664   Durbin-Watson:                   0.042
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          1329668.249
Skew:                           1.457   Prob(JB):                         0.00
Kurtosis:                      11.881   Cond. No.                         1.00
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Difference of means

In [211]:

```
ztest(x1=sample1['CNT_FAM_MEMBERS'], x2=sample2['CNT_CHILDREN'], value=1.8)
```

Out[211]:

```
(-5.180909056271185, 2.208071348369464e-07)
```

In [ ]: