UseCase_Customer Loan Prediction_Naive Bayes

August 13, 2018

0.1 Use Case - Customer Dataset : Naive Bayes Classifier

The ultimate aim of the usecase is to predict loan approval or denial status of the loan application by building the machine learning model using Naive Bayes Classifier. The customer rating dataset contains the following columns:

'APPLICATION.ID', 'DSA.ID', 'DEALER.ID', 'QUEUE.ID', 'CURRENT.STAGE', 'MARITAL.STATUS', 'GENDER', 'AGE', 'EDUCATION', 'RESIDENCE.TYPE', 'CITY', 'STATE', 'ZIP.CODE', 'EMPLOY.CONSTITUTION', 'PAN.STATUS', 'APPLICATION.SCORE', 'APPROVED.AMOUNT', 'APPLIED.AMOUNT', 'LOAN.TENOR', 'OWN.HOUSE.TYPE', 'PRIMARY.ASSET.CTG', 'PRIMARY.ASSET.MAKE', 'PRIMARY.ASSET.MODELNO', 'VOTER_ID', 'DRIVING_LICENSE', 'AADHAAR', 'PAN', 'BANK_PASSBOOK', 'APPLICATION.STATUS'

```
In [154]: import pandas as pd
          from sklearn.naive_bayes import GaussianNB
          from sklearn.metrics import classification_report
          from sklearn.model_selection import train_test_split
          import matplotlib.pyplot as plt

          df=pd.read_excel('customer_dataset.xlsx')
```

```
In [4]: df.apply(lambda x: [x.unique()])
```

```
Out[4]: APPLICATION.ID          [[27497000024, 25556001005, 27220000249, 27067...
        DSA.ID                  [[JB02005, RD02622, SK02345, DN30900, AS22782,...
        DEALER.ID               [[27497, 25556, 27220, 27067, 26189, 27793, 26...
        QUEUE.ID                     [[Straight  Through Process, Under.Writer]]
        CURRENT.STAGE           [[PD_DE, DCLN, APRV, SRNV, INV_GNR, LOS_DISB, ...
        MARITAL.STATUS                                    [[Single, Married]]
        GENDER                                               [[Male, Female]]
        AGE                     [[30, 38, 52, 57, 43, 28, 33, 23, 49, 21, 41, ...
        EDUCATION               [[GRADUATE, OTHERS, UNDER GRADUATE, POST-GRADU...
        RESIDENCE.TYPE          [[OWNED-BUNGLOW, PARENT OWNED-HOUSE, OWNED-ROW...
        CITY                    [[UDHANA, BHOPAL, RAIPUR, JAMNAGAR, HYDERABAD,...
        STATE                   [[GUJARAT, MADHYA PRADESH, CHHATTISGARH, TELAN...
        ZIP.CODE                [[394210, 462001, 492001, 361001, 492006, 5000...
        EMPLOY.CONSTITUTION     [[SELF-EMPLOYED, SALARIED, PARTNERSHIP, PRIVAT...
        PAN.STATUS                 [[Pan Not Submitted, ERROR, EXIST, NOT_FOUND]]
        APPLICATION.SCORE       [[69.0, 108.0, 143.0, 60.0, 62.0, 92.0, 3.0, 1...
```

```
APPROVED.AMOUNT          [[32321, 47000, 30000, 40000, 20000, 45000, 35...
APPLIED.AMOUNT           [[28000, 47000, 30000, 49000, 45000, 17900, 13...
LOAN.TENOR               [[10, 12, 18, 24, 20, 9, 14, 120, 15, 96, 240,...
OWN.HOUSE.TYPE           [[Self Owned, Parent Owned, Spouse Owned, Chil...
PRIMARY.ASSET.CTG        [[REF-FF HOME, TELEVISION, AIR CONDITIONER, HO...
PRIMARY.ASSET.MAKE       [[SAMSUNG, SONY, ELECTROLUX, TARGET, LG, INTEX...
PRIMARY.ASSET.MODELNO    [[RT30K3723S8/HL, KLV-29P423D, AIR CONDITIONER...
VOTER_ID                                                        [[F, T]]
DRIVING_LICENSE                                                 [[F, T]]
AADHAAR                                                         [[T, F]]
PAN                                                             [[F, T]]
BANK_PASSBOOK                                                   [[F, T]]
APPLICATION.STATUS                                [[Declined, Approved]]
dtype: object
```

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7759 entries, 0 to 7758
Data columns (total 29 columns):
APPLICATION.ID           7759 non-null int64
DSA.ID                   7759 non-null object
DEALER.ID                7759 non-null int64
QUEUE.ID                 7759 non-null object
CURRENT.STAGE            7759 non-null object
MARITAL.STATUS           7759 non-null object
GENDER                   7759 non-null object
AGE                      7759 non-null int64
EDUCATION                7759 non-null object
RESIDENCE.TYPE           7759 non-null object
CITY                     7759 non-null object
STATE                    7759 non-null object
ZIP.CODE                 7759 non-null int64
EMPLOY.CONSTITUTION      7759 non-null object
PAN.STATUS               7759 non-null object
APPLICATION.SCORE        7759 non-null float64
APPROVED.AMOUNT          7759 non-null int64
APPLIED.AMOUNT           7759 non-null int64
LOAN.TENOR               7759 non-null int64
OWN.HOUSE.TYPE           7759 non-null object
PRIMARY.ASSET.CTG        7759 non-null object
PRIMARY.ASSET.MAKE       7759 non-null object
PRIMARY.ASSET.MODELNO    7759 non-null object
VOTER_ID                 7759 non-null object
DRIVING_LICENSE          7759 non-null object
AADHAAR                  7759 non-null object
PAN                      7759 non-null object
BANK_PASSBOOK            7759 non-null object
```

```
APPLICATION.STATUS       7759 non-null object
dtypes: float64(1), int64(7), object(21)
memory usage: 1.7+ MB
```

```
In [6]: df.isnull().sum()
```

```
Out[6]: APPLICATION.ID           0
        DSA.ID                   0
        DEALER.ID                0
        QUEUE.ID                 0
        CURRENT.STAGE            0
        MARITAL.STATUS           0
        GENDER                   0
        AGE                      0
        EDUCATION                0
        RESIDENCE.TYPE           0
        CITY                     0
        STATE                    0
        ZIP.CODE                 0
        EMPLOY.CONSTITUTION      0
        PAN.STATUS               0
        APPLICATION.SCORE        0
        APPROVED.AMOUNT          0
        APPLIED.AMOUNT           0
        LOAN.TENOR               0
        OWN.HOUSE.TYPE           0
        PRIMARY.ASSET.CTG        0
        PRIMARY.ASSET.MAKE       0
        PRIMARY.ASSET.MODELNO    0
        VOTER_ID                 0
        DRIVING_LICENSE          0
        AADHAAR                  0
        PAN                      0
        BANK_PASSBOOK            0
        APPLICATION.STATUS       0
        dtype: int64
```

```
In [7]: df.isna().sum()
```

```
Out[7]: APPLICATION.ID           0
        DSA.ID                   0
        DEALER.ID                0
        QUEUE.ID                 0
        CURRENT.STAGE            0
        MARITAL.STATUS           0
        GENDER                   0
        AGE                      0
        EDUCATION                0
```

```
RESIDENCE.TYPE           0
CITY                     0
STATE                    0
ZIP.CODE                 0
EMPLOY.CONSTITUTION      0
PAN.STATUS               0
APPLICATION.SCORE        0
APPROVED.AMOUNT          0
APPLIED.AMOUNT           0
LOAN.TENOR               0
OWN.HOUSE.TYPE           0
PRIMARY.ASSET.CTG        0
PRIMARY.ASSET.MAKE       0
PRIMARY.ASSET.MODELNO    0
VOTER_ID                 0
DRIVING_LICENSE          0
AADHAAR                  0
PAN                      0
BANK_PASSBOOK            0
APPLICATION.STATUS       0
dtype: int64
```

In [8]:
```
df.describe()
```

Out[8]:

	APPLICATION.ID	DEALER.ID	AGE	ZIP.CODE	\
count	7.759000e+03	7759.000000	7759.000000	7759.000000	
mean	2.653510e+10	26535.095115	35.693259	520190.974481	
std	1.001756e+09	1001.756529	9.391776	134499.130446	
min	2.505000e+10	25050.000000	12.000000	110002.000000	
25%	2.564550e+10	25645.500000	28.000000	452001.000000	
50%	2.638500e+10	26385.000000	34.000000	520012.000000	
75%	2.749400e+10	27494.000000	42.000000	625016.000000	
max	2.877500e+10	28775.000000	67.000000	843325.000000	

	APPLICATION.SCORE	APPROVED.AMOUNT	APPLIED.AMOUNT	LOAN.TENOR
count	7759.000000	7759.000000	7759.000000	7759.000000
mean	46.343204	32321.418997	34211.627014	13.129656
std	34.835119	9988.441681	18952.163116	12.682123
min	-45.000000	7001.000000	10.000000	0.000000
25%	13.000000	32000.000000	21000.000000	10.000000
50%	42.000000	32321.000000	30000.000000	12.000000
75%	73.000000	32321.000000	43870.000000	12.000000
max	160.000000	300000.000000	800000.000000	360.000000

In [9]:
```
df.info()
```

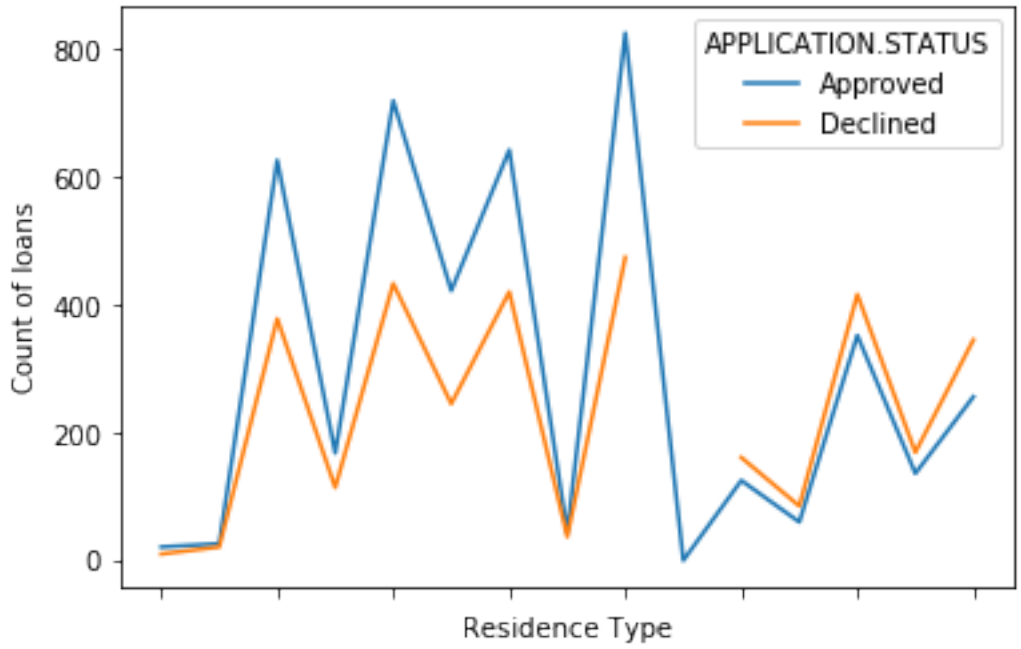
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7759 entries, 0 to 7758
Data columns (total 29 columns):
```

```
APPLICATION.ID           7759 non-null int64
DSA.ID                   7759 non-null object
DEALER.ID                7759 non-null int64
QUEUE.ID                 7759 non-null object
CURRENT.STAGE            7759 non-null object
MARITAL.STATUS           7759 non-null object
GENDER                   7759 non-null object
AGE                      7759 non-null int64
EDUCATION                7759 non-null object
RESIDENCE.TYPE           7759 non-null object
CITY                     7759 non-null object
STATE                    7759 non-null object
ZIP.CODE                 7759 non-null int64
EMPLOY.CONSTITUTION      7759 non-null object
PAN.STATUS               7759 non-null object
APPLICATION.SCORE        7759 non-null float64
APPROVED.AMOUNT          7759 non-null int64
APPLIED.AMOUNT           7759 non-null int64
LOAN.TENOR               7759 non-null int64
OWN.HOUSE.TYPE           7759 non-null object
PRIMARY.ASSET.CTG        7759 non-null object
PRIMARY.ASSET.MAKE       7759 non-null object
PRIMARY.ASSET.MODELNO    7759 non-null object
VOTER_ID                 7759 non-null object
DRIVING_LICENSE          7759 non-null object
AADHAAR                  7759 non-null object
PAN                      7759 non-null object
BANK_PASSBOOK            7759 non-null object
APPLICATION.STATUS       7759 non-null object
dtypes: float64(1), int64(7), object(21)
memory usage: 1.7+ MB
```

0.1.1 Data Exploration

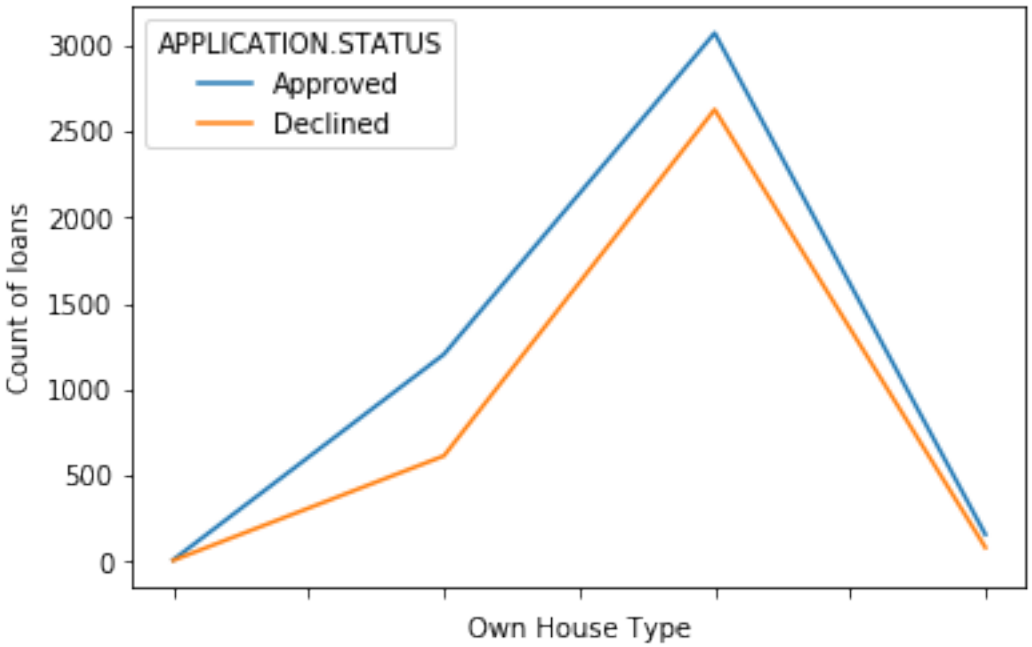
```
In [189]: pd.pivot_table(data=df,index='RESIDENCE.TYPE',values='APPLICATION.ID',columns='APPLICA
          plt.xlabel('Residence Type')
          plt.ylabel('Count of loans')

Out[189]: Text(0,0.5,'Count of loans')
```



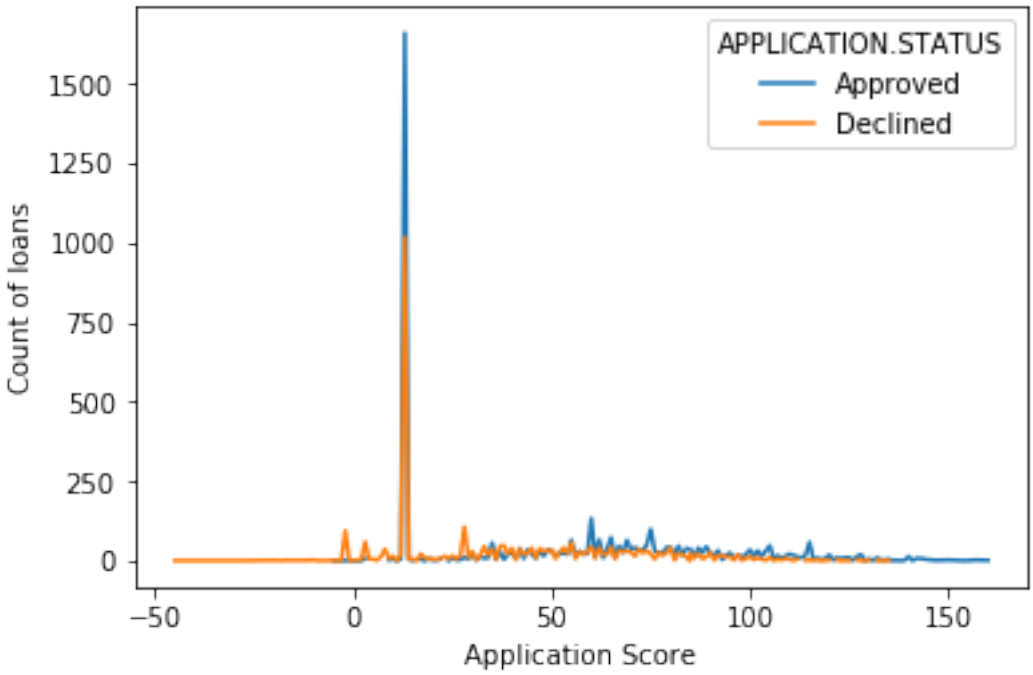
In [197]:
```
pd.pivot_table(data=df,index='OWN.HOUSE.TYPE',values='APPLICATION.ID',columns='APPLICA
plt.xlabel('Own House Type')
labels = df['OWN.HOUSE.TYPE'].unique()
plt.ylabel('Count of loans')
```

Out[197]:
```
Text(0,0.5,'Count of loans')
```



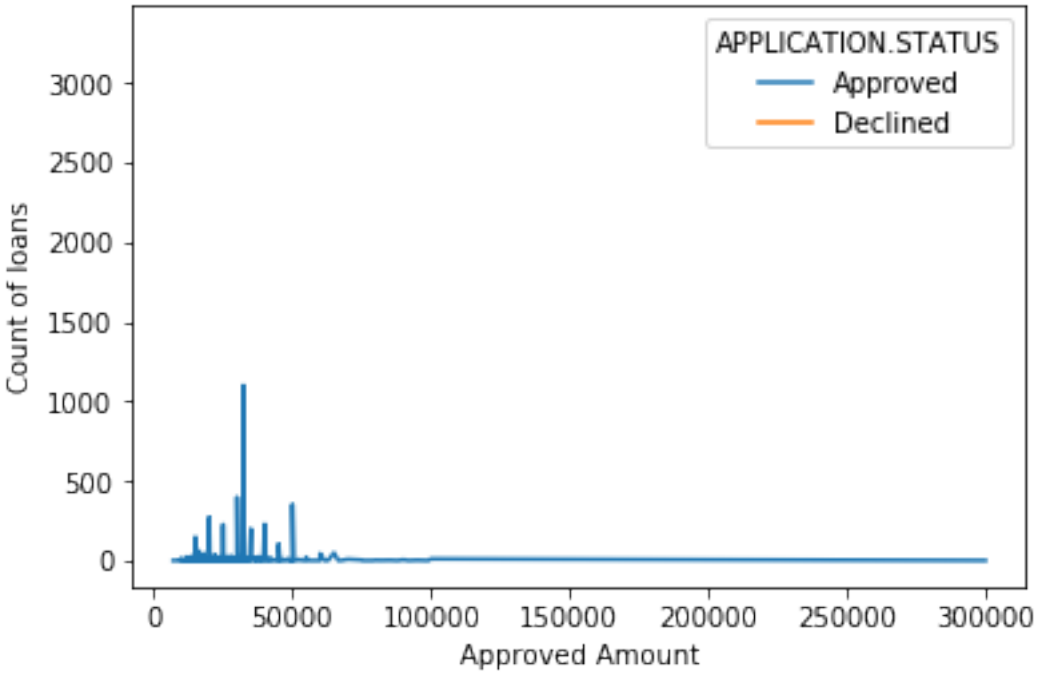
In [205]:
```
pd.pivot_table(data=df,index='APPLICATION.SCORE',values='APPLICATION.ID',columns='APPL
plt.xlabel('Application Score')
labels = df['OWN.HOUSE.TYPE'].unique()
plt.ylabel('Count of loans')
```

Out[205]:
```
Text(0,0.5,'Count of loans')
```



In [200]:
```
pd.pivot_table(data=df,index='APPROVED.AMOUNT',values='APPLICATION.ID',columns='APPLIC
plt.xlabel('Approved Amount')
labels = df['OWN.HOUSE.TYPE'].unique()
plt.ylabel('Count of loans')
```

Out[200]:
```
Text(0,0.5,'Count of loans')
```



In [19]:
```
## Bank Passbook - Not an important variable
df_temp=pd.pivot_table(data=df,index='BANK_PASSBOOK',values='APPLICATION.ID',columns='A
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[19]:

APPLICATION.STATUS	Approved	Declined	All	percent
BANK_PASSBOOK				
F	4264	3153	7417	57.49
T	173	169	342	50.58
All	4437	3322	7759	57.19

In [22]:
```
## PAN is also not a great differentiator
df_temp=pd.pivot_table(data=df,index='PAN',values='APPLICATION.ID',columns='APPLICATION
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[22]:

APPLICATION.STATUS	Approved	Declined	All	percent
AADHAAR				
F	1375	967	2342	58.71
T	3062	2355	5417	56.53
All	4437	3322	7759	57.19

In [24]:
```
## Aadhaar is also not a good differentiator
df_temp=pd.pivot_table(data=df,index='AADHAAR',values='APPLICATION.ID',columns='APPLICA
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[24]:

APPLICATION.STATUS	Approved	Declined	All	percent
AADHAAR				
F	1375	967	2342	58.71
T	3062	2355	5417	56.53
All	4437	3322	7759	57.19

In [25]:
```
## Driving License is also not a good differentiator
df_temp=pd.pivot_table(data=df,index='DRIVING_LICENSE',values='APPLICATION.ID',columns=
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[25]:

APPLICATION.STATUS	Approved	Declined	All	percent
DRIVING_LICENSE				
F	3926	2960	6886	57.01
T	511	362	873	58.53
All	4437	3322	7759	57.19

In [26]:
```
## VoterID is also not a good differentiator
df_temp=pd.pivot_table(data=df,index='VOTER_ID',values='APPLICATION.ID',columns='APPLIC
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[26]:

APPLICATION.STATUS	Approved	Declined	All	percent
VOTER_ID				
F	3774	2829	6603	57.16
T	663	493	1156	57.35
All	4437	3322	7759	57.19

In [27]:
```
## somewhat a good indicator.probably defining a term self owned or not would give more
df_temp=pd.pivot_table(data=df,index='OWN.HOUSE.TYPE',values='APPLICATION.ID',columns='
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[27]:

APPLICATION.STATUS	Approved	Declined	All	percent
OWN.HOUSE.TYPE				
Children Owned	7	4	11	63.64
Parent Owned	1203	612	1815	66.28
Self Owned	3072	2627	5699	53.90
Spouse Owned	155	79	234	66.24
All	4437	3322	7759	57.19

In [33]:
```
## PAN Status helps, if not found then low chance of loan
df_temp=pd.pivot_table(data=df,index='PAN.STATUS',values='APPLICATION.ID',columns='APPL
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[33]:

APPLICATION.STATUS	Approved	Declined	All	percent
PAN.STATUS				
ERROR	1043	799	1842	56.62

```
EXIST                    1348      1040  2388    56.45
NOT_FOUND                  84       114   198    42.42
Pan Not Submitted        1962      1369  3331    58.90
All                      4437      3322  7759    57.19
```

In [35]:
```
# Not much insights to gather here.
df_temp=pd.pivot_table(data=df,index='EMPLOY.CONSTITUTION',values='APPLICATION.ID',colu
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[35]:
```
APPLICATION.STATUS       Approved  Declined   All  percent
EMPLOY.CONSTITUTION
PARTNERSHIP                    10         7    17    58.82
PRIVATE LIMITED COMPANY        11        10    21    52.38
SALARIED                     1983      1426  3409    58.17
SELF-EMPLOYED                2428      1877  4305    56.40
TRUST                           5         2     7    71.43
All                          4437      3322  7759    57.19
```

In [37]:
```
df_temp=pd.pivot_table(data=df,index='STATE',values='APPLICATION.ID',columns='APPLICATI
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[37]:
```
APPLICATION.STATUS  Approved  Declined   All  percent
STATE
ANDHRA PRADESH         339.0     363.0   702    48.29
ASSAM                   23.0      16.0    39    58.97
BIHAR                    1.0       NaN     1   100.00
CHHATTISGARH           140.0      76.0   216    64.81
DELHI                  160.0     118.0   278    57.55
GUJARAT                791.0     444.0  1235    64.05
HARYANA                  7.0      19.0    26    26.92
KARNATAKA              130.0     173.0   303    42.90
MADHYA PRADESH         540.0     282.0   822    65.69
MAHARASHTRA            129.0     144.0   273    47.25
TAMIL NADU            1325.0    1032.0  2357    56.22
TELANGANA              332.0     403.0   735    45.17
UTTAR PRADESH           12.0      16.0    28    42.86
WEST BENGAL            508.0     236.0   744    68.28
All                   4437.0    3322.0  7759    57.19
```

In [42]:
```
# Rented Owners have low loan approval rates.
df_temp=pd.pivot_table(data=df,index='RESIDENCE.TYPE',values='APPLICATION.ID',columns='
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[42]:
```
APPLICATION.STATUS            Approved  Declined   All  percent
RESIDENCE.TYPE
COMPANY PROVIDED-FLAT             22.0      11.0    33    66.67
```

```
COMPANY PROVIDED-HOUSE            27.0      22.0    49    55.10
OWNED-BUNGLOW                    627.0     379.0  1006    62.33
OWNED-CHAWL                      169.0     115.0   284    59.51
OWNED-FLAT                       720.0     434.0  1154    62.39
OWNED-PENTHOUSE                  423.0     246.0   669    63.23
OWNED-ROWHOUSE                   643.0     421.0  1064    60.43
PARENT OWNED-FLAT                 45.0      38.0    83    54.22
PARENT OWNED-HOUSE               826.0     475.0  1301    63.49
RENTED-BACHELOR ACCOMODATION       1.0       NaN     1   100.00
RENTED-BUNGLOW                   126.0     162.0   288    43.75
RENTED-CHAWL                      61.0      86.0   147    41.50
RENTED-FLAT                      353.0     417.0   770    45.84
RENTED-PENTHOUSE                 137.0     170.0   307    44.63
RENTED-ROWHOUSE                  257.0     346.0   603    42.62
All                             4437.0    3322.0  7759    57.19
```

In [43]:
```
# Lower Education levels have low loan approval rates.
df_temp=pd.pivot_table(data=df,index='EDUCATION',values='APPLICATION.ID',columns='APPLI
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[43]:
```
APPLICATION.STATUS  Approved  Declined   All  percent
EDUCATION
DOCTORATE                 8         4    12    66.67
GRADUATE               2505      1752  4257    58.84
OTHERS                  637       590  1227    51.92
POST-GRADUATE           255       164   419    60.86
PROFESSIONAL             18        14    32    56.25
UNDER GRADUATE         1014       798  1812    55.96
All                    4437      3322  7759    57.19
```

In [46]:
```
# married person have high loan approval rates.
df_temp=pd.pivot_table(data=df,index='MARITAL.STATUS',values='APPLICATION.ID',columns='
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[46]:
```
APPLICATION.STATUS  Approved  Declined   All  percent
MARITAL.STATUS
Married                3629      2628  6257    58.00
Single                  808       694  1502    53.79
All                    4437      3322  7759    57.19
```

In [47]:
```
# Gender is not quite intuitive feature
df_temp=pd.pivot_table(data=df,index='GENDER',values='APPLICATION.ID',columns='APPLICAT
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[47]:
```
APPLICATION.STATUS  Approved  Declined   All  percent
GENDER
```

```
Female                    737       533  1270    58.03
Male                     3700      2789  6489    57.02
All                      4437      3322  7759    57.19
```

In [48]:
```
# Gender is not quite intuitive feature
df_temp=pd.pivot_table(data=df,index='CURRENT.STAGE',values='APPLICATION.ID',columns='A
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[48]:
```
APPLICATION.STATUS  Approved  Declined   All  percent
CURRENT.STAGE
APRV                1107.0       NaN  1107   100.00
DCLN                   NaN    2123.0  2123      NaN
INV_GNR              856.0       NaN   856   100.00
LOS_BDE               86.0       NaN    86   100.00
LOS_DISB             729.0       1.0   730    99.86
LOS_ERROR              5.0       1.0     6    83.33
PD_DE               1359.0    1197.0  2556    53.17
SRNV                 295.0       NaN   295   100.00
All                 4437.0    3322.0  7759    57.19
```

In [53]:
```
# Gender is not quite intuitive feature
df_temp=pd.pivot_table(data=df,index='QUEUE.ID',values='APPLICATION.ID',columns='APPLIC
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[53]:
```
APPLICATION.STATUS          Approved  Declined   All  percent
QUEUE.ID
Straight  Through Process       3454      1918  5372    64.30
Under.Writer                     983      1404  2387    41.18
All                             4437      3322  7759    57.19
```

In [52]:
```
# Straight through Process
df_temp=pd.pivot_table(data=df,index='QUEUE.ID',values='APPLICATION.ID',columns='APPLIC
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[52]:
```
APPLICATION.STATUS          Approved  Declined   All  percent
QUEUE.ID
Straight  Through Process       3454      1918  5372    64.30
Under.Writer                     983      1404  2387    41.18
All                             4437      3322  7759    57.19
```

In [54]:
```
### Application score would help
df_temp=pd.pivot_table(data=df,index='APPLICATION.SCORE',values='APPLICATION.ID',column
df_temp['percent']=round(df_temp['Approved']*100/df_temp['All'],2)
df_temp
```

Out[54]:

APPLICATION.SCORE	Approved	Declined	All	percent
-45.0	NaN	1.0	1	NaN
-32.0	NaN	1.0	1	NaN
-27.0	NaN	1.0	1	NaN
-22.0	NaN	2.0	2	NaN
-20.0	NaN	2.0	2	NaN
-18.0	NaN	1.0	1	NaN
-17.0	NaN	2.0	2	NaN
-15.0	1.0	2.0	3	33.33
-12.0	NaN	2.0	2	NaN
-10.0	NaN	3.0	3	NaN
-9.0	1.0	1.0	2	50.00
-7.0	NaN	1.0	1	NaN
-6.0	NaN	2.0	2	NaN
-5.0	1.0	1.0	2	50.00
-3.0	1.0	2.0	3	33.33
-2.0	1.0	95.0	96	1.04
-1.0	2.0	1.0	3	66.67
0.0	1.0	2.0	3	33.33
2.0	1.0	4.0	5	20.00
3.0	8.0	60.0	68	11.76
4.0	NaN	6.0	6	NaN
5.0	2.0	6.0	8	25.00
6.0	NaN	2.0	2	NaN
7.0	2.0	14.0	16	12.50
8.0	NaN	37.0	37	NaN
9.0	3.0	7.0	10	30.00
10.0	6.0	15.0	21	28.57
11.0	1.0	1.0	2	50.00
12.0	7.0	19.0	26	26.92
13.0	1660.0	1018.0	2678	61.99
...	...	...	...	...
121.0	3.0	NaN	3	100.00
122.0	10.0	3.0	13	76.92
123.0	8.0	1.0	9	88.89
124.0	10.0	1.0	11	90.91
125.0	11.0	1.0	12	91.67
126.0	2.0	NaN	2	100.00
127.0	14.0	4.0	18	77.78
128.0	20.0	1.0	21	95.24
129.0	1.0	1.0	2	50.00
130.0	7.0	2.0	9	77.78
131.0	1.0	NaN	1	100.00
132.0	10.0	1.0	11	90.91
133.0	4.0	1.0	5	80.00
134.0	4.0	1.0	5	80.00
135.0	6.0	1.0	7	85.71

```
136.0                    1.0       NaN     1   100.00
137.0                    2.0       1.0     3    66.67
138.0                    1.0       NaN     1   100.00
139.0                    2.0       NaN     2   100.00
140.0                   14.0       NaN    14   100.00
141.0                    1.0       NaN     1   100.00
142.0                    8.0       NaN     8   100.00
143.0                    8.0       1.0     9    88.89
145.0                    4.0       NaN     4   100.00
147.0                    2.0       NaN     2   100.00
150.0                    3.0       NaN     3   100.00
154.0                    1.0       NaN     1   100.00
157.0                    3.0       NaN     3   100.00
160.0                    2.0       NaN     2   100.00
All                   4437.0    3322.0  7759    57.19

[168 rows x 4 columns]
```

0.1.2 Model Preparation

```
In [214]: def featureselection():
              df=pd.read_excel('customer_dataset.xlsx')
              df.drop(['APPLICATION.ID','DSA.ID','DEALER.ID','ZIP.CODE','PRIMARY.ASSET.CTG','PRI
              df=df[['APPLICATION.SCORE','CURRENT.STAGE','MARITAL.STATUS','EDUCATION','RESIDENCE
              df_enc=pd.get_dummies(df,columns=['CURRENT.STAGE','MARITAL.STATUS','EDUCATION','RE
              x=df_enc.drop(['APPLICATION.STATUS_Declined'],axis=1)
              y=df_enc['APPLICATION.STATUS_Declined']
              return x,y

In [215]: x,y=featureselection()

          x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=.2,random_state=30)
          x_train,x_valid,y_train,y_valid=train_test_split(x_train,y_train,test_size=.3,random_s

          model=GaussianNB()
          y_valid_pred=model.fit(x_train,y_train).predict(x_valid)
          print("Validation Dataset")
          print(classification_report(y_valid, y_valid_pred))

          print("Test Dataset")
          y_pred=model.predict(x_test)
          print(classification_report(y_test, y_pred))
```

```
Validation Dataset
              precision    recall  f1-score   support

          0       1.00      0.76      0.86      1043
          1       0.76      1.00      0.86       820
```

```
avg / total       0.89      0.86      0.86      1863

Test Dataset
             precision    recall  f1-score   support

          0       1.00      0.76      0.86       891
          1       0.75      1.00      0.86       661

avg / total       0.89      0.86      0.86      1552
```