Houseloan-data-analysis-using python for simplilearn-

Data Analysis (Western Michigan University)

In [55]:
```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets pre
# You can also write temporary files to /kaggle/temp/, but they won't be saved outsi
```

```
/kaggle/input/house-loan-data-analysis/loan_data.csv
```

In [56]:
```
import pandas as pd
import sklearn
import numpy as np
import matplotlib.pyplot as plt
import os
import warnings
import seaborn as sns
from sklearn.preprocessing import OneHotEncoder
from sklearn.datasets import make_blobs
from sklearn.impute import SimpleImputer
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import StandardScaler
from sklearn.svm import LinearSVC
from sklearn.metrics import roc_auc_score
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import roc_auc_score
from sklearn.calibration import CalibratedClassifierCV
from sklearn.metrics import confusion_matrix
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.linear_model import SGDClassifier
import plotly.offline as py
import plotly.graph_objs as go
from plotly.offline import init_notebook_mode, iplot
from sklearn.model_selection import train_test_split
init_notebook_mode(connected=True)
import cufflinks as cf
cf.go_offline()
import pickle
import gc
import lightgbm as lgb
warnings.filterwarnings('ignore')
%matplotlib inline
```

In [57]:
```
house_loan=pd.read_csv('../input/house-loan-data-analysis/loan_data.csv')
house_loan.describe()
```

Out[57]:

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANI
count	307511.000000	307511.000000	307511.000000	3.075110e+05	3.075110e+05	307499.(
mean	278180.518577	0.080729	0.417052	1.687979e+05	5.990260e+05	27108.5
std	102790.175348	0.272419	0.722121	2.371231e+05	4.024908e+05	14493.7
min	100002.000000	0.000000	0.000000	2.565000e+04	4.500000e+04	1615.5
25%	189145.500000	0.000000	0.000000	1.125000e+05	2.700000e+05	16524.(
50%	278202.000000	0.000000	0.000000	1.471500e+05	5.135310e+05	24903.(
75%	367142.500000	0.000000	1.000000	2.025000e+05	8.086500e+05	34596.(
max	456255.000000	1.000000	19.000000	1.170000e+08	4.050000e+06	258025.5

8 rows × 106 columns

In [58]:

```
house_loan.columns
```

Out[58]:

```
Index(['SK_ID_CURR', 'TARGET', 'NAME_CONTRACT_TYPE', 'CODE_GENDER',
       'FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL',
       'AMT_CREDIT', 'AMT_ANNUITY',
       ...
       'FLAG_DOCUMENT_18', 'FLAG_DOCUMENT_19', 'FLAG_DOCUMENT_20',
       'FLAG_DOCUMENT_21', 'AMT_REQ_CREDIT_BUREAU_HOUR',
       'AMT_REQ_CREDIT_BUREAU_DAY', 'AMT_REQ_CREDIT_BUREAU_WEEK',
       'AMT_REQ_CREDIT_BUREAU_MON', 'AMT_REQ_CREDIT_BUREAU_QRT',
       'AMT_REQ_CREDIT_BUREAU_YEAR'],
      dtype='object', length=122)
```

In [59]:

```
house_loan.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Columns: 122 entries, SK_ID_CURR to AMT_REQ_CREDIT_BUREAU_YEAR
dtypes: float64(65), int64(41), object(16)
memory usage: 286.2+ MB
```

In [60]:

```
house_loan.isnull().sum()
```

Out[60]:

```
SK_ID_CURR                        0
TARGET                            0
NAME_CONTRACT_TYPE                0
CODE_GENDER                       0
FLAG_OWN_CAR                      0
                              ...
AMT_REQ_CREDIT_BUREAU_DAY     41519
AMT_REQ_CREDIT_BUREAU_WEEK    41519
AMT_REQ_CREDIT_BUREAU_MON     41519
AMT_REQ_CREDIT_BUREAU_QRT     41519
AMT_REQ_CREDIT_BUREAU_YEAR    41519
Length: 122, dtype: int64
```

In [61]:

```
house_loan.head()
```

Out[61]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REAI
0	100002	1	Cash loans	M	N	

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REAI
1	100003	0	Cash loans	F	N	
2	100004	0	Revolving loans	M	Y	
3	100006	0	Cash loans	F	N	
4	100007	0	Cash loans	M	N	

5 rows × 122 columns

In [62]:

```
defaulters=(house_loan.TARGET==1).sum()
payers=(house_loan.TARGET==0).sum()
print((defaulters/payers)*100)
```

```
8.781828601345662
```

In [63]:

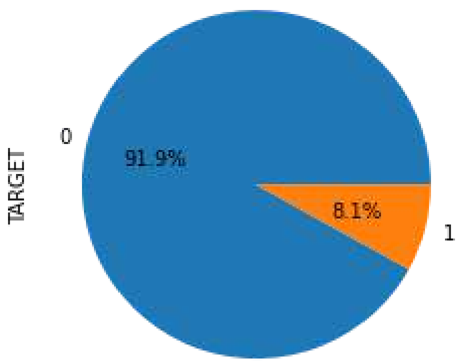
```
without_id=[column for column in house_loan.columns if column!='SK_ID_CURR']

#check for duplicate values
na=house_loan[house_loan.duplicated(subset=without_id,keep=False)]
print("Duplicates are: ",na.shape[0])
```

```
Duplicates are:  0
```

In [64]:

```
house_loan.TARGET.value_counts().plot(kind='pie',autopct='%1.1f%%')
```

Out[64]: <AxesSubplot:ylabel='TARGET'>



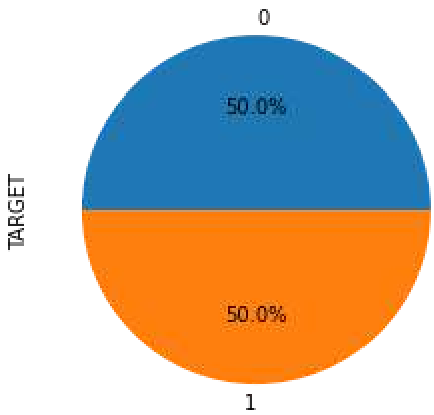
In [65]:

```
import matplotlib as plt
```

In [66]:

```
shuffled_data=house_loan.sample(frac=1,random_state=3)
unpaid_home_loan=shuffled_data.loc[shuffled_data['TARGET']==1]
paid_home_loan=shuffled_data.loc[shuffled_data['TARGET']==0].sample(n=24825,random_s
normalised_home_loan=pd.concat([unpaid_home_loan,paid_home_loan])
normalised_home_loan.TARGET.value_counts().plot(kind='pie',autopct="%1.1f%%")
```

Out[66]: <AxesSubplot:ylabel='TARGET'>



In [67]:
```
import tensorflow as tf
```

In [68]:
```
normalised_home_loan.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 49650 entries, 207339 to 121862
Columns: 122 entries, SK_ID_CURR to AMT_REQ_CREDIT_BUREAU_YEAR
dtypes: float64(65), int64(41), object(16)
memory usage: 46.6+ MB
```

In [69]:
```
normalised_home_loan.head
```

Out[69]:
```
<bound method NDFrame.head of          SK_ID_CURR  TARGET NAME_CONTRACT_TYPE CODE_GEN
DER FLAG_OWN_CAR  \
207339      340318       1         Cash loans           F            N
8756        110186       1         Cash loans           M            Y
230344      366811       1         Cash loans           F            N
178329      306645       1         Cash loans           M            Y
55586       164407       1         Cash loans           M            N
...            ...     ...                ...         ...          ...
130947      251878       0         Cash loans           F            Y
40467       146875       0         Cash loans           F            N
187004      316791       0         Cash loans           M            N
131755      252811       0         Cash loans           F            N
121862      241287       0         Cash loans           M            N

       FLAG_OWN_REALTY  CNT_CHILDREN  AMT_INCOME_TOTAL  AMT_CREDIT  \
207339              N             0          112500.0    405000.0
8756                N             0          135000.0    544491.0
230344              Y             0          112500.0    225000.0
178329              Y             0          157500.0    595273.5
55586               N             0          157500.0    521451.0
...               ...           ...               ...         ...
130947              Y             0          135000.0    770913.0
40467               N             2          360000.0    260640.0
187004              Y             1          180000.0    688500.0
131755              Y             2          202500.0    312840.0
121862              N             0           58500.0    254700.0

        AMT_ANNUITY  ...  FLAG_DOCUMENT_18 FLAG_DOCUMENT_19 FLAG_DOCUMENT_20  \
207339      21969.0  ...                0                0                0
8756        17563.5  ...                0                0                0
230344      17905.5  ...                0                0                0
178329      29083.5  ...                0                0                0
55586       35406.0  ...                0                0                0
...             ...  ...              ...              ...              ...
130947      24997.5  ...                0                0                0
40467       29475.0  ...                0                0                0
187004      22752.0  ...                0                0                0
```

```
131755      18090.0  ...               0                0                0
121862      13446.0  ...               0                0                0

       FLAG_DOCUMENT_21 AMT_REQ_CREDIT_BUREAU_HOUR AMT_REQ_CREDIT_BUREAU_DAY  \
207339                0                        0.0                       0.0
8756                  0                        0.0                       0.0
230344                0                        NaN                       NaN
178329                0                        NaN                       NaN
55586                 0                        0.0                       0.0
...                 ...                        ...                       ...
130947                0                        0.0                       0.0
40467                 0                        0.0                       0.0
187004                0                        0.0                       0.0
131755                0                        0.0                       0.0
121862                0                        0.0                       0.0

        AMT_REQ_CREDIT_BUREAU_WEEK  AMT_REQ_CREDIT_BUREAU_MON  \
207339                         0.0                        0.0
8756                           0.0                        0.0
230344                         NaN                        NaN
178329                         NaN                        NaN
55586                          0.0                        0.0
...                            ...                        ...
130947                         0.0                        1.0
40467                          0.0                        0.0
187004                         0.0                        0.0
131755                         0.0                        0.0
121862                         0.0                        0.0

        AMT_REQ_CREDIT_BUREAU_QRT  AMT_REQ_CREDIT_BUREAU_YEAR
207339                        0.0                         3.0
8756                          0.0                         0.0
230344                        NaN                         NaN
178329                        NaN                         NaN
55586                         0.0                         1.0
...                           ...                         ...
130947                        1.0                         1.0
40467                         0.0                         0.0
187004                        0.0                         0.0
131755                        1.0                         3.0
121862                        0.0                         0.0

[49650 rows x 122 columns]>
```

In [70]:

```
normalised_home_loan.dropna(axis=0)
normalised_home_loan.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 49650 entries, 207339 to 121862
Columns: 122 entries, SK_ID_CURR to AMT_REQ_CREDIT_BUREAU_YEAR
dtypes: float64(65), int64(41), object(16)
memory usage: 46.6+ MB
```

In [71]:

```
normalised_home_loan.isnull().sum()
```

Out[71]:

```
SK_ID_CURR                       0
TARGET                           0
NAME_CONTRACT_TYPE               0
CODE_GENDER                      0
FLAG_OWN_CAR                     0
                              ...
AMT_REQ_CREDIT_BUREAU_DAY     7648
AMT_REQ_CREDIT_BUREAU_WEEK    7648
AMT_REQ_CREDIT_BUREAU_MON     7648
AMT_REQ_CREDIT_BUREAU_QRT     7648
AMT_REQ_CREDIT_BUREAU_YEAR    7648
Length: 122, dtype: int64
```

In [72]:

```
#print(normalised_home_loan.apply())
```

In [73]:

```
print(pd.unique(normalised_home_loan.AMT_REQ_CREDIT_BUREAU_DAY))
print(pd.unique(normalised_home_loan.AMT_REQ_CREDIT_BUREAU_WEEK))
print(pd.unique(normalised_home_loan.AMT_REQ_CREDIT_BUREAU_MON))
print(pd.unique(normalised_home_loan.AMT_REQ_CREDIT_BUREAU_QRT))
print(pd.unique(normalised_home_loan.AMT_REQ_CREDIT_BUREAU_YEAR))
```

```
[ 0. nan  1.  2.  4.  3.  9.]
[ 0. nan  1.  2.  4.  3.  5.  6.]
[ 0. nan  1.  3.  5.  9.  2.  6.  8.  4. 11. 12.  7. 13. 10. 17. 15. 14.
 16. 18. 27.]
[ 0. nan  2.  3.  1.  4.  5.  6. 19.  7.]
[ 3.  0. nan  1.  5.  4.  2.  6.  7.  8.  9. 10. 14. 13. 12. 11. 22. 16.
 23. 17.]
```

In [74]:

```
normalised_home_loan.dropna(axis=0)
```

Out[74]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN
279124	423360	1	Cash loans	M	Y	
216116	350411	1	Cash loans	M	Y	
133687	255050	1	Cash loans	M	Y	
4159	104863	1	Cash loans	M	Y	
208602	341779	1	Cash loans	F	Y	
...	...	...	...	...	...	
108677	226053	0	Cash loans	M	Y	
258603	399273	0	Revolving loans	M	Y	
51880	160079	0	Cash loans	M	Y	
282820	427561	0	Cash loans	F	Y	
207101	340051	0	Revolving loans	F	Y	

1230 rows × 122 columns

In [75]:

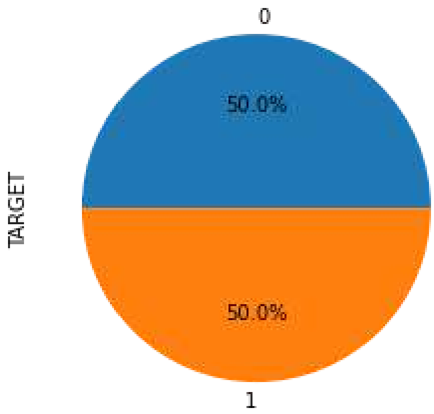
```
print(normalised_home_loan.info())
print(normalised_home_loan.isnull().sum())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 49650 entries, 207339 to 121862
Columns: 122 entries, SK_ID_CURR to AMT_REQ_CREDIT_BUREAU_YEAR
dtypes: float64(65), int64(41), object(16)
memory usage: 46.6+ MB
None
SK_ID_CURR                     0
TARGET                         0
NAME_CONTRACT_TYPE             0
CODE_GENDER                    0
FLAG_OWN_CAR                   0
                            ...
AMT_REQ_CREDIT_BUREAU_DAY   7648
AMT_REQ_CREDIT_BUREAU_WEEK  7648
AMT_REQ_CREDIT_BUREAU_MON   7648
```

```
AMT_REQ_CREDIT_BUREAU_QRT     7648
AMT_REQ_CREDIT_BUREAU_YEAR    7648
Length: 122, dtype: int64
```

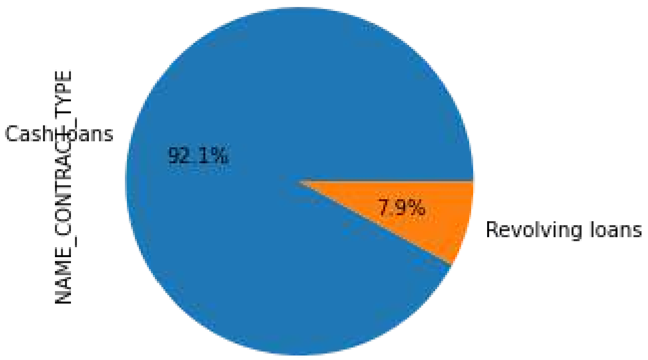
In [76]:
```
normalised_home_loan.TARGET.value_counts().plot(kind='pie',autopct="%1.1f%%")
```

Out[76]: <AxesSubplot:ylabel='TARGET'>



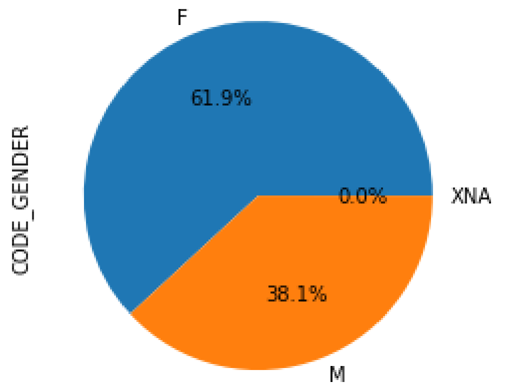
In [77]:
```
normalised_home_loan.NAME_CONTRACT_TYPE.value_counts().plot(kind='pie',autopct="%1.1
#high amount of cash loans
```

Out[77]: <AxesSubplot:ylabel='NAME_CONTRACT_TYPE'>



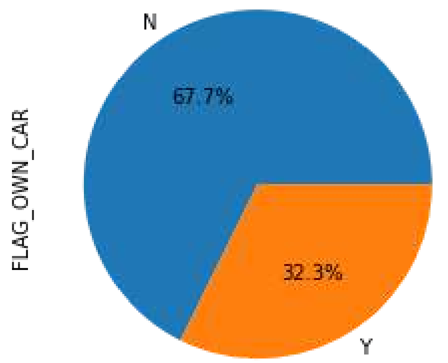
In [78]:
```
normalised_home_loan.CODE_GENDER.value_counts().plot(kind='pie',autopct="%1.1f%%")
#roughly equal amount
```

Out[78]: <AxesSubplot:ylabel='CODE_GENDER'>



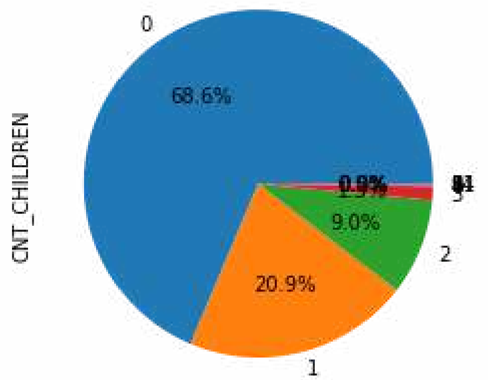
In [79]:
```
normalised_home_loan.FLAG_OWN_CAR.value_counts().plot(kind='pie',autopct="%1.1f%%")
```

Out[79]: <AxesSubplot:ylabel='FLAG_OWN_CAR'>

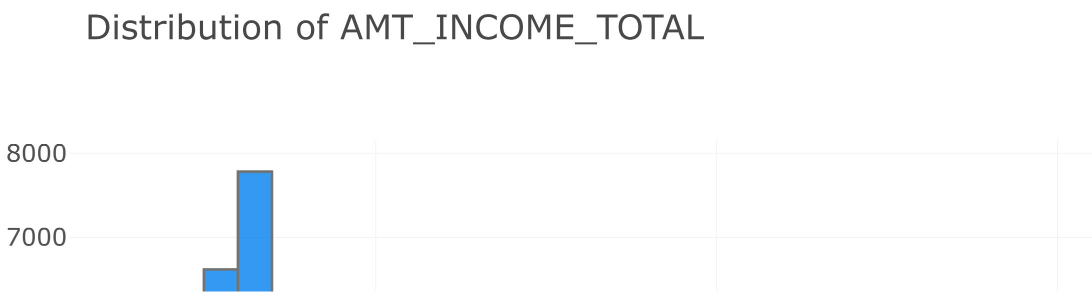


In [80]:
```
normalised_home_loan.CNT_CHILDREN.value_counts().plot(kind='pie',autopct="%1.1f%%")
```

Out[80]: <AxesSubplot:ylabel='CNT_CHILDREN'>



In [81]:
```
#!pip install chart_studio


cf.set_config_file(theme='polar')

normalised_home_loan[normalised_home_loan['AMT_INCOME_TOTAL'] < 2000000]['AMT_INCOME
   xTitle = 'Total Income', yTitle ='Count of applicants',
             title='Distribution of AMT_INCOME_TOTAL')
```



In [82]:
```
(normalised_home_loan[normalised_home_loan['AMT_INCOME_TOTAL']>1000000]['TARGET'].va
```

Out[82]:
```
0    64.864865
1    35.135135
Name: TARGET, dtype: float64
```

In [83]:
```
#print((normalised_home_loan[normalised_home_loan['CNT_CHILDREN']>1]['TARGET'].value
print((normalised_home_loan[normalised_home_loan['CNT_CHILDREN']>2]['TARGET'].value_
print((normalised_home_loan[normalised_home_loan['CNT_CHILDREN']>5]['TARGET'].value_
#as number of children is increasing lone defaulters are increasing
```

```
1    57.047872
0    42.952128
Name: TARGET, dtype: float64
1    81.818182
0    18.181818
Name: TARGET, dtype: float64
```

In [84]:
```
print((normalised_home_loan[normalised_home_loan['FLAG_OWN_CAR']=='N']['TARGET'].val
print((normalised_home_loan[normalised_home_loan['FLAG_OWN_CAR']=='Y']['TARGET'].val

#people with own cars are slighlty more likely to repay back the loan
```

```
1    51.350064
0    48.649936
Name: TARGET, dtype: float64
0    52.823962
1    47.176038
Name: TARGET, dtype: float64
```

In [85]:

```
print((normalised_home_loan[normalised_home_loan['CODE_GENDER']=='M']['TARGET'].valu
print((normalised_home_loan[normalised_home_loan['CODE_GENDER']=='F']['TARGET'].valu

#men more likely to default in payment of loans
```

```
1    56.280372
0    43.719628
Name: TARGET, dtype: float64
0    53.867691
1    46.132309
Name: TARGET, dtype: float64
```

In [86]:

```
print((normalised_home_loan[normalised_home_loan['NAME_CONTRACT_TYPE']=='Cash loans'
print((normalised_home_loan[normalised_home_loan['NAME_CONTRACT_TYPE']=='Revolving l

#cash loans have a higher percent of defaulters
```

```
1    50.802923
0    49.197077
Name: TARGET, dtype: float64
0    59.309995
1    40.690005
Name: TARGET, dtype: float64
```

In [87]:

```
normalised_home_loan=normalised_home_loan.sample(frac=1,random_state=5)
```

In [88]:

```
from sklearn.preprocessing import OrdinalEncoder

ordenc=OrdinalEncoder()
normalised_home_loan['NAME_CONTRACT_TYPE_CODE']=ordenc.fit_transform(normalised_home
print(normalised_home_loan[['NAME_CONTRACT_TYPE','NAME_CONTRACT_TYPE_CODE']].head(20
print(normalised_home_loan['NAME_CONTRACT_TYPE_CODE'].value_counts())
```

```
       NAME_CONTRACT_TYPE  NAME_CONTRACT_TYPE_CODE
302218         Cash loans                      0.0
167526         Cash loans                      0.0
159305         Cash loans                      0.0
275427         Cash loans                      0.0
8837           Cash loans                      0.0
192094         Cash loans                      0.0
235115    Revolving loans                      1.0
79051          Cash loans                      0.0
123267    Revolving loans                      1.0
5517           Cash loans                      0.0
128624         Cash loans                      0.0
187583         Cash loans                      0.0
143193         Cash loans                      0.0
288269         Cash loans                      0.0
44320          Cash loans                      0.0
256898         Cash loans                      0.0
118237         Cash loans                      0.0
5980      Revolving loans                      1.0
96475          Cash loans                      0.0
249976         Cash loans                      0.0
0.0    45708
1.0     3942
Name: NAME_CONTRACT_TYPE_CODE, dtype: int64
```

In [89]:

```
normalised_home_loan['CODE_GENDER_CODE']=ordenc.fit_transform(normalised_home_loan[[
print(normalised_home_loan[['CODE_GENDER','CODE_GENDER_CODE']].head(20))
print(normalised_home_loan['CODE_GENDER_CODE'].value_counts())
```

```
        CODE_GENDER  CODE_GENDER_CODE
302218            M               1.0
167526            F               0.0
159305            M               1.0
275427            F               0.0
8837              M               1.0
192094            M               1.0
235115            F               0.0
79051             F               0.0
123267            M               1.0
5517              F               0.0
128624            M               1.0
187583            F               0.0
143193            M               1.0
288269            F               0.0
44320             F               0.0
256898            F               0.0
118237            F               0.0
5980              M               1.0
96475             F               0.0
249976            F               0.0
0.0    30716
1.0    18932
2.0        2
Name: CODE_GENDER_CODE, dtype: int64
```

In [90]:

```
#2 other values in code_gender
normalised_home_loan.loc[normalised_home_loan['CODE_GENDER_CODE']==2]
```

Out[90]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN
83382	196708	0	Revolving loans	XNA	N	
189640	319880	0	Revolving loans	XNA	Y	

2 rows × 124 columns

In [91]:

```
normalised_home_loan['FLAG_OWN_CAR_CODE']=ordenc.fit_transform(normalised_home_loan[
print(normalised_home_loan[['FLAG_OWN_CAR','FLAG_OWN_CAR_CODE']].head(20))
print(normalised_home_loan['FLAG_OWN_CAR_CODE'].value_counts())
```

```
        FLAG_OWN_CAR  FLAG_OWN_CAR_CODE
302218             N                0.0
167526             N                0.0
159305             N                0.0
275427             N                0.0
8837               N                0.0
192094             N                0.0
235115             N                0.0
79051              N                0.0
123267             N                0.0
5517               N                0.0
128624             N                0.0
187583             N                0.0
143193             N                0.0
288269             Y                1.0
44320              Y                1.0
256898             N                0.0
118237             N                0.0
5980               Y                1.0
96475              N                0.0
249976             N                0.0
0.0    33591
```

```
1.0    16059
Name: FLAG_OWN_CAR_CODE, dtype: int64
```

In [92]:
```
normalised_home_loan['CNT_CHILDREN_CODE']=ordenc.fit_transform(normalised_home_loan[
print(normalised_home_loan[['CNT_CHILDREN_CODE','CNT_CHILDREN']].head(20))
print(normalised_home_loan['CNT_CHILDREN_CODE'].value_counts())
```

```
        CNT_CHILDREN_CODE  CNT_CHILDREN
302218                0.0             0
167526                0.0             0
159305                2.0             2
275427                0.0             0
8837                  0.0             0
192094                0.0             0
235115                0.0             0
79051                 0.0             0
123267                1.0             1
5517                  0.0             0
128624                0.0             0
187583                1.0             1
143193                0.0             0
288269                0.0             0
44320                 0.0             0
256898                0.0             0
118237                2.0             2
5980                  0.0             0
96475                 0.0             0
249976                0.0             0
0.0     34073
1.0     10381
2.0      4444
3.0       642
4.0        89
5.0        10
6.0         6
8.0         2
7.0         1
9.0         1
10.0        1
Name: CNT_CHILDREN_CODE, dtype: int64
```

In [93]:
```
normalised_home_loan=normalised_home_loan.sample(frac=1,random_state=45)
```

In [94]:
```
normalised_home_loan['TARGET'].value_counts()
```

Out[94]:
```
0    24825
1    24825
Name: TARGET, dtype: int64
```

In [95]:
```
y=normalised_home_loan.TARGET
```

In [96]:
```
#y=y.sample(frac=1,random_state=45)
```

In [97]:
```
normalised_home_loan_features=['SK_ID_CURR','NAME_CONTRACT_TYPE_CODE','CNT_CHILDREN_
```

In [98]:
```
from sklearn.model_selection import train_test_split
```

In [99]:
```
X=normalised_home_loan[normalised_home_loan_features]
```

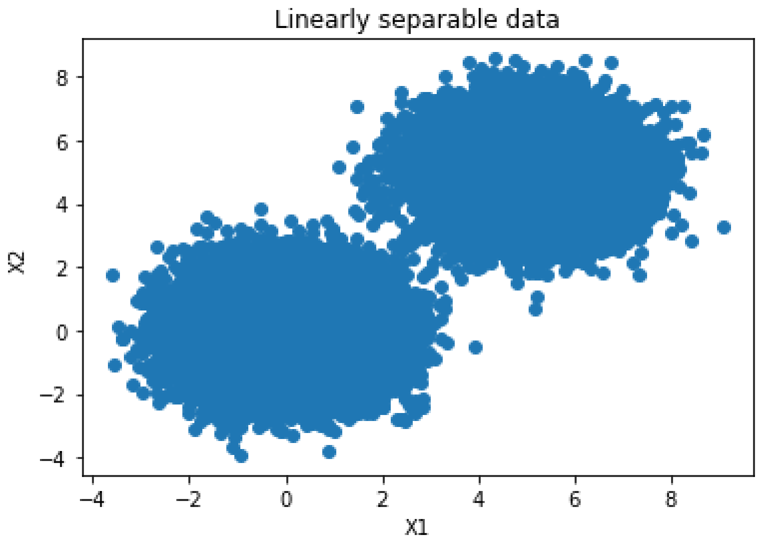
In [100…
```
#X=X.sample(frac=1,random_state=45)
```

In [101…
```
blobs_random_seed = 42
centers = [(0,0), (5,5)]
cluster_std = 1
frac_test_split = 0.33
num_features_for_samples = 2
num_samples_total = 49650

# Generate data
inputs, targets = make_blobs(n_samples = num_samples_total, centers = centers, n_fea

X_train,X_test,y_train,y_test=train_test_split(inputs,targets,test_size=0.33,random_
```

In [102…
```
print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)
```

(33265, 2) (16385, 2) (33265,) (16385,)

In [103…
```
plt.pyplot.scatter(X_train[:,0], X_train[:,1])
plt.pyplot.title('Linearly separable data')
plt.pyplot.xlabel('X1')
plt.pyplot.ylabel('X2')
plt.pyplot.show()
```



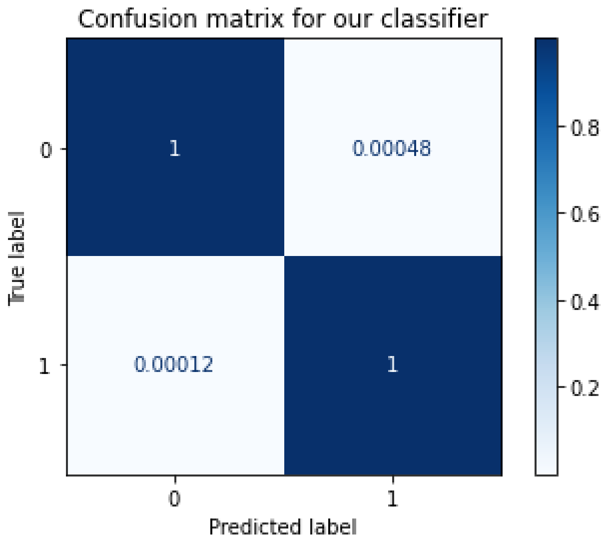
In [104…
```
from sklearn import svm
from sklearn.metrics import plot_confusion_matrix
```

In [105…
```
clf=svm.SVC(kernel='linear')
```

In [106…
```
clf=clf.fit(X_train,y_train)
```

In [107…
```
predictions = clf.predict(X_test)

# Generate confusion matrix
matrix = plot_confusion_matrix(clf, X_test, y_test,
                               cmap=plt.cm.Blues,
```

```
                                normalize='true')
plt.pyplot.title('Confusion matrix for our classifier')
plt.pyplot.show(matrix)
plt.pyplot.show()
```



In [118…

```
from sklearn.metrics import precision_score, recall_score,f1_score
```

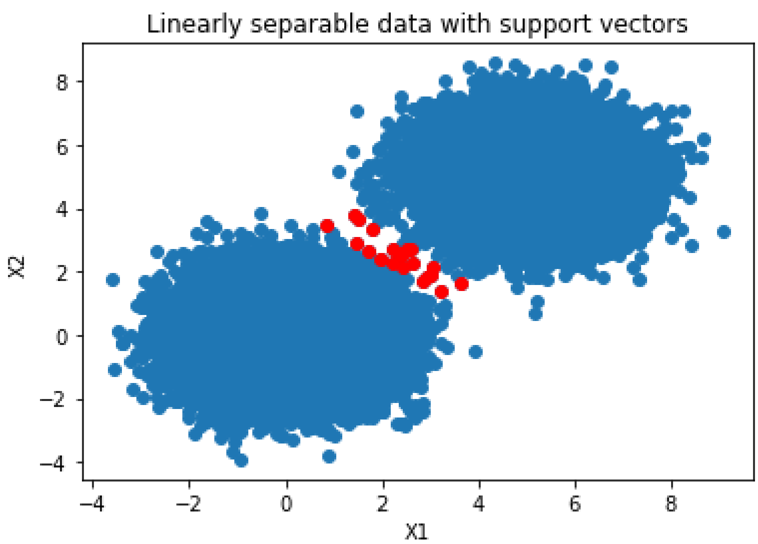
In [119…

```
print(precision_score(y_test, predictions))
print(recall_score(y_test, predictions))
print(f1_score(y_test,predictions,average=None))
```

```
0.9995034140285537
0.9998758072528564
[0.99969993 0.99968958]
```

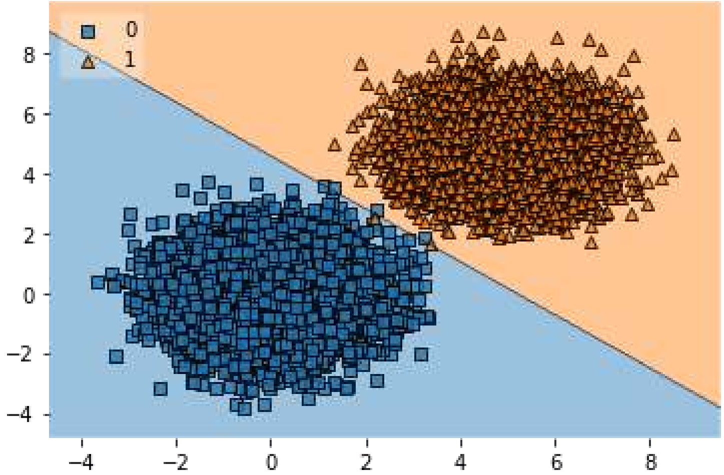
In [108…

```
support_vectors = clf.support_vectors_

# Visualize support vectors
plt.pyplot.scatter(X_train[:,0], X_train[:,1])
plt.pyplot.scatter(support_vectors[:,0], support_vectors[:,1], color='red')
plt.pyplot.title('Linearly separable data with support vectors')
plt.pyplot.xlabel('X1')
plt.pyplot.ylabel('X2')
plt.pyplot.show()
```



In [109…
```
from mlxtend.plotting import plot_decision_regions
```

In [112…
```
plot_decision_regions(X_test, y_test, clf=clf, legend=2)
plt.pyplot.show()
```



In [ ]: