ImbalancedData

May 23, 2022

[1]:
```
# Strategies to deal with imbalanced data
```

[2]:
```
import pandas as pd
```

[3]:
```
bankData = pd.read_csv('https://raw.githubusercontent.com/fenago/datasets/main/
↪bank-full.csv',sep=';')
```

[4]:
```
bankData.sample(5)
```

[4]:

	age	job	marital	education	default	balance	housing	loan	\
27405	42	management	married	tertiary	no	36	no	no	
10459	39	services	married	secondary	no	733	no	no	
16473	45	admin.	married	secondary	no	524	yes	no	
41034	58	retired	married	secondary	no	1227	no	no	
22095	33	management	married	tertiary	no	0	no	no	

	contact	day	month	duration	campaign	pdays	previous	poutcome	y
27405	cellular	21	nov	664	3	-1	0	unknown	yes
10459	unknown	16	jun	83	4	-1	0	unknown	no
16473	cellular	23	jul	808	1	-1	0	unknown	yes
41034	cellular	14	aug	182	2	37	2	failure	no
22095	cellular	21	aug	102	2	-1	0	unknown	no

[5]:
```
from sklearn.preprocessing import RobustScaler
rob_scaler = RobustScaler()
```

[6]:
```
# Converting each of the columns to scaled version
bankData['ageScaled'] = rob_scaler.fit_transform(bankData['age'].values.
↪reshape(-1,1))
bankData['balScaled'] = rob_scaler.fit_transform(bankData['balance'].values.
↪reshape(-1,1))
bankData['durScaled'] = rob_scaler.fit_transform(bankData['duration'].values.
↪reshape(-1,1))
```

[7]:
```
bankData.drop(['age','balance','duration'],axis = 1, inplace=True)
```

[8]:
```
bankData.head()
```

[8]:
```
            job  marital  education default housing loan  contact  day month  \
0    management  married   tertiary      no     yes   no  unknown    5   may
1    technician   single  secondary      no     yes   no  unknown    5   may
2  entrepreneur  married  secondary      no     yes  yes  unknown    5   may
3   blue-collar  married    unknown      no     yes   no  unknown    5   may
4       unknown   single    unknown      no      no   no  unknown    5   may

   campaign  pdays  previous poutcome   y  ageScaled  balScaled  durScaled
0         1     -1         0  unknown  no   1.266667   1.250000   0.375000
1         1     -1         0  unknown  no   0.333333  -0.308997  -0.134259
2         1     -1         0  unknown  no  -0.400000  -0.328909  -0.481481
3         1     -1         0  unknown  no   0.533333   0.780236  -0.407407
4         1     -1         0  unknown  no  -0.400000  -0.329646   0.083333
```

[9]:
```
# Converting all the categorical variables to dummy variables
bankCat = pd.
 ↪get_dummies(bankData[['job','marital','education','default','housing','loan','contact','mon
```

[10]:
```
# Seperating the numerical data
bankNum =␣
 ↪bankData[['ageScaled','balScaled','day','durScaled','campaign','pdays','previous']]
bankNum.shape
```

[10]:
```
(45211, 7)
```

[11]:
```
# Merging with the original data frame
# Preparing the X variables
X = pd.concat([bankCat, bankNum], axis=1)
print(X.shape)
# Preparing the Y variable
Y = bankData['y']
print(Y.shape)
X.head()
```

```
(45211, 51)
(45211,)
```

[11]:
```
   job_admin.  job_blue-collar  job_entrepreneur  job_housemaid  \
0           0                0                 0              0
1           0                0                 0              0
2           0                0                 1              0
3           0                1                 0              0
4           0                0                 0              0

   job_management  job_retired  job_self-employed  job_services  job_student  \
0               1            0                  0             0            0
1               0            0                  0             0            0
```

```
2               0            0                  0            0            0
3               0            0                  0            0            0
4               0            0                  0            0            0

   job_technician  …  poutcome_other  poutcome_success  poutcome_unknown  \
0               0  …               0                 0                 1
1               1  …               0                 0                 1
2               0  …               0                 0                 1
3               0  …               0                 0                 1
4               0  …               0                 0                 1

   ageScaled  balScaled  day  durScaled  campaign  pdays  previous
0   1.266667   1.250000    5   0.375000         1     -1         0
1   0.333333  -0.308997    5  -0.134259         1     -1         0
2  -0.400000  -0.328909    5  -0.481481         1     -1         0
3   0.533333   0.780236    5  -0.407407         1     -1         0
4  -0.400000  -0.329646    5   0.083333         1     -1         0

[5 rows x 51 columns]
```

[13]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
# Splitting the data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.3,␣
 ↪stratify=Y, random_state=123)
# Defining the LogisticRegression function
bankModel = LogisticRegression()
bankModel.fit(X_train, y_train)
```

```
/opt/conda/lib/python3.8/site-packages/sklearn/linear_model/_logistic.py:762:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[13]:
```
LogisticRegression()
```

[14]:
```
pred = bankModel.predict(X_test)
print('Accuracy of Logistic regression model prediction on test set: {:.2f}'.
 ↪format(bankModel.score(X_test, y_test)))
```

```
Accuracy of Logistic regression model prediction on test set: 0.90
```

[15]:
```
# Confusion Matrix for the model
from sklearn.metrics import confusion_matrix
confusionMatrix = confusion_matrix(y_test, pred)
print(confusionMatrix)
from sklearn.metrics import classification_report
print(classification_report(y_test, pred))
#good at predicting no --> precision and recall for no's are great
```

```
[[11718   259]
 [ 1077   510]]
              precision    recall  f1-score   support

          no       0.92      0.98      0.95     11977
         yes       0.66      0.32      0.43      1587

    accuracy                           0.90     13564
   macro avg       0.79      0.65      0.69     13564
weighted avg       0.89      0.90      0.89     13564
```

[16]:
```
print('Percentage of positive class :',(y_train[y_train=='yes'].value_counts()/
 ↪len(y_train) ) * 100)
print('Percentage of negative class :',(y_train[y_train=='no'].value_counts()/
 ↪len(y_train) ) * 100)
```

```
Percentage of positive class : yes    11.697791
Name: y, dtype: float64
Percentage of negative class : no    88.302209
Name: y, dtype: float64
```

[ ]:
```
# Three ways to deal with imbalanced data
# 1) Get more data
# 2) Undersample or remove data
# 3) Create fake data
# 4) Hybrid of 2 & 3
```

[17]:
```
from sklearn.model_selection import train_test_split
# Splitting the data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.3,␣
 ↪random_state=123)
```

[18]:
```
# let us first join the train_x and train_y for ease of operation
trainData = pd.concat([X_train,y_train],axis=1)
trainData.head()
```

[18]:

	job_admin.	job_blue-collar	job_entrepreneur	job_housemaid	\
19100	1	0	0	0	

```
37958           1                0                0             0
12451           0                1                0             0
18263           0                0                0             0
5128            0                0                0             0

       job_management  job_retired  job_self-employed  job_services  \
19100               0            0                  0             0
37958               0            0                  0             0
12451               0            0                  0             0
18263               1            0                  0             0
5128                0            0                  0             1

       job_student  job_technician  …  poutcome_success  poutcome_unknown  \
19100            0               0  …                 0                 1
37958            0               0  …                 0                 0
12451            0               0  …                 0                 1
18263            0               0  …                 0                 1
5128             0               0  …                 0                 1

       ageScaled  balScaled  day  durScaled  campaign  pdays  previous   y
19100   0.800000  -0.162979    5   0.236111         1     -1         0  no
37958   0.733333  -0.238938   14   0.865741         2    289        19  no
12451   0.000000   0.385693    1   1.347222         3     -1         0  no
18263   1.333333  -0.330383   31  -0.592593         8     -1         0  no
5128   -0.466667  -0.142330   21  -0.435185         2     -1         0  no

[5 rows x 52 columns]
```

[19]:
```
# Finding the indexes of the sample data set where the propensity is 'yes'
ind = trainData[trainData['y']=='yes'].index
print(len(ind))
```

```
3723
```

[20]:
```
# Seperate the minority classes
minData = trainData.loc[ind]
print(minData.shape)
```

```
(3723, 52)
```

[21]:
```
ind1 = trainData[trainData['y']=='no'].index
print(len(ind1))
```

```
27924
```

[25]:
```
majData = trainData.loc[ind1]
print(majData.shape)
```

```
majData.head()
```

```
(27924, 52)
```

[25]:

	job_admin.	job_blue-collar	job_entrepreneur	job_housemaid	\
19100	1	0	0	0	
37958	1	0	0	0	
12451	0	1	0	0	
18263	0	0	0	0	
5128	0	0	0	0	

	job_management	job_retired	job_self-employed	job_services	\
19100	0	0	0	0	
37958	0	0	0	0	
12451	0	0	0	0	
18263	1	0	0	0	
5128	0	0	0	1	

	job_student	job_technician	…	poutcome_success	poutcome_unknown	\
19100	0	0	…	0	1	
37958	0	0	…	0	0	
12451	0	0	…	0	1	
18263	0	0	…	0	1	
5128	0	0	…	0	1	

	ageScaled	balScaled	day	durScaled	campaign	pdays	previous	y
19100	0.800000	-0.162979	5	0.236111	1	-1	0	no
37958	0.733333	-0.238938	14	0.865741	2	289	19	no
12451	0.000000	0.385693	1	1.347222	3	-1	0	no
18263	1.333333	-0.330383	31	-0.592593	8	-1	0	no
5128	-0.466667	-0.142330	21	-0.435185	2	-1	0	no

```
[5 rows x 52 columns]
```

[27]:
```
majSample = majData.sample(n=len(ind),random_state = 123)
```

[28]:
```
print(majSample.shape)
majSample.head()
```

```
(3723, 52)
```

[28]:

	job_admin.	job_blue-collar	job_entrepreneur	job_housemaid	\
17387	0	0	0	0	
34679	0	1	0	0	
26572	1	0	0	0	
3280	0	0	0	0	
4434	0	0	0	0	

```
       job_management  job_retired  job_self-employed  job_services  \
17387               1            0                  0             0
34679               0            0                  0             0
26572               0            0                  0             0
3280                0            1                  0             0
4434                1            0                  0             0

       job_student  job_technician  …  poutcome_success  poutcome_unknown  \
17387            0               0  …                 0                 1
34679            0               0  …                 0                 0
26572            0               0  …                 0                 1
3280             0               0  …                 0                 1
4434             0               0  …                 0                 1

       ageScaled  balScaled  day  durScaled  campaign  pdays  previous   y
17387   0.666667   0.752212   28  -0.425926         3     -1         0  no
34679   0.800000   0.086283    5  -0.106481         7    250         3  no
26572   0.466667   1.785398   20  -0.134259         2     -1         0  no
3280    1.200000   1.972714   15  -0.009259         1     -1         0  no
4434   -0.133333   2.011062   20  -0.055556         1     -1         0  no

[5 rows x 52 columns]
```

[29]:
```
# Concatinating both data sets and then shuffling the data set
balData = pd.concat([minData,majSample],axis = 0)
print('balanced data set shape',balData.shape)
# Shuffling the data set
from sklearn.utils import shuffle
balData = shuffle(balData)
balData.head()
```

```
balanced data set shape (7446, 52)
```

[29]:
```
       job_admin.  job_blue-collar  job_entrepreneur  job_housemaid  \
39882           1                0                 0              0
41306           0                0                 0              0
39609           0                0                 0              0
40522           0                0                 0              0
41709           0                1                 0              0

       job_management  job_retired  job_self-employed  job_services  \
39882               0            0                  0             0
41306               1            0                  0             0
39609               0            0                  0             0
40522               1            0                  0             0
41709               0            0                  0             0
```

```
       job_student  job_technician  …  poutcome_success  poutcome_unknown  \
39882            0               0  …                 0                 0
41306            0               0  …                 0                 0
39609            1               0  …                 0                 1
40522            0               0  …                 0                 0
41709            0               0  …                 0                 0

       ageScaled  balScaled  day  durScaled  campaign  pdays  previous    y
39882   0.000000   0.246313    2  -0.083333         2     28        15   no
41306  -0.733333  -0.165929   27  -0.106481         2    119         1  yes
39609  -0.866667  -0.256637   26   1.226852         1     -1         0  yes
40522  -0.533333  -0.103982    8   0.837963         1    229         2  yes
41709  -0.133333   0.115044    7   0.884259         1    495         1  yes

[5 rows x 52 columns]
```

[30]:
```
# Making the new X_train and y_train
X_trainNew = balData.iloc[:,0:51]
X_trainNew.head()
y_trainNew = balData['y']
y_trainNew.head()
```

[30]:
```
39882     no
41306    yes
39609    yes
40522    yes
41709    yes
Name: y, dtype: object
```

[31]:
```
# Defining the LogisticRegression function
bankModel1 = LogisticRegression()
bankModel1.fit(X_trainNew, y_trainNew)
# Predicting on the test
pred = bankModel1.predict(X_test)
print('Accuracy of Logisticr regression model prediction on test set for␣
 ↪balanced data set: {:.2f}'.format(bankModel1.score(X_test, y_test)))
```

```
Accuracy of Logisticr regression model prediction on test set for balanced data
set: 0.83

/opt/conda/lib/python3.8/site-packages/sklearn/linear_model/_logistic.py:762:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
```

```
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[32]:
```
# Confusion Matrix for the model
from sklearn.metrics import confusion_matrix
confusionMatrix = confusion_matrix(y_test, pred)
print(confusionMatrix)
```

```
[[9969 2029]
 [ 278 1288]]
```

[34]:
```
# Confusion Matrix for the model
from sklearn.metrics import confusion_matrix
confusionMatrix = confusion_matrix(y_test, pred)
print(confusionMatrix)
```

```
[[9969 2029]
 [ 278 1288]]
```

[35]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test, pred))
```

```
              precision    recall  f1-score   support

          no       0.97      0.83      0.90     11998
         yes       0.39      0.82      0.53      1566

    accuracy                           0.83     13564
   macro avg       0.68      0.83      0.71     13564
weighted avg       0.91      0.83      0.85     13564
```

[36]:
```
#Overfitting: SMOTE ... MSMOTE
```

[37]:
```
!pip install smote-variants
```

```
WARNING: The directory '/home/jovyan/.cache/pip' or its parent directory is
not owned or is not writable by the current user. The cache has been disabled.
Check the permissions and owner of that directory. If executing pip with sudo,
you may want sudo's -H flag.
Collecting smote-variants
  Downloading smote_variants-0.4.0-py3-none-any.whl (134 kB)
     |                | 134 kB 18.7 MB/s eta 0:00:01
Requirement already satisfied: numpy>=1.13.0 in
/opt/conda/lib/python3.8/site-packages (from smote-variants) (1.18.5)
Collecting statistics
```

```
  Downloading statistics-1.0.3.5.tar.gz (8.3 kB)
Requirement already satisfied: pandas in /opt/conda/lib/python3.8/site-packages
(from smote-variants) (1.1.4)
Requirement already satisfied: joblib in /opt/conda/lib/python3.8/site-packages
(from smote-variants) (0.17.0)
Collecting keras
  Downloading keras-2.9.0-py2.py3-none-any.whl (1.6 MB)
     |                | 1.6 MB 31.5 MB/s eta 0:00:01
Collecting mkl
  Downloading mkl-2022.1.0-py2.py3-none-manylinux1_x86_64.whl (256.4 MB)
     |                | 256.4 MB 52.4 MB/s eta 0:00:01    |
| 14.4 MB 47.9 MB/s eta 0:00:06     |                               | 44.8 MB
47.9 MB/s eta 0:00:05                | 58.2 MB 43.4 MB/s eta 0:00:05
| 75.4 MB 43.4 MB/s eta 0:00:05               | 80.7 MB 43.4 MB/s eta 0:00:05
| 89.8 MB 62.9 MB/s eta 0:00:03     |                          | 119.9 MB
62.9 MB/s eta 0:00:03
Requirement already satisfied: tensorflow in
/opt/conda/lib/python3.8/site-packages (from smote-variants) (2.3.1)
Requirement already satisfied: scipy in /opt/conda/lib/python3.8/site-packages
(from smote-variants) (1.5.3)
Requirement already satisfied: scikit-learn in /opt/conda/lib/python3.8/site-
packages (from smote-variants) (0.23.2)
Collecting minisom
  Downloading MiniSom-2.3.0.tar.gz (8.8 kB)
Collecting docutils>=0.3
  Downloading docutils-0.18.1-py2.py3-none-any.whl (570 kB)
     |                | 570 kB 73.3 MB/s eta 0:00:01
Requirement already satisfied: python-dateutil>=2.7.3 in
/opt/conda/lib/python3.8/site-packages (from pandas->smote-variants) (2.8.1)
Requirement already satisfied: pytz>=2017.2 in /opt/conda/lib/python3.8/site-
packages (from pandas->smote-variants) (2020.4)
Collecting tbb==2021.*
  Downloading tbb-2021.6.0-py2.py3-none-manylinux1_x86_64.whl (4.0 MB)
     |                | 4.0 MB 60.6 MB/s eta 0:00:01
Collecting intel-openmp==2022.*
  Downloading intel_openmp-2022.1.0-py2.py3-none-manylinux1_x86_64.whl (10.7 MB)
     |                | 10.7 MB 49.9 MB/s eta 0:00:01
Requirement already satisfied: gast==0.3.3 in
/opt/conda/lib/python3.8/site-packages (from tensorflow->smote-variants) (0.3.3)
Requirement already satisfied: wheel>=0.26 in /opt/conda/lib/python3.8/site-
packages (from tensorflow->smote-variants) (0.35.1)
Requirement already satisfied: protobuf>=3.9.2 in /opt/conda/lib/python3.8/site-
packages (from tensorflow->smote-variants) (3.13.0)
Requirement already satisfied: opt-einsum>=2.3.2 in
/opt/conda/lib/python3.8/site-packages (from tensorflow->smote-variants) (3.3.0)
Requirement already satisfied: astunparse==1.6.3 in
/opt/conda/lib/python3.8/site-packages (from tensorflow->smote-variants) (1.6.3)
Requirement already satisfied: tensorboard<3,>=2.3.0 in
```

```
/opt/conda/lib/python3.8/site-packages (from tensorflow->smote-variants) (2.3.0)
Requirement already satisfied: tensorflow-estimator<2.4.0,>=2.3.0 in
/opt/conda/lib/python3.8/site-packages (from tensorflow->smote-variants) (2.3.0)
Requirement already satisfied: google-pasta>=0.1.8 in
/opt/conda/lib/python3.8/site-packages (from tensorflow->smote-variants) (0.2.0)
Requirement already satisfied: keras-preprocessing<1.2,>=1.1.1 in
/opt/conda/lib/python3.8/site-packages (from tensorflow->smote-variants) (1.1.2)
Requirement already satisfied: six>=1.12.0 in /opt/conda/lib/python3.8/site-
packages (from tensorflow->smote-variants) (1.15.0)
Requirement already satisfied: termcolor>=1.1.0 in
/opt/conda/lib/python3.8/site-packages (from tensorflow->smote-variants) (1.1.0)
Requirement already satisfied: grpcio>=1.8.6 in /opt/conda/lib/python3.8/site-
packages (from tensorflow->smote-variants) (1.33.2)
Requirement already satisfied: h5py<2.11.0,>=2.10.0 in
/opt/conda/lib/python3.8/site-packages (from tensorflow->smote-variants)
(2.10.0)
Requirement already satisfied: wrapt>=1.11.1 in /opt/conda/lib/python3.8/site-
packages (from tensorflow->smote-variants) (1.12.1)
Requirement already satisfied: absl-py>=0.7.0 in /opt/conda/lib/python3.8/site-
packages (from tensorflow->smote-variants) (0.11.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/opt/conda/lib/python3.8/site-packages (from scikit-learn->smote-variants)
(2.1.0)
Requirement already satisfied: setuptools in /opt/conda/lib/python3.8/site-
packages (from protobuf>=3.9.2->tensorflow->smote-variants)
(49.6.0.post20201009)
Requirement already satisfied: requests<3,>=2.21.0 in
/opt/conda/lib/python3.8/site-packages (from
tensorboard<3,>=2.3.0->tensorflow->smote-variants) (2.24.0)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in
/opt/conda/lib/python3.8/site-packages (from
tensorboard<3,>=2.3.0->tensorflow->smote-variants) (1.7.0)
Requirement already satisfied: markdown>=2.6.8 in /opt/conda/lib/python3.8/site-
packages (from tensorboard<3,>=2.3.0->tensorflow->smote-variants) (3.3.3)
Requirement already satisfied: google-auth<2,>=1.6.3 in
/opt/conda/lib/python3.8/site-packages (from
tensorboard<3,>=2.3.0->tensorflow->smote-variants) (1.23.0)
Requirement already satisfied: werkzeug>=0.11.15 in
/opt/conda/lib/python3.8/site-packages (from
tensorboard<3,>=2.3.0->tensorflow->smote-variants) (1.0.1)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in
/opt/conda/lib/python3.8/site-packages (from
tensorboard<3,>=2.3.0->tensorflow->smote-variants) (0.4.2)
Requirement already satisfied: idna<3,>=2.5 in /opt/conda/lib/python3.8/site-
packages (from requests<3,>=2.21.0->tensorboard<3,>=2.3.0->tensorflow->smote-
variants) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in
/opt/conda/lib/python3.8/site-packages (from
```

```
requests<3,>=2.21.0->tensorboard<3,>=2.3.0->tensorflow->smote-variants)
(2020.6.20)
Requirement already satisfied: chardet<4,>=3.0.2 in
/opt/conda/lib/python3.8/site-packages (from
requests<3,>=2.21.0->tensorboard<3,>=2.3.0->tensorflow->smote-variants) (3.0.4)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/opt/conda/lib/python3.8/site-packages (from
requests<3,>=2.21.0->tensorboard<3,>=2.3.0->tensorflow->smote-variants)
(1.25.11)
Requirement already satisfied: rsa<5,>=3.1.4; python_version >= "3.5" in
/opt/conda/lib/python3.8/site-packages (from google-
auth<2,>=1.6.3->tensorboard<3,>=2.3.0->tensorflow->smote-variants) (4.6)
Requirement already satisfied: cachetools<5.0,>=2.0.0 in
/opt/conda/lib/python3.8/site-packages (from google-
auth<2,>=1.6.3->tensorboard<3,>=2.3.0->tensorflow->smote-variants) (4.1.1)
Requirement already satisfied: pyasn1-modules>=0.2.1 in
/opt/conda/lib/python3.8/site-packages (from google-
auth<2,>=1.6.3->tensorboard<3,>=2.3.0->tensorflow->smote-variants) (0.2.8)
Requirement already satisfied: requests-oauthlib>=0.7.0 in
/opt/conda/lib/python3.8/site-packages (from google-auth-
oauthlib<0.5,>=0.4.1->tensorboard<3,>=2.3.0->tensorflow->smote-variants) (1.3.0)
Requirement already satisfied: pyasn1>=0.1.3 in /opt/conda/lib/python3.8/site-
packages (from rsa<5,>=3.1.4; python_version >= "3.5"->google-
auth<2,>=1.6.3->tensorboard<3,>=2.3.0->tensorflow->smote-variants) (0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in /opt/conda/lib/python3.8/site-
packages (from requests-oauthlib>=0.7.0->google-auth-
oauthlib<0.5,>=0.4.1->tensorboard<3,>=2.3.0->tensorflow->smote-variants) (3.0.1)
Building wheels for collected packages: statistics, minisom
  Building wheel for statistics (setup.py) … done
  Created wheel for statistics: filename=statistics-1.0.3.5-py3-none-
any.whl size=7453
sha256=8afc9a611d2918be55f569d875b7f1a243e7f947cd3c7ba9b0d176b519b9c007
  Stored in directory: /tmp/pip-ephem-wheel-cache-
ip5yxgfz/wheels/36/4b/c7/6af97584669b756c0d60c5ff05d5fb1f533a4e4d96e5ee92b9
  Building wheel for minisom (setup.py) … done
  Created wheel for minisom: filename=MiniSom-2.3.0-py3-none-any.whl
size=9018
sha256=a8aa20408d8dfb65375b961f99c4da6d06c02428f22d8fba9067aa7e5b5258a1
  Stored in directory: /tmp/pip-ephem-wheel-cache-
ip5yxgfz/wheels/6d/4e/9e/a95c14a232a196c22d9c04b221ff5d25461a1a4c55339c61db
Successfully built statistics minisom
Installing collected packages: docutils, statistics, keras, tbb, intel-openmp,
mkl, minisom, smote-variants
Successfully installed docutils-0.18.1 intel-openmp-2022.1.0 keras-2.9.0
minisom-2.3.0 mkl-2022.1.0 smote-variants-0.4.0 statistics-1.0.3.5 tbb-2021.6.0
```

[38]:
```
# Splitting the data into train and test sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.3,␣
 ↪random_state=0)
print("Before OverSampling count of yes: {}".format(sum(y_train=='yes')))
print("Before OverSampling count of no: {} \n".format(sum(y_train=='no')))
```

```
Before OverSampling count of yes: 3694
Before OverSampling count of no: 27953
```

[39]:
```
import smote_variants as sv
import numpy as np
# Instantiating the SMOTE class
oversampler= sv.SMOTE()
```

[40]:
```
# Creating new training set
X_train_us, y_train_us = oversampler.sample(np.array(X_train), np.
 ↪array(y_train))
```

```
2022-05-23 13:30:11,230:INFO:SMOTE: Running sampling via ('SMOTE',
"{'proportion': 1.0, 'n_neighbors': 5, 'n_jobs': 1, 'random_state': None}")
```

[41]:
```
# Shape after oversampling
print('After OverSampling, the shape of train_X: {}'.format(X_train_us.shape))
print('After OverSampling, the shape of train_y: {} \n'.format(y_train_us.
 ↪shape))
print("After OverSampling, counts of label 'Yes': {}".
 ↪format(sum(y_train_us=='yes')))
print("After OverSampling, counts of label 'no': {}".
 ↪format(sum(y_train_us=='no')))
```

```
After OverSampling, the shape of train_X: (55906, 51)
After OverSampling, the shape of train_y: (55906,)

After OverSampling, counts of label 'Yes': 27953
After OverSampling, counts of label 'no': 27953
```

[42]:
```
# Training the model with Logistic regression model
# Defining the LogisticRegression function
bankModel2 = LogisticRegression()
bankModel2.fit(X_train_us, y_train_us)
# Predicting on the test set
pred = bankModel2.predict(X_test)
# Printing accuracy
print('Accuracy of Logistic regression model prediction on test set for Smote␣
 ↪balanced data set: {:.2f}'.format(bankModel2.score(X_test, y_test)))
```

```
# Confusion Matrix for the model
from sklearn.metrics import confusion_matrix
confusionMatrix = confusion_matrix(y_test, pred)
print(confusionMatrix)
# Classification report for the model
from sklearn.metrics import classification_report
print(classification_report(y_test, pred))
```

```
/opt/conda/lib/python3.8/site-packages/sklearn/linear_model/_logistic.py:762:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

```
Accuracy of Logistic regression model prediction on test set for Smote balanced
data set: 0.84
[[10097  1872]
 [  326  1269]]
              precision    recall  f1-score   support

          no       0.97      0.84      0.90     11969
         yes       0.40      0.80      0.54      1595

    accuracy                           0.84     13564
   macro avg       0.69      0.82      0.72     13564
weighted avg       0.90      0.84      0.86     13564
```

[43]:
```
# Splitting the data into train and test sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.3,␣
 ↪random_state=0)
print("Before OverSampling count of yes: {}".format(sum(y_train=='yes')))
print("Before OverSampling count of no: {} \n".format(sum(y_train=='no')))
```

```
Before OverSampling count of yes: 3694
Before OverSampling count of no: 27953
```

[44]:
```
import smote_variants as sv
import numpy as np
# Instantiating the SMOTE class
oversampler= sv.MSMOTE()
```

```
# Creating new training sts
X_train_us, y_train_us = oversampler.sample(np.array(X_train), np.
 ↪array(y_train))
# Shape after oversampling
print('After OverSampling, the shape of train_X: {}'.format(X_train_us.shape))
print('After OverSampling, the shape of train_y: {} \n'.format(y_train_us.
 ↪shape))
print("After OverSampling, counts of label 'Yes': {}".
 ↪format(sum(y_train_us=='yes')))
print("After OverSampling, counts of label 'no': {}".
 ↪format(sum(y_train_us=='no')))
```

```
2022-05-23 13:44:18,047:INFO:MSMOTE: Running sampling via ('MSMOTE',
"{'proportion': 1.0, 'n_neighbors': 5, 'n_jobs': 1, 'random_state': None}")

After OverSampling, the shape of train_X: (55906, 51)
After OverSampling, the shape of train_y: (55906,)

After OverSampling, counts of label 'Yes': 27953
After OverSampling, counts of label 'no': 27953
```

[45]:
```
# Fitting model
# Training the model with Logistic regression model
# Defining the LogisticRegression function
bankModel2 = LogisticRegression()
bankModel2.fit(X_train_us, y_train_us)
# Predicting on the test
pred = bankModel2.predict(X_test)
print('Accuracy of Logistic regression model prediction on test set for Smote␣
 ↪balanced data set: {:.2f}'.format(bankModel2.score(X_test, y_test)))
# Confusion Matrix for the model
from sklearn.metrics import confusion_matrix
confusionMatrix = confusion_matrix(y_test, pred)
print(confusionMatrix)
from sklearn.metrics import classification_report
print(classification_report(y_test, pred))
```

```
/opt/conda/lib/python3.8/site-packages/sklearn/linear_model/_logistic.py:762:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

```
Accuracy of Logistic regression model prediction on test set for Smote balanced
data set: 0.83
[[10055  1914]
 [  340  1255]]
              precision    recall  f1-score   support

          no       0.97      0.84      0.90     11969
         yes       0.40      0.79      0.53      1595

    accuracy                           0.83     13564
   macro avg       0.68      0.81      0.71     13564
weighted avg       0.90      0.83      0.86     13564
```

[ ]: