```
import xgboost as xgb
from sklearn.ensemble import ExtraTreesClassifier
import matplotlib.image as mpimg
import numpy as np
import sys
import os
import gzip
import math
import matplotlib.pyplot as plt
import random
import copy
from sklearn.decomposition import PCA, KernelPCA
import csv
import pandas as pd
import seaborn as sns
import cv2
import glob
import pickle
import scipy
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.decomposition import PCA
from sklearn.neural_network import MLPClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.metrics import roc_auc_score
from sklearn import metrics
from sklearn.neighbors import KNeighborsClassifier
import scipy
import scipy.misc
import re
from sklearn.svm import SVC
from PIL import Image
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_score

from skimage import io
from skimage.color import rgb2gray
from skimage.transform import resize
from skimage.feature import hog
from sklearn import tree
from sklearn.linear_model import Lasso
import sklearn as sk
from sklearn import svm
from sklearn.metrics import confusion_matrix
import pandas as pd
import os
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.feature_selection import VarianceThreshold
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import QuantileTransformer
import matplotlib.pyplot as plt
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
from sklearn.feature_selection import RFE
from sklearn.linear_model import LogisticRegression
from sklearn.feature_selection import RFECV
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import StratifiedKFold
import pandas as pd
from sklearn.model_selection import cross_validate
import numpy as np
from sklearn.svm import LinearSVC
from sklearn.linear_model import Lasso
from sklearn.feature_selection import SelectFromModel
from sklearn.metrics import matthews_corrcoef
from sklearn.ensemble import GradientBoostingClassifier
```

```
xdata=pd.read_csv("./train/train.csv")
```

```
xdata=pd.read_csv("./train/train.csv")
ydata=pd.read_csv("./test/test.csv")
```

```
xlab=xdata['Label']
xid=xdata['id']
yid=ydata['id']
pdA=xdata.drop(['Label','id'], axis = 1)
pdB=ydata.drop(['id'], axis = 1)
```

```
print(np.array(xdata).shape)
print(np.array(ydata).shape)
A=np.array(pdA)
B=np.array(pdB)
xlab=np.array(xlab)
xid=np.array(xid)
yid=np.array(yid)
print(A.shape)
print(B.shape)
```

```
(279, 60485)
(71, 60484)
(279, 60483)
(71, 60483)
```

```
constant_filter = VarianceThreshold(0.1)
constant_filter.fit(np.concatenate((pdA,pdB)))

print(len(pdA.columns[constant_filter.get_support()]))

constant_columns = [column for column in pdA.columns
                    if column not in pdA.columns[constant_filter.get_support()]]
pdA.drop(labels=constant_columns, axis=1, inplace=True)
pdB.drop(labels=constant_columns, axis=1, inplace=True)
C=np.array(pdA)
D=np.array(pdB)
print(C.shape,D.shape)
```

```
17936
(279, 17936) (71, 17936)
```

```
corr_matrix = pdA.corr().abs()
upper = corr_matrix.where(np.triu(np.ones(corr_matrix.shape), k=1).astype(np.bool))
to_drop = [column for column in upper.columns if any(upper[column] > 0.95)]
print(len(to_drop))
pdA.drop(labels=to_drop, axis=1, inplace=True)
pdB.drop(labels=to_drop, axis=1, inplace=True)
G=np.array(pdA)
H=np.array(pdB)
print(G.shape,H.shape)
```

```
82
(279, 17854) (71, 17854)
```

```
print(G.shape,H.shape)
```

```
(279, 17854) (71, 17854)
```

```
pca = KernelPCA(n_components=95)
pca.fit(np.concatenate((G,H)))
GG = pca.transform(G)
HH = pca.transform(H)
print(GG.shape,HH.shape)
```

```
(279, 95) (71, 95)
```

```
clf= MLPClassifier(max_iter=2000)
from sklearn.model_selection import ShuffleSplit
ss=ShuffleSplit(n_splits=20, test_size=0.5, random_state=10)
scores = cross_val_score(clf, GG,xlab, cv=ss,n_jobs=-1, verbose=1)
print(np.mean(scores))
print(scores)
clf.fit(GG, xlab)
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  20 out of  20 | elapsed:    2.8s finished
0.5746428571428572
[0.63571429 0.53571429 0.59285714 0.62857143 0.55       0.57857143
 0.6        0.6        0.55       0.56428571 0.54285714 0.62857143
 0.60714286 0.6        0.56428571 0.50714286 0.56428571 0.56428571
 0.52857143 0.55      ]
MLPClassifier(activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9,
              beta_2=0.999, early_stopping=False, epsilon=1e-08,
              hidden_layer_sizes=(100,), learning_rate='constant',
              learning_rate_init=0.001, max_fun=15000, max_iter=2000,
              momentum=0.9, n_iter_no_change=10, nesterovs_momentum=True,
              power_t=0.5, random_state=None, shuffle=True, solver='adam',
              tol=0.0001, validation_fraction=0.1, verbose=False,
              warm_start=False)
```

```python
pred= clf.predict(HH)
print (len(pred))
ids= np.arange(3001, 3072)
df= pd.DataFrame(ids, columns= ["id"])
df['Label']= pred
df.to_csv('./subm.csv', index= False)
```

```
71
```