In [4]:
```
!pip install sklearn
```

In [6]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import make_classification
import pandas as pd
```

In [7]:
```
import numpy as np
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, r2_score
from sklearn.model_selection import train_test_split
import collections
```

In [8]:
```
X=pd.read_csv('dz_pro_train.csv')
X_t=pd.read_csv('dz_pro_test.csv')
X_aug=pd.read_csv("dz_pro_train_aug.csv")
X_t_aug=pd.read_csv("dz_pro_test_aug.csv")
```

Vectorizing

In [9]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
def do_nothing(x):
    return x
def create_features(train_data, test_data):
    vect = TfidfVectorizer(analyzer='word',tokenizer=do_nothing, preprocessor=do_nothing, token_pattern=None)

    tweet_words = []
    for tweet in train_data["tweet"]:
        tweet_words.append(tweet)
    labels = []
    for label in train_data["label"]:
        labels.append(label)

    train_labels = np.asarray(labels)
    train_features = vect.fit_transform(train_data["tweet"])
    test_features = vect.transform(test_data["tweet"])

    return train_features, train_labels, test_features
```

In [10]:
```
X["tweet"]
```

Out[10]:
```
0                  عنديش مزية كشعب نستحقوش النظافة النظام
1       زعما نتا مول العقل أسي الغزواني هذاك راه حساب ...
2       يعمري غاضتني بصح لبنات بلا استثناء واااو براف...
3                     خدمات فاشلة تقول عاملين علينا مزية
4       اه علابالي الصحراء تقدر ترجعها جنة بصح المشكل ...
                              ...
1465                          آش داني وعلاش مشيت هههههههه
1466                           اخطونا بك بحكومتك بمسؤوليك
1467    العمرة ساعتين والحج تروح الصباح ترجع العشية ال...
1468                                    يا اخي كنبغيك بزاف
1469    التنظيم زعما غادي فمستوى عالي دورة وهران الالع...
Name: tweet, Length: 1470, dtype: object
```

In [68]:
```
train_features, train_labels, test_features = create_features(X, X_t)
```

In [69]:
```
train_labels.shape
test_features
```

Out[69]:
```
<180x233 sparse matrix of type '<class 'numpy.float64'>'
        with 2998 stored elements in Compressed Sparse Row format>
```

Logistic regression

In [70]:
```
# Import the logistic regression model from sklearn
from sklearn.linear_model import LogisticRegression

# Define the model
model = LogisticRegression(random_state=0, solver='lbfgs',
                           multi_class='multinomial')
# Train model
model.fit(train_features, train_labels)
```

Out[70]:
```
LogisticRegression(multi_class='multinomial', random_state=0)
```

In [74]:
```
labels = []
for label in X_t["label"]:
    labels.append(label)
test_labels = np.asarray(labels)
```

In [75]:
```
model.score(test_features, test_labels)
```

Out[75]:
```
0.4777777777777778
```

Random forest

In [73]:
```
# We train two models: random forest and logistic regression
from sklearn.ensemble import RandomForestClassifier
# Initialize a Random Forest classifier with 500 trees
forest = RandomForestClassifier(n_estimators = 500, max_depth = None, min_samples_split=2, min_samples_leaf =1,
                                bootstrap = True, random_state=0)
# Train the model
forest = forest.fit(train_features, train_labels)
# Print score of model(using test dataset)
print(forest.score(test_features, test_labels))
```

```
0.46111111111111114
```

Augmented data

In [ ]:
```

```

In [13]:
```
train_features_aug, train_labels_aug, test_features_aug = create_features(X_aug, X_t_aug)
```

Logistic regression

In [14]:
```
# Import the logistic regression model from sklearn
from sklearn.linear_model import LogisticRegression

# Define the model
model = LogisticRegression(random_state=0, solver='lbfgs',
                           multi_class='multinomial')
# Train model
model.fit(train_features_aug, train_labels_aug)
labels = []
for label in X_t_aug["label"]:
    labels.append(label)
test_labels_aug = np.asarray(labels)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarning: lbfgs fai
led 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(
```

In [15]:
```
model.score(test_features_aug, test_labels_aug)
```

Out[15]:
```
0.5696969696969697
```

Random forest

In [17]:
```
# We train two models: random forest and logistic regression
from sklearn.ensemble import RandomForestClassifier
# Initialize a Random Forest classifier with 500 trees
forest = RandomForestClassifier(n_estimators = 500, max_depth = None, min_samples_split=2, min_samples_leaf =1,
                                bootstrap = True, random_state=0)
# Train the model
forest = forest.fit(train_features_aug, train_labels_aug)
# Print score of model(using test dataset)
```

In [18]:
```
print(forest.score(test_features_aug, test_labels_aug))
```

```
0.6181818181818182
```

In [ ]:
```

```