```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
# tred count vectorizer , tf-idf used seperately,ngrams , linearsvc, logistic

import numpy as np
from sklearn import preprocessing
import csv
from sklearn.svm import LinearSVC
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import accuracy_score, f1_score
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.multiclass import OneVsRestClassifier

# Here is the file path to give

# This is cleaning data converting into matrices form
# Here stripped of whitespaces and converted into lower case



with open("/content/drive/MyDrive/Colab Notebooks/DF_project/Full_set.csv") as file:
  reader = csv.reader(file, delimiter=',')
  next(reader)
  count = 0
  set1 = set()
  X_tmp =[]
  y_tmp =[]
  for line in reader:
    #print(line)
    y1 =[]
    X_tmp.append(line[0].lower().strip())
    count = count +1
    #new = line[1].split("|")
    #print(new)
    y1.append(line[1].lower().strip())
    set1.add(line[1].lower().strip())
    if line[2] != '':
      y1.append(line[2].lower().strip())

    y_tmp.append(y1)

    #if count == 10:
      #break

print("X_Samples = ",len(X_tmp))
print("y_Samples = ",len(y_tmp))




#print(len(y_tmp))

print(f"Labels ={set1}")
print("count_labels=", len(set1))

#print(y_tmp)
```

```
X_Samples =  1000
y_Samples =  1000
Labels ={'edu', 'util', 'dei', 'govt data', 'public', 'cyber', 'connect', 'other', 'ag', 'mobility', 'enviro', 'iot'}
count_labels= 12
```

```
# Now splitting into train and test data

X_txt_train, X_txt_test, y_train_text, y_test_text = train_test_split(X_tmp, y_tmp, test_size=0.2, random_state=42)
```

```
# converting into numpy arrays

X_train = np.array(X_txt_train)
X_test = np.array(X_txt_test)



from sklearn.metrics import precision_score, recall_score, f1_score

lb = preprocessing.MultiLabelBinarizer(classes=('other', 'iot', 'mobility', 'connect', 'edu', 'util', 'public', 'ag', 'cyber', 'govt data', '
y_train = lb.fit_transform(y_train_text)
y_test = lb.transform(y_test_text)
print(y_test)

#print(y_test)
print(y_train.shape)
print(y_test.shape)

classifier = Pipeline([
('vectorizer', CountVectorizer()),
('tfidf', TfidfTransformer()),
('clf', OneVsRestClassifier(LinearSVC()))])

classifier.fit(X_train, y_train)
predicted = classifier.predict(X_test)
print(predicted)
print("Accuracy Score: ",accuracy_score(y_test, predicted))
precision = precision_score(predicted,y_test,average="macro") # Get scores using svm_test_predictions and y_test with the precision_score met
recall = recall_score(predicted,y_test,average="macro")
f1 = f1_score(predicted,y_test,average="macro")
print("Precision: {:.4f}".format(precision))
print("Recall: {:.4f}".format(recall))
print("F1: {:.4f}".format(f1))
```

```
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 1]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 1 ... 0 0 0]
 [0 0 1 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
(800, 12)
(200, 12)
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 1 0 ... 0 0 0]
 [1 0 0 ... 0 0 0]]
Accuracy Score:  0.145
Precision: 0.1467
Recall: 0.3897
F1: 0.1991
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedMetricWarning: Precision is ill-defined and bei
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedMetricWarning: Recall is ill-defined and being
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1580: UndefinedMetricWarning: F-score is ill-defined and being
  _warn_prf(average, "true nor predicted", "F-score is", len(true_sum))
```

```
from sklearn.metrics import precision_score, recall_score, f1_score
from sklearn.feature_extraction.text import TfidfVectorizer

lb = preprocessing.MultiLabelBinarizer(classes=('other', 'iot', 'mobility', 'connect', 'edu', 'util', 'public', 'ag', 'cyber', 'govt data', '
y_train = lb.fit_transform(y_train_text)
y_test = lb.transform(y_test_text)

#print(y_test)
print(y_train.shape)
print(y_test.shape)

classifier = Pipeline([
('vec2', TfidfVectorizer(analyzer='word' , stop_words='english')),
('tfidf', TfidfTransformer()),
```

```
('clf', OneVsRestClassifier(LinearSVC()))])

classifier.fit(X_train, y_train)
predicted = classifier.predict(X_test)
print(predicted)
print("Accuracy Score: ",accuracy_score(y_test, predicted))
precision = precision_score(predicted,y_test,average="macro") # Get scores using svm_test_predictions and y_test with the precision_score met
recall = recall_score(predicted,y_test,average="macro")
f1 = f1_score(predicted,y_test,average="macro")
print("Precision: {:.4f}".format(precision))
print("Recall: {:.4f}".format(recall))
print("F1: {:.4f}".format(f1))
```

```
(800, 12)
(200, 12)
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 1 0 ... 0 0 0]
 [1 0 0 ... 0 0 0]]
Accuracy Score:  0.12
Precision: 0.1352
Recall: 0.3779
F1: 0.1849
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedMetricWarning: Precision is ill-defined and bei
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedMetricWarning: Recall is ill-defined and being
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1580: UndefinedMetricWarning: F-score is ill-defined and being
  _warn_prf(average, "true nor predicted", "F-score is", len(true_sum))
```

Colab paid products - Cancel contracts here