گزارش پروژه نهایی درس هوش مصنوعی و سیستم‌های خبره

موضوع: تشخیص زبان از طریق متن ورودی

استاد: دکتر مهرنوش شمس‌فرد

دانشجو: امیر حلاجی بیدگلی

بهار ۱۴۰۰

سرفصل مطالب

کتابخانه‌ها

```
import pandas as pd
import csv
import numpy as np
from sklearn.utils import shuffle
from sklearn.linear_model import SGDClassifier
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn import metrics
from hazm import *
```

در این خط، کتابخانه‌هایی را که در طول پروژه به آن نیاز داریم، **import** می‌کنیم. این قسمت نکته‌ی خاص دیگری ندارد.

خواندن داده‌ها

در این قسمت داده‌های ۶ زبان را می‌خوانیم. داده‌های زبان فارسی، متونی از متون خبری اخبار صدا و سیما بودند که هر کدام در فایل‌های جدا نگه‌داری می‌شدند. با استفاده از تکّه کد زیر، تمام متن‌ها را در یک فایل به فرمت **csv** می‌ریزیم.

```
data = ''
for i in range(0, 8):
    for j in range(1, 10):
        with open(('0' + str(i) + str(j) + '.txt'), 'r', encod
ing='utf-8') as f:
            data += f.read()
            # data += '\n'

with open('output1.csv', mode='w', encoding='utf-8') as f:
    f.write(data)
```

حذف کلمات توقف

Stopwords های زبان فارسی در یک فایل تکست که در پوشه‌های فایل‌های آپلود شده قرار داده شده وجود داد. با استفاده از تکه کد زیر، آن‌ها را در یک لیست می‌ریزیم.

```
normalizer = Normalizer()

stopwords_list = []
stopwords_url = 'drive/MyDrive/Artificial Intelligence-
Spring 2021/project/stopwords.txt'
f = open(stopwords_url, mode='r')
stopwords_list.append(f.read())
persian_stopwords = []
for i in stopwords_list:
  persian_stopwords.append(i)
```

برداری کردن کلمات جملات

```
kaggle_task = pd.read_csv('drive/MyDrive/Artificial Intelligence-
Spring 2021/project/task.csv')
help_me = kaggle_task.Id

vectorizer = TfidfVectorizer(min_df=11, max_df=0.07, decode_error='strict'
, ngram_range=(1, 2), binary=0.5, sublinear_tf=True)
train_x = vectorizer.fit_transform(all_language_dataset.Id)
labelencoder = LabelEncoder()
train_y = labelencoder.fit_transform(all_language_dataset.Category)

kaggle_task_x = vectorizer.transform(kaggle_task.Id)
```

یک vectorizer با ویژگی‌های زیر می‌سازیم:

```
min_df=11, max_df=0.07, decode_error='strict',
ngram_range=(1, 2)
```

توضیح موارد زیر:

Min_df: هنگام وکتورایز کردن، مقادیر کمتر از این مقدار را نادیده می‌گیرد.

Max_df: هنگام وکتورایز کردن، مقادیر تا حداکثر این مقدار را وکتورایز می‌کند؛ درغیر اینصورت، آن را نادیده می‌گیرد.

Decode_error: اگر در decode کردن کلمات به مشکل بخورد، ارور را نادیده نمی‌گیرد.

Ngram_range: از unigram و bigram استفاده می‌کند.

تقسیم داده‌ها به داده یادگیری و تست

```
x_train, x_test, y_train, y_test = train_test_split(train_x, train_y, test
_size=0.25, random_state=0)
```

۷۵ درصد داده را مخصوص یادگیری و ۲۵ درصد باقی را به داده‌ی تست اختصاص می‌دهیم.

ساختن مدل با استفاده ار داده‌های یادگیری

```
def benchmark(clf):
  print("**", clf, "***")
  clf.fit(x_train, y_train)
  prediction = clf.predict(x_test)
  score = metrics.accuracy_score(y_test, prediction)
  print("accuracy:  %0.3f",  score)

results = []
clf = SGDClassifier(loss="modified_huber", max_iter=1500, l1_ratio=0.15, a
verage=True, shuffle=False)
name = "SGD Classifier"
print(name)
results.append(benchmark(clf))
```

مدل Stochastic gradient decsent را پارامترهای بالا می‌سازیم.

تابع هزینه: modified_huber

$$L_\delta(y, f(x)) = \begin{cases} \frac{1}{2}(y - f(x))^2 & \text{for} |y - f(x)| \le \delta, \\ \delta\left(|y - f(x)| - \frac{1}{2}\delta\right), & \text{otherwise.} \end{cases}$$

بیشینه‌ی iteration: تعداد اپوک‌های اجرا که در اینجا ۱۵۰۰ گذاشته‌ام.

امتحان کردن مدل روی داده‌های تست

```
kaggle_results = []
print("***  KAGGLE TEST  ***")
classifier = SGDClassifier(loss="modified_huber", penalty='l2', max_iter=1
500, l1_ratio=0.15, class_weight=None, average=True, shuffle=False)
classifier.fit(x_train, y_train)

kaggle_prediction = classifier.predict(kaggle_task_x)
```

ثبت نتایج در فایل

```
preLabels = ['Id', 'Category']
labels_y = labelencoder.inverse_transform(kaggle_prediction)
labels_x = vectorizer.inverse_transform(kaggle_task_x)


result_file = 'drive/MyDrive/Artificial Intelligence-
Spring 2021/project/result.csv'
with open(result_file, mode='w') as result_file:
  wr = csv.writer(result_file, delimiter=',')
  wr.writerow(preLabels)
  for (id, item) in enumerate(labels_y, start=0):
    wr.writerow((help_me[id] , item))
```

ابتدا باید یک **header** با نام **Id** و **Category** بسازیم. سپس تبدیل معکوس **vectorize** را برای **x** و **y** انجام دهیم. سپس نتیجه را طبق کد بالا در فایل **result.csv** می‌ریزیم.

پیاده‌سازی logistic regression برای خروجی چند کلاسه

```
print(train_x[0])
y1 = np.zeros([train_x.shape[0], 6])
y1 = pd.DataFrame(y1)
y1
```

```
def hypothesis(theta, X):
    return 1 / (1 + np.exp(-(np.dot(theta, X.T)))) - 0.0000001
```

```
def cost(X, y, theta):
    y1 = hypothesis(X, theta)
    return -(1/len(X)) * np.sum(y*np.log(y1) + (1-y)*np.log(1-y1))

def gradient_descent(X, y, theta, alpha, epochs):
    m = len(all_language_dataset)
    for i in range(0, epochs):
        for j in range(0, 10):
            theta = pd.DataFrame(theta)
            h = hypothesis(theta.iloc[:,j], X)
            for k in range(0, theta.shape[0]):
                theta.iloc[k, j] -= (alpha/m) * np.sum((h-
y.iloc[:, j])*X.iloc[:, k])
            theta = pd.DataFrame(theta)
    return theta, cost


theta = np.zeros([train_x.shape[1]+1, y1.shape[1]])
theta = gradient_descent(train_x, y1, theta, 0.02, 1500)
```

لینک منابع استفاده شده

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html

https://towardsdatascience.com/multiclass-classification-algorithm-from-scratch-with-a-project-in-python-step-by-step-guide-485a83c79992

https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

لینک داده‌های استفاده شده:

داده‌ی زبان فارسی: http://dataheart.ir/article/3397 –

https://drive.google.com/file/d/1mBeSSrEnajB2qxYs67tQbEDWm
/https://bigdata-ir.com – pRMZ0U0/view

داده‌ی زبان عربی: https://data.mendeley.com/datasets/v524p5dhpj/2

داده‌ی زبان انگلیسی، ترکی، آلمانی . فرانسوی:

https://www.kaggle.com/basilb2s/language-detection