Projenin Amacı

Proje kapsamında, twitter üzerinden haber paylaşımı yapan hesapların, attığı twitlerini alıp hangi kategoriye ait olduğuna göre kategorize eden bir program yazılım geliştirmek amaçlanmıştır. Geliştirilen uygulamada hedef twitter adresini ve çekilecek tweet sayısını değiştirebilmek mümkündür. Bu işlemleri yaparken minimum %50 başarı oranını tutturmak hedeflenmiştir.

Projenin Tasarımı

Veri Seti:

Projeye başlarken ilk adımımız kendimize uygun bir veri seti bulmak veya oluşturmaktı. Sıfırdan veri seti oluşturmanın zorluğunu, harcatacağı zamanı ve verimliliğini göz önüne aldığımızda bu proje için en doğru tercihin hazır bir veri seti üzerinden ilerlemek olduğuna karar kıldık. Twitterda Machine Learning Engineer olarak çalışan Rishabh Misra tarafından hazırlanmış olan bir veri setini seçtik. Seçtiğimiz veri seti 200.000'in üzerindeki haber örneğini ve 41 farklı kategoriyi içermekte ve projenin gerekliliklerini tam olarak karşılamaktadır. Veri setinin içeriğine göz atabilmemizi sağlayan birkaç kod satırını çalıştırıyoruz.

```
news_articles = pd.read_json("News_Category_Dataset_v2.json", lines = True)
news_articles.info()
news_articles.head()

print("The total number category:>",news_articles['category'].nunique())
category=news_articles['category'].value_counts()
print(category)
```

Elde ettiğimiz çıktı bize veriseti hakkında genel bir bilgi sunuyor. Çıktımız şu şekilde:

```
RangeIndex: 200853 entries, 0 to 200852
Data columns (total 6 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   category           200853 non-null  object
 1   headline           200853 non-null  object
 2   authors            200853 non-null  object
 3   link               200853 non-null  object
 4   short_description  200853 non-null  object
 5   date               200853 non-null  datetime64[ns]
dtypes: datetime64[ns](1), object(5)
memory usage: 9.2+ MB
```

Aynı zamanda bize toplam kategori sayısını ve hangi kategoriye ait kaç haber olduğu bilgisini de sunuyor:

```
The total number category present here----------> 41
POLITICS          32739
WELLNESS          17827
ENTERTAINMENT     16058
TRAVEL             9887
STYLE & BEAUTY     9649
PARENTING          8677
HEALTHY LIVING     6694
QUEER VOICES       6314
FOOD & DRINK       6226
BUSINESS           5937
COMEDY             5175
SPORTS             4884
BLACK VOICES       4528
HOME & LIVING      4195
PARENTS            3955
THE WORLDPOST      3664
WEDDINGS           3651
WOMEN              3490
IMPACT             3459
DIVORCE            3426
CRIME              3405
MEDIA              2815
WEIRD NEWS         2670
GREEN              2622
WORLDPOST          2579
RELIGION           2556
STYLE              2254
SCIENCE            2178
WORLD NEWS         2177
TASTE              2096
TECH               2082
MONEY              1707
ARTS               1509
FIFTY              1401
GOOD NEWS          1398
ARTS & CULTURE     1339
ENVIRONMENT        1323
COLLEGE            1144
LATINO VOICES      1129
CULTURE & ARTS     1030
EDUCATION          1004
```

Veri setinin bilgi içeriği:

@dataset{dataset,
author = {Misra, Rishabh},
year = {2018},
month = {06},
pages = {},
title = {News Category Dataset},
doi = {10.13140/RG.2.2.20331.18729}
}

Yapısı ve içeriği itibariyle twitter üzerinde kullanmaya oldukça uyumlu bir veri seti olduğunu gözlemledik. Elimizdeki diğer veri setleriyle de kıyas yaptığımızda ve küçük yazılım testleri uyguladığımızda aldığımız sonuçlar neticesinde kesin olarak ilgili dataseti kullanmaya karar verdik.

Kullanıma daha uygun hale getirme amacıyla çok kısa olan ve kendini tekrar eden içerikleri veri setinden çıkarma işlemi uyguladık.

```
news_articles = news_articles[news_articles['headline'].apply(lambda x:
len(x.split())>5)]
print("Total number of articles after removal of headlines with short
title:", news_articles.shape[0])
news_articles.sort_values('headline',inplace=True, ascending=False)
duplicated_articles_series = news_articles.duplicated('headline', keep =
False)
news_articles = news_articles[~duplicated_articles_series]
```

Bu işlem sonrası elimizde kalan verinin yeni haline göz atmak gerekirse:

```
print("Total number of articles after removing duplicates:",
news_articles.shape[0])
news_articles.isna().sum()
print("Total number of articles : ", news_articles.shape[0])
print("Total number of authors : ", news_articles["authors"].nunique())
print("Total number of unqiue categories : ",
news_articles["category"].nunique())
```

Çıktısı:

```
Total number of articles after removal of headlines with short title: 97318
Total number of articles after removing duplicates: 97008
Total number of articles :  97008
Total number of authors :  13874
Total number of unqiue categories :  31
```

İşlemler sonrası elimizde toplamda 31 farklı kategoriye ait 97008 adet verimiz kaldı. Son kez elimizde kalan kategorilere ve o kategorilere ait veri sayısına göz atmak gerekirse:

```
The total number category:> 31
POLITICS          28058
ENTERTAINMENT     12084
HEALTHY LIVING     4232
QUEER VOICES       4022
THE WORLDPOST      3600
SPORTS             3352
BLACK VOICES       3350
COMEDY             3204
BUSINESS           2728
PARENTS            2704
WOMEN              2398
CRIME              2379
WEIRD NEWS         2315
MEDIA              2228
WORLD NEWS         2106
GREEN              1678
STYLE              1674
RELIGION           1472
TASTE              1461
IMPACT             1437
TRAVEL             1319
ARTS & CULTURE     1310
TECH               1195
WORLDPOST          1182
GOOD NEWS          1065
LATINO VOICES       975
SCIENCE             972
COLLEGE             693
FIFTY               668
ARTS                575
EDUCATION           572
```

Yazılım Dili:

Kullanılacak yazılım dili için Python üzerinde karar kıldık. Bu alandaki çalışmaların, örneklerin çoğunun Python üzerinden olması ve kütüphane zenginliği bu dili seçmemizin ana sebepleriydi. Yazma ve derleme işlemlerini PyCharm üzerinden gerçekleştirdik.

Kullanılan Kütüphaneler:

```
# Temel işlem kütüphaneleri
import random
import os
import math
import time
import numpy as np
import pandas as pd

# Ana işlemlerin gerçekleşmesi için gerekli kütüphaneler
from gensim.models import Word2Vec
from gensim.models import KeyedVectors
import gensim
import pickle

# Tablo, grafik dökümü için gerekli kütüphaneler
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.figure_factory as ff
import plotly.graph_objects as go
import plotly.express as px

# Aşağıdaki kütüphaneler veriyi temizleme amacıyla kullanılıyor. Stop
words, tokenize ve lemmatization işlemleri için gereklidir.
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer

# SKLearn içindeki vektör işlemleri için gereklidir
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer

# SKLearn içindeki benzerlik matrisleri için gereklidir.
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.metrics import pairwise_distances

# Twitter işlemleri için gereklidir.
import tweepy
import re
```

Yöntem:

Projenin çözüm aşamasında problem 3 ana bölümde ele alındı. İlk twitter üzerinden ilgili içeriklerin çekilmesi için çözüm üretildi. Bu amaçla "Twipy" kütüphanesinden yararlanılarak çözüm üretildi. Twitter üzerinden "geliştirici hesabı" elde etmek için başvuru yapıldı. Twitter'dan veri çekme işlemi "newsTweets(tweets)" isimli fonksiyon içerisine tanımlandı.

İlk olarak geliştirici hesabın bilgileri ile program üzerindne twitter'a giriş yaptık. *(Güvenlik sebepleriyle ilgili bilgiler *** şeklinde rapora aktarılmıştır.)*

```
consumer_key = ********************
consumer_secret = ******************
access_key = **********************
access_secret = ********************

try:
    auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_key, access_secret)
    auth.get_authorization_url()
    api = tweepy.API(auth)
except tweepy.TweepError:
    print('Hata')
```

İkinci adımda verilerin çekileceği twitter adresleri ve kaçar tane tweet çekileceği programa yazıldı. Adres ve tweet sayısını istenilen şekillerde bu kısımdan değiştirebiliriz.

```
public_tweets = api.user_timeline(screen_name="nytimes", count=20)
public_tweets2 = api.user_timeline(screen_name="CNN", count=20)
```

Ve tweetleri alıp bir dizi üzerine kaydetme işlemi başladı.

```
longtext = []

for tweet in public_tweets:
    tmp = re.sub(r"http\S+", "", tweet.text)
    longtext.append(tmp)

for tweet in public_tweets2:
    tmp = re.sub(r"http\S+", "", tweet.text)
    longtext.append(tmp)
    tweets = longtext.copy()
tweets = newsTweets(tweets)

for i in range(len(tweets)):
    print(i," ",tweets[i])
```

Projenin çözüm aşamasının ikinci kısmında veri setinin tanımlanması, incelenmesi ve düzenlenmesi kısımları var. İlk olarak veri setini programa yükleyip veri setinin özelliklerini yazdırmaya başlıyoruz. Ardından içeriğine göz

atıp, kullanmaya uygun olmayan dataları siliyoruz. Raporun başlangıç kısmında bu kod satırlarının, çıktılarının ve sonuç olarak elimizde kalan yeni veri setinin bilgisine ulaşabilirsiniz.

Veri setinin düzenlenmesi için yapılan adımlardan sonra text preprocessing işlemlerine başladık. Burada önce stopwords temizliği ve ardından lemmatization işlemleri gerçekleştirildi. Böylece verimiz classification işlemine hazır bir hale geldi.

Stopwords:

```
stop_words = set(stopwords.words('english'))

for i in range(len(news_articles_temp["headline"])):
    string = ""
    for word in news_articles_temp["headline"][i].split():
        word = ("".join(e for e in word if e.isalnum()))
        word = word.lower()
        if not word in stop_words:
          string += word + " "
    news_articles_temp.at[i,"headline"] = string.strip()
```

Lemmatization:

```
lemmatizer = WordNetLemmatizer()

for i in range(len(news_articles_temp["headline"])):
    string = ""
    for w in word_tokenize(news_articles_temp["headline"][i]):
        string += lemmatizer.lemmatize(w,pos = "v") + " "
    news_articles_temp.at[i, "headline"] = string.strip()
```

Twitterdan alınan veriler ve elimizdeki veri setinin içeriği hazır hale geldikten sonra kategorize işlemlerine başladık. Çözüm yöntemleri arasında yaptığımız testler sonrasında en başarılı sonucu “Word2Vec” yaklaşımı ile elde ettik. Bu sebeple programı bu yaklaşım üzerine geliştirdik.

Word2Vec, kelimeleri vektör uzayında ifade etmeye çalışan unsupervised (no labels) ve tahmin temelli(prediction-based) bir modeldir. Google araştırmacı Tomas Mikolov ve ekibi tarafından 2013 yılında icat edilmiştir.

Bu işleme başlamadan önce Word2Vec modelini uygulamaya yüklememiz gerekmektedir. Yükleme ve tanımlama işlemlerinden sonra her şey uygulamayı çalıştırmaya ve test etmeye hazır duruma gelecektir.

```
pickle.format_version
loaded_model =
gensim.models.KeyedVectors.load_word2vec_format('./GoogleNews-vectors-
negative300.bin', binary=True)


vocabulary = loaded_model.vocab
w2v_headline = []
for i in news_articles_temp['headline']:
    w2Vec_word = np.zeros(300, dtype="float32")
    for word in i.split():
        if word in vocabulary:
            w2Vec_word = np.add(w2Vec_word, loaded_model[word])
    w2Vec_word = np.divide(w2Vec_word, len(i.split()))
    w2v_headline.append(w2Vec_word)
w2v_headline = np.array(w2v_headline)
```

Programın kategorize işlemini gerçekleştirdiği fonksiyon:

```
def avg_w2v_based_model(row_index, num_similar_items):
    couple_dist = pairwise_distances(w2v_headline,
w2v_tweets_tmp[row_index].reshape(1,-1))
    indices = np.argsort(couple_dist.ravel())[0:num_similar_items]
    df = pd.DataFrame({'publish_date':

news_articles['date'][indices].values,
               'headline':news_articles['headline'][indices].values,
                'Category': news_articles['category'][indices].values,
                'Euclidean similarity with the queried article':
couple_dist[indices].ravel()})

    #return df.iloc[1:,1]
    return df.iloc[1:,[1,2]]
```

Kategorilerin alınacağı kısım:

```
def categoryCalculator(x):

    dizi = []
    for i in x['Category']:
        dizi.append(i)
    max= 0
    result = ""
    for i in dizi:
        maxTmp = dizi.count(i)
        if maxTmp>max:
            max = maxTmp
            result = i
    return result
```

Programın çalıştığı ve çıktı ürettiği kısım:

```
Categories = []
print ("      News Tweets
Recomended Category")
for i in range(len(w2v_tweets_tmp)):
    Categories.append(categoryCalculator(avg_w2v_based_model(i, 20)))
    print(i," ",tweets[i], "   ", Categories[i])
```

Hazırlanan yazılımın başarısını test edebilmek için de bir test kısmı hazırladık.:

```
def avg_w2v_based_model_test(row_index, num_similar_items):
    couple_dist = pairwise_distances(w2v_headline,
w2v_headline[row_index].reshape(1,-1))
    indices = np.argsort(couple_dist.ravel())[0:num_similar_items]
    df = pd.DataFrame({'publish_date':
news_articles['date'][indices].values,
               'headline':news_articles['headline'][indices].values,
                'Category': news_articles['category'][indices].values,
                'Euclidean similarity with the queried article':
couple_dist[indices].ravel()})

    return df.iloc[1:,[1,2]]
test = []

testCount = 0
for i in range(0,100):
    index = random.randint(0, 97000)
    testCategory = categoryCalculator(avg_w2v_based_model_test(index, 25))

    if testCategory.strip(" ").lower() ==
news_articles['category'][index].strip(" ").lower():
        testCount+=1

print("Total news number : 97000" )
print("Total number of news selected for category calculating : 25")
print("Total random selected news ( for calculating success rate ) : 100")
print("Success Rate is : %",testCount)
```

Burada program 100 tweeti kategorize edip, daha sonrasında kaçını doğru tahmin ettiğini kontrol ederek bize bir başarı yüzdesi veriyor. Yaptığımız denemeler sonucunda %40-%70 arasında değişen bir başarı yüzdesi ile programın çalıştığını gözlemledik. Proje başında gerçekleştirdiğimiz %50 başarı yüzdesine ulaşarak projeyi sonlandırdık.

Test Sonuçları:

Yaptığımız 10 farklı testin sonucunda elde ettiğimiz başarı yüzdeleri:

```
Total news number : 97000
Total number of news selected for category calculating : 25
Total random selected news ( for calculating success rate ) : 100
Success Rate is : % 57
```

```
Total news number : 97000
Total number of news selected for category calculating : 50
Total random selected news ( for calculating success rate ) : 100
Success Rate is : % 51
```

```
Total news number : 97000
Total number of news selected for category calculating : 100
Total random selected news ( for calculating success rate ) : 100
Success Rate is : % 45
```

```
Total news number : 97000
Total number of news selected for category calculating : 25
Total random selected news ( for calculating success rate ) : 100
Success Rate is : % 59
```

```
Total news number : 97000
Total number of news selected for category calculating : 25
Total random selected news ( for calculating success rate ) : 100
Success Rate is : % 52
```

```
Total news number : 97000
Total number of news selected for category calculating : 50
Total random selected news ( for calculating success rate ) : 100
Success Rate is : % 61
```

```
Total news number : 97000
Total random selected news ( for calculating success rate ) : 1000
Success Rate is : % 47
```

```
Total news number : 97000
Total number of news selected for category calculating : 25
Total random selected news ( for calculating success rate ) : 100
Success Rate is : % 56
```

```
Total news number : 97000
Total number of news selected for category calculating : 25
Total random selected news ( for calculating success rate ) : 100
Success Rate is : % 62
```

```
Total news number : 97000
Total number of news selected for category calculating : 25
Total random selected news ( for calculating success rate ) : 100
Success Rate is : % 64
```

Yapılan 10 testin sonucunda ortalama %55.4 başarı oranıyla program amacına ulaşmıştır.

Linkler:
Kullanılan Google vector tablosu:
https://drive.google.com/file/d/0B7XkCwpI5KDYNlNUTTlSS21pQmM/edit

Kullanılan Dataset:
https://drive.google.com/file/d/1sTGXy8xMVPYOdHYfWraWzOiQvbvArW43

Dataseti hazırlayan kişi:
https://rishabhmisra.github.io/experience/

Proje linki:
https://drive.google.com/file/d/1NcE6Ih9U12LOJwtRv3Fxd-Veu0i8htjn

Kullanılan Kaynaklar:
https://medium.com
https://stackabuse.com
http://docs.tweepy.org/en/latest/
https://realpython.com/python-keras-text-classification/
https://www.kaggle.com
https://scikit-learn.org/stable/
https://developers.google.com/machine-learning/guides
https://monkeylearn.com/text-classification/
http://docs.tweepy.org/en/latest/
https://realpython.com/python-keras-text-classification/
https://www.tensorflow.org/tutorials/text/word2vec
https://docs.python.org/3/library/pickle.html