T.C.
ONDOKUZ MAYIS ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Makine Öğrenmesine Giriş

Proje Raporu

Kopya Yakalama Aracı

15060765
Ebubekir Doğan

Danışman

Prof.Dr – Erdal KILIÇ

OCAK / 2019

SAMSUN

İÇİNDEKİLER

ÖNSÖZ

Makine öğrenimi,bilgisayarların algılayıcı verisi ya da veritabanları gibi veri türlerine dayalı öğrenimini olanaklı kılan algoritmaların tasarım ve geliştirme süreçlerini konu edinen bilim dalıdır.Makine öğrenimi araştırmalarının odaklandığı konu bilgisayarlara karmaşık örüntüleri algılama ve veriye dayalı akılcı kararlar verebilme becerisi kazandırmaktır.İstatistik,olasılık kuramı,veri madenciliği,örüntü tanıma,yapay zeka,uyarlamalı denetim ve kuramsal bilgisayar bilimi gibi alanlarla yakından ilintili olduğunu göstermektedir.

ÖZET

Teknolojinin gelişmesi ile her geçen gün insanlar bu gelişime paralel olarak çeşitli uygulamalara ihtiyaç duymaktadır.Makine Öğrenmesi(Machine Learning)'de bunlardan sadece bir tanesi.Son yıllarda kullanım alanı ve popülerliği artan makine öğrenmesi sayesinde çeşitli problemler hem daha kısa sürede hesaplanmakta hemde zeka gerektiren uygulamalar bilgisayarlar aracılığıyla hesaplanabilir hale gelmiştir.İnsan gibi öğrenme ve eğitilmeye müsait olan makine öğrenmesi düzgün eğitildiği halde bir insanın yapabileceği şeyleri daha doğru ve hızlı hesaplar hale gelmiştir.Bu uygulamada ise kopya yakalama aracı,makine öğrenimi algoritmaları kullanarak yapılmıştır.

1) Amaç

Uygulamadaki temel amaç kopya yakalamaktır.Biraz daha açarsak bu tanımı;

Yüzlerce binlerce adet verinin/dosyanın elimizde bulunduğunu varsayalım.Bu verilerin/dosyaların birbirleri arasında ilişki var mı,bireysel mi yapılmış yoksa birbirinden mi kopyalanmış bunun kararını vermek,ayrıca bunun kararını verirken de verilerin/dosyaların birbirlerine ne oranlarda benzerlik bulunduğunu,bu uygulamada yapmak amaçlanmıştır.

2) Kullanılan Kütüphaneler

a) Algoritma : KNN Algoritması
b) Vector Bulma : Count Vectorizer Kütüphanesi
c) Bağlaç vs. Gereksiz Kelimelerin Metinden Atılması : nltk.corpus > stopwords
d) Vectorden 0 içerikli değerlerin atılması : PCA
e) Kelimele Kökleri : TurkishStemer Kütüphanesi
f) Cosinus Uzaklığı
g) Tkinder Kütüphanesi

a)KNN Algoritması

Bu algoritmanın adı K komşu algoritmasıdır.Çalışma mantığı nesnelerin birbiri arasındaki yakınlık ilişkilerine göre kümeleme işlemi yapar.Doğrusal ayrıştırma yöntemi ile koordinat düzleminde çalışır.

b) Count Vectorizer Kütüphanesi

Bu kütüphane verilen dizideki kelimelerin kaç adet geçtiğini ve bunun sonucunun bize yine bir dizi olarak geri döndürülmesini sağlar.

Aşağıdaki örnekten daha iyi anlayabiliriz.

```
>>> from sklearn.feature_extraction.text import CountVectorizer
>>> corpus = ['Ben Ebubekir Dogan','19 Mayis Universitesinde okuyorum']
>>> vectorizer = CountVectorizer()
>>> X = vectorizer.fit_transform(corpus)
>>> print(vectorizer.get_feature_names())
['19', 'ben', 'dogan', 'ebubekir', 'mayis', 'okuyorum', 'universitesinde']
>>> print(X.toarray())
[[0 1 1 1 0 0 0]
 [1 0 0 0 1 1 1]]
>>>
```

c) NLTK stop words

Bu kütüphanemiz ise bize Türkçe'deki bağlaç veya fazlalık kelimelerin bulunmasını sağlayan kütüphanedir.

```
>>> from nltk.corpus import stopwords
>>>
>>> stop_words = set(stopwords.words('turkish'))
>>> print(stop_words)
{'tüm', 'her', 'hepsi', 'için', 'defa', 'az', 'nerde', 'birkaç', 'acaba', 'biri', 'bu', 'aslında', 'diye', 'hem', 'birşey',
 'kim', 'çok', 'o', 'kez', 'ne', 'gibi', 'de', 'belki', 'siz', 'ya', 'şey', 'hep', 'niçin', 'ise', 'sanki', 'da', 'mu', 'mü
', 've', 'ile', 'biz', 'eğer', 'nasıl', 'ama', 'mı', 'veya', 'nereye', 'niye', 'nerede', 'şu', 'hiç', 'daha', 'ki', 'yani',
 'neden', 'en', 'çünkü', 'bazı'}
>>>
```

Bu kütüphane sayesinde dosyalarımızı okuduktan sonra elde ettiğimiz içerik verilerinden stop_words kelimelerini yazdığım method sayesinde elimizdeki içerik verilerinden bu kelimeleri atıyoruz ve bize diğer kelimeler kalıyor.

d) PCA(Principal component analysis)

Verilerin tekil değer ayrışımını kullanarak doğrusal boyutsallığın azaltılması, onu daha düşük boyutlu bir alana yansıtmak için kullanılır.

Yani kısacası bizim yüklediğimiz verilerimizdeki boyutun azaltılmasını sağlar.Daha açık bir tabir ile söylemek gerekirse dizideki 0 sayılarının diziden atılmasını sağlar.Birkaç veride sıkıntı olmaz ancak veri boyutu büyüdükçe dizinin bellekte tutacağı yer artacağından dolayı PCA bir nebzede olsa bize kolaylık sağlar.

e) Turkish Stemmer

Kelimelerin köklerini bulmamızı sağlayan bu kütüphanede ise dizi olarak verdiğimiz kelime listesinin bize köklerine ayrılmış halini verir.Örnek çalışma şekli ;

```
C:\Users\EbubekirDgn>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:32:41) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from TurkishStemmer import TurkishStemmer
>>> stemmer = TurkishStemmer()
>>> stemmer.stem("bugünlerde")
'bugün'
>>>
```

f) Cosinus Uzaklığı

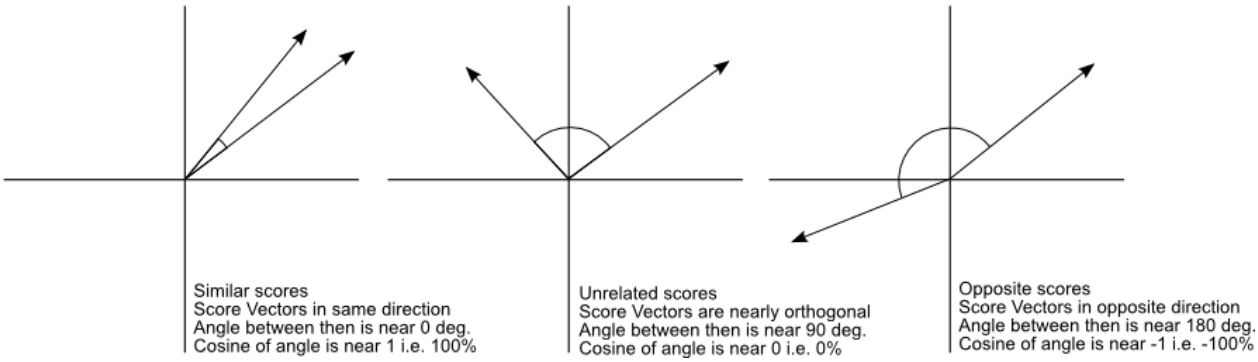
Uygulamada, kosinüs benzerliği, iki metin / belgenin ne kadar benzer olduğunu belirlemeye çalışan kütüphanedir.

İki vektör arasındaki kosinüs benzerliği (veya Vektör Uzayındaki iki belge), aralarındaki açının kosinüsünü hesaplayan bir ölçüdür. Bu metrik, bir yönelim ölçüsüdür, normalize edilmiş bir alandaki belgeler arasında bir karşılaştırma olarak görülebilir, çünkü her bir belgenin her kelime sayısının (countVectorizer)yalnızca büyüklüğünü dikkate almıyoruz, ancak belgeler arasındaki açı. Kosinüs benzerlik denklemini oluşturmak için yapmamız gereken nokta ürününün denklemini aşağıdakiler için çözmektir $\cos\theta$:

$$\vec{a} \cdot \vec{b} = \|\vec{a}\|\|\vec{b}\| \cos\theta$$

$$\cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\|\|\vec{b}\|}$$

İşte bu, bu kosinüs benzerliği formülü. Cosine Benzerliği, aşağıdaki örneklerde olduğu gibi, büyüklük yerine açıya bakarak iki belgenin ne kadar ilişkili olduğunu söyleyen bir ölçüm üretecektir:



Farklı belgeler için Cosine Benzerlik değerleri, 1 (aynı yön), 0 (90 °), -1 (zıt yönler).

Başka bir vektörden uzak bir noktaya işaret eden bir vektöre sahip olsak bile, yine de küçük bir açıya sahip olabileceğini ve bu Cosine Benzerliğinin kullanımındaki merkezi noktadır, ölçümün belgelerdeki yüksek terim sayısını göz ardı etme eğiliminde olduğunu unutmayın. “Gökyüzü” kelimesinin 200 kez göründüğü bir belgeye ve “gökyüzü” kelimesinin 50 göründüğü bir belgeye sahip olduğumuzu ve aralarındaki Öklid mesafesinin daha yüksek olacağını, ancak açının hala küçük olacağını, belgeleri karşılaştırırken bunun bizim için önemli olduğudur.

Projede cosinus hesapladığımız kod :nbrs = NearestNeighbors(n_neighbors=number, algorithm='auto', **metric='cosine'**).fit(x[testSayisi:])

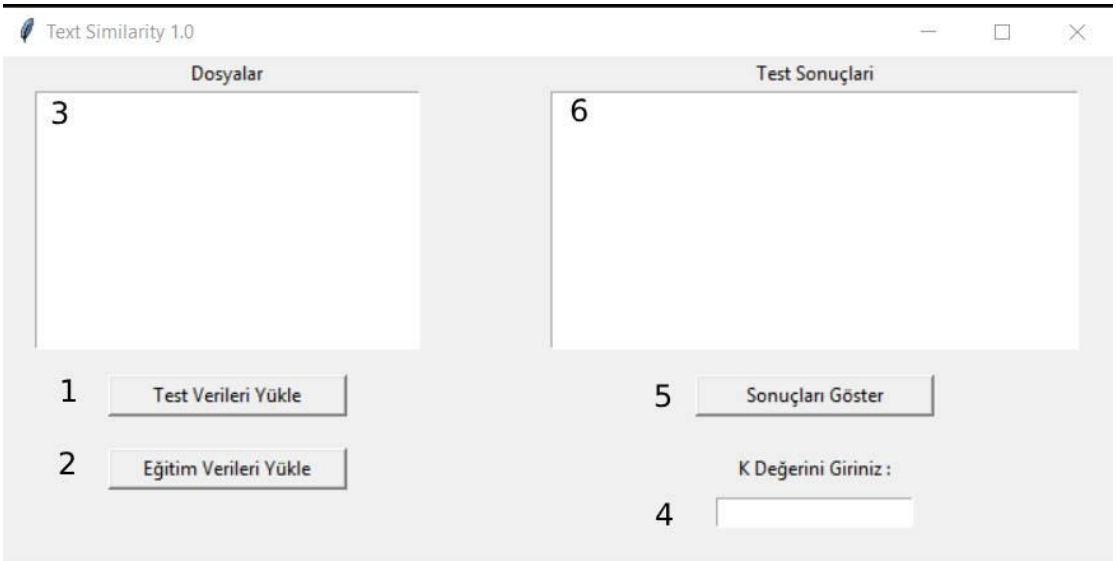
Metric değeri olarak cosine verdiğimizde KNN algoritmamıza gerekli işlemleri kendisi yapmaktadır.

g) Tkinder Kütüphanesi

Tkinter, Python programlama dili ile birlikte gelen grafiksel kullanıcı arayüzü aracıdır. Python'la birlikte gelmesi ve basit bir yapıya sahip olması, Tkinter'in yaygın kullanımına neden olmuştur.

Proje dosyasının en üstüne from tkinder import * diyerek projemize ekleyip grafiksel ekran oluşturmamızı sağlar

3) Uygulama Nasıl Çalışıyor?



1. **Test Verilerini Yükle butonu:** Test verilerini çoklu(multiple) yüklediğimiz buton.
2. **Eğitim Verilerini Yükle butonu:** Eğitim verilerini çoklu(multiple) yüklediğimiz buton.
3. **Dosyalar :** Burada test ve eğitim verisi olarak yüklediğimiz dosyalar listeleniyor.
4. Knn algoritmamızdan istediğimiz en yakın K verisinin sayısını istiyoruz.
5. **Sonuçları Göster butonu**: K değerini alıp gerekli işlemler sonrasında Test Sonuçlarını listeliyor.
6. **Test Sonuçları :** 'Sonuçları Göster' butonuna bastığımızda sonuçları burada gösteriyor.

Uygulamamızın arayüzü yukarıdaki gibidir.Bu uygulamada kelimelerin benzerliği yol haritası ile Kopya Yakalama Aracı yapılmaya çalışıldı.

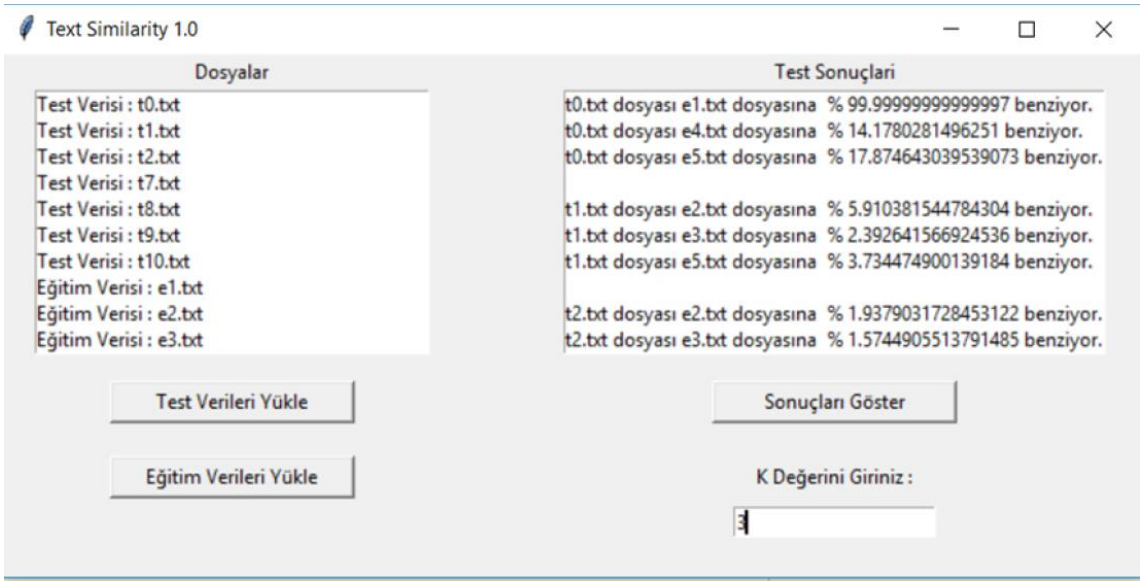
Uygulamada ilk önce kullanıcı test verilerini sisteme yükledikten sonra,bu verilerin vectorleri bulup bir diziye kaydedildi.Sonra kullanıcımız 'Eğitim Verileri Yükle' butonuna tıklayarak bunlarında vectorleri de ayrı bir diziye kaydediliyor.

Sonra bu iki dizinin birbirleri ile cosinüs uzaklıkları bulup birbileri ile karşılaştırılıyor.Bunun sonucunda bize 0-1 arasında benzerlik değeri üretiyor.Bunun sonucunda hangi eğitim verimiz,hangi test verimizie ne kadar oranda benziyor bulmuş oluyoruz.Bunun üzerinde kısa bir hesaplama işlemi yapıp bu 0- 1 arasındaki benzerlik oranını yüzdelik(%) dilim olarak bulmuş oluyoruz.

4) Uygulama Veri Seti

Uygulama içerisinde kullandığımız verileri haber sitelerinin köşe yazarlarından rastgele seçilmiş olup birbirleri ile alakaları yoktur.Ancak kimi konular(teknoloji,siyaset,magazin) benzer nitelikli olduğu için bu uygulamada bunlar arasındaki temel benzerlik oranlarını bulmayı hedefledik.

5)Örnek Çalışma Görüntüsü



Örnek ekran görüntüsünde yukarıda görüldüğü üzere **7 adet test verisi** üzerinde ve **5 adet eğitim verisi** üzerinde işlemler yapmış bulunuyoruz.

K değerimizi 3 verdiğimizde ; aşağıdaki sonuçları elde ediyoruz.

Test Sonuçları adlı ListBox'ın içerisinin tamamı ;

t0.txt dosyası e1.txt dosyasına % 99.99999999999997 benziyor.

t0.txt dosyası e4.txt dosyasına % 14.17802814962512 benziyor.

t0.txt dosyası e5.txt dosyasına % 17.874643039539052 benziyor.

t1.txt dosyası e2.txt dosyasına % 5.910381544784238 benziyor.

t1.txt dosyası e3.txt dosyasına % 2.392641566924558 benziyor.

t1.txt dosyası e5.txt dosyasına % 3.7344749001392064 benziyor.

t2.txt dosyası e2.txt dosyasına % 1.9379031728452012 benziyor.

t2.txt dosyası e3.txt dosyasına % 1.5744905513792373 benziyor.

t2.txt dosyası e5.txt dosyasına % 5.928729943347721 benziyor.

t7.txt dosyası e2.txt dosyasına % 1.0743966765323965 benziyor.
t7.txt dosyası e5.txt dosyasına % 3.824250058050649 benziyor.
t7.txt dosyası e3.txt dosyasına % 7.11598809729721 benziyor.

t8.txt dosyası e5.txt dosyasına % 4.469131420443095 benziyor.
t8.txt dosyası e3.txt dosyasına % 5.511687720099623 benziyor.
t8.txt dosyası e2.txt dosyasına % 13.981273204963585 benziyor.

t9.txt dosyası e3.txt dosyasına % 2.419081331851536 benziyor.
t9.txt dosyası e2.txt dosyasına % 7.096286821580033 benziyor.
t9.txt dosyası e1.txt dosyasına % 10.75313696777318 benziyor.

t10.txt dosyası e1.txt dosyasına % 68.53759656949467 benziyor.
t10.txt dosyası e5.txt dosyasına % 17.06431689716672 benziyor.
t10.txt dosyası e4.txt dosyasına % 17.786181479483187 benziyor.

Şeklinde sonuç veriyor.

KAYNAKÇA

https://github.com/otuncelli/turkish-stemmer-python

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.pairwise.cosine_similarity.html

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

https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

https://kenanatmaca.com/knn-k-nearest-neigborhood-algoritmasi/https://pythonspot.com/nltk-stop-words/

https://www.yazilimbilisim.net/python/python-tkinter-ornekleri/

https://tr.wikipedia.org/wiki/Tkinter