

[**Mühendislik Fakültesi**](https://www.konya.edu.tr/muhendislikvemimarlik)

[**Bilgisayar Mühendisliği Bölümü**](https://www.konya.edu.tr/bilgisayarmuhendisligi)

**Doğal Dil İşlemeye Giriş Dersi Proje Formu**

|  |
| --- |
| **Proje Başlığı** |
| Ödev Benzerlik Tespiti Uygulaması |

|  |  |
| --- | --- |
| **Öğrenci Bilgileri** | |
| **Öğrenci No** | 19010011007 |
| **Öğrenci Ad Soyad** | Muhammed Said TAYLAN |

|  |
| --- |
| **Arş. Gör. Dr.**  **Sinem ÇINAROĞLU** |

**Kasım 2022**

**Konya**

İçindekiler

[1. Problemin Belirlenmesi 2](#_Toc121781795)

[2. Problemin Çözümü ve Özgün Değer 2](#_Toc121781796)

[3. Projenin Amacı ve Hedefi 2](#_Toc121781797)

[4. Projenin iş-zaman çizelgesi 3](#_Toc121781798)

[5. Proje Adımları 4](#_Toc121781799)

[5.1. Doğal Dil İşlemenin Anlaşılması 4](#_Toc121781800)

[5.2. Doğal Dil İşleme Kavramlarının Öğrenilmesi 4](#_Toc121781801)

[5.3. Proje Araştırması ve Proje Tanıtımı 4](#_Toc121781802)

[5.4. Yapım Aşamaları 6](#_Toc121781803)

[6. SONUÇ 10](#_Toc121781804)

[KAYNAKÇA 10](#_Toc121781805)

# Problemin Belirlenmesi

Üniversiteler başta olmak üzere okullarda ödev tesliminde öğrenciler arasında kopya durumları sıkça meydana gelmektedir. Üniversitede bir öğretim üyesinin ya da okulda bir öğretmenin tüm ödevlerin (ne kadar) benzer olup olmadığını tespit etmesi oldukça zor bir durumdur. Kopya tespiti zor olduğu için kopya durumunun takip edilmemesi ise öğrencileri tembelliğe ve haksızlığa itmektedir. Birçok öğrenci kopyanın yanlışlığını ve doğuracağı sonuçları bilmesine rağmen bunları görmezden gelerek ödevde daha yüksek puan almayı hedeflemektedir. Bu yüzden dış bir etkenin zorlayıcı bir unsur olarak kullanılmasının kaçınılmaz olduğu görülmektedir.

# Problemin Çözümü ve Özgün Değer

Kopya sorununun engellenebilmesi için dış bir etkenin zorlayıcı bir unsur olarak kullanılmasının kaçınılmaz olduğu görülüyor. Bunun için ödevler arasında benzerliğin olup olmadığının, varsa ne ölçüde olduğunun tespitinin yapılması gereklidir. Bu tespit, yükü öğretmen üyesi ya da öğretmenlere yüklemeksizin bilgisayar yardımıyla yapılacaktır. Uygulamaya “metin dosyası” olarak girilen herbir öğrencinin ödevi diğer herbir öğrencinin ödevi ile manuel olarak tek tek karşılaştırılacaktır. İki metnin karşılaştırılması ise, Python dilinde en popüler doğal dil işleme kütüphanesi olan NLTK küttüphanesi kullanılarak gerçekleştirilecektir.

Uygulama, Django framework kullanılarak inşa edilmiş bir web arayüzü üzerinden kullanıcıya sunulur. Benzerlikleri bulunmak istenen iki dosya sisteme yüklenir ve işlem gerçekleştirilir.

Projede bir Python kütüphanesi olan Gensim kullanılmaktadır. Böylece dökümandaki tüm kelimeler indexlenerek bir Corpus veritabanı oluşturulmaktadır. Bu da sorgu işlemlerinin oldukça hızlı olacağı anlamına gelmektedir.

# Projenin Amacı ve Hedefi

Eğitim hayatında öğrenciler arasında ödevlerde kopya(benzerlik) olup olmadığını, varsa benzerlik oranını tespit edip kopya çektiği tespit edilen öğrencilere gerekli cezanın uygulanarak kopya çekme durumunun azaltılması hedeflenmektedir.

# Projenin iş-zaman çizelgesi

İş-zaman çizelgesi

|  |  |  |  |
| --- | --- | --- | --- |
| İP.  No. | İş Paketlerinin Adı ve Hedefleri | Zaman Aralığı (Haftalık) | Başarı Ölçütü |
| 1 | Doğal Dil İşlemenin Ne Olduğunu Anlaşılması ve Kavranması:  Doğal dil işlemenin ne işe yaradığı ve ne gibi faydaları olduğu öğrenilecektir. | 1 - 4. haftalar | Doğal dil işlemenin ne olduğu bilinmeden proje hakkında bir fikir dahi yürütülmesi mümkün olmayacağı için doğal dil işlemenin ne olduğunu anlaşılması ve kavranmasının projeye katkısı 20% oranındadır. |
| 2 | Doğal Dil İşleme Kavramlarının Öğrenilmesi:  Bir proje yapabilmek için elbette o konuya hakimiyet gerekmektedir. Bu yüzden doğal dil işlemenin temel kavramları öğrenilecek ve proje için yeterli donanıma sahip olunacaktır. | 5 - 8 hafta | Her şeyde olduğu gibi doğal dil işlemin de kendini özgü kavramları olacaktır. Bunların öğrenilmesi algoritmanın oluşturulmasına neden olacağı ve projenin kendisine etki edeceğinden, kavramların öğrenilmesinin projeye katkısı 30% oranındadır. |
| 3 | Projenin başlatılıp tamamlanması:  Proje, doğal dil işleme için en çok kullanılan Python diline tamamlanacaktır. Python’ın, doğal dil işleme için en popüler kütüphanesi olan NLTK kullanılacaktır. Bir web sitesi aracılığıyla uygulama kullanıcıya sunuculacaktır. | 9 – 11. haftalar | Projenin tamamlanmasını projeye katkısı 50% oranındadır. Proje başarılı olamasa dahi birçok şeyin öğrenilmesine fırsat verdiği için projenin başarı ölçütünün 100%’ünü kaplamamaktadır. |

# Proje Adımları

## Doğal Dil İşlemenin Anlaşılması

Doğal dil işlemenin tam olarak ne olduğunu ve ve amaçla kullanıldığını anlaşılabilmesi için Python’da bunun için kullanılan NLTK[1] ve spacy[2] kütüphaneleri araştırıldı. Bu kütüphanelerden NLTK, doğal dil işlemeyi gerçekleştirmek için kolayca kullanılabilen metotlar sunar. Bunların arasında cümleleri token’lerine ayırma, verilen cümlelerden sözlük oluşturma, verilen bir cümlenin token’lerine ayrıştırılıp sözlük içinde sorgulanması…

Yukarıda sayılan işlemler gibi pek çok işlem görüldükçe ve doğal dil işlemenin kullanım alanlarına araştırıldıkça, ayrıca”Doğal Dil İşlemeye Giriş” dersi boyunca görülen algoritmalar ve kullanım alanları da görüldükçe doğal dil işlemenin ne olduğu ve hangi amaçlarla kullanıldığı anlaşıldı. Önceki süreçte doğal dil işlemenin, yapa zekanın sıkı sıkıya bir türevi olduğu sanılıyordu. Bu sebeple, yapay zeka ile cümleler ayrıştırılabilecekken bir cümleyi basit fonksiyonlarla ögelerine ayırmak anlamsız geliyordu. Ancak doğal dil işleme sürecinin bundan farklı olduğu öğrenildi. Buradan çıkarımla, doğal dil işleme ile yapay zekaya bağlı olmaksızın insan dilinin en iyi şekilde bilgisayara aktarılıp işlemler yapılabilmesini kapsadığı anlaşılmış oldu.

## Doğal Dil İşleme Kavramlarının Öğrenilmesi

Doğal Dil İşlemeye Giriş dersi kapsamında anlatılan konularda dahil olan kavramlar, derste anlatılanın dışında detaylıca araştırıldı. Corpus/sözlük, stem, lemma, type, token bu kavramlardan bazılarıdır. Adım 1’de bahsedildiği üzere NLTK kütüphanesinde bu kelimeler sıklıkla kullanılmaktadır. Örneğin bir cümlenin kelimelerine ayrılması işlemine “sentence tokenize” adı verilmektedir. Token kavramına vakıf olunmadan NLTK kütüphanesinin bu fonksiyonunu kavrayabilmek mümkün değildir.

Doğal Dil İşlemeye Giriş dersinin zaman geçtikçe daha iyi anlaşılması, sorunların daha iyi tespit edilmesi bu süreci kolaylaştıran etkenlerdendi.

## Proje Araştırması ve Proje Tanıtımı

Adım 1 ve 2’de anlatıldığı üzere doğal dil işlemeye zaman geçtikçe daha aşina olunsa da bu sürede oluşan birikim, deneyimin mevcut olmadığı Python dilinde bir proje geliştirecek kadar yüksek değil. Bu sebeple mevcut bir proje üzerinde incelemeler ve bazı değişiklikler yapılarak doğal dil işleme ile bir uygulama ortaya konuldu. Uygun bir proje arama sürecinde Github, Gitlab ve Kaggle dahil olmak üzere birçok platform tarandı. Ancak karşılaşılan projeler genellikle oldukça büyük, kavranması zor, pek çok farklı dosyadan oluşmaktaydı. Bu yüzden farklı forum sitelerine, blog’lara da bakıldı. “Doğal Dil İşleme proje fikirleri” diye bir arama yapıldığında, arama sonuçlarında çıkan pek çok site Şekil 1’deki gibi bir içeriğe sahipti.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil

Durum her ne kadar böyle olsa da uygun bir proje bulunmakta uzun süre zorluk çekildi. Uzun süren araştırmalar sonucunda bir teknoloji blog yazısına[3] denk gelindi. Örnek projeler, bunların kodları ve (mevcutsa) eğitim videolarını paylaşıyordu. Yapılabilirliğinin en yüksek olduğu düşünülen bir projeye tıklanıldığında bir sayfaya[4] yönlendirildi. Bu sayfada projenin kodları mevcut ve kodlar yazılı olarak anlatılıyordu. Projede yazılım dili olarak Python ve arayüz olarak Python dilinin Framework’ü olan Django kullanılmaktaydı. Bir Django geçmişine sahip olunduğu üzere bu projede durulmaya karar verildi. Proje, bir web sayfasına yüklenen iki dosyanın benzerliğini karşılaştırmaktadır.

Proje tamamen Python dili ile inşa edilmiştir. Arayüz dışında doğal dil işleme kısmı için üç kütüphane kullanmaktadır: NLTK ve Gensim.

NLTK: Doğal dil işleme sürecine yardımcı olan, içinde pek çok hazır fonksiyon barındıran bir kütüphanedir.

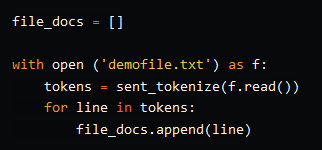
Gensim: Gensim, büyük korpora ile konu modelleme, belge indeksleme ve benzerlik alma için bir Python kütüphanesidir.[5]

NumPy: Matematiksel işlemler için hazır fonksiyonlar barındıran bir kütüphanedir.

NLTK kütüphanesi, web sayfası aracılığıyla gelen metinleri cümlelerine, ardından kelimelerine ayırdıktan sonra bu kelimelerden bir corpus(sözlük) oluşturmakta; Gensim kütüphanesi ise bu sözlükten bir Bag of Words oluşturup kelimelerin bulunma sıklıklarına göre diskte, index bazlı bir sorgu dokümanı oluşturmaktadır. Numpy kütüphanesi, listede toplama işlemi ve benzerlik ortalaması için kullanılmaktadır.

## Yapım Aşamaları

* + 1. Web arayüzünden gelen ilk dosyanın içeriğindeki cümleleri tokenlerine ayırmak yani içeriği cümlelere ayırmak için NLTK kütüphanesindeki “sent\_tokenize” fonksiyonu kullanıldı. Metindeki, ayrılan her bir cümle bir listeye eklendi.



Şekil

* + 1. Bir önceki adımda, metindeki cümlelerden oluşan listedeki her bir cümle, NLTK kütüphanesinin word\_tokenize fonksiyonu kullanılarak kelimelerine(token) ayrılarak yeni bir liste oluşturuldu. Sadece alfa karakterleden oluşan parçalar değil; kesme işareti, nokta gibi işaretler de birer token sayılmaktadır.

metin, siyah, portakal, karanlık içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil

* + 1. Cümlelerdeki tokenlerden oluşturulan listeden, Gensim kütüphanesinin corpora.Dictonary metodu kullanılarak bir sözlük üretildi. Bu sözlükteki her bir token’in eşsiz bir id bilgisine sahiptir.



Şekil



Şekil

* + 1. Oluşturulan sözlükten bir Bag of Words(corpus) üretildi. Bag of Word; her bir cümlede/paragrafta, bir kelimenin geçme sıklığıdır. Temel olarak her bir kelimeyi index’ler yani her bir kelimeye bir token atar. Böylece sözlükte, kelimelerin id bilgileri varken bundan sonra her bir kelimenin, metinde bulunma sıklığı da bag of word içinde mevcuttur.



Şekil

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil

* + 1. Üretilen corpus’un TF-IDF durumu hesaplanır. TFIDF(Terim Frekansı - Ters Belge Frekansı); bir bag of word modelidir. Terim frekansı(TF), sözcüğün belgede ne sıklıkta göründüğüdür ve ters belge sıklığı(IDF), sözcüğün korpusta ne kadar nadir olduğuna göre değeri ölçeklendirir. Basit bir ifadeyle, belgeler arasında daha sık ortaya çıkan kelimeler daha küçük ağırlıklar alır. Böylece web uygulaması tarafından sağlanan belgelerdeki benzerlik oranları TF-IDF ile kullanılarak iki dökümandaki benzer kelimelerin oranına göre hesaplanmaktadır. Eğer bir kelime iki dökümanda da benzer sayıda geçiyorsa bu ağırlık azalmaktadır. Eğer bir kelime, iki dökümanda da aynı sayıda geçiyorsa bu kelime, TF-IDF corpus’undan silinmektedir.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil

Yukarıda gösterilen örnekte, her üç cümlede de “This” ve “is” kelimeleri eşit sayıda bulunduğu için corpus’tan silinmiştir.

5.4.5. Bir önceki adımda oluşturulan TF-IDF objesi, Gensim tarafından sağlanan Similarity metodu ile alt indexlere ayrılarak bir matris şeklinde diske kaydedilir. İki döküman arasındaki sorgulama, kaydedilen dosya üzerinden yapılacaktır.



Şekil

* + 1. Diske kaydedilen corpus ile karşılaştırılmak istenen ikinci doküman, TF-IDF corpus’u kullanılarak Şekil 11’de gösterildiği gibi karşılaştırılabilmektedir.



Şekil

* + 1. Şekil 12 ve Şekil 13’te gösterildiği gibi iki farklı doküman olduğu varsayılırsa Şekil 14’teki gibi sonuç olarak üç farklı benzerlik değeri çıkacaktır. Bunlardan hangisinin alınması gerektiğini hesaplamak için bir ortalamaya başvurulur. Tam bu noktada NumPy kütüphanesi gitmektedir.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil



Şekil



Şekil

* + 1. Ortalamayı hesaplamak için ortaya çıkan benzerlik oranlarını Şekil 15’te gösterildiği gibi ilk olarak toplamak, ardından Şekil 16’daki gibi benzerlik sayısına bölmek ve yüzdeliğini hesaplamak gerekmektedir.



Şekil

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil

* + 1. Bunun sonucu olarak belirli değerler elde edilir. Son değer olarak ise istenildiği gibi iki dökümanın birbirine benzerlik oranı elde edilir.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil

* + 1. Bu adıma kadar anlatılanlar, gerçek olmayan basit örnekler üzerinden anlatıldı. Ancak uygulamanın gerçek benzerlik Şekil 18’de gösterildiği gibidir. Şekil 19 ve Şekil 20’deki iki metin karşılaştırılarak Şekil 21’de birden çok benzerlik sonucu elde edilmiş ve Şekil 22’de bunların ortalaması alınmış, ardından Şekil 23’de gösterildiği gibi bu sonuç Django framework’ü kullanılarak arayüz kullanılarak sunulmuştur.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil

metin, portakal, karanlık içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil



Şekil

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil



Şekil



Şekil

# SONUÇ

Ödevler arasında benzerlik tespti yapan bir uygulama üzerinden doğal dil işleme anlaşılmaya çalışılmış ve büyük ölçüde başarılı olunmuştur.

# KAYNAKÇA

1. [www.nltk.org](http://www.nltk.org)
2. [www.spacy.io](http://www.spacy.io)
3. [www.theclickreader.com/open-source-nlp-projects-with-tutorials/](http://www.theclickreader.com/open-source-nlp-projects-with-tutorials/)
4. [www.dev.to/thepylot/compare-documents-similarity-using-python-nlp-4odp](http://www.dev.to/thepylot/compare-documents-similarity-using-python-nlp-4odp)
5. [www.github.com/RaRe-Technologies/gensim](http://www.github.com/RaRe-Technologies/gensim)