Kocaeli Üniversitesi

Bilgisayar Mühendisliği Bölümü
Programlama Laboratuvarı II
Proje III

Emirhan Koç

200202068

I. GİRİŞ

Proje Python dilinde Microsoft Visual Studio Code IDE'si kullanılarak yazılmıştır.

Proje tasarlanırken Multithreading,Pandas kütüphanesi ve PyQt5 kütüphanesi ile ilgili teknik bilgiler araştırılmış ve gerekli algoritmaların kurulmasında karar kılınmıştır.

Müşteri şikayetleri kayıtlarının tutulduğu bir veri seti içerisindeki benzer kayıtlar tespit edilecek ve tespit edilen kayıtlar masaüstü uygulamasında gösterilecektir. Multithreading kullanarak benzerlik arama süresini düşürmek amaçlanmaktadır.

Amaç :

- Veri seti içerisindeki arama işlem süresini multithreading kullanılarak azaltmak.
- Belirtilen sütun/sütunlar için her bir satırdaki kayıtların birbiriyle kelime bazlı karşılaştırılması ve aralarındaki benzerliğin tespit edilmesi.
- Uygulama içerisinde istenen özelliklere göre kayıtları filtrelemek ve kullanıcıya göstermek.
- Masaüstü uygulama geliştirme hakkında bilgi ve beceriye sahip olmak.

Multithreading (çok iş parçacıklı çalışma), bir merkezi işlem biriminin (CPU) (veya çok çekirdekli bir işlemcideki tek bir çekirdeğin) aynı anda işletim sistemi tarafından desteklenen birden çok yürütme iş parçacığı sağlama yeteneğidir.

Bu tür programlamada birden çok iş parçacığı aynı anda çalışır. Çok iş parçacıklı model, sorgulamalı olay döngüsü kullanmaz. CPU zamanı boşa harcanmaz. Boşta kalma süresi minimumdur. Daha verimli programlarla sonuçlanır. Herhangi bir nedenle bir iş parçacığı duraklatıldığında, diğer iş parçacıkları normal şekilde çalışır.

Veri seti; finansal ürünler ve hizmetler hakkında alınan gerçek dünya şikayetlerini içermektedir. Veri seti, müşterilerin Kredi Raporları, Öğrenci Kredileri, Para Transferi vb. gibi finans sektöründeki birden fazla ürün ve hizmet hakkında yaptığı şikayetlerin farklı bilgilerini içermektedir.

Veri seti aşağıdaki kurallara uygun olacak şekilde yeniden düzenlenmelidir:

- Elde edilen tabloda 6 farklı sütun bulunmalıdır: Product (Ürün), Issue (Konu), Company (Şirket), State, Complaint ID, Zip Code.
- Null değer içeren kayıtlar bulunmamalıdır.
- Kayıtlardaki noktalama işaretleri kaldırılmalıdır.
- Kayıtlardaki stop word'ler kaldırılmalıdır (nltk kütüphanesi kullanılabilir).

Geliştirilecek projede tüm kayıtlar arasındaki benzerlik ilişkisinin incelenmesi beklenmektedir.Bu nedenle her bir kaydın diğer bir kayıtla karşılaştırılması gerekmektedir. Karşılaştırmanın mümkün olduğunca hızlı olması için multithread

kullanılmalıdır. Benzerlik, kayıtların içerdikleri ortak kelime sayısına göre olmalıdır.

- Verilen veri seti istenen şekilde yeniden düzenlenmelidir.
- Düzenlenmiş veri setindeki kayıtlar arasında benzerlik kontrolü yapılmalıdır. Kontrol sırasında mutlaka multithreading kullanılmalıdır. Multithreading için kullanılacak thread sayısı uygulama arayüzünden girilmelidir.
- Her thread'in çalışma zamanı ve tüm thread'ler için toplam çalışma zamanı bilgileri uygulama arayüzünde gösterilmelidir.
- İstenilen sütun ya da sütunlar arasındaki girilen benzerlik oranı (threshold) ve üzerinde benzerliğe sahip kayıtlar masaüstü uygulamasında gösterilmelidir.

- Uygulamanızı sunmak üzere basit bir arayüz geliştirmeniz istenmektedir.

II. YÖNTEM

- Başlangıçta proje detaylı bir şekilde analiz edildi.

- Python dili,multihreading,PyQt5 ve PyQt5 Designer hakkında gerekli araştırmalar yapıldı.

- Yapılması istenen multihreading yapısı ve bu yapıya atılacak olan fonksiyonlar dizayn edildi.

III. YALANCI KOD

- import csv.
- csv dosyasında işlem yapabilmek için kütüphane ekle.
- import string
- string değerlerle işlem yapabilmek için kütüphane ekle.
- import pandas
- import threading.
- csv dosyasını oku.
- csv dosyasını sınırlandır.
- def karsilastirma(ilkindeks,ikinciindeks).
- tanımla enbuyuk.
- if (len(ikinciindeks) ¿ len(ilkindeks))
- enbuyuk = len(ikinciindeks)
- tanımla benzersayisi.
- for item1 in ilkindeks:
- for item2 in ikinciindeks:
- if (item1 == item2):
- benzersayisi = benzersayisi + 1
- sonuc = (benzersayisi/enbuyuk) * 100
- return sonuc.

- def productkiyaslama():
- product = dataFrame['Product']
- tanımla ilkindeks.
- ilkindeks = product.iloc[sayi].lower().strip()
- tanımla parcalananbirinci
- parcalananbirinci = ilkindeks.split()
- for sayi2 in range(len(product)):
- tanımla ikinciindeks.
- ikinciindeks = product.iloc[sayi2].lower().strip()
- tanımla parcalananikinci.
- parcalananikinci = ikinciindeks.split(" ")
- benzerlikorani = karsilastirma(parcalananbirinci, parcalananikinci)
- bastır ilkindeks.
- bastır ikinciindeks.
- bastır benzerlikorani.
- bastır bosluk.
- def issuekiyaslama():
- tanımla issue.
- issue = dataFrame['Issue']
- for sayi in range(len(issue)):
- tanımla ilkindeks.
- ilkindeks = issue.iloc[sayi].lower().strip()
- tanımla parcalananbirinci.
- parcalananbirinci = ilkindeks.split()
- for sayi2 in range(len(issue)):
- tanımla ikinciindeks.
- ikinciindeks = issue.iloc[sayi2].lower().strip()
- tanımla parcalananikinci.
- parcalananikinci = ikinciindeks.split(" ")
- bastır ilkindeks.
- bastır ikinciindeks.

- bastır benzerlikorani.
- bastir bosluk.
- def companykiyaslama():.
- company = dataFrame['Company'].
- for sayi in range(len(company)):.
- ilkindeks = company.iloc[sayi].lower().strip().
- parcalananbirinci = ilkindeks.split().
- for sayi2 in range(len(company)):
- ikinciindeks = company.iloc[sayi2].lower().strip()
- parcalananikinci = ikinciindeks.split(" ").
- bastır ilkindeks.
- bastır ikinciindeks.
- bastır benzerlikorani.

IV. DENEYSEL SONUÇLAR

Yapılan işlemler ve gerçekleştirilien algoritmalar sonucunda ortaya çıkan sonuçlar şu şekildedir :

- Yazılan kıyaslama kodlarının multihreadinge uyum sağlayabilmesi için fonksiyonlaştırılması gerektiği görüldü.

- Aynı thread üzerinde tek bir çağırma işlemi yapılabildiğinden her işlem için birden fazla thread oluşturulması gerektiği görüldü.

- Yapılan işlemler sayesinde isterlerin gerçeklenmesi sağlandı.

V. SONUÇ

Yapılan işlemler sonucunda bizden istenilen veri düzenlemesi tamamen başarıyla gerçekleştirildi.Sonrasında düzenli veri üzerinde istenilen karşılaştırma algoritması başarıyla kuruldu ve fonksiyonların her birine başarıyla uygulandı.Düzenli verinin her bir sütunu için farklı bir kıyaslama kodu yazıldı ve fonksiyonlaştırılarak multihreading işlemine uygun hale getirildi.Aynı şekilde her bir sütun için ayrı ayrı multithreading kodu yazıldı ve gerekli görülen yerlerde çağrıldı.

VI. KAYNAKÇA

- https://mertmekatronik.com/thread-ve-multithread-nedir

- https://www.tutorialspoint.com/operatingsystem/osmultithreading.htm

- https://www.javatpoint.com/multithreading-in-java

- https://www.geeksforgeeks.org/multithreading-python-set-1/

- https://www.youtube.com/watch?v=QAVoXJtQ9QM

- https://www.youtube.com/watch?v=tRSHp8jsZdwt=162s

- https://www.youtube.com/watch?v=ZHSXaFfx84I