***candidate.py***

Özgeçmişin (resume) metnini alıp, belirli bir klasördeki iş ilanlarıyla karşılaştırarak hangi iş ilanlarının özgeçmişe en uygun olduğunu belirleyen bir Streamlit uygulaması oluşturur. Uygulama, metin işleme, doğal dil işleme (NLP), görselleştirme ve metin benzerliği hesaplamaları yapar.

Betiğin genel akışı şu şekildedir:

**• Streamlit Kullanıcı Arayüzü (UI)**:

• Kullanıcıya özgeçmiş yükleme ve iş ilanlarının bulunduğu klasör yolunu girmesi için arayüz sağlar.

• Kullanıcıya özgeçmişin metnini girme veya bir PDF olarak yükleme seçeneği sunar.

• Özgeçmiş ve iş ilanları arasında dil uyumluluğunu kontrol eder ve dil uyuşmazlığı durumunda uyarı verir.

• Karşılaştırma işlemi için bir buton sağlar.

**• Metin Çıkarma**:

**• textract** ve **pdfplumber** kütüphaneleri kullanılarak PDF belgelerinden metin çıkarılır.

**• pytesseract** kullanılarak PDF'den görüntüye dönüştürülen sayfalardan metin çıkarılır.

**• Dil İşleme**:

**• spacy** kullanılarak İngilizce ve Almanca dilleri için dil modeli yüklenir ve metinden terimler çıkarılır.

**• langdetect** kullanılarak metinlerin hangi dile ait olduğu tespit edilir.

**• Metin Benzerliği ve Görselleştirme**:

**• textdistance** kütüphanesi ile çeşitli benzerlik metrikleri hesaplanır.

**• WordCloud** ve **Plotly** kütüphaneleri ile görselleştirme yapılır.

**• SentenceTransformer** kullanılarak cümleler vektör haline getirilir ve kosinüs benzerliği hesaplanır.

**• Streamlit Kontrolleri**:

• Kullanıcı seçimlerine göre uygulamanın akışını yönlendirir.

• Sidebar üzerinde dil ve özgeçmiş formatı seçenekleri sunar.

• Kullanıcı tarafından sağlanan dosya ve klasör yollarını işler.

**• Uygulama Mantığı**:

• İş ilanlarıyla özgeçmiş arasında benzerlik skorları hesaplar.

• En yüksek skora sahip iş ilanının içeriğini görselleştirir.

**• Yardımcı Fonksiyonlar**:

• Metin formatlama, temizleme ve çıkarım işlemleri için yardımcı fonksiyonlar içerir.

**• Ana Fonksiyon (main)**:

• Uygulamanın giriş noktasıdır ve yukarıdaki tüm işlevleri koordine eder.

**• Metrik Tanımları**:

• Kullanılan metriklerin tanımları ve açıklamaları sağlar.

**Güvenlik ve Performans Notları:**

• Tesseract için yol sabit bir şekilde tanımlanmıştır, bu güvenlik açısından risk oluşturabilir ve portatif olmamaktadır.

• Uygulama, işlediği verilerin boyutuna ve iş ilanlarının sayısına bağlı olarak önemli miktarda sistem kaynağı tüketebilir.

**Kullanım Öncesi Yapılacaklar:**

• Gerekli kütüphanelerin yüklenmiş olması gerekir.

• Tesseract OCR yolu (**pytesseract.pytesseract.tesseract\_cmd**) sistemdeki Tesseract kurulum yoluna göre ayarlanmalıdır.

**• spacy** modelleri (**en\_core\_web\_lg** ve **de\_core\_news\_lg**)

**• spacy** modelleri (**en\_core\_web\_lg** ve **de\_core\_news\_lg**) indirilmiş ve kurulmuş olmalıdır.

**• SentenceTransformer** kütüphanesi için kullanılan model isimleri (**bert-base-nli-mean-tokens** ve **T-Systems-onsite/german-roberta-sentence-transformer-v2**) sistemde önceden indirilmeli veya bu modellerin yüklenmesi için internet erişimi gereklidir.

**Programın Çalışması:**

Programın çalışma zamanında gerçekleştireceği adımlar aşağıdaki gibidir:

**• Kullanıcı Arayüzü Hazırlığı**:

• Kullanıcıya sunulan seçenekler ve girdi alanları, Streamlit arayüzü üzerinden sağlanır. Kullanıcı, özgeçmişi metin olarak girebilir veya PDF formatında yükleyebilir. İş ilanlarının bulunduğu klasör yolu da kullanıcıdan alınır.

**• Metin Çıkarımı ve Dil Tespiti**:

• Kullanıcının yüklediği özgeçmiş metni ve iş ilanları, belirtilen fonksiyonlar aracılığıyla işlenir. Metinlerin hangi dile ait olduğu belirlenir.

**• Anahtar Terimlerin Çıkarılması**:

**• spacy** kullanılarak belirlenen dil modeline göre metinlerden isim, sıfat ve özel isimlerden oluşan anahtar terimler çıkarılır.

**• Benzerlik Hesaplamaları ve Görselleştirme**:

• Hesaplanan benzerlik metriklerine göre iş ilanlarının özgeçmişle uyumu değerlendirilir. En yüksek uyum gösteren iş ilanları görselleştirme araçları ile kullanıcıya sunulur.

**• Metrik Tanımları Sayfası**:

• Kullanıcının metrikler hakkında bilgi edinmesi için tanımlar ve açıklamalar sağlanır.

**• Dil Seçimi ve İş Akışı Kontrolü**:

• Kullanıcı tarafından seçilen dile (Almanca veya İngilizce) göre farklı iş akışları çalıştırılır. Her dil için farklı **SentenceTransformer** modelleri kullanılır.

**• Sonuçların Görselleştirilmesi ve Sunulması**:

• İş ilanları ile özgeçmiş arasındaki benzerlik puanları ve terimler görselleştirilerek kullanıcıya sunulur.

**• İş Akışı Yönlendirme**:

• Kullanıcının seçimlerine göre **run\_english\_version** veya **run\_german\_version** fonksiyonları çağrılır ve ilgili iş akışı gerçekleştirilir.

**• Çalışma Zamanı**:

**• if \_\_name\_\_ == "\_\_main\_\_":** satırı, betiğin doğrudan çalıştırılması durumunda **main** fonksiyonunun çağrılmasını sağlar. Bu, betiğin bir komut satırı aracılığıyla veya bir Python yorumlayıcısı üzerinden çalıştırılabilmesi için standart bir Python idiyomudur.

Bu betiğin başarılı bir şekilde çalışabilmesi için yukarıda belirtilen önkoşulların sağlanması ve gerekli kütüphane ve bağımlılıkların sistemde kurulu olması gerekmektedir. Betiğin kullanımı için Streamlit'in başlatılması ve bir web tarayıcısı üzerinden erişilmesi beklenir.

***Not:***

Bu koddaki benzerlik skoru hesaplama işlemi, özgeçmiş metni ve iş tanımları (job descriptions - JD) arasındaki ilişkiyi farklı metrikler kullanarak ölçmek için tasarlanmıştır. İşte adım adım açıklaması:

**• Anahtar Terimlerin Çıkarılması**:

• İş tanımı metninden (**JD**), **extract\_terms\_english** fonksiyonu kullanılarak anahtar terimler çıkarılır. Bu fonksiyon, metindeki isimleri, sıfatları ve özel isimleri (kendi adları) seçer.

**• Ortak Terimlerin Belirlenmesi**:

• Çıkarılan iş tanımı terimleri (**job\_description\_terms**) ile özgeçmiş terimleri (**resume\_terms**) arasındaki kesişim hesaplanarak ortak terimlerin kümesi (**common\_terms**) oluşturulur.

**• Metin Gömme (Embedding) İşlemi**:

Özgeçmiş metni ve iş tanımı metni, **SentenceTransformer** modeli kullanılarak vektörleştirilir (embed edilir). Bu işlem, metinlerin çok boyutlu sayısal gösterimini oluşturur ve bu sayede metinlerin anlam benzerliği hesaplanabilir.

**• Kosinüs Benzerliği Skoru**:

Oluşturulan gömme vektörleri arasındaki kosinüs benzerliği, **util.pytorch\_cos\_sim** fonksiyonu ile hesaplanır. Bu skor, iki vektör arasındaki açının kosinüsü alınarak bulunur ve 0 ile 1 arasında bir değerdir. 1'e yakın değerler yüksek benzerlik anlamına gelir.

**• Diğer Benzerlik Metrikleri**:

**• textdistance** kütüphanesi kullanılarak çeşitli benzerlik metrikleri hesaplanır:

**• Jaccard Benzerliği**: İki kümenin kesişiminin birleşimine oranıdır.

**• Sorensen-Dice Benzerliği**: İki kümenin kesişiminin iki katının, küme eleman sayılarının toplamına oranıdır.

**• Kozin Benzerliği**: İki metin kümesinin vektörleştirilmiş terim frekansları arasındaki kosinüs mesafesidir.

**• Örtüşme (Overlap) Benzerliği**: İki kümenin kesişiminin daha küçük kümedeki eleman sayısına oranıdır.

**• Genel Benzerlik Skoru**:

Hesaplanan tüm benzerlik metrikleri (**j**, **s**, **c**, **o**, ve **cosine\_similarity\_score**) toplanır ve 5'e bölünerek ortalama bir benzerlik skoru (**similarity\_score**) elde edilir. Bu skor, farklı ölçütleri dikkate alarak daha dengeli ve kapsamlı bir benzerlik değerlendirmesi sunar.

**• Sonuçların Kaydedilmesi**:

Hesaplanan skorlar ve diğer bilgiler, her bir iş ilanı için bir sözlükte (**dictionary**) toplanır. Bu sözlükler daha sonra **results** ve **advanced\_similarity\_scores** listelerine eklenir.

**• Sonuçların DataFrame'e Dönüştürülmesi ve Gösterimi**:

Toplanan sonuçlar, **pandas** DataFrame'ine dönüştürülür ve ortak terim sayısına göre sıralanır. **st.table** fonksiyonu kullanarak bu sonuçlar Streamlit arayüzünde bir tablo olarak gösterilir.

***Jobs.py***

Bu Python kodu, özgeçmişleri iş tanımıyla karşılaştırmak için tasarlanmış bir Streamlit uygulamasının bir parçası gibi görünüyor ve hem İngilizce hem de Almanca dillerini destekliyor. Uygulama, kullanıcıların iş tanımlarını girmelerini ve en iyi eşleşmeleri belirlemek için özgeçmiş yüklemelerini sağlayacak şekilde bir web uygulaması olarak çalışacak şekilde tasarlanmıştır. Kodun adım adım açıklaması şöyle:

**• Kütüphaneleri İçe Aktarma**: Dosya işlemleri, metin çıkarma, doğal dil işleme (NLP), veri görselleştirme ve makine öğrenimi gibi farklı görevleri ele almak için çeşitli Python kütüphaneleri içe aktarılır.

**• Konfigürasyon**:

• Tesseract, görüntülerden metin çıkarmak için bir OCR motoru olarak ayarlanır.

• İngilizce ve Almanca dillerindeki yaygın kelimeleri filtrelemek için kullanışlı olmayan kelimeleri ayıklamak üzere NLTK durak kelimeleri (stopwords) indirilir.

**• Yardımcı Fonksiyonlar**:

• Metin dilini tespit etmek, PDF'den metin çıkarmak, terimleri çıkarmak ve ortak terimleri bulmak gibi çeşitli görevler için yardımcı fonksiyonlar tanımlanır.

**• Görselleştirme Fonksiyonları**:

• Kelime bulutu, pasta grafikleri ve çubuk grafikleri oluşturmak için görselleştirme fonksiyonları tanımlanır.

**• Ana Fonksiyonlar**:

**• run\_english\_version** ve **run\_german\_version** fonksiyonları, uygulamanın İngilizce ve Almanca versiyonlarını çalıştırmak için kullanılır. Bu fonksiyonlar, kullanıcı girdilerini işler, özgeçmişleri iş tanımlarıyla karşılaştırır ve sonuçları görselleştirir.

**• Ana Akış**:

**• main** fonksiyonu, uygulamanın giriş noktasıdır ve kullanıcının dil seçimine göre uygun fonksiyonları çalıştırır.

**• Streamlit Öğeleri**:

• Kullanıcı arayüzü bileşenleri, Streamlit fonksiyonları kullanılarak oluşturulur. Kullanıcılar, kenar çubuğu üzerinden özgeçmiş klasör yolu girebilir, iş tanımı formatını seçebilir ve PDF veya metin olarak yükleyebilirler.

**• Dil İşleme ve Benzerlik Skoru Hesaplama**:

• Uygulama, özgeçmişleri iş tanımlarıyla karşılaştırmak için çeşitli NLP işlemleri ve benzerlik metrikleri kullanır. Örneğin, **SentenceTransformer** modeli kullanılarak metinlerin gömülü vektörleri oluşturulur ve bunların kozin benzerliği hesaplanır.

• Jaccard, Sorensen-Dice, Kozin ve Örtüşme benzerlik skorları hesaplanır ve özgeçmişlerin iş tanımlarına ne kadar benzediğini belirlemek için kullanılır.

**• Sonuçların Gösterilmesi**:

• Hesaplanan sonuçlar, Streamlit tabloları ve grafikler kullanılarak gösterilir. Kullanıcılar, özgeçmişlerin iş tanımına olan benzerliklerini çeşitli görsellerle inceleyebilirler.

**• Metrik Tanımları**:

• Kullanılan metriklerin tanımları ve matematiksel formülleri **metrics\_definitions** fonksiyonu ile açıklanır. Kullanıcılar, her bir metriğin ne anlama geldiğini ve nasıl hesaplandığını bu bölümden öğrenebilirler.

**• Kullanıcı Arayüzü Akışı**:

**main** fonksiyonu, uygulamanın başladığı yerdir ve kullanıcılara "Home" ve "Metrics Definitions" olmak üzere iki sayfa seçeneği sunar.

Kullanıcı "Home" sayfasını seçerse, bir dil seçimi yapması istenir ve seçimine bağlı olarak **run\_english\_version** veya **run\_german\_version** fonksiyonları çalıştırılır.

"Metrics Definitions" sayfasında, benzerlik metrikleri hakkında detaylı bilgiler sunulur.

**Dil Seçeneği ve Uygulama Akışı**:

• Kullanıcı tarafından belirtilen dil seçeneğine göre, uygulama İngilizce veya Almanca sürümünü çalıştırır. Her dil için, özgeçmişlerin ve iş tanımının diline uygun olarak metin işleme ve terim çıkarma işlemleri yapılır.

**Benzerlik Skorları ve Görselleştirme**:

Özgeçmişler ve iş tanımı arasındaki benzerlik skorları hesaplandıktan sonra, bu skorlar çeşitli görselleştirme teknikleriyle sunulur: pasta grafikleri, çubuk grafikleri ve yan yana grafikler.

Ayrıca, her özgeçmiş için gelişmiş benzerlik metrikleri de hesaplanır ve gösterilir.

**En Yüksek Skorlu Özgeçmişin Gösterilmesi**:

En yüksek benzerlik skoruna sahip özgeçmiş, kullanıcıya özel olarak belirtilir ve içeriğini görüntülemek için genişletilebilir bir panel sağlanır.

**Streamlit Uygulamasının Çalıştırılması**:

Eğer kod doğrudan çalıştırılırsa, yani **if \_\_name\_\_ == "\_\_main\_\_":** koşulu sağlanırsa, **main** fonksiyonu çağrılır ve uygulama başlatılır.

**Dil İşleme:**

Uygulama, PDF formatındaki özgeçmişleri ve iş tanımlarını işlemek için birkaç adım izler:

**• extract\_text\_from\_pdf**: PDF'den metin çıkarmak için kullanılır. **pdfplumber** kütüphanesi, PDF dosyalarından sayfa sayfa metin çıkarılmasını sağlar.

**• extract\_terms\_english** ve **extract\_terms\_german**: Bu fonksiyonlar, **spacy** kütüphanesi kullanılarak İngilizce veya Almanca metinlerden önemli terimleri (isimler, sıfatlar ve özel isimler) çıkarmak için kullanılır.

**• get\_common\_terms**: İki metin arasındaki ortak terimleri bulmak için kullanılır. Ayrıca, dil seçimine bağlı olarak durdurma kelimelerini (stop words) çıkarmak için **nltk** kütüphanesinden durdurma kelime listeleri kullanılır.

**Benzerlik Skoru Hesaplama:**

**• sentence\_transformers\_model.encode**: Bu adımda, cümle veya belge düzeyinde gömülü vektörler (embeddings) üretmek için **SentenceTransformer** modeli kullanılır. Özgeçmiş ve iş tanımı metinleri bu modele verilerek, her biri için yüksek boyutlu vektörler elde edilir.

**• util.pytorch\_cos\_sim**: Elde edilen gömülü vektörler arasındaki kosinüs benzerliği bu fonksiyon kullanılarak hesaplanır. Bu skor, 0 ile 1 arasında bir değer alır, 1'e yakın değerler yüksek benzerliği ifade eder.

Ayrıca, gelişmiş metrikler hesaplanır:

**• textdistance** kütüphanesi: Jaccard, Sorensen-Dice, Cosine ve Overlap benzerlik skorlarını hesaplamak için kullanılır.

**• textacy.extract.keyterms.textrank**: Özgeçmiş ve iş tanımı metinlerinden anahtar terimleri çıkarmak için kullanılır.

**• Word2Vec** ve **WmdSimilarity**: Word Mover's Distance (WMD) benzerlik skorunu hesaplamak için kullanılır. Bu, terimlerin anlamına dayalı olarak metinler arasındaki "mesafeyi" ölçer.

Hesaplanan tüm bu skorlar, her bir özgeçmiş için bir 'similarity\_score' olarak birleştirilir. Bu skor, çeşitli benzerlik metriklerinin ortalaması alınarak hesaplanır ve özgeçmişlerin iş tanımına ne kadar yakın olduğunu gösterir.

Sonuçlar, kullanıcıya sunulmadan önce bir pandas DataFrame içinde toplanır ve uygun görselleştirme fonksiyonları ile gösterilir.

***general.py***

Bu program, bir iş açıklaması ile bir özgeçmiş arasındaki uyumu değerlendiren ve bunu görselleştiren bir Streamlit uygulamasıdır. Programın çalışma akışı şu şekildedir:

**• Kütüphaneler ve Araçlar:** Program, PDF dosyaları işlemek, dil tespit etmek, metin ön işleme yapmak, dil modellerini kullanmak, benzerlik skoru hesaplamak ve görselleştirme yapmak için çeşitli Python kütüphanelerini ve araçlarını kullanır.

**• Dil Tercihi ve Dosya Yükleme:** Kullanıcı, değerlendirmek üzere bir dil seçer (İngilizce veya Almanca) ve hem iş açıklaması hem de özgeçmiş için PDF dosyalarını yükler.

**• Dil Tespiti:** Yüklenen PDF'lerin dili, **detect\_pdf\_language** fonksiyonu kullanılarak tespit edilir.

**• Metin Çıkarma:** PDF dosyalarından **extract\_text\_from\_pdf** fonksiyonu ile metin çıkarılır.

**• Ön İşleme:** Çıkarılan metinler, **preprocess** fonksiyonu ile küçük harfe dönüştürülür ve gereksiz karakterler temizlenir.

**• Benzerlik Değerlendirmesi:** İş açıklaması ve özgeçmiş metinleri, **evaluate\_similarity** fonksiyonunda çeşitli metrikler kullanılarak karşılaştırılır:

**• Ortak Terimlerin Bulunması:** İki metindeki ortak terimler **get\_common\_terms** fonksiyonu ile belirlenir.

**• Word Mover’s Distance (WMD):** **Word2Vec** modeli kullanılarak iki metin arasındaki anlam bazlı uzaklık ölçülür.

**• BERT-tabanlı Kosinüs Benzerliği:** BERT dil modelinden alınan gömülüler arasında kosinüs benzerliği hesaplanır.

**• Sentence Transformers ile Kosinüs Benzerliği:** Cümle dönüştürücüler kullanılarak metinlerin gömülüler arasında kosinüs benzerliği hesaplanır.

**• Sonuçların Ağırlıklandırılması:** Farklı benzerlik ölçümleri, belirlenen ağırlıklara göre ortalamalanır ve nihai benzerlik skoru elde edilir.

**• Sonuçların Gösterilmesi:** Hesaplanan benzerlik skorları ve diğer metrikler, kullanıcıya bir pandas DataFrame içinde tablo olarak sunulur.

**• Kelime Bulutu Görselleştirmesi:** Ortak terimlerin görselleştirilmesi için kelime bulutu oluşturulur ve bu Streamlit uygulaması arayüzünde gösterilir.

**• Streamlit Uygulaması:**

• Kullanıcı arayüzü, Streamlit ile oluşturulur ve kullanıcılar burada dosyalarını yükleyip, dilleri seçip, karşılaştırma yapabilirler.

• Eğer yüklenen iş açıklaması ve özgeçmiş farklı dillerdeyse, kullanıcıya bir uyarı mesajı gösterilir.

Bu uygulama, işe alım süreçlerini otomatikleştirmek ve özgeçmişlerin iş tanımlarına ne kadar uyduğunu objektif bir şekilde değerlendirmek için kullanılabilir.