**HAVELSAN Ekspres 2024 Yönerge Dokümanı Doğal Dil İşleme Kategorisi**

**Sunum Raporu**

1. Çok-sınıflı Metin Sınıflandırma:

Bu case için neler yaptığımız, hangi methodları kullandığımız aşama aşama eklenmiştir.

1. **Veri Hazırlama ve Yükleme:**
   * Verilerinizi Google Drive'dan yükleyip işlemeye başlıyoruz. Bu, veri kaynağınızın nerede olduğunu ve nasıl erişildiğini gösterir.
   * Veri yükleme işlemi sırasında, klasörleri dolaşıyor ve her bir dosyayı okuyarak içeriğini alıyoruz. Bu, veri toplama ve işleme adımlarını gösterir.
2. **NLP Modelinin ve Tokenizer'ın Yüklenmesi:**
   * **transformers** kütüphanesinden gerekli bileşenleri yüklüyoruz: model, tokenizer ve diğer yardımcı araçlar.
   * Model ve tokenizer'ı Türkçe duyarlı bir model ve tokenizer olarak yükleyerek, Türkçe metinlerle çalıştığınızı belirtiyoruz.
3. **Veri Ön İşleme:**
   * Veri içerisindeki stopwords ve special characterler temizleniyor.
   * Data augmentation(arttırma) aşamasında SMOTE kütüphanesini kullanarak sentetik verilerle f1 score minimuma indirmeyi hedefliyoruz.
   * Verileri kategorilere göre gruplandırıyoruz ve bunları bir DataFrame'e dönüştürüyoruz.
   * Kategorilerin dağılımını gösteren bir grafik oluşturuyoruz. Bu, veri setinizin dengesiz olup olmadığını gösterir.
4. **Etiket ve ID Eşleşmesi Oluşturma:**
   * Veri setindeki kategorilerin her biri için birer ID oluşturuyoruz. Bu, modelin kategoriler arasında dönüşüm yapabilmesi için gereklidir.
   * Ayrıca, ID'leri kategorilere geri dönüştüren bir eşleşme de oluşturuyoruz.
5. **Model Eğitimi İçin Veri Hazırlığı:**
   * Veri setinizi eğitim, doğrulama ve test alt kümelerine bölmek için işlemler yapıyoruz.
   * Tokenizer'ı kullanarak metin verilerinizi modele uygun hale getiriyoruz.
6. **Özelleştirilmiş DataLoader Oluşturma:**
   * PyTorch'ta kullanılmak üzere özelleştirilmiş bir DataLoader sınıfı tanımlıyoruz. Bu, modeli eğitirken ve değerlendirirken verileri işlemek için kullanılacaktır.
7. **Model Eğitimi:**
   * **Trainer** sınıfını kullanarak modelinizi eğitiyoruz. Bu adım, modelin veri setiniz üzerindeki performansını artırmak için ağırlıklarını güncelleyecektir.
8. **Değerlendirme ve Metrik Hesaplama:**
   * Modelin performansını ölçmek için belirli metrikleri hesaplamak için bir fonksiyon tanımlıyoruz.
   * Eğitim sonuçlarını değerlendiriyoruz ve bu metriklerin nasıl işlediğini görüyoruz.
9. **Modeli Saklama ve Yeniden Yükleme:**
   * Eğitilmiş modeli ve tokenizer'ı kaydediyoruz. Bu, ileride modeli yeniden kullanmak veya paylaşmak için önemlidir.
10. **Tahmin Yapma:**
    * Modeli kullanarak yeni metin verileri üzerinde tahminlerde bulunuyoruz.
    * Tahminlerin nasıl yapıldığını ve sınıflandırma sonuçlarını nasıl yorumladığınızı gösteriyoruz.

Kullanılan Model ve Kütüphaneler:

1. **BertForSequenceClassification**: Bu sınıf, BERT (Bidirectional Encoder Representations from Transformers) modelinin önceden eğitilmiş bir versiyonudur ve metin sınıflandırma görevleri için kullanılır. Metinlerin sınıflandırılması için bir dizi gizli katmanı ve sınıflandırma için bir çıkış katmanını içerir. Eğitim sırasında, bu model metinlerin önceden belirlenmiş bir dizi etiket arasından hangi kategoriye ait olduğunu belirlemek için kullanılır.
2. **BertTokenizerFast**: Bu sınıf, metinleri belirli bir maksimum uzunluğa ve modele uygun bir şekilde işlemek için kullanılan bir tokenizer'dır. BERT modeli, metinleri bir dizi belirli tokenlere dönüştürür ve bu dönüştürme işlemi, metinlerin BERT modeli tarafından anlaşılabilir hale gelmesini sağlar. BertTokenizerFast, bu işlemi hızlı bir şekilde gerçekleştirecek optimize edilmiş bir sınıftır.
3. **TrainingArguments**: Bu sınıf, modelin eğitimi sırasında kullanılacak çeşitli argümanları içerir. Örneğin, eğitim sürecinde hangi optimizasyon stratejisinin kullanılacağı, kaç epoch boyunca eğitimin gerçekleştirileceği, hangi metriklerin değerlendirileceği vb. gibi bilgileri içerir.
4. **Trainer**: Bu sınıf, modelin eğitimini ve değerlendirmesini kolaylaştıran bir arayüz sağlar. Trainer sınıfı, TrainingArguments içerisinde belirtilen ayarlar doğrultusunda modeli eğitir ve değerlendirir. Ayrıca, eğitim ve değerlendirme sırasında ilerlemeyi kaydetmek, loglamak ve modeli kaydetmek gibi işlevleri de gerçekleştirir.
5. **compute\_metrics() Fonksiyonu**: Bu fonksiyon, modelin performansını değerlendirmek için kullanılan metrikleri hesaplar. Genellikle doğruluk, hassasiyet, geri çağırma ve F1 skoru gibi metriklerin hesaplanmasını içerir.
6. **predict() Fonksiyonu**: Bu fonksiyon, modeli kullanarak belirli bir giriş metni için tahminlerde bulunur. Metin girişi, modelin sınıflandırması gereken bir metin olabilir. Bu fonksiyon, belirli bir metnin hangi kategoriye ait olduğunu tahmin etmek için modeli kullanır.
7. Anlamsal Arama

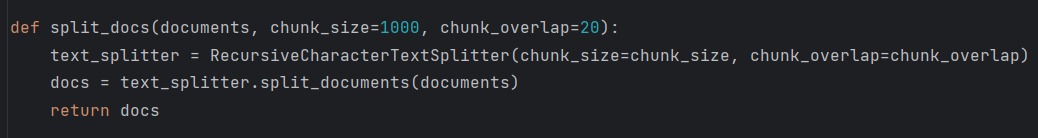
Bu sorunu çözerken kullanılan adımlar ve kavramlar:

1. **Veri Yükleme ve İşleme:** Google Drive'da bulunan bir CSV dosyasını yükleme ve işleme adımları. Bu adımlarda pandas kütüphanesi kullanılarak CSV dosyası okunur ve belirli sütunlardan (URL, başlık, metin) gerekli bilgiler alınır.

A computer screen with white text

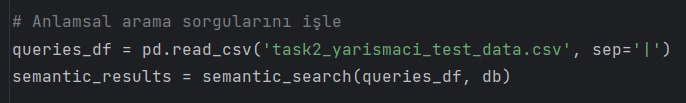
Description automatically generated

1. **Belge Parçalama (Chunking)**  Belge metinlerinin daha iyi işlenebilmesi için uygun boyutlara bölünmesi işlemi. Bu, metinlerin daha küçük parçalara ayrılmasını sağlar ve daha iyi analiz imkanı sunar.



1. **Metin Gömme (Embedding):** Belge metinlerinden metin gömme vektörlerinin oluşturulması adımı. Bu vektörler, belgelerin anlamsal özelliklerini temsil eder.
2. **Gömme Modelinin Yüklenmesi:**SentenceTransformerEmbeddings sınıfı, metinleri bir gömme uzayına dönüştürmek için kullanılan bir modeli yükler. Bu işlev, cümlenin anlamsal özelliklerini temsil eden vektörler elde etmek için kullanılır. Burada "all-MiniLM-L6-v2" modeli kullanılmış.
3. **Benzerlik Araması:** Oluşturulan metin gömme vektörlerini kullanarak belirli bir sorgu metni için en benzer belgelerin bulunması. Bu, belirli bir konuyla ilgili belgelerin keşfedilmesini sağlar.
4. **Anlamsal Arama Sorgularının İşlenmesi:**

semantic\_search işlevi, bir dizi anlamsal arama sorgusunu işler ve bu sorguları vektör veritabanında arar. Her sorgu için en benzer belgeleri bulur ve bunları bir sonuç çerçevesinde toplar. Bu işlev, sorgu ve sonuçlar arasındaki eşleşmeyi sağlar.



1. **Sonuçların İncelenmesi:** Verilen bir sorgu için en benzer belgelerin ve ilgili içeriklerinin incelenmesi adımı. Bu, kullanıcının sorgusuna en uygun belgeleri bulmasına yardımcı olur.

Kübra Çelebi-MCBU Bilgisayar Mühendisliği

Yasemin Koç-MCBU Bilgisayar Mühendisliği