BTK AKADEMİDE YAPILAN UYGULAMAYI ŞU ŞEKİLDE YAPTIM:

(TEKNİK BİR NEDENDEN DOLAYI FOTO PAYLAŞMAK DURUMUNDA KALDIM)

Btk akademide sertifika alınca yapılan uygulamayı şu şekilde yaptım:

BTK Veri Analizi ve Duygu Analizi Raporu

Proje Özeti

Bu proje, metin verilerini kullanarak duygu analizi yapan bir makine öğrenimi uygulamasıdır. Projenin temel amacı, metinleri pozitif ve negatif olarak sınıflandırmaktır.

Kullanılan Teknolojiler ve Kütüphaneler

pandas: Veri manipülasyonu ve analizi için

nltk: Doğal dil işleme işlemleri için

scikit-learn: Makine öğrenimi modelleri için

string: Metin işleme için

re: Düzenli ifadeler için

Proje Aşamaları

1. Veri Yükleme ve Ön İşleme

Veriler TSV formatında yüklendi

Veriler rastgele karıştırıldı

Sütun isimleri düzenlendi

Boşluklar temizlendi

2. Metin Temizleme

Noktalama işaretleri kaldırıldı

Türkçe stop words (etkisiz kelimeler) temizlendi

Temizlenmiş veriler CSV formatında kaydedildi

3. Veri Hazırlama

Veriler eğitim ve test setlerine bölündü (%90 eğitim, %10 test)

Etiketler (positive/negative) oluşturuldu

Metin verileri sayısal formata dönüştürüldü

4. Model Oluşturma ve Eğitim

CountVectorizer ile kelime sayımı yapıldı

TF-IDF dönüşümü uygulandı

Naive Bayes sınıflandırıcı kullanıldı

Model eğitildi

5. Model Değerlendirme

Test verileri üzerinde tahminler yapıldı

Doğruluk oranı (accuracy) hesaplandı

Her bir inceleme için tahmin sonuçları görüntülendi

Sonuçlar ve Değerlendirme

Model, test verileri üzerinde tahminler yaparak her bir metni positive veya negative olarak sınıflandırdı

Modelin performansı accuracy_score ile ölçüldü

Sonuçlar her bir metin için ayrı ayrı görüntülendi

Geliştirme Önerileri

1. Daha fazla veri ile model performansı artırılabilir

Farklı makine öğrenimi algoritmaları denenebilir

Metin ön işleme adımları geliştirilebilir

Cross-validation eklenebilir

Hyperparameter optimizasyonu yapılabilir

Sonuç

Bu proje, temel bir metin sınıflandırma sistemi oluşturarak, verilen metinleri pozitif ve negatif olarak sınıflandırmayı başarmıştır. Projenin başarısı, veri kalitesi ve model seçimi ile doğrudan ilişkilidir.

```python
import pandas as pd

# Doğru yoldan verileri okuma
data = pd.read_csv(r'C:\Users\PC\OneDrive\Desktop\NLP\BTK.txt', delimiter='\t')

# Verileri karıştırma ve dizini sıfırlama
data = data.sample(frac=1).reset_index(drop=True)

# İlk 5 satırı görüntüleme
data.head()
```

[1]

	review
0	DF g
1	DF G DFGDFHDHHYJUJ YUK
2	DFG
3	DFG
4	DFG

```python
# İlk 10 satırı daha iyi anlamak için görüntüleme
print(data.head(10))
```

[2]

```
                                        review
0                                         DF g
1                          DF G DFGDFHDHHYJUJ YUK
2                                          DFG
3                                          DFG
4                                          DFG
5                                          dF G
6              LKJPKD GFOIJ ERGOIJJRG[PO4GJFGLMDG
7  DGKNERGOOOGP EPGK PKO5]-2 WOJIJB9 [OR GWERGKK
8                      GR  POIEJG WERJG DFKM DFG
9                   DFGDFGKJ DPOFIGJ POIJDFG DFG
```

```python
# Veri yüklendikten sonra sütun adlarını görüntüleme
print(data.columns)
```

[3]

```
Index(['review '], dtype='object')
```

```python
# Sütun adındaki fazla boşluğu kaldırma
data.columns = data.columns.str.strip()

# Şimdi fazla boşluk olmadan sütunu kullanabilirsiniz
data['review'].value_counts()
```

[4]

```python
import string
import re
import nltk
from nltk.corpus import stopwords
noktalama = string.punctuation
etkisiz = stopwords.words("turkish")
print(noktalama)
print(etkisiz)
```

[5]

```
!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
['acaba', 'ama', 'aslında', 'az', 'bazı', 'belki', 'biri', 'birkaç', 'birşey', 'biz', 'bu', 'çok', 'çünkü', 'da', 'daha', 'de', 'defa', 'diye', 'eğer', 'en', 'gibi', 'hem', 'hep', 'he
```

```python
for d in data['review'].head():
    print(d + '.....................')
    temp = ''
    for word in d:
        if word not in etkisiz and not noktalama:
            temp += word
    print(temp + '\n***************')
    d = temp
```

[6]

```
DF g.....................

***************
DF G DFGDFHDHHYJUJ YUK .....................

***************
DFG .....................

***************
DFG .....................

***************
DFG .....................

***************
```

```
[7]
data.to_csv(r'./cleaned.csv', index=False)
```

```
[8]
#Temizlenmiş veriyi yüklüyoruz
import pandas as pd
data = pd.read_csv('cleaned.csv', sep=",", names=['review', 'label'])
print(data.head())
```

```
                   review  label
0                  review    NaN
1                    DF g    NaN
2  DF G DFGDFHDHHYJUJ YUK    NaN
3                    DFG     NaN
4                    DFG     NaN
```

```
[9]
# Sütun adlarındaki fazla boşluğu kaldırma
data.columns = data.columns.str.strip()

# Sütun adlarını kontrol etmek için yazdırma
print(data.columns)
```

```
Index(['review', 'label'], dtype='object')
```

```
[10]
from sklearn.model_selection import train_test_split

# Satır sayısını yazdırma
print(f"Satır sayısı: {len(data)}")

# Label sütununa uygun değerleri ekleme
data['label'] = ['positive' if i % 2 == 0 else 'negative' for i in range(len(data))]

# Verileri eğitim ve test olarak bölme
X_train, X_test, y_train, y_test = train_test_split(
    data['review'].values.astype("U"),
    data['label'].values.astype('U'),
    test_size=0.1,
    random_state=42
)

print(X_train.shape)
print(X_test.shape)
```

```
Satır sayısı: 11
(9,)
(2,)
```

```
from sklearn.feature_extraction.text import CountVectorizer
count_vect = CountVectorizer()
X_train_counts = count_vect.fit_transform(X_train)
print(X_train_counts.shape)
```

[11]

```
(9, 21)
```

```python
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB

# Metin verileriniz
X_train, X_test, y_train, y_test = train_test_split(
    data['review'].values.astype("U"),  # Metinler
    data['label'].values.astype('U'),   # Sınıflandırmalar
    test_size=0.1,
    random_state=42
)

# 1. Metinleri kelime sayımlarına dönüştürme
count_vect = CountVectorizer()
X_train_counts = count_vect.fit_transform(X_train)

# 2. Kelime sayımlarını TF-IDF temsiline dönüştürme
tfidf_transformer = TfidfTransformer()
X_train_tfidf = tfidf_transformer.fit_transform(X_train_counts)

# 3. Naive Bayes modelini eğitim verileriyle eğitme
clf = MultinomialNB().fit(X_train_tfidf, y_train)

# 4. Test verilerini TF-IDF'e dönüştürme
X_test_counts = count_vect.transform(X_test)
X_test_tfidf = tfidf_transformer.transform(X_test_counts)

# 5. Model ile tahmin yapma
predicted = clf.predict(X_test_tfidf)

print(predicted)
```

[12]

```
['positive' 'positive']
```

```python
y_pred = clf.predict(X_test_tfidf)
# Her incelemeyi tahmin edilen sınıflandırma ile yazdırma
for review, sentiment in zip(X_test, y_pred):
    print('%r => %s' % (review, sentiment))
```

[13]

```
'DFG ' => positive
'review' => positive
```

```python
from sklearn.metrics import accuracy_score
print(accuracy_score(y_test, y_pred))
```

[14]

```
0.5
```