İlaç Yorumları Sınıflandırma ve Analiz Projesi

Mahire Zühal Özdemir

İçindekiler

Proje Planı

- Öncelikle rating değerini **3 sınıfa düşürelim** ve veri setini hazırlayalım.
    - Sınıf Etiketleme (Rating'in 3 Sınıfa Düşürülmesi)
    - Olumlu (Positive): rating >= 7 → Etiket: 2
    - Nötr (Neutral): 4 <= rating < 7 → Etiket: 1
    - Olumsuz (Negative): rating < 4 → Etiket: 0
- Sınıflandırma Modeli
    - Hedef: Metin (review) ile rating_class arasında ilişki kurarak sınıflandırma yapmak.
    - Metin Ön İşleme
    - Küçük harfe dönüştürme
    - Noktalama işaretlerini kaldırma
    - Stopword'leri temizleme
    - Lemmatization/Stemming
- Konu Modelleme (Topic Modeling)
    - Hedef: review metinlerinde condition için hangi konuların öne çıktığını belirlemek.
    - Konu modellemede, hastaların yorumlarındaki ana temaları çıkaracağız.

Veri Önişleme ve Kontrolü

İlk olarak, projenin temel verileri olan üç CSV dosyası kontrol edilmiştir. Bu dosyalar şunlardır:

Veriseti: **https://www.kaggle.com/datasets/mohamedabdelwahabali/drugreview**

Bu veri kümesi, belirli ilaçlar ve ilişkili sağlık durumlarına yönelik hasta yorumlarını içerir ve genel hasta memnuniyetini yansıtan bir **10 yıldızlı derecelendirme sistemi** sunar. Çevrimiçi ilaç inceleme sitelerinden toplanan bu veriler, ilaç deneyimleri üzerindeki çeşitli analizleri desteklemek için hazırlanmıştır.

Veri kümesi şu temel özelliklerden oluşur:

- drugName (Kategorik): İlaç adı.
- condition (Kategorik): İlaçla tedavi edilen sağlık durumu.
- review (Metin): Hastaların ilaç hakkındaki yorumları.
- rating (Sayısal): Genel hasta memnuniyetini yansıtan 10 yıldızlı değerlendirme.
- date (Tarih): İnceleme giriş tarihi.
- usefulCount (Sayısal): Yorumu yararlı bulan kullanıcı sayısı.

Sınıflandırma ve Yeni Sütun Eklenmesi

Bu adımda, verisetinde bulunan rating (derecelendirme) sütunundaki hasta değerlendirme skorlarını pozitif, nötr ve negatif olarak sınıflandırdık. Bu sınıflandırma sonucunu yeni bir sütun olarak ekleyerek veriyi daha analiz edilebilir hale getirdik.

Yapılan İşlemler

1. **Fonksiyon Tanımlama:**

- Derecelendirme skorlarını sınıflandırmak için bir Python fonksiyonu oluşturuldu. Bu fonksiyon, her bir skorun belirli bir aralıkta olup olmadığına göre sınıf atar.
- Sınıflandırma kriterleri:
  - **Pozitif Yorumlar:** rating >= 7 → Sınıf: **1**
  - **Nötr Yorumlar:** 4 <= rating < 7 → Sınıf: **0**
  - **Negatif Yorumlar:** rating < 4 → Sınıf: **-1**

2. **Yeni Sütun Eklenmesi:**

- apply fonksiyonu kullanılarak her bir satırdaki rating değerleri sınıflandırıldı ve sonuçlar rating_class adıyla yeni bir sütuna eklendi.

3. **Dosyanın Kaydedilmesi:**

- Yeni oluşturulan veri çerçevesi, "drug_reviews_with_classes.csv" adıyla bir CSV dosyasına kaydedildi.

Verisetini Dengeleme

Verisetinde bulunan pozitif negatif ve nötr sınıfları arasında veri dengesizliği gözlemlenmiştir. Verileri dengelemek için pozitif sınıftan veri azaltımı yapılmalıdır. Semantik analiz için kullanacağım condition değeri incelenir. Top 5 condition seçilerek geri kalan condition değerleri silinir ve sınıf 5 'e düşürülür. Pozitif sınıftan veri silimi buna göre yapılır.

	Pozitif	Nötr	Negatif
Birth Control	4790	1747	2720
Depression	1933	318	510
Acne	1239	191	294
Anxiety	1211	147	231
Pain	1238	125	227
Toplam	**10411**	**2528**	**3982**

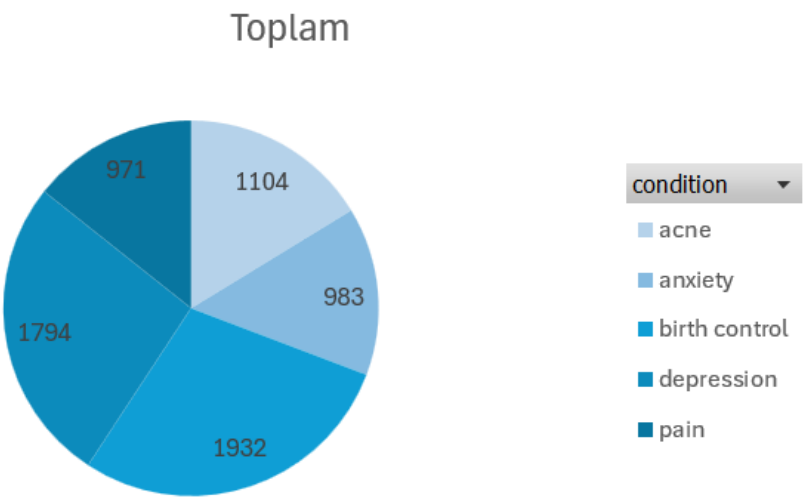
İlk olarak bu 5 consition dışında kalan conditionlar silinir.

Silinen conditionlar sonrası 16921 adet veri kaldı.

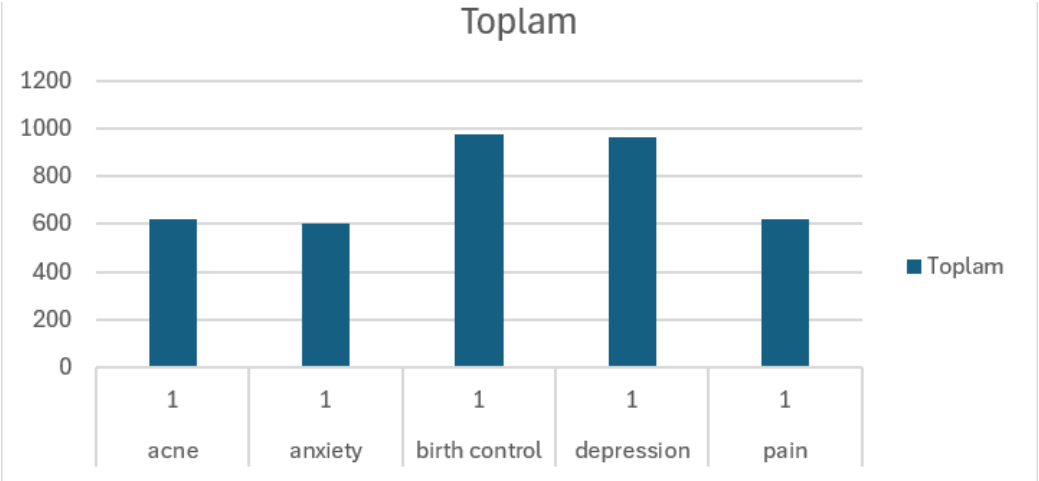
Pozitif sınıftan belirli condition değerlerini azaltalım. Birth control değerini yarıya düşürelim.

	Pozitif	Nötr	Negatif
Birth Control	978	500	454
Depression	966	318	510
Acne	619	191	294
Anxiety	605	147	231
Pain	619	125	227
Toplam	**3787**	**1281**	**1216**

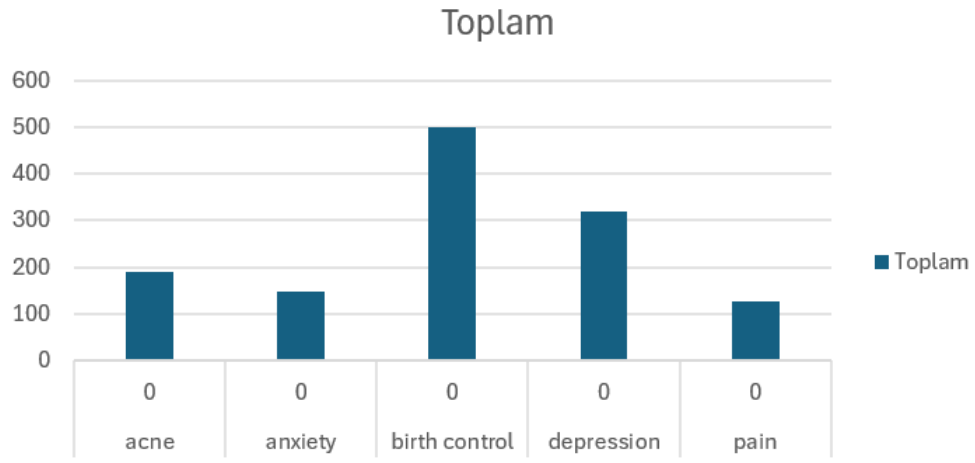
Toplamda 6284 veri kullanılarak modeller eğitilmiştir.



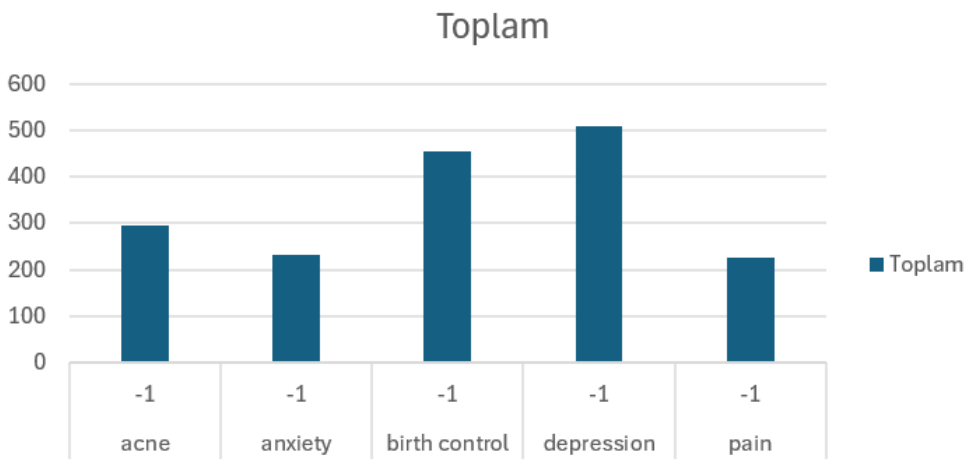
Şekil 1: Veriseti condition sayıları



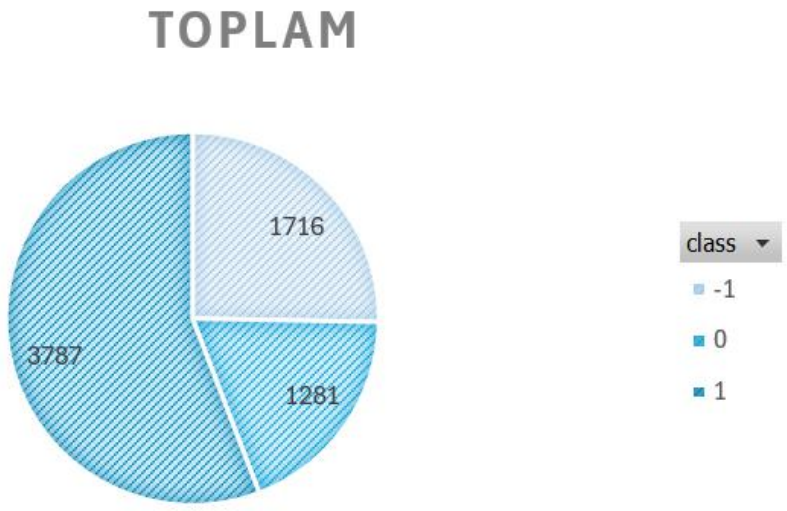
Şekil 2: Veriseti pozitif sınıf veri sayısı



Şekil 3: Veriseti nötr sınıf veri sayısı



Şekil 4: Veriseti negatif sınıf veri sayısı



Şekil 5: Veriseti sınıflara göre veri sayısı

Model Aşaması

Veri Hazırlığı: Metin Temizleme Süreci

İlk aşamada, veri setimizde bulunan kullanıcı yorumlarını (review text) temizleyerek modele uygun hale getirdik. Bu işlem sırasında şu adımlar gerçekleştirildi:

- **Küçük Harfe Dönüştürme**

Tüm metinler küçük harfe çevrildi. Bu sayede büyük-küçük harf farkından kaynaklanan tutarsızlıklar giderildi.

- **Noktalama İşaretlerini ve Özel Karakterleri Kaldırma**

Metinlerde yer alan noktalama işaretleri (.,!?) ve özel karakterler (@#$%) temizlendi. Böylece yalnızca anlamlı kelimeler üzerinde işlem yapılması sağlandı.

- **Durdurma Kelimelerinin (Stop Words) Çıkarılması**

İngilizce metinlerde sıkça geçen ancak modeli eğitmede anlamlı katkı sağlamayan kelimeler (the, is, in, vb.) çıkarıldı. Bu işlem için sklearn.feature_extraction.text.ENGLISH_STOP_WORDS kullanıldı.

- **Sonuçların Kaydedilmesi**

Temizlenmiş metinler, cleaned_text adında yeni bir sütun olarak aynı CSV dosyasına eklendi ve healthpulse_reviews_cleaned.csv adıyla kaydedildi.

Farklı metotlara göre sınıflandırma işlemi

Veri temizlendikten sonra lojistik regresyon ile sınıflandırma iişlemi yapıldı.

	precision	recall	f1-score	support
-1	0.66	0.73	0.70	797
0	0.52	0.31	0.39	556
1	0.72	0.82	0.77	990
accuracy			0.67	2343
macro avg	0.64	0.62	0.62	2343
weighted avg	0.65	0.67	0.65	2343

Şekil 6: Lojistik regresyon sınıflandırma sonuçları

Verilen bu değerler düşük bulundu. Farklı modeller denendi.

Random forest ile sonuçlar aşağıdaki gibi elde edildi.

	precision	recall	f1-score	support
-1	0.68	0.74	0.71	797
0	0.98	0.19	0.31	556
1	0.65	0.91	0.76	990
accuracy			0.68	2343
macro avg	0.77	0.61	0.59	2343
weighted avg	0.74	0.68	0.64	2343

Şekil 7: Random forest sınıflandırma sonuçları

SVM ile sonuçlar aşağıdaki gibi elde edildi.

	precision	recall	f1-score	support
-1	0.75	0.55	0.63	350
0	0.60	0.14	0.23	249
1	0.69	0.96	0.80	758
accuracy			0.70	1357
macro avg	0.68	0.55	0.56	1357
weighted avg	0.69	0.70	0.65	1357

Total Accuracy: 0.7008

Şekil 8: SVM sınıflandırma sonuçları

Model Semantik Analiz

Projede, her bir koşul (condition) için genel yorum analizi yapılması amacıyla BERT modeli kullanılmıştır. Bu model, her bir koşulun analiz edilerek genelleştirilmesini sağlamak için tercih edilmiştir. Yorum analizinin daha doğru ve anlamlı sonuçlar üretmesi için BERT'in güçlü dil işleme yeteneklerinden faydalanılmıştır. Eğitim sürecine ait ekran görüntüleri, kullanılan yöntemleri ve elde edilen sonuçları daha net bir şekilde göstermek adına aşağıda sunulmuştur.

```
Epoch 1/3: 100% 1172/1172 [13:56<00:00,  1.40it/s]
Epoch 1 - Training loss: 0.3553464668465193
Epoch 2/3: 100% 1172/1172 [13:58<00:00,  1.40it/s]
Epoch 2 - Training loss: 0.15075509828644879
Epoch 3/3: 100% 1172/1172 [13:57<00:00,  1.40it/s]
Epoch 3 - Training loss: 0.09343182577172679
Evaluating: 100% 293/293 [01:10<00:00,  4.13it/s]
              precision    recall  f1-score   support

           0       0.98      0.99      0.99      1376
           1       0.85      0.77      0.80       209
           2       0.85      0.92      0.88       377
           3       0.99      0.83      0.90       199
           4       0.93      0.95      0.94       182

    accuracy                           0.94      2343
   macro avg       0.92      0.89      0.90      2343
weighted avg       0.94      0.94      0.94      2343
```

Şekil 9: BERT eğitim aşaması - 1

```
 Original:  "i was on this pill for almost two years. it does work as far as not getting pregnant however my exp
Tokenized:  ['"', 'i', 'was', 'on', 'this', 'pill', 'for', 'almost', 'two', 'years', '.', 'it', 'does', 'work',
Token IDs:  [1000, 1045, 2001, 2006, 2023, 17357, 2005, 2471, 2048, 2086, 1012, 2009, 2515, 2147, 2004, 2521, 20
```

Şekil 10: BERT eğitim aşaması - 2

```
ertForSequenceClassification(
  (bert): BertModel(
    (embeddings): BertEmbeddings(
      (word_embeddings): Embedding(30522, 768, padding_idx=0)
      (position_embeddings): Embedding(512, 768)
      (token_type_embeddings): Embedding(2, 768)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (encoder): BertEncoder(
      (layer): ModuleList(
        (0-11): 12 x BertLayer(
          (attention): BertAttention(
            (self): BertSdpaSelfAttention(
              (query): Linear(in_features=768, out_features=768, bias=True)
              (key): Linear(in_features=768, out_features=768, bias=True)
              (value): Linear(in_features=768, out_features=768, bias=True)
              (dropout): Dropout(p=0.1, inplace=False)
```

Şekil 11: BERT eğitim aşaması - 3

```
            (dense): Linear(in_features=768, out_features=768, bias=True)
            (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
            (dropout): Dropout(p=0.1, inplace=False)
          )
        )
        (intermediate): BertIntermediate(
          (dense): Linear(in_features=768, out_features=3072, bias=True)
          (intermediate_act_fn): GELUActivation()
        )
        (output): BertOutput(
          (dense): Linear(in_features=3072, out_features=768, bias=True)
          (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
          (dropout): Dropout(p=0.1, inplace=False)
        )
      )
    )
  )
  (pooler): BertPooler(
    (dense): Linear(in_features=768, out_features=768, bias=True)
    (activation): Tanh()
  )
)
```

Şekil 12: BERT eğitim aşaması - 4

```
The BERT model has 201 different named parameters.

==== Embedding Layer ====

bert.embeddings.word_embeddings.weight                  (30522, 768)
bert.embeddings.position_embeddings.weight                (512, 768)
bert.embeddings.token_type_embeddings.weight                (2, 768)
bert.embeddings.LayerNorm.weight                              (768,)
bert.embeddings.LayerNorm.bias                                (768,)

==== First Transformer ====

bert.encoder.layer.0.attention.self.query.weight          (768, 768)
bert.encoder.layer.0.attention.self.query.bias                (768,)
bert.encoder.layer.0.attention.self.key.weight            (768, 768)
bert.encoder.layer.0.attention.self.key.bias                  (768,)
bert.encoder.layer.0.attention.self.value.weight          (768, 768)
bert.encoder.layer.0.attention.self.value.bias                (768,)
bert.encoder.layer.0.attention.output.dense.weight        (768, 768)
bert.encoder.layer.0.attention.output.dense.bias              (768,)
bert.encoder.layer.0.attention.output.LayerNorm.weight        (768,)
bert.encoder.layer.0.attention.output.LayerNorm.bias          (768,)
bert.encoder.layer.0.intermediate.dense.weight           (3072, 768)
```

Şekil 13: BERT eğitim aşaması - 5

```
==== First Transformer ====

bert.encoder.layer.0.attention.self.query.weight          (768, 768)
bert.encoder.layer.0.attention.self.query.bias                (768,)
bert.encoder.layer.0.attention.self.key.weight            (768, 768)
bert.encoder.layer.0.attention.self.key.bias                  (768,)
bert.encoder.layer.0.attention.self.value.weight          (768, 768)
bert.encoder.layer.0.attention.self.value.bias                (768,)
bert.encoder.layer.0.attention.output.dense.weight        (768, 768)
bert.encoder.layer.0.attention.output.dense.bias              (768,)
bert.encoder.layer.0.attention.output.LayerNorm.weight        (768,)
bert.encoder.layer.0.attention.output.LayerNorm.bias          (768,)
bert.encoder.layer.0.intermediate.dense.weight           (3072, 768)
bert.encoder.layer.0.intermediate.dense.bias                (3072,)
bert.encoder.layer.0.output.dense.weight                 (768, 3072)
bert.encoder.layer.0.output.dense.bias                        (768,)
bert.encoder.layer.0.output.LayerNorm.weight                  (768,)
bert.encoder.layer.0.output.LayerNorm.bias                    (768,)

==== Output Layer ====

bert.pooler.dense.weight                                  (768, 768)
bert.pooler.dense.bias                                        (768,)
classifier.weight                                           (5, 768)
classifier.bias                                               (5,)
```

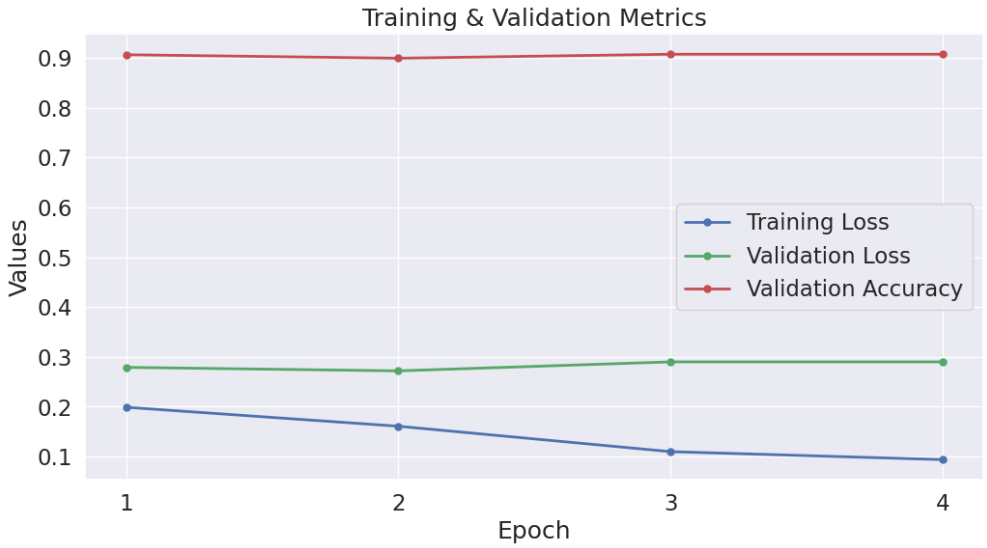
Şekil 14: BERT eğitim aşaması - 6

```
  Average training loss: 0.09
  Training epcoh took: 0:08:48

Running Validation...
  Accuracy: 0.91
  Validation Loss: 0.29
  Validation took: 0:00:19

Training complete!
Total training took 0:36:35 (h:mm:ss)
```

Şekil 15: BERT eğitim aşaması -7

epoch	Training Loss	Valid. Loss	Valid. Accur.	Training Time	Validation Time
1	0.199000	0.279000	0.906000	0:08:48	0:00:19
2	0.161000	0.272000	0.899000	0:08:48	0:00:19
3	0.110000	0.290000	0.907000	0:08:48	0:00:19
4	0.094000	0.290000	0.907000	0:08:48	0:00:19

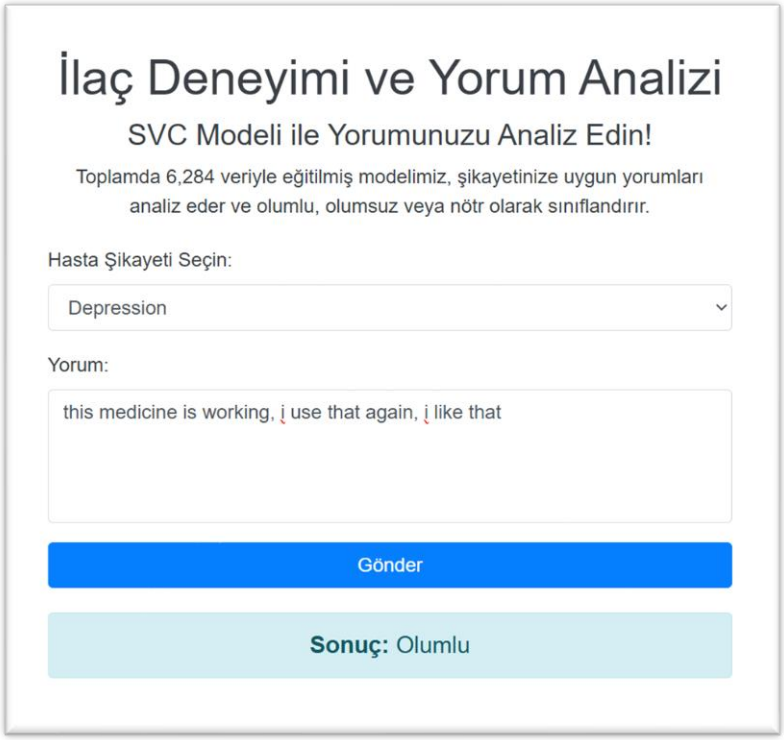
Şekil 16: BERT eğitim aşaması - 8



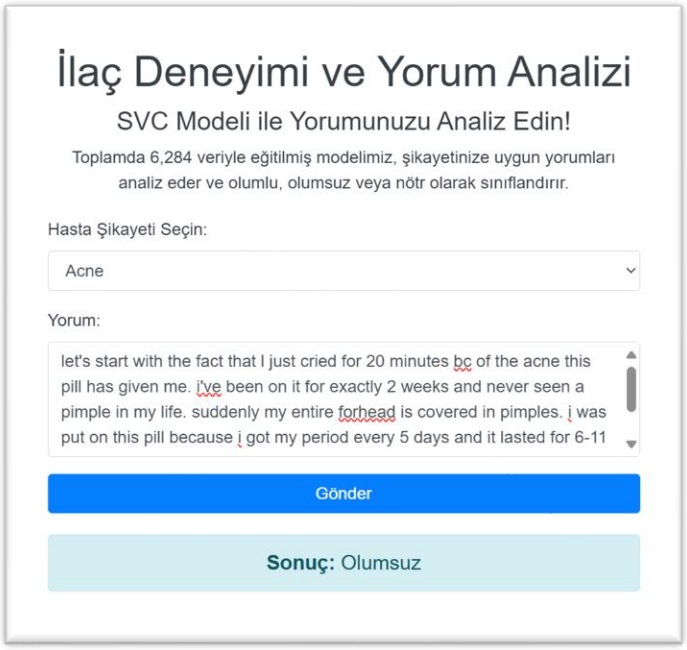
Şekil 17: BERT eğitim sonuçları

Projenin Çalışır Haline Ait ekran Görüntüleri

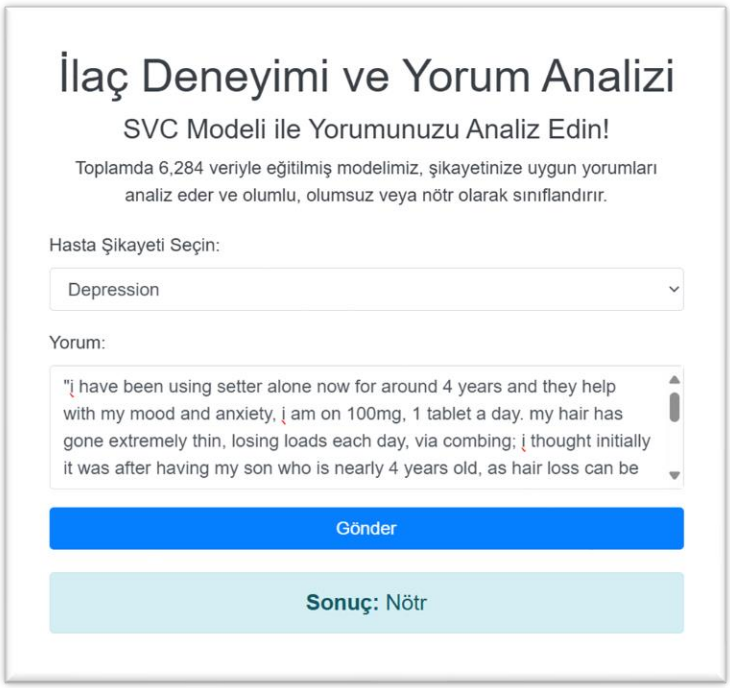
Projede, bir Flask framework'ü kullanılarak bir app.py dosyası oluşturulmuştur. Bu uygulama, kullanıcıların kolaylıkla erişebileceği ve veri işleme sürecini yönetecek bir arayüz sağlamak amacıyla geliştirilmiştir. Verilerin işlenmesi ve sınıflandırılması işlemi, önceden eğitilmiş bir modelin projeye entegre edilmesi yoluyla gerçekleştirilmiştir. Bu sayede, yüksek doğruluk oranı sağlayan bir sınıflandırma sistemi oluşturulmuş ve modelin eğitimi sırasında kazanılan bilgi, verilerin hızlı ve doğru bir şekilde sınıflandırılması için kullanılmıştır. Flask uygulaması, hem kullanıcı dostu bir arayüz sunmuş hem de arka planda model çağrısı ve veri işleme işlemlerini başarıyla yürütmüştür



Şekil 18: Uygulama görüntüsü - 1



Şekil 19: Uygulama görüntüsü - 2



Şekil 20: Uygulama görüntüsü - 3