


# Metin Üzerinden Mental Hastalık Tespiti



# Veri Seti

## Mental Health Corpus


Labeled sentences about depression and anxiety



[Data Card](#) [Code \(12\)](#) [Discussion \(3\)](#)

### About Dataset

The Mental Health Corpus is a collection of texts related to people with anxiety, depression, and other mental health issues. The corpus consists of two columns: one containing the comments, and the other containing labels indicating whether the comments are considered poisonous or not. The corpus can be used for a variety of purposes, such as sentiment analysis, toxic language detection, and mental health language analysis. The data in the corpus may be useful for researchers, mental health professionals, and others interested in understanding the language and sentiment surrounding mental health issues.

**Usability**   
10.00

**License**  
Attribution 4.0 International (CC ..)

**Expected update frequency**  
Never

**Tags**

[Text](#) [NLP](#)

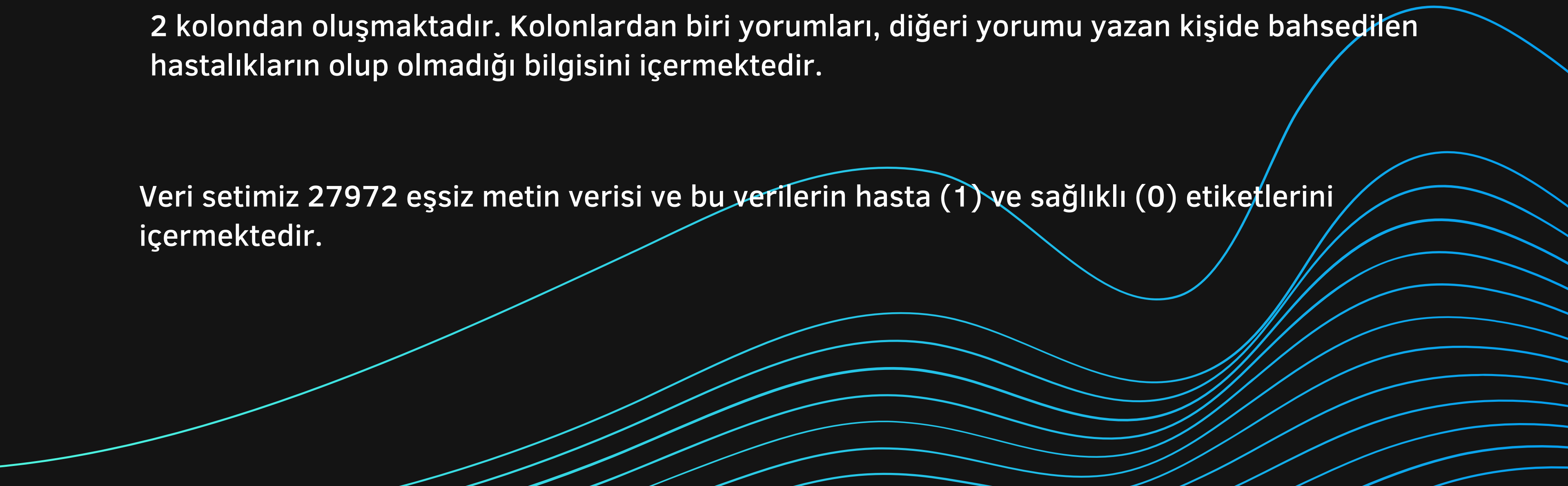
<https://www.kaggle.com/datasets/reihanenamdari/mental-health-corpus>

# Veri Seti

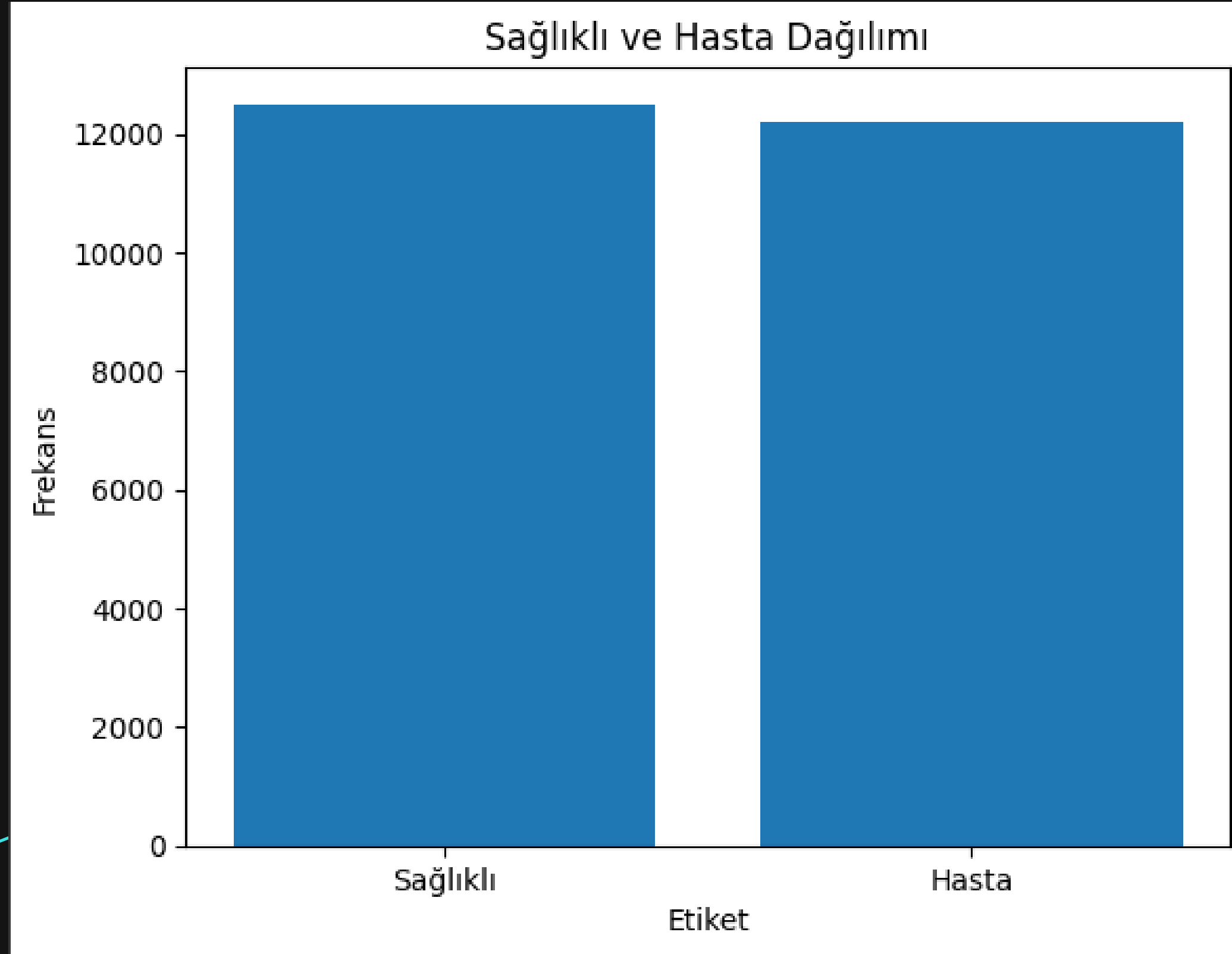
Anksiyete depresyon gibi mental sađlık hastalıklarına sahip olan ve olmayan insanların metinsel verilerinden oluşmaktadır.

2 kolondan oluşmaktadır. Kolonlardan biri yorumları, diğeri yorumu yazan kişide bahsedilen hastalıkların olup olmadığı bilgisini içermektedir.

Veri setimiz 27972 eşsiz metin verisi ve bu verilerin hasta (1) ve sağlıklı (0) etiketlerini içermektedir.



# Veri Seti



# Ön işleme Aşamaları

Metin madenciliğinde, ön işleme metin verilerini hazırlamak ve işlemek için kullanılan bir dizi teknik ve adımdır. Metin verilerinin analizi ve çıkarılması için önemlidir. Projemizde kullandığımız ön işleme aşamaları:

1-Metin Temizleme (Text Cleaning): Metin verilerindeki gereksiz karakterleri, noktalama işaretlerini, sayıları ve özel sembolleri kaldırmak için bu adım gerçekleştirilir. Ayrıca, büyük-küçük harf dönüşümü de yapılabilir.

2-Tokenizasyon: Metni küçük parçalara, yani "token"lara ayırmak için kullanılan bir yöntemdir. Genellikle kelimeler, cümleler veya paragraflar gibi dil birimleri için tokenlar oluşturulur.

3-Stop Words Kaldırma: Dilin yaygın ve anlamsız kelimeleri olan "stop words" (duraklama kelimeleri) kaldırılır. Örneğin, "bir", "ve", "ama", "veya" gibi kelimeler sıklıkla kaldırılan stop words'lardır.

4-N-Gram Oluşturma: Ardışık kelimelerin veya karakterlerin grupları olan n-gramlar oluşturulabilir. Bu, metin içindeki kelime veya karakter örüntülerini tanımak için kullanışlı olabilir.

5-Stemming veya Lemmatization: Stemming, kelimeleri kök formlarına dönüştürmeye yardımcı olur. Örneğin, "koşmak", "koşuyor" ve "koştı" kelimeleri "koş" köküne dönüştürülebilir. Lemmatization ise kelimeleri kelime köklerine indirgeme işlemidir.

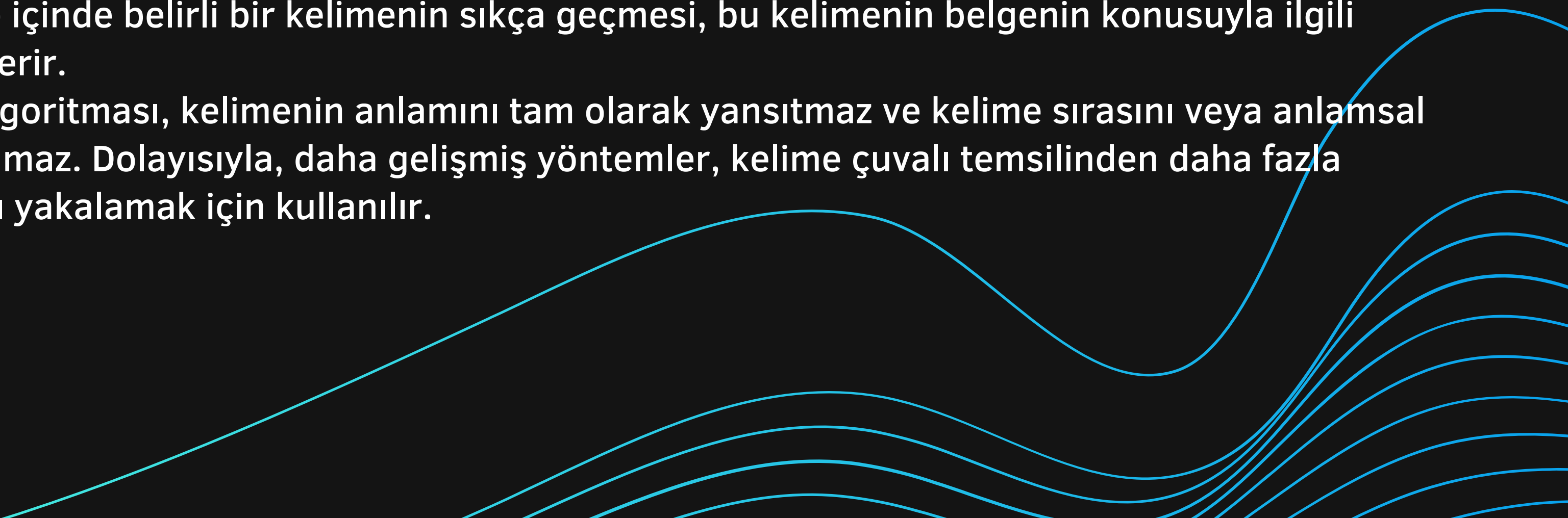
# Bag of Words Algoritması

"Bag of Words" algoritması, bir metni analiz etmek veya temsil etmek için kullanılan basit bir yöntemdir. Bu yöntemde, metin içeriği kelime düzeyinde işlenir ve her kelimenin sıklığı kaydedilir, ancak kelimenin sırası veya bağlamı göz ardı edilir.

Algoritmanın çalışma mantığı oldukça basittir: İlk adımda, metin parçalara ayrılır ve kelime düzeyinde tokenize edilir. Ardından, tüm kelimeler bir kelime çuvalına (bag) yerleştirilir ve her kelimenin metinde kaç kez geçtiği kaydedilir. Bu şekilde, metnin içeriği bir kelime frekans vektörü ile temsil edilir.

"Bag of Words" algoritması, kelime sıklıklarının metnin özelliklerini yansıttığı varsayımına dayanır. Örneğin, bir belge içinde belirli bir kelimenin sıkça geçmesi, bu kelimenin belgenin konusuyla ilgili olabileceğini gösterir.

"Bag of Words" algoritması, kelimenin anlamını tam olarak yansıtmaz ve kelime sırasını veya anlamsal ilişkileri dikkate almaz. Dolayısıyla, daha gelişmiş yöntemler, kelime çuvalı temsilinden daha fazla bağlamı ve anlamı yakalamak için kullanılır.





# TF-IDF Algoritması

"Tf-Idf" (term frequency-inverse document frequency) algoritması, bir belge koleksiyonunda belirli bir kelimenin önemini belirlemek için kullanılan istatistiksel bir yöntemdir. Tf-idf, bir kelimenin bir belgedeki frekansını (term frequency) ve tüm belgelerdeki yaygınlığını (inverse document frequency) dikkate alarak hesaplanır.

Tf-idf algoritmasının çalışma mantığı şu şekildedir: İlk olarak, bir belge koleksiyonu üzerinde metinler parçalara ayrılır ve her bir kelimenin belge içindeki sıklığı (term frequency) hesaplanır. Ardından, belge koleksiyonunda belirli bir kelimenin ne kadar yaygın olduğunu belirlemek için inverse document frequency (IDF) değeri hesaplanır. IDF, belirli bir kelimenin tüm belgelerdeki yaygınlığını ölçer ve nadir veya önemli kelimelerin daha yüksek skor almasını sağlar.

Son olarak, term frequency (TF) ve inverse document frequency (IDF) değerleri birleştirilerek tf-idf skoru hesaplanır. Bu skor, belirli bir kelimenin belge içindeki önemini temsil eder. Yüksek bir tf-idf skoru, belirli bir kelimenin belgedeki önemli bir terim olduğunu gösterirken, düşük bir skor ise o kelimenin belgedeki yaygın bir terim olduğunu gösterir.

Tf-idf algoritması, bilgi çıkarma, döküman sıralama, özetleme ve metin sınıflandırması gibi doğal dil işleme problemlerinde yaygın olarak kullanılır. Bu yöntem, belirli bir kelimenin belge içindeki önemini ve belgeler arasındaki farklılıkları vurgulamak için etkili bir şekilde kullanılır.

# Naive Bayes Algoritması

"Naive Bayes" algoritması, makine öğrenimi ve doğal dil işleme alanında yaygın olarak kullanılan bir olasılık tabanlı sınıflandırma yöntemidir. Naive Bayes, Bayes teoremi temel alınarak, girdi verilerini ve bu verilere dayalı sınıflandırma modellerini kullanarak tahminler yapar.

Naive Bayes algoritması, temel olarak sınıflandırma problemlerinde kullanılır ve verinin özelliklerini bağımsız olarak ele alır. "Naive" (saf) olarak adlandırılmasının sebebi, bu algoritmanın her özelliğin birbirinden bağımsız olduğunu varsayan basit bir varsayıma dayanmasıdır.

Multinomial Naive Bayes, metin sınıflandırması gibi kategorik özelliklerin olduğu problemlerde yaygın olarak kullanılır. Örneğin, bir belgenin içinde geçen kelime frekanslarını temsil eden vektörlerle çalışır.

Bernoulli Naive Bayes, sadece varlık veya yokluk gibi ikili özelliklere sahip verilerle çalışır. Örneğin, bir metinde belirli kelimelerin geçip geçmediğini temsil eden ikili vektörlerle kullanılır.

Her iki tür de Naive Bayes'in birer alt kümesidir ve verilerin yapısına ve problemin gerekliliklerine göre tercih edilebilirler.

Naive Bayes algoritmaları, basitlikleri ve hesaplama hızları nedeniyle yaygın olarak kullanılır. Ayrıca, eğitim veri seti boyutu büyüdükçe daha iyi performans gösterebilirler. Ancak, bağımsızlık varsayımı nedeniyle bazen bazı ilişkileri veya etkileşimleri yakalayamayabilirler.



# CNN Algoritması

"CNN" (Convolutional Neural Network - Evrişimli Sinir Ağı) algoritması, özellikle görüntü işleme alanında etkili bir derin öğrenme modelidir., özellikle görsel verilerin analizi için başarılı olan bir derin öğrenme modelidir. Ancak metin verilerinin analizi için de uyarlanabilir. CNN, metin verilerini işlemek ve özelliklerini çıkarmak için farklı boyutlardaki evrişim katmanlarından oluşur.

CNN'in metin verilerindeki çalışma prensibi şu şekildedir:

1. Gömme Katmanı (Embedding Layer): İlk olarak, metin verilerini sayısal bir formata dönüştürmek için gömme katmanı kullanılır. Bu katman, kelimeleri vektör temsillerine dönüştürür. Bu temsiller, kelime dağarcığındaki her bir kelimenin benzersiz bir sayısal vektörle temsil edildiği bir kelime dağarcığından alınır.
2. Evrişim Katmanı (Convolutional Layer): Evrişim katmanı, metin verilerini işlemek için kullanılır. Evrişim filtreleri, metin verilerinin farklı bölümlerine uygulanır ve özellik haritalarını çıkarır. Bu özellik haritaları, metindeki farklı özelliklerin algılanmasına yardımcı olur.
3. Havuzlama Katmanı (Pooling Layer): Havuzlama katmanı, evrişim katmanının çıkardığı özellik haritalarını özetler. Örneğin, maksimum havuzlama kullanarak, her özellik haritasının en büyük özellik değerini alır ve özet bir vektör oluşturur.
4. Tam Bağlantı Katmanları (Fully Connected Layers): Havuzlama katmanından gelen özet vektörü, tam bağlantı katmanlarında beslenir. Bu katmanlar, özelliklerin birleşimiyle metin verilerindeki daha yüksek seviye temsilleri öğrenir. Son tam bağlantı katmanında, metin verilerinin sınıflandırılması veya başka bir görev için çıktı üretilir.

# SVM Algoritması

"SVM" (Support Vector Machine - Destek Vektör Makinesi) algoritması, sınıflandırma ve regresyon problemlerinde kullanılan bir makine öğrenimi yöntemidir. SVM, veri noktalarını sınıflara ayırmak için optimal bir hiper düzlem bulmaya çalışır.

SVM algoritması, farklı çekirdek (kernel) fonksiyonları kullanarak çalışabilir.

**Linear Kernel (Doğrusal Çekirdek):** Doğrusal kernel, veriyi doğrusal olarak ayıran bir hiper düzlem bulmaya çalışır. Bu, veri noktalarının sınıflara göre ayrılabilmesi için bir doğru kullanır.

**Polynomial Kernel (Polinomial Çekirdek):** Polinomial kernel, verinin karmaşık bir şekilde ayrıldığı durumlarda kullanılır. Veriyi yüksek dereceli polinomlarla ayıran bir hiper düzlem bulmaya çalışır.

**Radial Basis Function (RBF) Kernel (Radyal Baz Fonksiyonu Çekirdeği):** RBF kernel, verinin non-linear olarak ayrıldığı durumlarda kullanılır. Veriyi bir merkez noktası etrafında radyal olarak ayrılan bir hiper düzlem bulmaya çalışır.

**Sigmoid Kernel:** Sigmoid kernel, sigmoid fonksiyonunu kullanarak veriyi ayırmaya çalışır. Bu kernel, non-linear sınıflandırma problemleri için kullanılabilir.

Her bir kernel, verinin yapısına ve problem gerekliliklerine göre tercih edilebilir. Kernel fonksiyonları, verinin farklı boyutlarda temsil edilmesini sağlayarak SVM'nin esnekliğini artırır.

SVM algoritması, verinin iyi ayrıldığı durumlarda etkili sonuçlar verir. Ayrıca, sınırlı veri noktalarıyla bile iyi performans gösterebilir ve aşırı öğrenmeye karşı dirençlidir. Ancak, büyük veri setleriyle çalışırken hesaplama yoğunluğu yüksek olabilir.

# Algortimaların Sonuçları

## Multinomial Naive Bayes - Bag of Words

Doğruluk (Accuracy): 0.8502501786990707  
F1 skoru: 0.8669418863131152

## Bernouilli Naive Bayes - Bag of Words

Doğruluk (Accuracy): 0.7993209435310936  
F1 skoru: 0.7703006749846594

## Support Vector Machine (linear kernel) Bag of Words

Doğruluk (Accuracy): 0.9054681915654038  
F1 skoru: 0.9037306642402184

## Support Vector Machine (polnominal kernel) Bag of Words

Doğruluk (Accuracy): 0.699964260185847  
F1 skoru: 0.5855344359417427

# Algortimaların Sonuçları

Support Vector Machine (RBF kernel)  
Bag of Words

Doğruluk (Accuracy): 0.9035025017869907  
F1 skoru: 0.9014238773274917

Support Vector Machine (sigmoid kernel)  
Bag of Words

Doğruluk (Accuracy): 0.759649749821301  
F1 skoru: 0.7605483354103614

Bernouilli Naive Bayes - TF-IDF

Doğruluk (Accuracy): 0.7993209435310936  
F1 skoru: 0.7703006749846594

Convolutional Neural Network - TF-IDF

Doğruluk (Accuracy): 0.5443937109673203  
F1 skoru: 0.5357209625379104

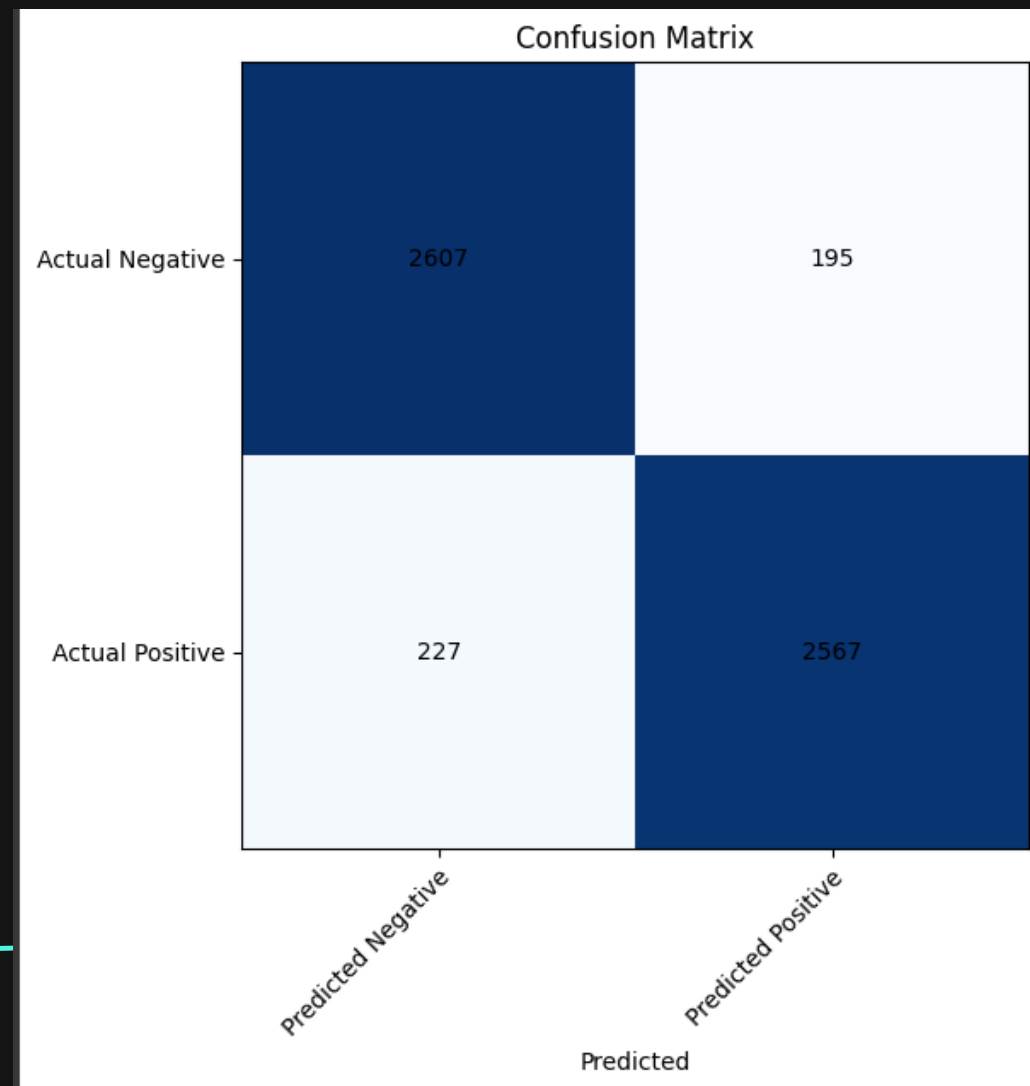
Multinomial Naive Bayes - TF-IDF

Doğruluk (Accuracy): 0.8557898498927805  
F1 skoru: 0.871843735111958

# Algortimaların Sonuçları

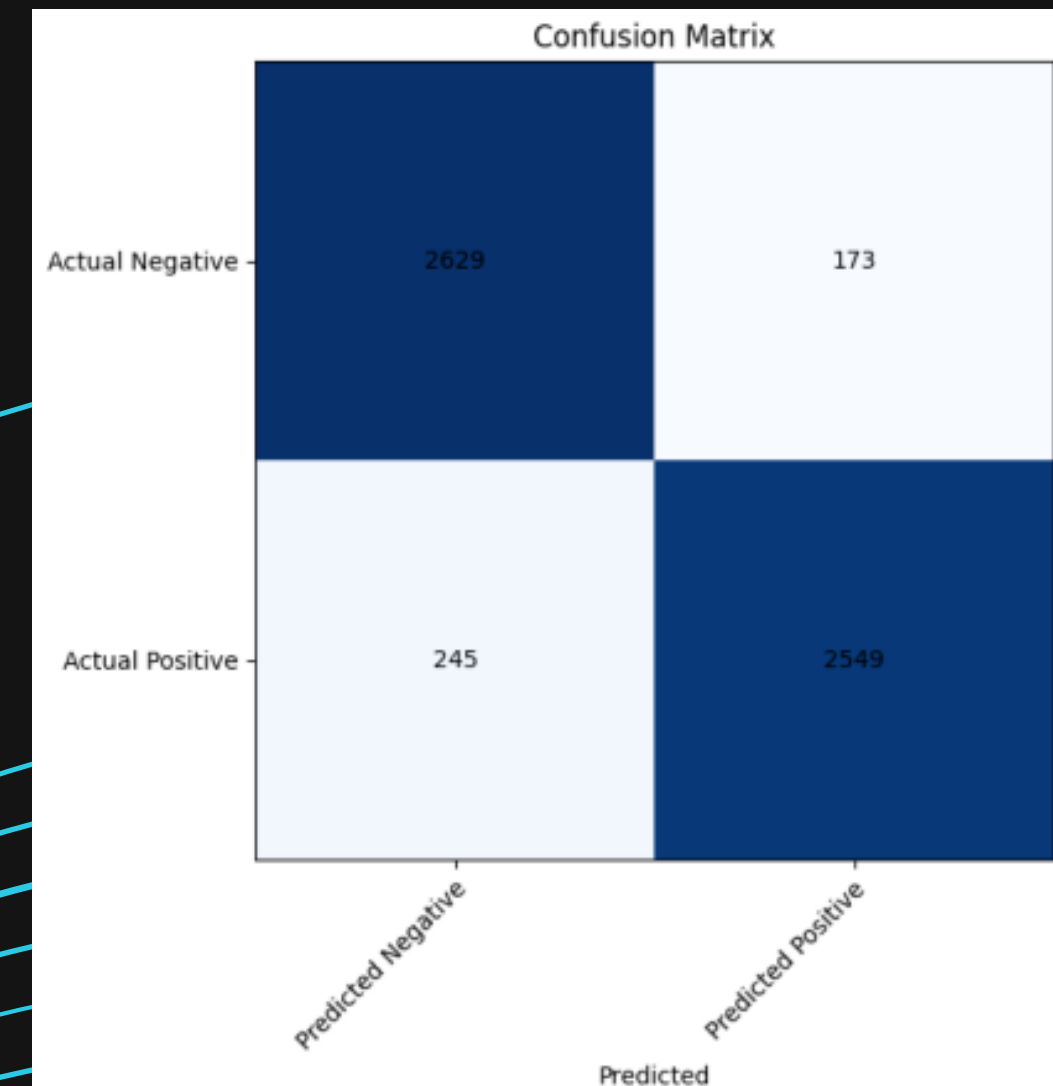
Support Vector Machine (linear kernel)  
TF-IDF

Doğruluk (Accuracy): 0.9245889921372409  
F1 skoru: 0.9240460763138949



Support Vector Machine (RBF kernel)  
TF-IDF

Doğruluk (Accuracy): 0.7993209435310936  
F1 skoru: 0.7703006749846594

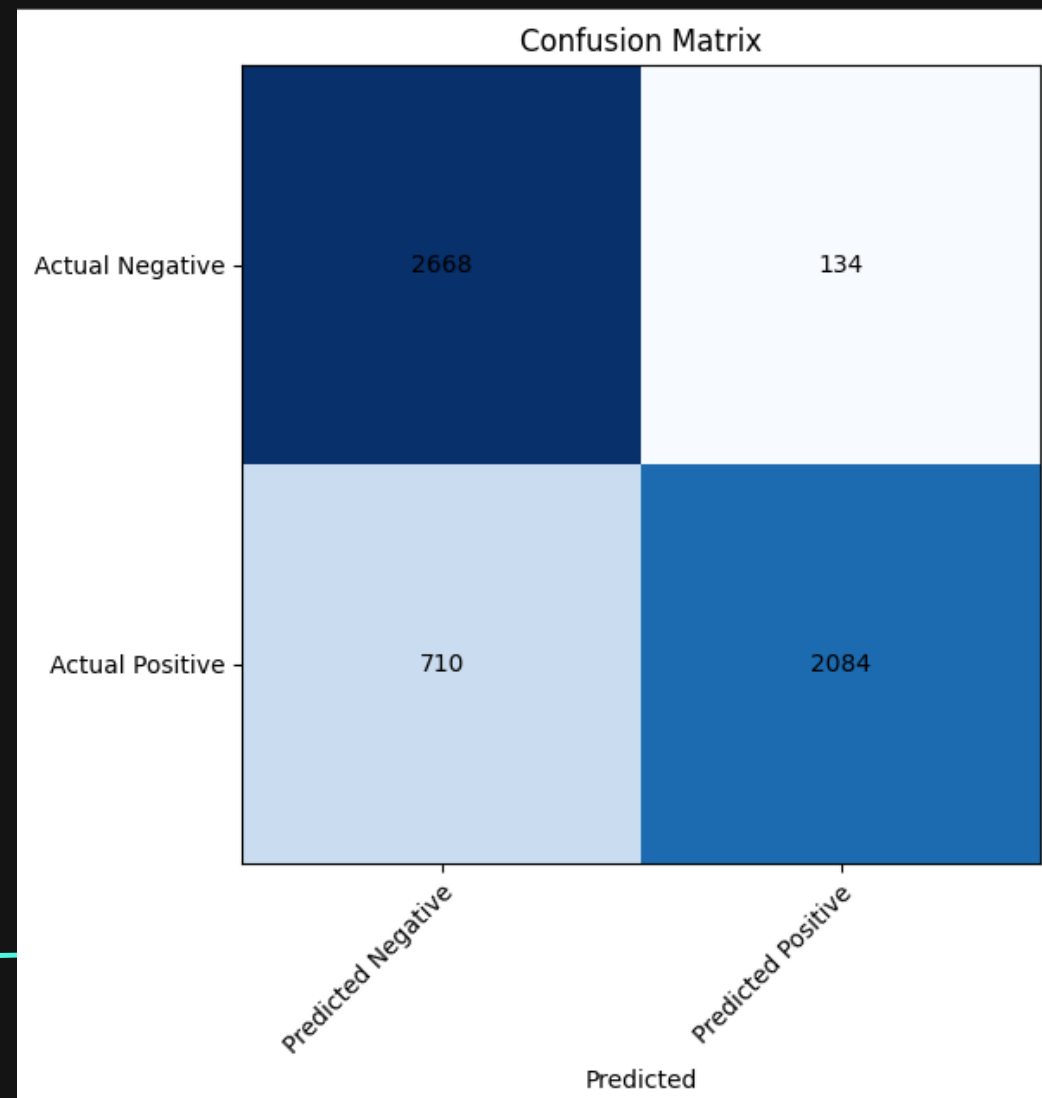




# Algortimaların Sonuçları

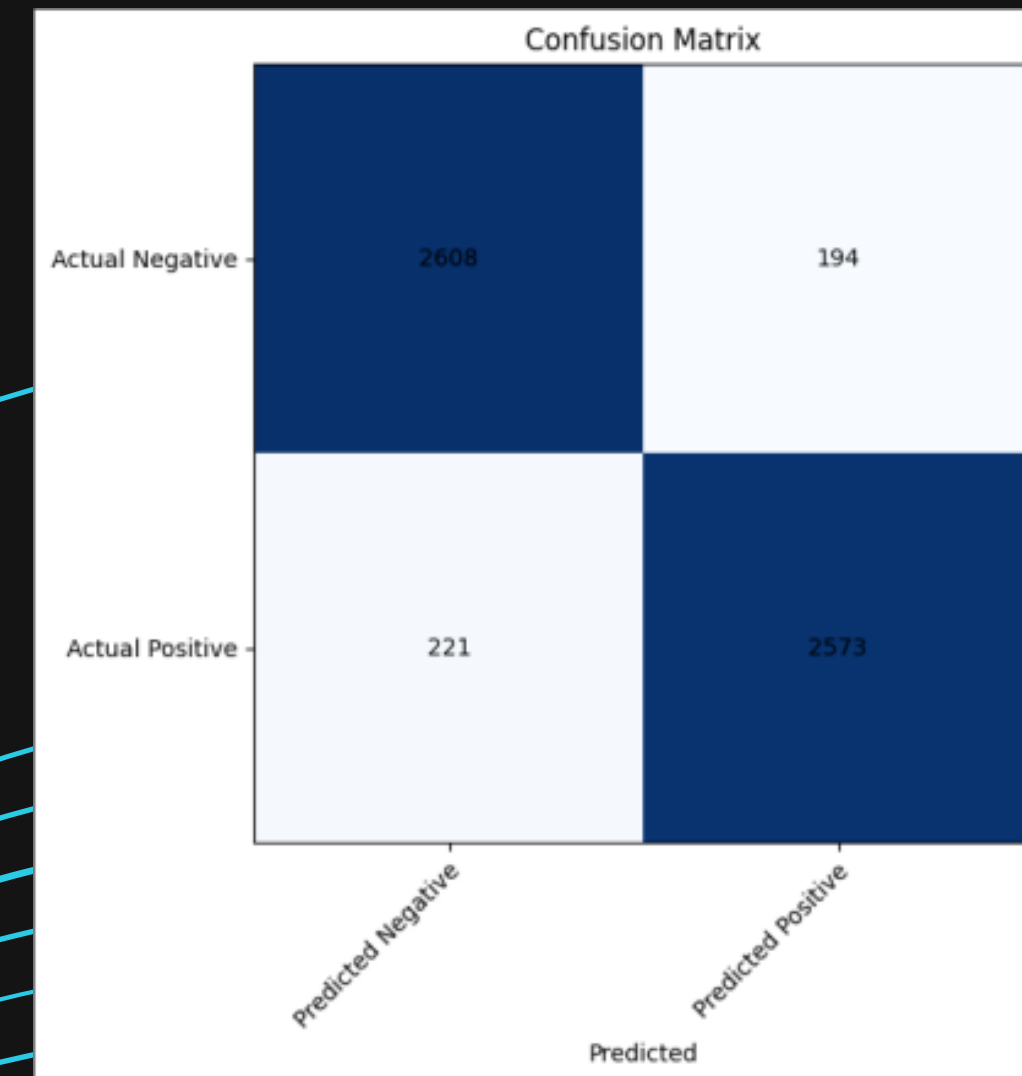
Support Vector Machine (polynomial kernel)  
TF-IDF

Doğruluk (Accuracy): 0.8491779842744818  
F1 skoru: 0.8316041500399042



Support Vector Machine (sigmoid kernel)  
TF-IDF

Doğruluk (Accuracy): 0.9258398856325947  
F1 skoru: 0.9253731343283581



# Sonuç

Aldığımız doğruluk (Accuracy) ve F1 skorlarını değerlendirdiğimizde TF-IDF ve sigmoid çekirdekli Destek Vektör Makinası (Support Vector Machine - SVM) algoritmalarını kullanarak elde ettiğimiz sonucun diğerlerinden önde olduğunu belirledik.