**1) Giriş :**

**Duygu analizinin ne olduğunun açıklanması:**

Duygu analizi, doğal dil işleme gibi yöntemler kullanılarak metinlerden duyguların tespit edilmesi ve sınıflandırılmasıdır. Sosyal medya, müşteri geri bildirimleri, haber makaleleri gibi pek çok alanda müşteri memnuniyetini izleme, yeni pazarlama stratejileri bulmak gibi amaçlarla kullanır.

**Projenin motivasyonunun açıklanması:**

Bu projede kullanıcı yorumlarını analiz ederek yeni yorumların olumlu mu yoksa olumsuz mu olduğunu makine öğrenmesi yöntemleri kullanarak analiz etmek, öznitelik seçimi uygulamadan ve farklı sayılarda öznitelik seçimi uygulanarak başarımların ölçülmesi ve en uygun parametrelerin belirlenmesi hedeflenmektedir. Proje öznitelik seçimi uygulanmadan yapılan analizlerle, öznitelik seçimi uygulanan analizlerin karşılaştırılması üzerine odaklanmaktadır.

**Projenin aşamalarının başlıklar halinde kısaca yazılması:**

1. Önişleme
   1. Metnin düzeltilmesi
   2. Kelimelerin köklerine ayrılması ve negatif olanların işaretlenmesi
   3. Emojilerin pozitif veya negatif olarak işaretlenmesi
   4. Noktalama işaretlerinin kaldırılması
   5. Fazla boşlukların kaldırılması.
2. K-Fold kullanılarak eğitim ve test kümelerinin oluşturulması
3. Özniteliklerin çıkartılması
4. TF, DF, IDF değerlerinin her küme için hesaplanması ve TFIDF hesaplanması
5. Farklı sınıflandırıcılar kullanarak ortalama accurary ve ortalama f1-score değerlerinin hesaplanması
6. Train.csv dosyasını girdi olarak alarak Chi-square veya Information Gain yöntemleriyle en ayırt edici 250, 500, 1000, 2500 ve 5000 özniteliklerin ayrı ayrı dosyalara kaydedilmesi
7. Öznitelik seçiminden sonra her bir öznitelik sayısı için TF, DF, IDF değerlerinin ve TFIDF hesaplanması
8. Farklı sınıflandırıcılar kullanarak her bir öznitelik sayısı için ortalama accurary ve ortalama f1-score değerlerinin hesaplanması
9. Elde edilen sonuçların raporlanması

**2) Ön İşleme: Uygulanan ön işleme adımları nelerdir ?**

1. Hatalı yazılan metinlerin düzeltilmesi.
2. Kelimelerin köklerine ayrılması ve negatif olanların işaretlenmesi.
3. Emojilerin pozitif veya negatif olarak işaretlenmesi. (NEGEMOTİON, POSEMOTİON)
4. Noktalama işaretlerinin kaldırılması.
5. Fazla boşlukların kaldırılması.

**3) Öznitelik çıkarımı: Öznitelik çıkarım yöntemi (kelime çantası) yaklaşımının açıklanması:**

Dokümanlarda bulunan unigram, bigram, n-gram özniteliklerin çıkartılması ve ardından her bir öznitelik için frekansların hesaplanması işlemidir. Bu sayede basitleştirilmiş bir temsil elde edilir (öznitelik vektörü gibi).

**4) Öznitelik ağırlıklandırma: Öznitelik ağırlıklandırma yönteminin (tf, idf, tf\*idf) açıklanması:**

TF: Her bir özniteliğin doküman içerisinde kaç defa geçtiğinin belirlenmesidir.

IDF: Öncelikle DF hesaplanır, DF her bir özniteliğin dokümanların kaç tanesinde geçtiğinin hesaplanmasıdır, daha sonra dokümanların çoğunda geçen terimlerin sınıflandırıcının modellenmesinde ayırt ediciliği az olduğundan bu terimlerin ağırlıklarının azaltılması gerekmektedir, idf bunu sağlar ve “idf = log(N/df)” formülüyle hesaplanır (N, toplam doküman sayısı), (df, doküman frekansı).

TF\*IDF: Özniteliklerin doküman içerisinde kaç defa geçtiği (tf) ile idf değerlerinin çarpılmasıyla elde edilir. Metinlerdeki önemli kelimelerin belirlenmesi için kullanılan bir ağırlıklandırma yöntemidir.

**5) Öznitelik seçimi: Öznitelik seçimi yönteminin ( Ki-kare, Information Gain gibi) açıklanması:**

Ki-kare (Chi-square): Kategorik bağımlı değişkenlerle ilişkilendirilmiş kategorik öznitelikler arasındaki ilişkiyi ölçer. Bir özniteliğin bağımsız değişken üzerindeki etkisi istatistiksel olarak test edilir.

Information Gain (Bilgi Kazancı): Özniteliklerin sınıflandırma veya regresyon problemlerinde ne kadar bilgi sağladığını ölçer. Karar ağaçları gibi algoritmalarla kullanılır ve bir özniteliğin sınıflandırma yaparken sağladığı bilgi miktarına göre sıralama yapar.

**6) Sınıflandırma: Kullanılan sınıflandırma yöntemlerinin açıklanması:**

**SVM:** Verileri en iyi şekilde ayırmak için bir hiper düzlem bulur. Özellikle ikili sınıflandırma problemlerinde kullanılır. Verileri destek vektörleriyle ayırdıktan sonra verilerin hangi bölgeye denk düştüğünü hesaplayarak sınıflandırır.

**Random Forest:** Birçok karar ağacının bir araya gelmesiyle oluşturulan bir sınıflandırma modelidir. Her ağaç rastgele örneklenmiş veri alt kümeleri üzerinde eğitilir ve tahminlerin ortalaması veya modu alınarak sınıflandırma yapılır.

**Logistic Regression:** Girdi değişkenlerini kullanarak eğitilir ve bir çizgi (denklem) oluşturur daha sonra girdilerin çıktısının hangi sınıfa ait olduğunu bu denklemi kullanarak tahmin eder.

**Gradient Boosting:** Zayıf tahmin edicileri sırayla birleştirerek güçlü bir tahminci oluşturur. Her ağaç önceki ağacın hatalarını düzeltmeye odaklanarak eğitilir ve gradyan inişi kullanılarak toplam hata fonksiyonu minimize edilir, karmaşık ilişkileri yakalayabilir ve yüksek doğruluk sağlayabilir.

**7) Sonuçlar: Sınıflandırma sonuçlarının (accuracy ve f-score) tablolar şeklinde gösterimi ve yorumlanması:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Öznitelik seçimi olmadan** | **250** | **500** | **1000** | **2500** | **5000** |
| **SVM - Accuracy** | 0.8042142857142858 | 0.8277142857142858 | 0.8399285714285715 | 0.8395714285714285 | 0.8372857142857143 | 0.8307142857142858 |
| **SVM - F1 Score** | 0.7936514237460492 | 0.8264801269749755 | 0.8363818588721281 | 0.8344471113917787 | 0.8319888093317447 | 0.8239067386550047 |
| **Random Forest - Accuracy** | 0.8033571428571429 | 0.786357142857143 | 0.8030000000000002 | 0.8065714285714287 | 0.815 | 0.8101428571428572 |
| **Random Forest - F1 Score** | 0.7967262113930803 | 0.7842637171317421 | 0.7999492056706083 | 0.8047167216771333 | 0.8120945649523478 | 0.8067420323667999 |
| **Logistic Regression - Accuracy** | 0.7922857142857144 | 0.832285714285714 | 0.8381428571428572 | 0.8381428571428571 | 0.8342857142857142 | 0.8339285714285714 |
| **Logistic Regression - F1 Score** | 0.7934100869482014 | 0.8333506229707112 | 0.8385959768072102 | 0.8392436539202697 | 0.836951356565043 | 0.8339860502165815 |
| **Gradient Boosting - Accuracy** | 0.7900714285714285 | 0.7878571428571429 | 0.7917142857142858 | 0.7907142857142857 | 0.7898571428571429 | 0.7902142857142855 |
| **Gradient Boosting - F1 Score** | 0.7875197061473981 | 0.786044360051848 | 0.787747665251528 | 0.7886631489788087 | 0.7873388898348515 | 0.787619246206728 |

**Yorumlar:**

Tablodaki değerlere ve grafikteki duruma bakarak, öznitelik seçimi olduğunda genellikle daha başarılı bir model elde edildiği söylenebilir fakat öznitelik seçimi olduğunda model bazen daha kötü sonuçlar da verebilmektedir, özellikle random forest algoritmasına bakıldığında 250 öznitelik yerine tüm özniteliklerin kullanılması durumunda daha başarılıdır fakat SVM kullanıldığında tam tersi durum söz konusudur.

Random forest ve gradient boosting gibi algoritmalar rastgelelik içerdiğinden sonuçlar her hesaplanmada biraz değişebilir. Genel olarak baktığımızda bu veri seti için 500 öznitelik seçimi yeterlidir, çünkü çoğu algoritma için bu sayı başarılıdır ve sayının daha büyük seçilmesi hem hesaplama süresini uzatmakta hem de başarımın bazı algoritmalarda düşmesine sebep olmaktadır.