Stress Level Guesser

Çağdaş Çetin
Bilgisayar Mühendisliği
Yıldız Teknik Üniversitesi
İstanbul, Türkiye
cagdas.cetin@std.yildiz.edu.tr

I. Giriş

Günümüzün yaşam tarzı göz önünde bulundurulduğunda insanlar, uykunun insan bedenine sağladığı faydaları unutarak uyumaktadır. Smart-Yoga Pillow (SaYoPillow), stres ve uyku arasındaki ilişkinin anlaşılmasın yardımcı olmak ve "Akıllı-Uyku" fikrini tam olarak gerçekleştirmek için önerilmiştir. Uyku sırasındaki bu değişiklere dayalı olarak bir sonraki gün için stres tahmini yapılır. Analiz edilen stres verilerinin ortalama fizyolojik değişikliklerle birlikte depolama için bir IoT bulutuna güvenli bir şekilde aktarımı yapılır. Herhangi bir verinin buluttan herhangi bir üçüncü taraf uygulamasına güvenli bir şekilde aktarılması da önerilir. SaYoPillow, %96'ya varan doğrulukla ve stresi azaltmak için uyku alışkanlıklarının dikkate alınmasıyla yeni bir teknolojidir.

SayoPillow data setinde [2], literatür taraması yapılarak oluşturulan horlama oranı, solunum oranı, vücut sıcaklığı, uzuv hareket oranı, kandaki oksijen seviyesi, göz hareketleri, uyunan saat, nabız değerleri ile stres seviyesi (0-az/normal, 1-orta az, 2-orta, 3-orta yüksek, 4-yüksek) arasındaki ilişki görülmektedir. Hiçbir insan denek göz önüne alınmamıştır. [3] [4]

A. Özniteliklerin Özellikleri

- snoring_rate = horlama oranı %0 oranında boş değer içerir, nümerik sütun
- respiration_rate = solunum oranı %0 oranında boş değer içerir, nümerik sütun
- body_temperature = vücut sıcaklığı %0 oranında boş değer içerir, nümerik sütun
- limb_movement = uzuv hareket oranı %0 oranında boş değer içerir, nümerik sütun
- blood_oxygen = kandaki oksijen seviyesi %0 oranında boş değer içerir, nümerik sütun
- eye_movement = göz hareketleri %0 oranında boş değer içerir, nümerik sütun
- sleep_hourt = uyunan saat %0 oranında boş değer içerir, nümerik sütun
- heart_rate = nabız %0 oranında boş değer içerir, nümerik sütun

II. Önişleme

Veri önişleme; veri madenciliği modelleri kurulmadan önce veri seti üzerinde yapılan bir takım düzeltme, eksik veriyi tamamlama, tekrarlanan verileri kaldırma, dönüştürme, bütünleştirme, temizleme, normalleştirme, boyut indirgeme vb. işlemlerdir [5].

İlk olarak veri setindeki verilerin içinde null değer kontrolü yapılmıştır. İkinci olarak ise aynı veriden birden fazla değerin olup olmadığına bakılmıştır. Sürekli olan değerler için normalizasyon işlemi gerçekleştirilmiştir.

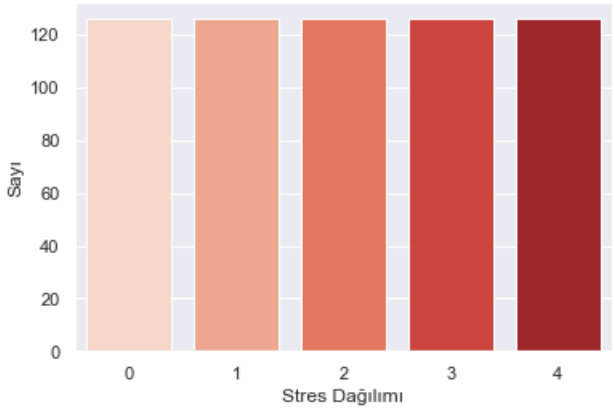

Fig. 1. Veri seti stres değeri dağılımı

III. Metot

Bu projede [1] sınıflandırma algoritmaları olan Random Forest, K-Nearest Neighbour ve Naive Bayes kullanılmıştır. Bu metotlar farklı özellikleri doğrultusunda daha iyi karşılaştırılabilmek için seçilmiştir.

Veri setimiz, metotlarımızda kullanılmak için **x** ve **y** değerlerine ayrılmıştır. Sonrasında ise, train_test_split() ile %80 eğitim kümesi, %20 test kümesi olacak şekilde x_train, x_test, y_train, y_test gruplarına ayrılmıştır.

A. Random Forest

Random Forest algoritması, denetimli sınıflandırma algoritmalarından biridir. Hem regresyon hem de sınıflandırma problemlerinde kullanılmaktadır. Algoritma, birden fazla karar ağacı üreterek sınıflandırma işlemi esnasında sınıflandırma değerini yükseltmeyi hedefler.

Random Forest algoritması birbirinden bağımsız olarak çalışan birçok karar ağacının bir araya gelerek aralarından en yüksek puan alan değerin seçilmesi işlemidir. Ağaç sayısı arttıkça kesin bir sonuç elde etme oranımız artmaktadır. Karar

ağaçları algoritması ile arasındaki temel fark, Random Forest algoritmasında kök düğümü bulma ve düğümleri bölme işleminin rastgele olmasıdır. [6]

Modelin parametresi olan ağaç sayısı(n_estimators), kodu her çalıştırdığımızda 1'den 50'ye kadar olan değerler denenerek maksimum doğruluk oranını veren değere göre belirlenmektedir.

Bu raporda örnek olarak kullanılan tablolarda ve uygulama değerlerinde kullanılmak üzere olarak test edildiğinde en yüksek doğruluk oranını veren ağaç sayısı 37 olarak çıkmıştır ve sonraki bütün Random Forest algoritmasına ait olan bilgiler bu sayıya göre olacaktır.

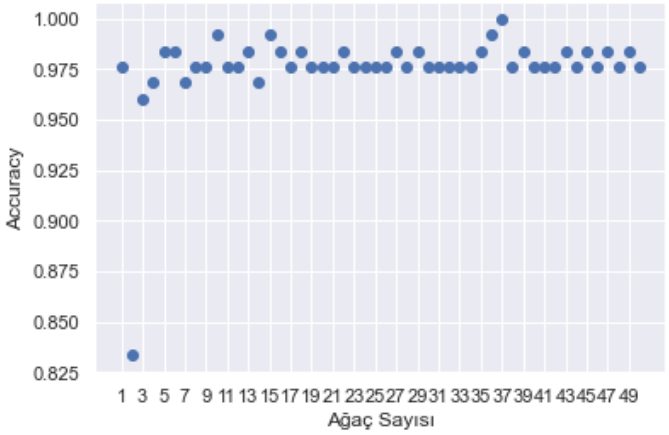

Fig. 2. Ağaç sayısı ve doğruluk arasındaki ilişki

B. K-Nearest Neighbors

K-Nearest Neighbor (KNN), denetimli öğrenme tekniğine dayalı en basit makine öğrenimi algoritmalarından biridir. KNN algoritması, yeni durum/veriler ile mevcut durumlar arasındaki benzerliğe bakar ve yeni durumu mevcut kategorilere en çok benzeyen kategoriye yerleştirir. KNN algoritması, mevcut tüm verileri saklar ve benzerliğe göre yeni bir veri noktasını sınıflandırır. Bu, yeni veriler geldiğinde, KNN algoritması kullanılarak kolayca en uygun kategoriye göre sınıflandırılabileceği anlamına gelir. KNN algoritması sınıflandırmanın yanı sıra regresyon için de kullanılabilir ancak daha çok sınıflandırma problemleri için kullanılır. KNN, parametrik olmayan bir algoritmadır, yani temel veriler üzerinde herhangi bir varsayımda bulunmaz. Tembel öğrenen algoritması olarak da adlandırılır çünkü eğitim kümesinden hemen öğrenmez, bunun yerine veri kümesini depolar ve sınıflandırma anında veri kümesi üzerinde bir işlem gerçekleştirir. KNN algoritması, eğitim aşamasında sadece veri setini saklar ve yeni veri aldığında bu veriyi yeni veriye çok benzeyen bir kategoride sınıflandırır. [7]

En uygun K değerini bulmak için önceden tanımlanmış istatistiksel yöntemler yoktur. Rastgele bir K değeri seçilir ve deneyerek optimize edilmeye çalışılır. Küçük bir K değeri seçmek, dengesiz karar sınırlarına yol açabilir. Tanımlanmış bir aralıkta hata oranı ve K değerlerini gösteren bir grafik türetilir. Ardından, minimum hata oranına sahip K değerini seçilir [8]

Modelin parametresi olan komşu sayısı(n_neighbors), kodu her çalıştırdığımızda 1'den 50'ye kadar olan değerler denenerek maksimum doğruluk oranını veren değere göre belirlenmektedir.

Bu raporda örnek olarak kullanılan tablolarda ve uygulama değerlerinde kullanılmak üzere olarak test edildiğinde en yüksek doğruluk oranını veren komşu sayısı 27 olarak çıkmıştır ve sonraki bütün K-Nearest Neighbors algoritmasına ait olan bilgiler bu sayıya göre olacaktır.

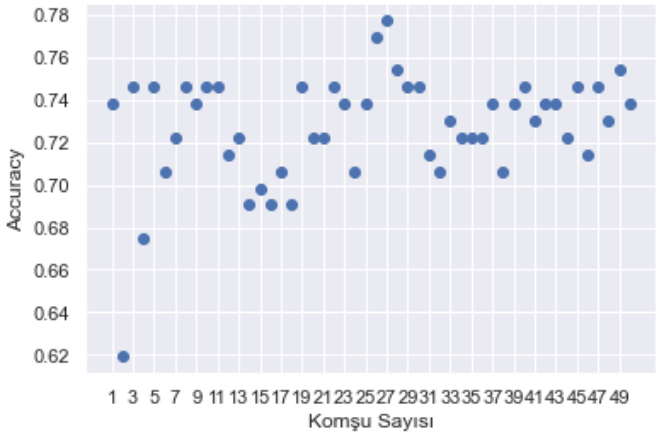

Fig. 3. Komşu sayısı ve doğruluk arasındaki ilişki

C. Naive Bayes

Öznitelikler arasında bağımsızlık varsayımı ile Bayes Teoremine dayalı bir sınıflandırma tekniğidir. Basit bir ifadeyle, bir Naive Bayes sınıflandırıcısı, bir sınıftaki belirli bir özelliğin, başka herhangi bir özellikle ilgisi olmadığını varsayar.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Fig. 4. Bayes teoremi

Bayes teoremini kullanarak, B gerçekleştiğine göre A'nın olma olasılığını bulabiliriz. Burada B kanıt, A ise hipotezdir. Burada yapılan varsayım, tahmin edicilerin/özelliklerin bağımsız olduğudur. Yani belirli bir özelliğin varlığı diğerini etkilemez. Bu nedenle naif denir.

Naive Bayes modelinin oluşturulması kolaydır ve özellikle çok büyük veri kümeleri için kullanışlıdır. Sadeliğin yanı sıra, Naive Bayes'in son derece karmaşık sınıflandırma yöntemlerinden bile daha iyi performans gösterdiği bilinmektedir.

Öznitelikler sürekli değerler aldığında ve ayrık olmadığında, bu değerlerin bir gauss dağılımından örneklendiğini varsayarız. [9]

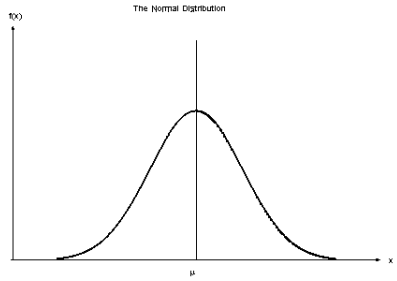

Fig. 5. Gaussian dağılımı(Normal dağılım)

Veri kümesinde değerlerin bulunma şekli değiştiğinden, koşullu olasılık formülü de şu şekilde değişir:

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp\left(-\frac{(x_i-\mu_y)^2}{2\sigma_y^2}\right)$$

Fig. 6. Gaussian koşullu olasılık formülü

IV. Uygulama

Makine öğrenmesi algoritmalarının, sınıflandırmanın ve regresyon algoritmalarının performansını değerlendirmek için kullanabileceğimiz çeşitli metrikler vardır. Makine öğrenimi performansını değerlendirmek için metrikler dikkatli bir şekilde seçilmelidir çünkü makine öğrenimi algoritmalarının performansının nasıl ölçüleceği ve karşılaştırılacağı, tamamen seçtiğiniz metriğe bağlı olacaktır. Sonuçta çeşitli özelliklerin önemini nasıl ağırlıklandırdığınız, tamamen seçtiğiniz metriğe göre değişecektir. [10]

Sınıflandırma algoritmalarının değerlendirme metrikleri arasında confusion matrix, accuracy, precision, sensitivity(recall), f1-score bulunur.

A. Confusion Matrix

Confusion Matrix(Karışıklık Matrisi), çıktının iki veya daha fazla sınıf olabileceği makine öğrenimi sınıflandırma problemi için bir performans ölçümüdür. Tahmini ve gerçek değerlerin farklı kombinasyonlarını içeren bir tablodur.

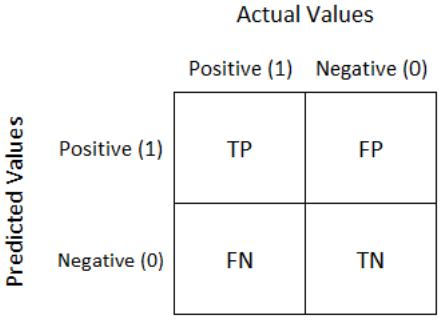

Fig. 7. Confusion Matrix

Tam olarak bir performans metriği değil, diğer metriklerin sonuçları değerlendirdiği bir tür temeldir. Her hücre bir değerlendirme faktörünü temsil eder. Bunlar:

- True Positive(TP): Pozitif tahmin edilen örneğin pozitif çıktığını belirtir. Doğru tahmindir.
- True Negative(TN): Negatif tahmin edilen örneğin negatif çıktığını belirtir. Doğru tahmindir.
- False Pozitif(FP): Pozitif tahmin edilen örneğin negatif çıktığını belirtir. Yanlış tahmindir. Bu faktör, istatistiksel terminolojide Tip 1 hatayı temsil eder.
- False Negatif(FN): Negatif tahmin edilen örneğin pozitif çıktığını belirtir. Yanlış tahmindir. Bu faktör, istatistiksel terminolojide Tip 2 hatayı temsil eder. [11]

Aşağıda kullanılan metotlara göre oluşan confusion matrix değerleri verilmiştir.

```
[[21  0  0  0  0]
 [ 1 26  1  0  0]
 [ 0  0 34  0  0]
 [ 0  0  0 15  1]
 [ 0  0  0  0 27]]
```

Fig. 8. Random Forest Confusion Matrix

```
[[18  3  0  0  0]
 [ 4 22  2  0  0]
 [ 0  6 24  4  0]
 [ 0  0  1 14  1]
 [ 0  0  0  7 20]]
```

Fig. 9. K-Nearest Neighbor Confusion Matrix

```
[[21  0  0  0  0]
 [ 0 28  0  0  0]
 [ 0  0 34  0  0]
 [ 0  0  0 15  1]
 [ 0  0  0  0 27]]
```

Fig. 10. Naive Bayes Confusion Matrix

B. Accuracy

Accuracy metriği, uygulanması en basit metriklerden birisidir ve yapılan bu projeye göre, stres seviyesi doğru tahmin edilen insanların, hem doğru hem de yanlış tahmin edilen insanların toplamına oranına accuracy(doğruluk) denir.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

Kullanılan metotlara göre elde edilen accuracy değerleri aşağıdaki gibidir.

- Random Forest Accuracy: 0.976190
- KNN Accuracy: 0.777778
- Naive Bayes Accuracy: 0.992063

C. Precision

Precision, gerçek pozitiflerin, tahmin edilen toplam pozitiflere oranıdır. .

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

Kullanılan metotlara göre elde edilen precision değerleri aşağıdaki gibidir.

- Random Forest Precision: 0.978052 (macro)
- KNN Precision: 0.785826 (macro)
- Naive Bayes Precision: 0.992857 (macro)

D. Recall

Recall ise, gerçek pozitiflerin temel gerçeklikteki tüm pozitiflere oranıdır.

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

Kullanılan metotlara göre elde edilen recall değerleri aşağıdaki gibidir.

- Random Forest Recall: 0.973214 (macro)
- KNN Recall: 0.792896 (macro)
- Naive Bayes Recall: 0.987500 (macro)

E. Diğer Metrikler

Aşağıda, kullanılan metotların fazladan metrikleri verilmiştir.

Random Forest Metrics:

	precision	recall	f1-score	support
0	0.95	1.00	0.98	21
1	1.00	0.93	0.96	28
2	0.97	1.00	0.99	34
3	1.00	0.94	0.97	16
4	0.96	1.00	0.98	27
accuracy			0.98	126
macro avg	0.98	0.97	0.97	126
weighted avg	0.98	0.98	0.98	126

Fig. 11. Random Forest Metrics

KNN Metrics:

	precision	recall	f1-score	support
0	0.82	0.86	0.84	21
1	0.71	0.79	0.75	28
2	0.89	0.71	0.79	34
3	0.56	0.88	0.68	16
4	0.95	0.74	0.83	27
accuracy			0.78	126
macro avg	0.79	0.79	0.78	126
weighted avg	0.81	0.78	0.78	126

Fig. 12. K-Nearest Neighbors Metrics

Naive Bayes Metrics:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	21
1	1.00	1.00	1.00	28
2	1.00	1.00	1.00	34
3	1.00	0.94	0.97	16
4	0.96	1.00	0.98	27
accuracy			0.99	126
macro avg	0.99	0.99	0.99	126
weighted avg	0.99	0.99	0.99	126

Fig. 13. Naive Bayes Metrics

F. Tartışma

KUllanılan 3 metodu karşılaştıracak olursak önce accuracy değerlerine bakarız. Accuracy değeri en yüksek olan Naive Bayes metodudur ve hemen ardından çok yakın bir değer ile Random Forest gelmektedir. KNN ise büyük bir fark ile en sonda yer almaktadır.

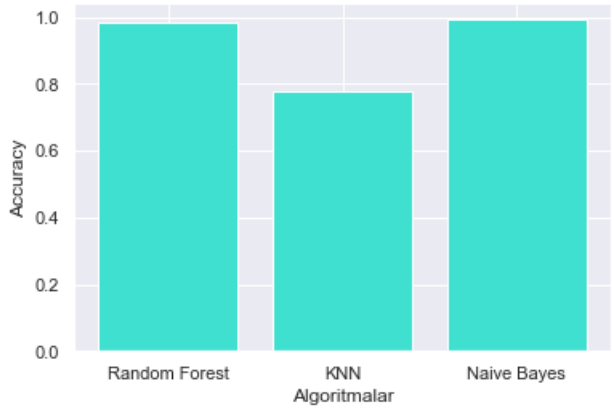

Fig. 14. Accuracy karşılaştırması

En uygun metodu seçmek için accuracy tek başına yeterli değildir. Bundan dolayı precision ve recall değerlerinin harmonik ortalaması olan, diğer metrikler arasında yer alan f1-score değerine bakmak uygun metodu seçme kararını vermemize katkı sağlayacaktır. Precision ve recall değerlerini macro olarak aldığımız için macro f1-score değerine baktığımızda en yüksek değere yine Naive Bayes metodunun sahip olduğunu görmektyiz. Bu durumdan yola çıkarak kullanılan en iyi metodun Naive Bayes olduğunu söyleyebiliriz.

Naive Bayes metodunun kullanılan veri seti için en uygun metot olmasının nedeni veri setindeki bütün niteliklerin birbirinden bağımsız olmasıdır. Naive Bayes metediu da bütün niteliklerin bağımsız olduğunu varsayarak çalışır.

İkinci sırada ise Random Forest gelmektedir. Alınan değerleri karşılaştırdığımızda bütün değerleri kendisinden sonraki sırada gelen KNN metodundan daha yüksektir. İyi olmasının sebebi veri setindeki nitelikler arasından rastgele seçimler yaparak karar ağaçları oluşturup bunlar arasında karar vermesidir. En iyi modeli seçmek için bir döngü içerisinde 1'den 50'ye kadar olan ağaç sayıları denenmiştir.

KNN metodu ise son sırada yer almaktadır. Bütün değerleri diğer iki metottan büyük bir farkla daha düşüktür. Bunun sebebi veri setindeki outlier değerleri olabilir.

G. Sonuçlar

Bu projede insanların uyku sırasında ölçülen değerlerine bakılarak stres seviyelerinin tahmininin yapılması amaçlanmıştır. İlk olarak veri setindeki verilerin içinde null değer kontrolü yapılmıştır. İkinci olarak ise aynı veriden birden fazla değerin olup olmadığına bakılmıştır. Sürekli olan değerler için normalizasyon işlemi gerçekleştirilmiştir. Daha sonra veri seti, eğitim ve test şeklinde 2 küme olacak şekilde ayrılmıştır. Eğitim kümesinin tüm veri setine oranı %80, test kümesinin ise %20 olarak belirlenmiştir. Random Forest metodunda bir döngü ile 1 ağaçtan 50 ağaca kadar bütün ormanlar denenmiştir ve en iyi sonucu veren ağaç sayısı alınmıştır. KNN metodunda ise Random Forest metodunda olduğu gibi yine bir döngü ile 1 komşudan 50 komşuya kadar bütün komşu sayıları denenmiştir ve en iyi sonucu veren komşu sayısı alınmıştır. Naive Bayes metodunda ise koşullu olasılık formülü ile olasılıksal durum hesapları yapılıp accuracy, precision ve recall değerleri bulunmuştur. Son olarak bulunan bütün değerler karşılaştırılıp, bu veri setinde en iyi çalışan metodun Naive Bayes olduğu sonucuna varılmıştır.

REFERENCES

[1] https://github.com/cagdascetin35/Stress-Level-Guesser
[2] https://www.kaggle.com/datasets/laavanya/human-stress-detection-in-and-through-sleep?select=SaYoPillow.csv
[3] L. Rachakonda, A. K. Bapatla, S. P. Mohanty, and E. Kougianos, "SaYoPillow: Blockchain-Integrated Privacy-Assured IoMT Framework for Stress Management Considering Sleeping Habits", IEEE Transactions on Consumer Electronics (TCE), Vol. 67, No. 1, Feb 2021, pp. 20-29.
[4] L. Rachakonda, S. P. Mohanty, E. Kougianos, K. Karunakaran, and M. Ganapathiraju, "Smart-Pillow: An IoT based Device for Stress Detection Considering Sleeping Habits", in Proceedings of the 4th IEEE International Symposium on Smart Electronic Systems (iSES), 2018, pp. 161–166.
[5] https://www.veribilimiokulu.com/buyuk-veri-on-isleme-makale-notlari/
[6] https://ece-akdagli.medium.com/makine-öğrenmesinde-random-forest-algoritması-a79b044bbb31
[7] https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning
[8] https://medium.com/swlh/k-nearest-neighbor-ca2593d7a3c4
[9] https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c
[10] https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_algorithms_performance_metrics.htm
[11] https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62