BIM305- Makine Öğrenmesi

MEHMET BURAK DURDU

210911059

MODEL 1: PCA OLMADAN

metin, yazı tipi, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Pandas Kütüphanesi: pandas kütüphanesi, veri manipülasyonu ve analizi için kullanılır. pd olarak kısaltılır.
2. Scikit-learn Kütüphanesi: model\_selection modülü, veri kümesini eğitim ve test kümelerine bölmek için kullanılır. linear\_model modülü, lineer regresyon modelini uygulamak için kullanılır. LogisticRegression sınıfı, lojistik regresyon modelini uygulamak için kullanılır. metrics modülü, model performansını değerlendirmek için kullanılır.
3. Matplotlib ve Seaborn Kütüphaneleri: matplotlib.pyplot kütüphanesi, çeşitli görselleştirmeler oluşturmak için kullanılır. seaborn kütüphanesi, Matplotlib üzerine inşa edilmiş istatistiksel veri görselleştirmesi için kullanılır. sns olarak kısaltılır.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. StressLevelModel adlı bir sınıfı tanımlar. Bu sınıf, stres seviyesi veri setini işlemek, analiz etmek ve modellemek için kullanılır.
2. Bu sınıfın \_\_init\_\_ (kurucu) metodu, sınıfın başlatılması sırasında çağrılır ve data\_path parametresi ile bir CSV dosyasını okuyarak bir Pandas DataFrame oluşturur. Ayrıca, lineer regresyon ve lojistik regresyon için kullanılacak özel sütun listelerini (kolonları) tanımlar.
3. linear\_cols: Lineer regresyon modeli için bağımsız değişkenlerin olduğu sütunlar.
4. logistic\_cols: Lojistik regresyon modeli için hem bağımlı hem de bağımsız değişkenlerin olduğu sütunlar.

metin, ekran görüntüsü, yazı tipi, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu fonksiyonun amacı, eğer veri çerçevesinde "Unnamed: 0" adında bir sütun varsa bu sütunu kaldırmaktır.
2. Bu fonksiyonun adı remove\_unnamed\_column, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. Bu metodun içindeki if şartı, "Unnamed: 0" adında bir sütunun self.df içinde bulunup bulunmadığını kontrol eder. Eğer bu sütun varsa, del anahtar kelimesiyle bu sütunu veri çerçevesinden kaldırır.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu fonksiyonun amacı, veri çerçevesindeki eksik değerleri (NaN, Not a Number) ortalama değerlerle doldurmaktır.
2. Bu fonksiyon adı fill\_missing\_values, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. self.df.fillna(self.df.mean(), inplace=True) ifadesi, veri çerçevesindeki her NaN değerini o sütundaki ortalama değer ile doldurur. inplace=True parametresi, bu değişiklikleri mevcut veri çerçevesine uygular ve yeni bir veri çerçevesi döndürmez. Eksik değerleri ortalama ile doldurmak, veri setini daha tutarlı hale getirmek ve model eğitimi için uygun hale getirmek için yaygın bir ön işleme adımıdır.

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, veri çerçevesindeki sütunlar arasındaki ilişkileri görselleştirmek amacıyla bir korelasyon ıs haritası (correlation heatmap) eklemektir.
2. Bu metodun adı add\_correlation\_heatmap, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. İlk olarak, bir Matplotlib figure'ü oluşturarak grafiğin boyutunu belirliyoruz (plt.figure(figsize=(8, 6))). Daha sonra, self.df.corr() ile veri çerçevesindeki sütunlar arasındaki korelasyon matrisini hesaplıyoruz. sns.heatmap ile bu matrisi renkli bir ıs haritası olarak görselleştiriyoruz. annot=True ile değerleri gösteriyoruz, cmap='coolwarm' ile renk haritasını belirliyoruz, fmt=".2f" ile değerlerin ondalık basamak sayısını belirliyoruz ve linewidths=0.5 ile kenar çizgilerinin kalınlığını ayarlıyoruz. Son olarak, grafiğin başlığını belirleyip plt.show() ile grafiği görüntülüyoruz. Bu tür bir korelasyon ıs haritası, veri setindeki değişkenler arasındaki ilişkileri anlamak için kullanışlı bir görselleştirme aracıdır.

metin, ekran görüntüsü, yazılım, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, hedef değişkene en düşük korelasyon katsayısına sahip olan sütunu bulmaktır.
2. Bu metodun adı find\_column\_with\_lowest\_correlation, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. Metodun içinde, self.df.corr() ile veri çerçevesindeki sütunlar arasındaki korelasyon matrisini hesaplıyoruz. correlation\_matrix.iloc[:-1, -1] ifadesi, hedef değişken hariç diğer değişkenlere karşı olan korelasyon katsayılarını alır. idxmin() metodu ile bu korelasyon katsayıları içinde en küçük olanın sütun adını (min\_corr\_col) buluyoruz. Son olarak, bulunan sütunun adını yazdırıyoruz ve bu adı geri döndürüyoruz.
3. Bu metodun kullanımı, veri setindeki değişkenler arasındaki ilişkiyi anlamak ve modelleme için en uygun sütunu seçmek için önemlidir.

metin, ekran görüntüsü, yazı tipi, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, hedef değişkene en düşük korelasyona sahip olan sütunu veri çerçevesinden kaldırmaktır.
2. Bu metodun adı drop\_column\_with\_lowest\_correlation, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. İlk olarak, self.find\_column\_with\_lowest\_correlation() metodunu kullanarak en düşük korelasyona sahip sütunu buluyoruz. Bu sütunun adını min\_corr\_col değişkenine atıyoruz.
3. Daha sonra, kaldırılacak sütunun adını ve bu sütunun en yakın korelasyon değerini yazdırıyoruz. Ardından, self.df.drop(columns=min\_corr\_col, inplace=True) ifadesi ile veri çerçevesinden bu sütunu kaldırıyoruz. inplace=True parametresi, bu değişikliği mevcut veri çerçevesine uygular ve yeni bir veri çerçevesi döndürmez.
4. Son olarak, bir sütun kaldırıldıktan sonra lineer sütun listesini güncellemek için self.update\_linear\_cols(min\_corr\_col) metodunu çağırıyoruz. Bu adım, lineer regresyon modeli için kullanılan sütun listesini güncel tutmak önemlidir, çünkü korelasyona dayalı olarak bir sütun kaldırıldığında bu modelin yeniden eğitilmesi gerekebilir.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, bir sütun kaldırıldıktan sonra lineer regresyon modeli için kullanılan sütun listesini güncellemektir.
2. Bu metodun adı update\_linear\_cols, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. Bu metodun içinde, self.linear\_cols listesindeki sütunlardan kaldırılan sütunu çıkarmak için bir liste anlama (list comprehension) kullanıyoruz. Yani, col değişkeni ile self.linear\_cols listesinde dönerken, col'un dropped\_column ile eşit olmadığı durumları yeni liste içinde tutuyoruz. Böylece, kaldırılan sütunu içermeyen yeni bir lineer sütun listesi elde ediyoruz. Bu adım, bir sütun kaldırıldığında lineer regresyon modelini güncellemek ve yeniden eğitmek için kullanılır.

metin, yazı tipi, multimedya yazılımı, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, veriyi lineer regresyon modeli için eğitim ve test setlerine bölmektir.
2. Bu metodun adı split\_data\_linear, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. İlk olarak, self.df[self.linear\_cols] ifadesi ile veri çerçevesindeki lineer regresyon için kullanılan sütunları X\_linear değişkenine atıyoruz. Hedef değişkeni olan 'stress\_level' sütununu da y\_linear değişkenine atıyoruz.
3. Daha sonra, train\_test\_split fonksiyonunu kullanarak veriyi eğitim ve test setlerine bölüyoruz. test\_size=0.2 parametresi ile verinin yüzde 20'sini test setine ayırıyoruz. random\_state=42 parametresi, bölmeyi tekrarladığımızda aynı sonuçları elde etmemizi sağlar.

metin, yazılım, multimedya yazılımı, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, veriyi lojistik regresyon modeli için eğitim ve test setlerine bölmektir. Ayrıca, belirli bir sınıfın geri kalanına karşı tahmin edileceği bir senaryoyu ele alır.
2. Bu metodun adı split\_data\_logistic, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. İlk olarak, self.df[self.logistic\_cols] ifadesi ile veri çerçevesindeki lojistik regresyon için kullanılan sütunları X\_logistic değişkenine atıyoruz.
3. Daha sonra, self.df['sleep\_quality'] == target\_class ifadesi ile 'sleep\_quality' sütunundaki değerleri target\_class ile karşılaştırıyoruz. Bu karşılaştırma sonucunda bir boolean Serisi elde ediyoruz ve astype(int) ile bu boolean değerleri 0 ve 1'e dönüştürüyoruz. Böylece, belirli bir sınıfı tahmin etmek üzere lojistik regresyon modeli için kullanılacak y\_logistic hedef değişkenini oluşturuyoruz.
4. Son olarak, train\_test\_split fonksiyonunu kullanarak veriyi eğitim ve test setlerine bölüyoruz. test\_size=0.2 parametresi ile verinin yüzde 20'sini test setine ayırıyoruz ve random\_state=42 parametresi ile tekrarlanabilirliği sağlıyoruz.
5. Bu metodun kullanımı, lojistik regresyon modelini eğitmek ve değerlendirmek için kullanılacak olan eğitim ve test setlerini elde etmek için önemlidir.

metin, ekran görüntüsü, yazılım, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, verilen özellikler (features) ve hedef değişken (target variable) üzerinde bir lineer regresyon modeli eğitmektir.
2. Bu metodun adı train\_linear\_model, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. Metodun içinde, LinearRegression() sınıfından bir model oluşturulur ve bu model fit metodunu kullanarak verilen eğitim özellikleri (X\_train) ve hedef değişkeni (y\_train) üzerinde eğitilir.
3. Son olarak, eğitilmiş model geri döndürülür.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, verilen özellikler (features) ve hedef değişken (target variable) üzerinde bir lojistik regresyon modeli eğitmektir.
2. Bu metodun adı train\_logistic\_model, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. Metodun içinde, LogisticRegression() sınıfından bir model oluşturulur ve bu model fit metodunu kullanarak verilen eğitim özellikleri (X\_train) ve hedef değişkeni (y\_train) üzerinde eğitilir.
3. Son olarak, eğitilmiş model geri döndürülür.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, verilen eğitilmiş bir lineer regresyon modelini test verisi üzerinde değerlendirmek ve değerlendirme sonuçlarını görselleştirmektir.
2. Bu metodun adı evaluate\_linear\_model, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. Metodun içinde, eğitilmiş model (model) kullanılarak test verisi (X\_test) üzerinde tahminler yapılır.
3. Daha sonra, self.calculate\_rmse metodunu kullanarak gerçek ve tahmin edilen değerler arasındaki kök ortalama kare hatası (RMSE) hesaplanır. RMSE, modelin tahminlerinin gerçek değerlerden ne kadar uzak olduğunu ölçen bir performans metriğidir.
4. Son olarak, self.visualize\_linear\_predictions metodunu kullanarak lineer regresyon tahminlerini görselleştiren bir grafik oluşturulur. Bu adım, modelin başarı ve hata paternlerini anlamak için önemlidir.
5. Bu metodun kullanımı, lineer regresyon modelinin performansını değerlendirmek ve bu değerlendirmeyi görselleştirmek için önemlidir.

metin, ekran görüntüsü, yazılım, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, gerçek ve tahmin edilen değerler arasındaki kök ortalama kare hatasını (Root Mean Squared Error - RMSE) hesaplamak ve bu değeri döndürmektir.
2. Bu metodun adı calculate\_rmse, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. Metodun içinde, mean\_squared\_error fonksiyonu kullanılarak gerçek (y\_true) ve tahmin edilen (y\_pred) değerler arasındaki karesel hata ortalaması hesaplanır. Ardından, bu karesel hatanın karekökü alınarak RMSE elde edilir.
3. Son olarak, RMSE değeri ekrana yazdırılır ve bu değer geri döndürülür. RMSE, bir regresyon modelinin tahmin başarısını ölçen bir metrik olduğu için bu değer, modelin ne kadar iyi performans gösterdiğini anlamak için kullanılabilir.

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, lineer regresyon modelinin gerçek ve tahmin edilen değerleri arasındaki ilişkiyi görselleştirmektir.
2. Bu metodun adı visualize\_linear\_predictions, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. Metodun içinde, gerçek (y\_true) ve tahmin edilen (y\_pred) değerlerin birbirine göre scatter plot (nokta grafiği) ile görselleştirilir.
3. Ayrıca, bir doğru olan kırmızı bir kesikli çizgi eklenir. Bu çizgi, mükemmel bir tahmin durumunu temsil eder ve eğer tüm tahminler bu çizgi üzerinde olursa, modelin mükemmel bir performans gösterdiği anlamına gelir.
4. Son olarak, grafik başlık, x ve y eksen etiketleri gibi görsel öğeler eklenir ve plt.show() fonksiyonu ile grafik görüntülenir. Bu adım, modelin gerçek ve tahmin edilen değerleri arasındaki ilişkiyi daha iyi anlamak için önemlidir.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, verilen eğitilmiş bir lojistik regresyon modelini test verisi üzerinde değerlendirmek, sınıflandırma metriklerini hesaplamak ve bu değerleri görselleştirmektir.
2. Bu metodun adı evaluate\_logistic\_model, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. Metodun içinde, eğitilmiş model (model) kullanılarak test verisi (X\_test) üzerinde tahminler yapılır.
3. Daha sonra, self.calculate\_classification\_metrics metodunu kullanarak sınıflandırma metrikleri hesaplanır. Bu metrikler genellikle doğruluk (accuracy), karmaşıklık matrisi (confusion matrix) ve sınıflandırma raporu (classification report) içerir.
4. Ardından, self.print\_classification\_metrics metodunu kullanarak hesaplanan metrikler ekrana yazdırılır. Bu adım, modelin performansını anlamak için önemlidir.
5. Son olarak, self.visualize\_roc\_curve metodunu kullanarak ROC eğrisini görselleştiren bir grafik oluşturulur. ROC eğrisi, lojistik regresyon modelinin duyarlılık ve özgüllük performansını gösteren bir grafiktir.
6. Bu metodun kullanımı, lojistik regresyon modelinin performansını değerlendirmek, sınıflandırma metriklerini görselleştirmek ve modelin hassasiyetini anlamak için önemlidir.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, sınıflandırma metriklerini (doğruluk, karmaşıklık matrisi, sınıflandırma raporu) hesaplamak ve bu değerleri bir demet içinde döndürmektir.
2. Bu metodun adı calculate\_classification\_metrics, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. Metodun içinde, accuracy\_score, confusion\_matrix, ve classification\_report fonksiyonları kullanılarak sınıflandırma metrikleri hesaplanır.
3. Hesaplanan metrikler, doğruluk (accuracy), karmaşıklık matrisi (confusion matrix) ve sınıflandırma raporu (classification report) olmak üzere üç ayrı değer olarak bir demet içinde döndürülür.
4. Bu metrikler, bir sınıflandırma modelinin performansını değerlendirmek için kullanılır. Doğruluk, modelin doğru tahminlerin yüzdesini ölçerken, karmaşıklık matrisi ve sınıflandırma raporu, modelin sınıflar arasındaki performansını daha ayrıntılı bir şekilde değerlendirir.

metin, ekran görüntüsü, yazılım, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, sınıflandırma metriklerini (doğruluk, karmaşıklık matrisi, sınıflandırma raporu) ekrana yazdırmaktır.
2. Bu metodun adı print\_classification\_metrics, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. Metodun içinde, basit bir metin tablosu şeklinde doğruluk, karmaşıklık matrisi ve sınıflandırma raporu ekrana yazdırılır.
3. Bu bilgiler, bir sınıflandırma modelinin performansını daha anlamak ve yorumlamak için kullanılır. print fonksiyonu ile ekrana yazdırılan bu bilgiler, modelin hangi sınıfları ne kadar iyi tahmin ettiği ve hangi metriklerin kullanıldığı hakkında bilgi verir.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, eğitilmiş bir lojistik regresyon modelinin ROC (Receiver Operating Characteristic) eğrisini görselleştirmektir.
2. Bu metodun adı visualize\_roc\_curve, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. Metodun içinde, roc\_curve fonksiyonu kullanılarak lojistik regresyon modelinin ROC eğrisi için false positive rate (fpr), true positive rate (tpr) ve eşik değerleri (thresholds) hesaplanır.
3. Daha sonra, bu değerler kullanılarak ROC eğrisi plt.plot fonksiyonu ile çizilir. ROC eğrisinin altında kalan alan (AUC, Area Under the Curve), modelin performansını ölçen bir değerdir. Bu alan ne kadar büyükse, modelin performansı o kadar iyidir.
4. Grafik üzerinde, ROC eğrisi, rastgele bir modelin performansını temsil eden çizgi ile karşılaştırılır. Eğer ROC eğrisi bu çizginin üzerinde ise, modelin performansı iyidir. plt.show() fonksiyonu ile grafik görüntülenir.

metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, veriyi alt grafiklerle görselleştirmektir.
2. Bu metodun adı visualize\_data, ve bir sınıf metodu olduğu için self parametresi ile bir sınıf örneğine bağlı olarak çalışır. Metodun içinde, matplotlib ve seaborn kütüphaneleri kullanılarak veriyi görselleştirmek için alt grafikler (subplots) oluşturulur.
3. Her alt grafikte, verinin bir özelliği (kolonu) için ya yoğunluk grafiği (kernel density plot) ya da sütun grafiği (count plot) çizilir. plt.subplot fonksiyonu ile alt grafik pozisyonları belirlenir, ve plt.xlabel, plt.ylabel fonksiyonları ile ekseni etiketlenir.
4. Son olarak, plt.tight\_layout() fonksiyonu ile alt grafikler arasındaki boşlukları düzenler ve plt.show() fonksiyonu ile grafik görüntülenir.

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, bir dizi işlemi sırasıyla gerçekleştirmektir. Her bir adımda, sınıfın diğer metodları kullanılarak veri görselleştirmesi, korelasyon analizi, veri ön işleme, doğrusal regresyon modeli eğitimi ve değerlendirmesi, lojistik regresyon modeli eğitimi ve değerlendirmesi, son olarak ise bir ilişkinin görselleştirilmesi işlemleri yapılır.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

bu yapı bir dosya hem bir modül olarak içe aktarılabilir hem de bir bağımsız betik olarak çalıştırılabilir olmasını sağlamak için kullanılır.

SONUÇLAR:

metin, diyagram, plan, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, kalıp, desen, düzen, renklilik içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, çizgi, ekran görüntüsü, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

MODEL 2 : PCA İLE

metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. pandas, numpy, ve matplotlib.pyplot veri manipülasyonu ve görselleştirme işlemleri için kullanılır.
2. sklearn kütüphanesinden çeşitli alt modüller içe aktarılır. Bu modüller, makine öğrenimi modelleri oluşturmak ve değerlendirmek için kullanılacaktır.
3. seaborn görselleştirmeleri geliştirmek için kullanılır.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. StressLevelModel sınıfını tanımlar. Bu sınıf, bir stres seviyesi veri setini içeren bir CSV dosyasından veri yükleyen ve bu veri seti üzerinde çeşitli makine öğrenimi modellerini eğiten bir araç sağlar.
2. Constructor (\_\_init\_\_) Metodu:
3. data\_path parametresi, CSV dosyasının yolu olarak alınır.
4. pd.read\_csv(data\_path) ile belirtilen CSV dosyası pandas kütüphanesi kullanılarak okunur ve self.df özelliğine atanır (veri setini temsil eden bir DataFrame).
5. self.linear\_cols ve self.logistic\_cols özellikleri, sırasıyla lineer regresyon ve logistic regresyon modelleri için kullanılacak sütun isimlerini içerir.
6. Veri Seti ve Sütunlar:
7. self.df: Veri setini içeren bir DataFrame.
8. self.linear\_cols: Lineer regresyon modellerinde kullanılacak sütun isimlerini içeren bir liste.
9. self.logistic\_cols: Logistic regresyon modellerinde kullanılacak sütun isimlerini içeren bir liste.
10. Bu sınıfın amaçları, veri setini yüklemek, belirli sütunları seçmek ve çeşitli işlemler için kullanılacak temel veri yapısını sağlamaktır. Sınıfın geri kalanı, bu temel yapı üzerine inşa edilen çeşitli metotlar içerir

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, self.df DataFrame'inde "Unnamed: 0" adlı bir sütun varsa bu sütunu kaldırmaktır. Veri seti bir CSV dosyasından okunduğunda veya başka bir şekilde oluşturulduğunda, bazen indeks sütunu olarak "Unnamed: 0" veya benzeri bir sütun eklenir.
2. if "Unnamed: 0" in self.df.columns:: Eğer "Unnamed: 0" adlı bir sütun self.df DataFrame'inin sütunları arasında bulunuyorsa,
3. del self.df["Unnamed: 0"]: O zaman bu sütunu ("Unnamed: 0") self.df DataFrame'inden sil.
4. Bu, sütunun varlığını kontrol eder ve eğer varsa onu kaldırır.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, self.df DataFrame'indeki eksik (NaN) değerleri sütun bazında ortalama değerlerle doldurmaktır. Eksik değerler, veri setlerinde sıkça karşılaşılan bir sorundur ve bu tür bir doldurma stratejisi, eksik değerleri genellikle sütunun ortalamasıyla doldurarak basit bir yaklaşım sunar.
2. self.df.fillna(self.df.mean(), inplace=True): Bu satırda, self.df DataFrame'indeki tüm eksik değerler, ilgili sütunun ortalamasıyla doldurulur. inplace=True parametresi, bu işlemin DataFrame üzerinde doğrudan yapılmasını sağlar, yani ayrı bir kopya oluşturulmaz ve orijinal DataFrame güncellenir.
3. Örneğin, eğer "depression" sütununda bir eksik değer varsa, bu değer "depression" sütununun ortalaması ile doldurulacaktır.

metin, yazı tipi, multimedya yazılımı, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, self.df DataFrame'indeki sütunlar arasındaki korelasyonu gösteren bir ısı haritasını görselleştirmektir. Korelasyon, iki değişken arasındaki ilişkinin gücünü ve yönünü ölçen bir istatistiksel ölçüdür. Bu görselleştirme, korelasyon matrisinin renkli bir haritasını oluşturarak bu ilişkileri anlamayı kolaylaştırır.
2. plt.figure(figsize=(8, 6)): Bu satırda, matplotlib kütüphanesini kullanarak bir figure (şekil) oluşturulur ve bu figürün boyutları belirlenir.
3. correlation\_matrix = self.df.corr(): Veri setindeki sütunlar arasındaki korelasyonları içeren bir korelasyon matrisi oluşturulur.
4. sns.heatmap(correlation\_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5): seaborn kütüphanesinin heatmap fonksiyonu kullanılarak korelasyon matrisi ısı haritasına çevrilir. annot=True parametresi, hücrelerin içine korelasyon değerlerini yerleştirir. cmap='coolwarm' parametresi, kullanılan renk haritasını belirler. fmt=".2f" parametresi, değerlerin ondalık kısmını iki basamakla sınırlar. linewidths=0.5 parametresi, hücreler arasındaki çizgi kalınlığını belirler.
5. plt.title('Correlation Heatmap'): Şeklin başlığı "Correlation Heatmap" olarak ayarlanır.
6. plt.show(): Şekil ekrana çizilir ve görüntülenir.
7. Bu metodun çağrılması, veri setindeki sütunlar arasındaki korelasyonu hızlı bir şekilde görselleştirmenizi sağlar.

metin, ekran görüntüsü, yazı tipi, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, hedef değişkenle en düşük korelasyon katsayısına sahip olan sütunu bulmaktır. Korelasyon katsayısı, iki değişken arasındaki ilişkinin gücünü ve yönlendirilmesini ölçen bir istatistiksel ölçüdür. Bu metot, veri setindeki sütunlar arasındaki bu korelasyonları kullanarak hedef değişkenle en düşük korelasyona sahip olan sütunu belirler.
2. correlation\_matrix = self.df.corr(): Veri setindeki sütunlar arasındaki korelasyonları içeren bir korelasyon matrisi oluşturulur.
3. min\_corr\_col = correlation\_matrix.iloc[:-1, -1].idxmin(): Korelasyon matrisinin son sütunu, hedef değişkenle olan korelasyonları içerir. idxmin() metodu, en küçük değere sahip olan sütunun indeksini (sütun adını) döndürür. -1 kullanılarak hedef değişkenle olan korelasyon sütunu seçilir.
4. print(f"Column with the lowest correlation to the target variable: {min\_corr\_col}"): Bulunan sütunun adını ekrana yazdırır.
5. return min\_corr\_col: Bulunan sütunun adını döndürür.

metin, ekran görüntüsü, yazılım, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, hedef değişkenle en düşük korelasyona sahip olan sütunu veri setinden çıkarmaktır. Bu adım, modelleme sürecinde kullanılacak sütunların belirlenmesi ve gereksiz sütunların çıkarılması için önemlidir.
2. min\_corr\_col = self.find\_column\_with\_lowest\_correlation(): Daha önce tanımlanan find\_column\_with\_lowest\_correlation metodunu kullanarak, hedef değişkenle en düşük korelasyona sahip olan sütunun adını bulur.
3. print(f"About to drop column with closest correlation to 0: {min\_corr\_col}"): Çıkarılacak sütunun adını ekrana yazdırır.
4. self.df.drop(columns=min\_corr\_col, inplace=True): DataFrame üzerinde, belirlenen sütunu çıkarır. inplace=True parametresi, bu işlemin orijinal DataFrame üzerinde yapılmasını sağlar.
5. print(f"Dropped column with closest correlation to 0: {min\_corr\_col}"): Çıkarılan sütunun adını ekrana yazdırır.
6. self.update\_linear\_cols(min\_corr\_col): Çıkarılan sütunun, lineer regresyon modellerinde kullanılan sütun listesinden (self.linear\_cols) çıkarılması için update\_linear\_cols metodunu çağırır.
7. Bu metodun çağrılması, hedef değişkenle en düşük korelasyona sahip olan sütunu çıkarır ve modelleme sürecinde kullanılacak veri setini günceller.

metin, yazı tipi, multimedya yazılımı, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, bir sütun çıkarıldıktan sonra lineer regresyon modellerinde kullanılacak sütun listesini (self.linear\_cols) güncellemektir. Çıkarılan sütun, bu listeden çıkarılır.
2. self.linear\_cols = [col for col in self.linear\_cols if col != dropped\_column]: List comprehension kullanarak, çıkarılan sütunu (dropped\_column) içermeyen bir yeni liste oluşturulur ve bu liste self.linear\_cols özelliğine atanır.

metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, PCA (Principal Component Analysis) yöntemini kullanarak veri setindeki boyutları azaltmaktır. PCA, veri setindeki değişken sayısını azaltarak özgün değişkenlerin ana bileşenlerini (principal components) bulur.
2. original\_linear\_cols = self.linear\_cols.copy(): Lineer regresyon modelleri için kullanılan orijinal sütun listesini (self.linear\_cols) kopyalar.
3. pca = PCA(n\_components=n\_components): n\_components parametresiyle belirtilen sayıda ana bileşeni koruyan bir PCA nesnesi oluşturur.
4. X\_linear = self.df[self.linear\_cols]: Lineer regresyon modellerinde kullanılan sütunlardan oluşan bir alt veri çerçevesi oluşturur.
5. X\_linear\_pca = pca.fit\_transform(X\_linear): PCA dönüşümünü uygular ve orijinal sütunları, ana bileşenlerle değiştirir.
6. pca\_column\_names = [f'pca\_{i+1}' for i in range(n\_components)]: Yeni PCA bileşenlerinin isimleri oluşturulur.
7. self.df.drop(columns=self.linear\_cols, inplace=True): Orijinal lineer sütunları veri çerçevesinden çıkarır.
8. self.df[pca\_column\_names] = X\_linear\_pca: Yeni PCA bileşenlerini veri çerçevesine ekler.
9. self.linear\_cols = pca\_column\_names: Lineer regresyon modellerinde kullanılan sütun listesini yeni PCA bileşen isimleriyle günceller.
10. print(f"Applied PCA to reduce dimensionality. Number of components kept: {n\_components}"): Konsola, boyut azaltma işleminin başarıyla gerçekleştirildiğine dair bir mesaj yazdırır.
11. print(f"Columns used for PCA: {original\_linear\_cols}"): Konsola, PCA için kullanılan orijinal sütunların isimlerini yazdırır.
12. print(f"Columns dropped after PCA: {list(self.df.columns.difference(original\_linear\_cols))}"): Konsola, PCA uygulandıktan sonra çıkarılan sütunların isimlerini yazdırır.
13. Bu metodun çağrılması, PCA'nın uygulanmasıyla veri setindeki boyutları azaltır ve bu işlemin sonuçlarını görsel ve bilgi tabanlı olarak sunar.

metin, multimedya yazılımı, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, veri setini lineer regresyon modeli için eğitim ve test setlerine bölmektir.
2. X\_linear = self.df[self.linear\_cols]: Lineer regresyon modellerinde kullanılacak sütunlardan oluşan bir özellik matrisi oluşturur.
3. y\_linear = self.df['stress\_level']: Hedef değişkeni oluşturur.
4. train\_test\_split(X\_linear, y\_linear, test\_size=0.2, random\_state=42): train\_test\_split fonksiyonu kullanılarak, belirlenen oranda (burada %80 eğitim, %20 test) veriyi rastgele iki alt sete böler. random\_state=42 parametresi, rastgele bölme işleminin tekrarlanabilir olmasını sağlar. Fonksiyon, eğitim özellik matrisi (X\_linear\_train), test özellik matrisi (X\_linear\_test), eğitim hedef değişkeni (y\_linear\_train), ve test hedef değişkeni (y\_linear\_test) olmak üzere dört alt set döndürür.
5. Bu metodun çağrılması, veri setini eğitim ve test setlerine ayırmak için kullanılır. Bu ayrım, modelin eğitim sırasında kullanılacak veriyi belirlemenin yanı sıra, modelin performansını değerlendirmek için ayrı bir veri setinin bulunmasını sağlar.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, veri setini logistic regresyon modeli için eğitim ve test setlerine bölmektir. Logistic regresyon, bir sınıflandırma algoritmasıdır ve genellikle iki sınıflı (binary) problemlerde kullanılır. Bu metodun kullanımı, logistic regresyon modelini eğitmek ve değerlendirmek için veri setini hazırlamak içindir.
2. X\_logistic = self.df[self.logistic\_cols]: Logistic regresyon modellerinde kullanılacak sütunlardan oluşan bir özellik matrisi oluşturur.
3. y\_logistic = (self.df['sleep\_quality'] == target\_class).astype(int): Hedef değişkeni oluşturur. target\_class parametresi, logistic regresyonun öğrenme sürecinde bir sınıfı diğerlerine karşı tahmin etmek için kullanılır. astype(int) ifadesi, True/False değerlerini 1/0'a dönüştürür.
4. train\_test\_split(X\_logistic, y\_logistic, test\_size=0.2, random\_state=42): train\_test\_split fonksiyonu kullanılarak, belirlenen oranda (burada %80 eğitim, %20 test) veriyi rastgele iki alt sete böler. random\_state=42 parametresi, rastgele bölme işleminin tekrarlanabilir olmasını sağlar. Fonksiyon, eğitim özellik matrisi (X\_logistic\_train), test özellik matrisi (X\_logistic\_test), eğitim hedef değişkeni (y\_logistic\_train), ve test hedef değişkeni (y\_logistic\_test) olmak üzere dört alt set döndürür.
5. Bu metodun çağrılması, veri setini logistic regresyon modeli için eğitim ve test setlerine ayırmak içindir.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, lineer regresyon modelini eğitmektir. Lineer regresyon, bir bağımlı değişkenin bağımsız değişkenlerle doğrusal bir ilişkisini modelleyen bir regresyon tekniğidir. Model, eğitim verileri üzerinde uyarlanarak, bağımsız değişkenlerin lineer bir kombinasyonu ile bağımlı değişkeni tahmin etmeyi öğrenir.
2. model = LinearRegression(): Lineer regresyon modelini oluşturur.
3. model.fit(X\_train, y\_train): Oluşturulan modeli eğitim verileri (X\_train, y\_train) üzerinde uyarlar. Bu adım, modelin bağımsız değişkenlerle bağımlı değişken arasındaki ilişkiyi öğrenmesini sağlar.
4. return model: Eğitilmiş lineer regresyon modelini döndürür.
5. Bu metodun çağrılması, lineer regresyon modelini eğitim verileri üzerinde uyarır ve eğitilmiş modeli döndürür.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, logistic regresyon modelini eğitmektir. Logistic regresyon, bir sınıflandırma algoritmasıdır ve genellikle iki sınıflı (binary) sınıflandırma problemlerinde kullanılır. Model, eğitim verileri üzerinde uyarlanarak, bağımsız değişkenlerin kombinasyonunu kullanarak belirli bir sınıfa ait olma olasılığını tahmin etmeyi öğrenir.
2. model = LogisticRegression(): Logistic regresyon modelini oluşturur.
3. model.fit(X\_train, y\_train): Oluşturulan modeli eğitim verileri (X\_train, y\_train) üzerinde uyarlar. Bu adım, modelin bağımsız değişkenlerle belirli bir sınıfa ait olma olasılığını tahmin etmeyi öğrenmesini sağlar.
4. return model: Eğitilmiş logistic regresyon modelini döndürür.
5. Bu metodun çağrılması, logistic regresyon modelini eğitim verileri üzerinde uyarır ve eğitilmiş modeli döndürür.

metin, ekran görüntüsü, ekran, görüntüleme, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, eğitilmiş bir lineer regresyon modelini değerlendirmektir. Değerlendirme işlemi, modelin test verileri üzerinde nasıl performans gösterdiğini anlamak için kullanılır.
2. y\_pred = model.predict(X\_test): Eğitilmiş modeli kullanarak, test verileri üzerinde tahminler yapar. Bu adım, modelin test verilerini kullanarak bağımlı değişkeni tahmin etmesini sağlar.
3. rmse = self.calculate\_rmse(y\_test, y\_pred): Hata metriği olarak root mean squared error (RMSE) hesaplar. Bu adım, gerçek ve tahmini değerler arasındaki farkı ölçer. RMSE ne kadar düşükse, modelin tahminleri o kadar iyi demektir.
4. self.visualize\_linear\_predictions(y\_test, y\_pred): Lineer regresyonun gerçek ve tahmini değerlerini görselleştirir. Bu adım, modelin tahminlerini gerçek değerlerle karşılaştırmak için bir scatter plot çizer.
5. Bu metodun çağrılması, eğitilmiş lineer regresyon modelinin performansını değerlendirir ve sonuçları görsel olarak sunar.

metin, ekran görüntüsü, yazılım, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, gerçek ve tahmini değerler arasındaki root mean squared error (RMSE) hesaplamaktır. RMSE, bir regresyon modelinin tahminlerinin gerçek değerlerden ne kadar uzak olduğunu ölçen bir hata metriğidir.
2. rmse = mean\_squared\_error(y\_true, y\_pred, squared=False): mean\_squared\_error fonksiyonunu kullanarak RMSE hesaplanır. squared=False parametresi, RMSE'nin karesiz (root) değerini döndürmesini sağlar. Bu adım, gerçek ve tahmini değerler arasındaki karesel hataların ortalamasının karekökünü alarak RMSE'yi hesaplar.
3. print('Linear Regression RMSE:', rmse): Hesaplanan RMSE'yi ekrana yazdırır.
4. return rmse: Hesaplanan RMSE'yi döndürür.
5. Bu metodun çağrılması, gerçek ve tahmini değerler arasındaki RMSE'yi hesaplar ve sonucu döndürür. RMSE ne kadar düşükse, modelin tahminleri o kadar iyi demektir.

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, lineer regresyonun gerçek ve tahmini değerlerini görselleştirmektir. Görselleştirme, modelin tahminlerinin gerçek değerlerle karşılaştırılmasını sağlar.
2. plt.scatter(y\_true, y\_pred, alpha=0.5): Scatter plot kullanarak, gerçek ve tahmini değerlerin dağılımını gösterir. Bu adım, her bir gerçek değerin karşılık gelen tahmini değerle ne kadar uygun olduğunu görsel olarak gösterir.
3. plt.plot([min(y\_true), max(y\_true)], [min(y\_true), max(y\_true)], color='red', linestyle='--', lw=2): 45 derece açılı bir kırmızı kesikli çizgi çizer. Bu çizgi, ideal durumu temsil eder; yani, gerçek ve tahmini değerler arasındaki mükemmel uyumu gösterir.
4. plt.title('Linear Regression: Actual vs Predicted'): Grafik başlığını belirler.
5. plt.xlabel('Actual Stress Level'): X ekseni etiketini belirler.
6. plt.ylabel('Predicted Stress Level'): Y ekseni etiketini belirler.
7. plt.grid(True): Izgara çizgilerini ekler.
8. plt.show(): Grafikleri gösterir.
9. Bu metodun çağrılması, lineer regresyonun gerçek ve tahmini değerlerini içeren bir scatter plot çizer.

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, eğitilmiş bir logistic regresyon modelini değerlendirmektir. Değerlendirme işlemi, modelin test verileri üzerinde nasıl performans gösterdiğini anlamak için kullanılır.
2. y\_pred = model.predict(X\_test): Eğitilmiş modeli kullanarak, test verileri üzerinde tahminler yapar. Bu adım, modelin test verilerini kullanarak bağımlı değişkenin sınıfını tahmin etmesini sağlar.
3. accuracy, conf\_matrix, classification\_rep = self.calculate\_classification\_metrics(y\_test, y\_pred): Sınıflandırma metriklerini hesaplar. Bu adım, doğruluk (accuracy), confusion matrix ve classification report gibi metrikleri hesaplar. Bu metrikler, modelin sınıflandırma performansını ölçer.
4. self.print\_classification\_metrics(accuracy, conf\_matrix, classification\_rep): Hesaplanan sınıflandırma metriklerini ekrana yazdırır.
5. self.visualize\_roc\_curve(model, X\_test, y\_test): ROC eğrisini görselleştirir. Bu adım, modelin sınıflandırma performansını farklı kesim noktalarında gösteren bir ROC eğrisi çizer.
6. Bu metodun çağrılması, eğitilmiş logistic regresyon modelinin performansını değerlendirir, sonuçları ekrana yazdırır ve ROC eğrisini görselleştirir.

metin, ekran görüntüsü, yazılım, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, sınıflandırma metriklerini (doğruluk, confusion matrix, classification report) hesaplamaktır.
2. accuracy = accuracy\_score(y\_true, y\_pred): Doğruluk (accuracy) metriğini hesaplar. Doğruluk, doğru tahmin edilen gözlemlerin toplam gözlem sayısına oranını ifade eder.
3. conf\_matrix = confusion\_matrix(y\_true, y\_pred): Confusion matrix'i hesaplar. Confusion matrix, sınıflandırma modelinin gerçek ve tahmin edilen sınıflarını karşılaştıran bir matristir.
4. classification\_rep = classification\_report(y\_true, y\_pred): Classification report'u hesaplar. Classification report, doğruluk, hassasiyet, duyarlılık ve F1 skoru gibi sınıflandırma metriklerini içeren bir rapordur.
5. Bu metodun çağrılması, sınıflandırma metriklerini hesaplar ve bu metrikleri doğruluk, confusion matrix ve classification report olarak döndürür.

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, sınıflandırma metriklerini ekrana yazdırmaktır.
2. print('Logistic Regression Classification Model:'): Başlık olarak "Logistic Regression Classification Model"i ekrana yazdırır.
3. print('Accuracy:', accuracy): Doğruluk (accuracy) metriğini ekrana yazdırır.
4. print('Confusion Matrix:'): Confusion matrix'i ekrana yazdırır.
5. print(conf\_matrix): Confusion matrix'i ekrana yazdırır.
6. print('Classification Report:'): Classification report'u ekrana yazdırır.
7. print(classification\_rep): Classification report'u ekrana yazdırır.
8. Bu metodun çağrılması, sınıflandırma metriklerini ekrana yazdırır ve modelin sınıflandırma performansı hakkında bilgi verir.

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, logistic regresyon modelinin ROC eğrisini görselleştirmektir. ROC eğrisi, bir sınıflandırma modelinin hassasiyet ve özgüllük performansını değerlendiren bir grafiktir.
2. fpr, tpr, thresholds = roc\_curve(y\_test, model.predict\_proba(X\_test)[:, 1]): ROC eğrisinin çizimi için gerekli olan false positive rate (FPR), true positive rate (TPR) ve eşik (thresholds) değerlerini hesaplar. model.predict\_proba(X\_test)[:, 1], modelin sınıf olasılıklarının ikinci sütununu (positif sınıfın olasılıkları) alır.
3. roc\_auc = auc(fpr, tpr): ROC eğrisi altındaki alanı (AUC) hesaplar. AUC, ROC eğrisinin altındaki alanın büyüklüğünü ölçer ve bir modelin sınıflandırma performansını özetler.
4. plt.figure(figsize=(8, 8)): 8x8 boyutunda bir figura oluşturur.
5. plt.plot(fpr, tpr, color='darkorange', lw=2, label='ROC curve (area = {:.2f})'.format(roc\_auc)): ROC eğrisini çizer. Eğri rengi darkorange, çizgi kalınlığı lw=2, etiket (label) alanı AUC değerini içerir.
6. plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--'): ROC eğrisinin referans çizgisini çizer. Bu çizgi, bir modelin rasgele tahmin yapması durumunda beklenen performansı gösterir.
7. plt.xlabel('False Positive Rate'): X ekseni etiketini belirler.
8. plt.ylabel('True Positive Rate'): Y ekseni etiketini belirler.
9. plt.title('Receiver Operating Characteristic (ROC) Curve'): Grafik başlığını belirler.
10. plt.legend(loc='lower right'): Grafikteki legend'ı sağ alt köşeye yerleştirir.
11. plt.show(): Grafikleri gösterir.
12. Bu metodun çağrılması, logistic regresyon modelinin ROC eğrisini çizer ve görselleştirir.

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, veriyi alt grafiklerle görselleştirmektir.
2. r = 4, c = 6, it = 1: Alt grafiklerin satır, sütun ve iterator (konum belirlemek için) değerlerini belirler.
3. plt.figure(figsize=(18, 12)): Genel figür boyutunu belirler.
4. for i in self.df.columns:: DataFrame'in sütunları üzerinde döngü başlatır.
5. plt.subplot(r, c, it): Alt grafik konumunu belirler.
6. if self.df[i].nunique() > 6:: Eğer sütunun benzersiz değer sayısı 6'dan fazlaysa, kernel density estimate (KDE) plotunu çizer.
7. sns.kdeplot(self.df[i]): KDE plotunu çizer.
8. plt.grid(): Izgara ekler.
9. else:: Eğer sütunun benzersiz değer sayısı 6 veya daha azsa, count plotunu çizer.
10. sns.countplot(x=self.df[i]): Count plotunu çizer.
11. plt.xlabel(i): X ekseni etiketini belirler.
12. plt.ylabel('Count' if self.df[i].nunique() <= 6 else 'Density'): Y ekseni etiketini belirler, eğer benzersiz değer sayısı 6'dan küçükse 'Count', değilse 'Density' olarak belirlenir.
13. it += 1: Iterator değerini bir artırır.
14. plt.tight\_layout(): Grafikleri düzenler.
15. plt.show(): Grafikleri gösterir.
16. Bu metodun çağrılması, veriyi alt grafiklerle görselleştirir ve her sütun için count plot veya KDE plot çizer.

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Bu metodun amacı, tüm iş akışını çalıştırmaktır.
2. self.visualize\_data(): Veriyi alt grafiklerle görselleştiren metodun çağrılması.
3. self.add\_correlation\_heatmap(): Verinin korelasyon ısı haritası ile görselleştirilmesi.
4. self.remove\_unnamed\_column(): Veri çerçevesinde "Unnamed: 0" sütunu varsa, bu sütunun kaldırılması.
5. self.fill\_missing\_values(): Veri çerçevesindeki eksik değerlerin ortalama değerlerle doldurulması.
6. self.drop\_column\_with\_lowest\_correlation(): Hedef değişkene en düşük korelasyona sahip olan sütunun kaldırılması.
7. self.apply\_pca(n\_components=10): PCA'nın uygulanarak boyut indirgenmesi. 10 bileşeni saklamak üzere ayarlanmıştır, bu sayı ihtiyaca göre değiştirilebilir.
8. X\_linear\_train, X\_linear\_test, y\_linear\_train, y\_linear\_test = self.split\_data\_linear(): Veriyi lineer regresyon modeli için eğitim ve test setlerine bölen metodun çağrılması.
9. linear\_model = self.train\_linear\_model(X\_linear\_train, y\_linear\_train): Lineer regresyon modelinin eğitilmesi.
10. self.evaluate\_linear\_model(linear\_model, X\_linear\_test, y\_linear\_test): Eğitilen lineer regresyon modelinin değerlendirilmesi.
11. target\_class = 1: Logistic regresyon modeli için hedef sınıfın belirlenmesi (diğer sınıflara karşı bu sınıfın tahmin edilmesi).
12. X\_logistic\_train, X\_logistic\_test, y\_logistic\_train, y\_logistic\_test = self.split\_data\_logistic(target\_class): Veriyi logistic regresyon modeli için eğitim ve test setlerine bölen metodun çağrılması.
13. logistic\_model = self.train\_logistic\_model(X\_logistic\_train, y\_logistic\_train): Logistic regresyon modelinin eğitilmesi.
14. self.evaluate\_logistic\_model(logistic\_model, X\_logistic\_test, y\_logistic\_test): Eğitilen logistic regresyon modelinin değerlendirilmesi.
15. plt.figure(figsize=(10, 6)): Büyük bir figür oluşturulması.
16. scatter = plt.scatter(...): Scatter plot ile gerçek ve tahmin edilen değerlerin görselleştirilmesi.
17. plt.colorbar(scatter, label='Stress level'): Renk skalası eklenmesi.
18. plt.plot([0, 1], [1, 0], color='red', linestyle='--', lw=2): Referans için kırmızı kesik çizgi çizilmesi.
19. plt.title('Logistic Regression: Actual Sleep Quality vs. Predicted Stress Level'): Başlık eklenmesi.
20. plt.xlabel('Actual Sleep Quality (Class 1)'): X ekseni etiketi eklenmesi.
21. plt.ylabel('Predicted Stress Level'): Y ekseni etiketi eklenmesi.
22. plt.grid(True): Izgara eklenmesi.
23. plt.show(): Grafiklerin gösterilmesi.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. bu yapı genellikle bir dosya hem bir modül olarak içe aktarılabilir hem de bir bağımsız betik olarak çalıştırılabilir olmasını sağlamak için kullanılır.

SONUÇLAR:

metin, diyagram, plan, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, diyagram, renklilik içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, çizgi, diyagram, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, çizgi, diyagram içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu