

**Hasan Ferdi Turgutlu Teknoloji Fakültesi**

**Yazılım Mühendisliği Bölümü**

**Veri Madenciliği**

**Proje Raporu**

**182802034 Hasan SOFİOĞLU  
182802038 İnci KÜÇÜK**

**Manisa, 2021**

**ALZAYMIR HASTALIK TESPİTİ**

**(ALZHEIMER DISEASE DETECTION)**

İnci KÜÇÜK 182802038

Hasan SOFİĞOĞLU 182802034

**ÖZET**

Projenin amacı Alzaymır hastalığının belirtilerini taşıyan yada taşımayan hastaların tespitini, makine öğrenmesi algoritmalarını kullanarak olası tahmin yürütmektir.Alzaymır hastalığının tespitinde bu hastalığa ilişkin sağlık ölçüt değerleri(sağlık terimleri),kişinin yaş,eğitim durumu gibi istatiksel bilgilerini içeren bir veri seti üzerinden makine öğrenmesi algoritmaları aracılığıyla seçilmiş bir hastada bu hastalığın olup olmadığının tahmini yürütülmeye çalışılmıştır. Projede 2 ayrı algoritma kullanılmış, eğitimlerden ve testlerden geçirilmiş şekilde en doğru sonuca en hızlı ulaşan algoritma tespit edilmiştir.

**ANAHTAR KELİMELER**

**Alzaymır, bayes, regrasyon, ölçütler, tahmin**

**1)Giriş**

Geliştirilecek mobil tabanlı Psikoloji Tahmin uygulamasında giriş işlemi sağlanması durumunda kullanıcılar yaş ve cinsiyet bilgilerini girip uygulamanın sorduğu sorulara verdikleri yanıtlar ile hangi rahatsızlığa eğilim gösterdiğini yüzdelik olarak görebilmesidir. Kullanıcı ayrıca uzman psikologlarca yazılmış makalelere ulaşabilir ve bu duruma göre bir uzmana başvurma ihtiyacı olacağını kestirebilir. Veriseti: Öncelikli hedefimiz öncelikle bir hastalık üzerinden veriseti oluşturup bu veri setiyle deneyler yapmak. Çıkan istatistik sonuç başarılı görünüyorsa hastalık sayısını gittikçe artırmak. Elle tutulur iki çeşit verimiz olacaktır biri hasta tecrübeleri biri de literatür(Bilimsel olarak kanıtlanmış ve okullarda kullanılan hangi yaşta kişilerin hastalağa yakalanma oranını, cinsiyete göre oranını, hisse göre oranını vs. gösteren kısımlar.). Biz bir hastalık için içerisinde en çok geçen kelimeleri ayrıştırıp belirli istatistiklere ayıracağız. Daha sonra kullanıcıdan aldığımız metnin içinde bu kelimelerin istatisliklerini arayıp ona göre kullanıcıya yüzdelik olarak bu hastalığa ne kadar yakın olduğunu söyleyeceğiz. Veri setini en verimli şekilde kullanıp veri madenciliği yöntemleri için uygun hale getireceğiz. Veri madenciliği yöntemlerinden bizim kısım için kullanabileceğimiz en uygun olan seçeneği deneyeceğiz. Literatür Araştırması

Bu düşündüğümüz mobil platformda Psikoloojik rahatsızlığı tahmin eden ve kullanıcların verilerini tecrübe olarak kullanıp kendini geliştiren herkes tarafından kolayca ulaşılabilen ilk projedir.

**2)Verilerin Anlatımı**

Proje amacımız doğrultusunda kullandığımız veri seti herhangi bir sağlık merkezi veya kliniğin bu hastalık üzerindeki elde etmiş olduğu sonuçlar yer almaktadır. Bu veri seti 10 farklı özellikteki içermektedir,ve bunun sonucunda ise kişide bu hastalığın bulunması(demented) ya da bulunmaması(non demented) şeklinde çıkarımları içeren bir veri setidir.

Veri Setinde Bulunan Özellikler:

Group 🡪Class (Sınıf): Bu özellik kişinin bu hastalığı taşıdığı ya da taşımadığını gösteren sonuçtur.  
Age 🡪 Age (Yaş) : Veri setinde bulunan kişilerin yaşlarını göstermektedir.  
EDUC 🡪 Years of Education (Eğitim Yılı) : Veri setinde bulunan kişilerin eğitim yılını içermektedir.  
SES 🡪 Socioeconomic Status / 1-5 (Sosyoekonomik Durum /1-5) : Hem toplumsal hem ekonomik alanı ve bunlar arasındaki ilişkileri ilgilendiren  
MMSE 🡪Mini Mental State Examination (Mini Mental Durum Muayenesi) : Mini mental test esas olarak Alzheimer hastalığı taramasında ve bir kişinin bilişsel performansının zaman içindeki değişimin ölçülmesidir.  
CDR 🡪Clinical Dementia Rating (Klinik Demans Derecelendirmesi) :Kliniğin belirlediği demans ölçüm krirteri  
eTIV 🡪Estimated total intracranial volüme (Tahmini Toplam Kafa içi hacim) : Klinik tarafından belirlenmiş tahmini toplam kafa iç hacminin ölçüm sonucunu verir.  
nWBV 🡪Normalize Whole Brain Volume (Tüm Beyin Hacmini Normalleştir) : Klinik sonuçların hesaplanmasıyla elde edilmiş beyin hacim verilerinin normalleştirilmesiyle ortaya çıkan değerdir.  
ASF 🡪Atlas Scaling Factor (Atlas Ölçekleme Faktörü) : Etıv verisinin otomatik olarak türetmek için kullanılan tıbbi bir terimdir.

**3)Önerilen Yöntemler**

Bu projeyi gerçekleştirmek için sınıflandırma yaklaşımını ele aldık. Sınıflandırmada bulunan lojistik regresyon analizi ve naive bayes algortimalarını kullandık. Bu 2 ayrı algorritma ile ayrı ayrı veri setimizi eğittik ve elde ettiğimiz sonuçları karşılaştırdık. Sisteme her iki tahmin sonucuna değerlendirme ölçütleri uyguladık(accularcy,sensisvity,specifiatly gibi)

**3.1) Veri Normalizasyon Adımlarının Yapılması**

Projede kullanmak istediğimiz veri setinde birçok eksik veri, sistem eğitiminde işleme alamayıp zorluk çıkarabilecek verileri tespit ettik ve uygun işlemleri uygulayarak veri setini eğitime hazır hale getirdik.

|  |
| --- |
| veri=pd.read\_csv("desktop/alzheimer.csv") |
| veri.head() |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Group | M/F | Age | EDUC | SES | MMSE | CDR | eTIV | nWBV | ASF |
| 0 | Nondemented | M | 87 | 14 | 2.0 | 27.0 | 0.0 | 1987 | 0.696 | 0.883 |
| 1 | Nondemented | M | 88 | 14 | 2.0 | 30.0 | 0.0 | 2004 | 0.681 | 0.876 |
| 2 | Demented | M | 75 | 12 | NaN | 23.0 | 0.5 | 1678 | 0.736 | 1.046 |
| 3 | Demented | M | 76 | 12 | NaN | 28.0 | 0.5 | 1738 | 0.713 | 1.010 |
| 4 | Demented | M | 80 | 12 | NaN | 22.00 | 0.5 | 1698 | 0.701 | 1.034 |

(3.1 Tablo 1 –Veri Normalizasyon Adımları Öncesi)

|  |
| --- |
| sns.countplot(x="Group",data=veri) |
| <AxesSubplot:xlabel='Group', ylabel='count'> |

Veri Setinde Bulunan Demented Nondemented converted olan kişi sayısı dağılım grafiği yukarda verilen kod tarafından uygulanılarak aşağıdaki grafik elde edilir. Bu işlemde veriler işlenebilir hale getirilir.

(3.1 Grafik 1 –Kişi Sayısı Dağılım Grafiği)

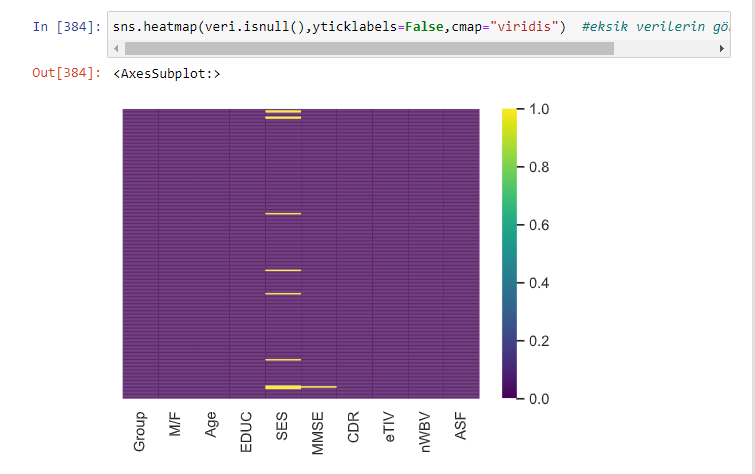
|  |
| --- |
| sns.countplot(x="Group", hue="M/F",data=veri) |
| <AxesSubplot:xlabel='Group', ylabel='count'> |

Yukardaki kod ile Demented ve nondemented ya da converted olan kişilerin cinsiyet dağılım grafiğini elde ediyoruz(3.1 Grafik 2). “M” harfi erkek cinsiyetini, “F” harfi kadın cinsiyetini temsil etmektedir.

(3.1 Grafik 2 –Cinsiyet Dağılım Grafiği)

|  |
| --- |
| sns.heatmap(veri.isnull(),yticklabels=False,cmap="viridis")  #eksik verilerin görselleştirilmesi |

Yukarıdaki kod ile veri normalizasyon işlemleri için eksik veriler tespit edilir.Grafikte (3.1 Grafik 3) Sarı ile gösterilen alanlarda boş değerler mevcuttur. Boş değerler verilerin işlenmesini engeller. Bu değerlerin hangi kategoriler altında olduğunu tespit edebilmek için görsel bir grafik kullanmak istedik. Uygun komutu yazarak 2 değerin yoğunluklu olarak boş değerlerin bulunduğunu tespit ettik(Sarı renkli alanlar bu boş değerlerin olduğu alanları göstermektedir

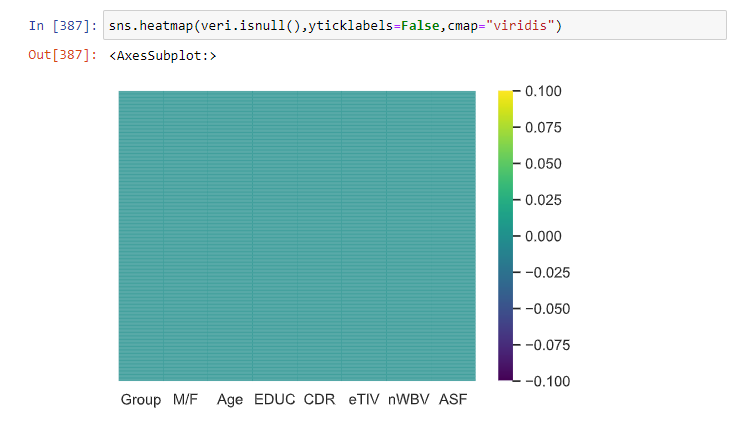


(3.1 Grafik 3 –Normalizasyonda Eksik Verilerin Tespiti)

Bu eksik değer barındıran özellikler verisetinden aşaıdaki kod komutu ile çıkartılır.ve yeniden görselleştirme kullanarak başka sorunlu veri var mı kontrol edilir.

|  |
| --- |
| veri.drop(["SES","MMSE"],axis=1,inplace=True)  #eksik veriler veri setinden kaldırılır |

Aşağıdaki grafikte(3.1 Grafik 4) görüldüğü gibi Veri setindeki boş eksik vb. kullanamadığımız değerleri veri setimizden çıkartmış olduk.



**(3.1 Grafik 3 –Normalizasyondan Sonra Veri Eksikliği Takibi)**

Veri setinde bulunan string ifadeler mevcuttur.(demented, M,F gibi.).Bu değerlere de işlemler uygulanarak işleme uygun hale getirilmelidir.

İlgili komutlardan sonra tekrar grafiğimize dönerek boş değer sütunlarının veri setinden kaldırılıp kaldırılmadığını kontrol ettik.

Grafikte yeşil alan veri setindeki boş eksik vb. kullanamadığımız değerleri veri setimizden çıkartmış olduğumuz anlamına gelmektedir.

Ardından veri setimizdeki değişken tiplerini işlem gereken herhangi bir değer olup olmadığını kontrol etmek tekrar inceledik.

Veri setinde bulunan string ifadeler mevcuttur (Group, M/F gibi). Bu değerlere de işlemler uygulanarak işleme uygun hale getirilmelidir. Tıbbi terimler hakkında herhangi bir isim değişikliğine gitmedik. Çünkü kalıp ifadeler ve günlük bir dile çevirmenin pek uygun olmayacağını, bu alanda uğraşan doktorların anlayabileceği terimler olduğu için isim terimlerin adlarını günlük dile çevirmedik.

|  |
| --- |
| Group=pd.get\_dummies(veri["Group"],drop\_first=True)  #uygun tipte olmayan veriler dummy değişkeni ile değiştirilir  Group.head() |

|  |  |  |
| --- | --- | --- |
|  | Demented | Nondemented |
| 0 | 0 | 1 |
| 1 | 0 | 1 |
| 2 | 1 | 0 |
| 3 | 1 | 0 |
| 4 | 1 | 0 |

**(3.1 Tablo 2 –Veri Normalizasyon Adımları Uygun Tipte olmayan verilerin değişimi)**

|  |
| --- |
| sex=pd.get\_dummies(veri["M/F"],drop\_first=True)  sex.head() |

|  |  |
| --- | --- |
|  | M |
| 0 | 1 |
| 1 | 1 |
| 2 | 1 |
| 3 | 1 |
| 4 | 1 |

**(3.1 Tablo 3 –Cinsiyetleri işlenebilir hale getirmek)**

|  |
| --- |
| Group=pd.get\_dummies(veri["Group"],drop\_first=True)  sex=pd.get\_dummies(veri["M/F"],drop\_first=True)  veri.drop(["Group","M/F"],axis=1,inplace=True)  veri=pd.concat([veri,Group,sex],axis=1) |

Tablo 3 te de görüldüğü gibi cinsiyetleri ifadesel olarak 0 ve 1 olacak şekilde düzenlemiş olduk. Veri ön işlemeden geçti. Artık veri setimiz eğitim için uygun hale gelmiştir.

Veri setinde son kez incelemeden geçirdik. Object tipinde vb. veri türü bulunmamaktadır.

Verilerin tamamen işlenmeye hazır halini aşağıda tablo 4’te görebiliyoruz.

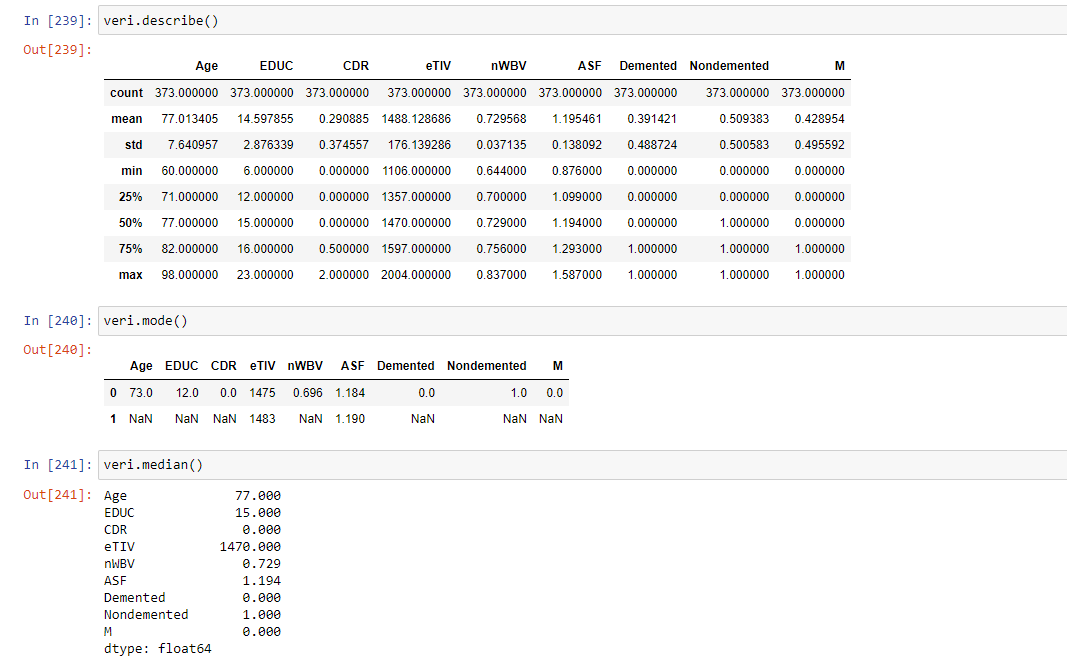
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Age | EDUC | CDR | eTIV | nWBV | ASF | Demented | NonDemented | M/F |
| 0 | 87 | 14 | 0.0 | 1987 | 0.696 | 0.883 | 0 | 1 | 1 |
| 1 | 88 | 14 | 0.0 | 2004 | 0.681 | 0.876 | 0 | 1 | 1 |
| 2 | 75 | 12 | 0.5 | 1678 | 0.736 | 1.046 | 1 | 0 | 1 |
| 3 | 76 | 12 | 0.5 | 1738 | 0.713 | 1.010 | 1 | 0 | 1 |
| 4 | 80 | 12 | 0.5 | 1698 | 0.701 | 1.034 | 1 | 0 | 1 |
| … | … | … | … | … | … | … | … | … | … |
| 368 | 82 | 16 | 0.5 | 1693 | 0.694 | 1.037 | 1 | 0 | 1 |
| 369 | 86 | 16 | 0.5 | 1688 | 0.675 | 1.040 | 0 | 0 | 1 |
| 370 | 61 | 13 | 0.0 | 1319 | 0.801 | 1.331 | 0 | 1 | 0 |

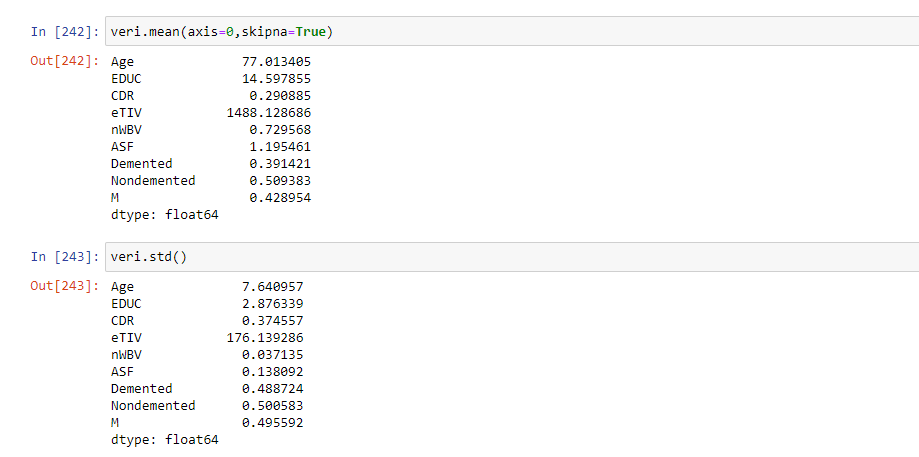
[373 rows x 9 columns]

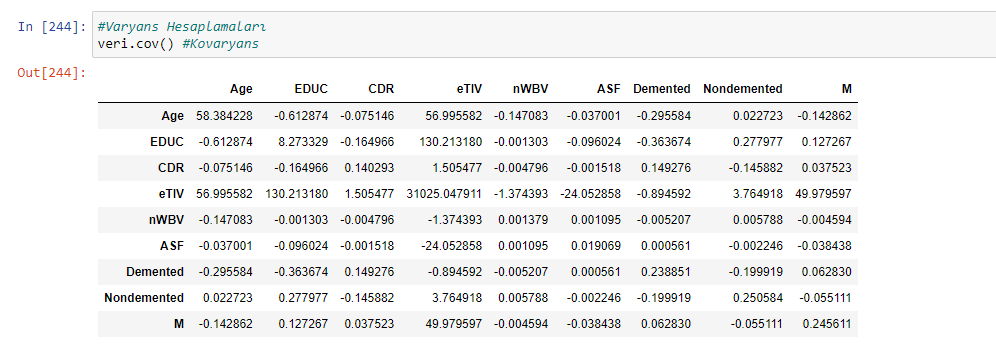
(3.1 Tablo 4 – Veri seti Son Hal)

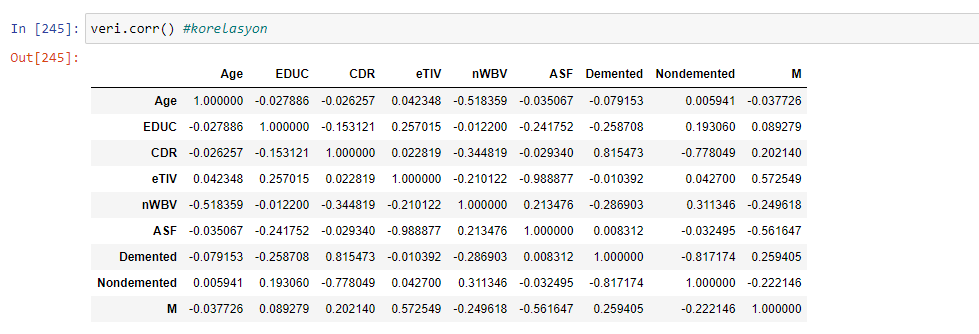
**3.2) Veri Seti Temel İstatistiksel Değerleri**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Age | EDUC | CDR | eTIV | nWBV | ASF | Demented | NonDemented | M/F |
| count | 373.0 | 373.0 | 373.0 | 373.0 | 373.0 | 373.0 | 0373.0 | 373.0 | 373.0 |
| Mean | 77.013 | 14.5978 | 0.2308 | 1488.126 | 0.729568 | 1.195461 | 0.391421 | 0.509383 | 0.428954 |
| std | 7.6409 | 2.8763 | 0.3745 | 176.1392 | 0.0371 | 0.138092 | 0.488724 | 0.500583 | 0.495592 |
| Min | 60.0000 | 6.0000 | 0.000 | 1106.00 | 0.64400 | 0.876600 | 0.00 | 0.00 | 0.00 |
| 25% | 71.00 | 12.00 | 0.0 | 1357.00 | 0.700 | 1.099 | 0.00 | 0.00 | 0.00 |
| 50% | 77.00 | 15.00 | 0.00 | 1470.00 | 0.729 | 1.194 | 0.00 | 1.00 | 0.00 |
| 75% | 82.00 | 16.00 | 0.5 | 1597.00 | 0.75600 | 1.293100 | 1.00 | 1.00 | 1.000 |
| max | 98 | 23.00 | 2.00 | 2004.00 | 0.83700 | ..58700 | 1.00 | 1.00 | 1.00 |









**3.2) Lojistik Regreasyon Analizi (Logistic Regression Analysis)**

Sınıflandırma algoritmalarından biri olan lojistik regrasyon algoritmasını kullanarak hedef tahmin işlemlerini gerçekleştirdik.

Veri setimizde bulunan hedef değişkenlerimiz olan demented ve nondemented olan değişkenleri x ve y ye eşitledik.

|  |
| --- |
| x=veri.drop("Demented",axis=1)  #hedef değişken "demented",geri kalan bileşenler özniteliklerdir  y=veri["Demented"]  #girdi ve çıktı verileri ayarlandı  #Train ve test işlemleri için veri seti parçalara ayıralım  from sklearn.model\_selection import train\_test\_split  x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.25,                                                 random\_state=100)  #tahmin etmesi için kullanıdıgımız hedef için ikili lojistik regresyon analizi kullanılır  from sklearn.linear\_model import LogisticRegression  lg\_model=LogisticRegression(max\_iter=1000)  #eğitim verisi kullnarak modeli eğitelim  lg\_model.fit(x\_train,y\_train)  -- output LogisticRegression(max\_iter=1000) |

Tahmin yapabilmek için veri setimizi parçalara ayırdık. Ardından gerekli komutları yazarak modelimizi fit ettik.

Modeli fit ettikten sonra elde edilen sistem tahmin değerleri:

|  |
| --- |
| #tahmin etmesi için kullanıdıgımız hedef için ikili lojistik regresyon analizi kullanılır  from sklearn.linear\_model import LogisticRegression  --output lg\_model=LogisticRegression(max\_iter=1000)  #eğitim verisi kullnarak modeli eğitelim  lg\_model.fit(x\_train,y\_train)  --out LogisticRegression(max\_iter=1000)  lg\_model.score(x\_test,y\_test)  #buradan anlaşılıyor ki modelimiz yeni veri tahmini %96 başarı ile tahmin edecek  --out 0.9680851063829787  lg\_model.score(x\_train,y\_train) #modelin doğruluk skoru  --out 0.9390681003584229  lg\_model=LogisticRegression(C=0.1,max\_iter=1000)  lg\_model.fit(x\_train,y\_train)  --out LogisticRegression(C=0.1, max\_iter=1000)  print(lg\_model.score(x\_test,y\_test))  print(lg\_model.score(x\_train,y\_train))  #modelin eğitim verisi üzerindeki skoru  --0.9680851063829787 0.931899641577061 |

**lg\_model.score** komutunun verdiği sonuç makine öğrenmesi uyguladığımız veri setinin bundan sonraki tahmin işlemlerinde sistemin yüzde kaç doğruluk ile bu tahmin işlemini gerçekleştireceğinin sonucunu vermektedir. Model başarı sonucuna göre sistem bir sonraki eklenecek veriyi %96 başarı ile tahmin edecektir.

**3.2.1)Model Değerlendirme**

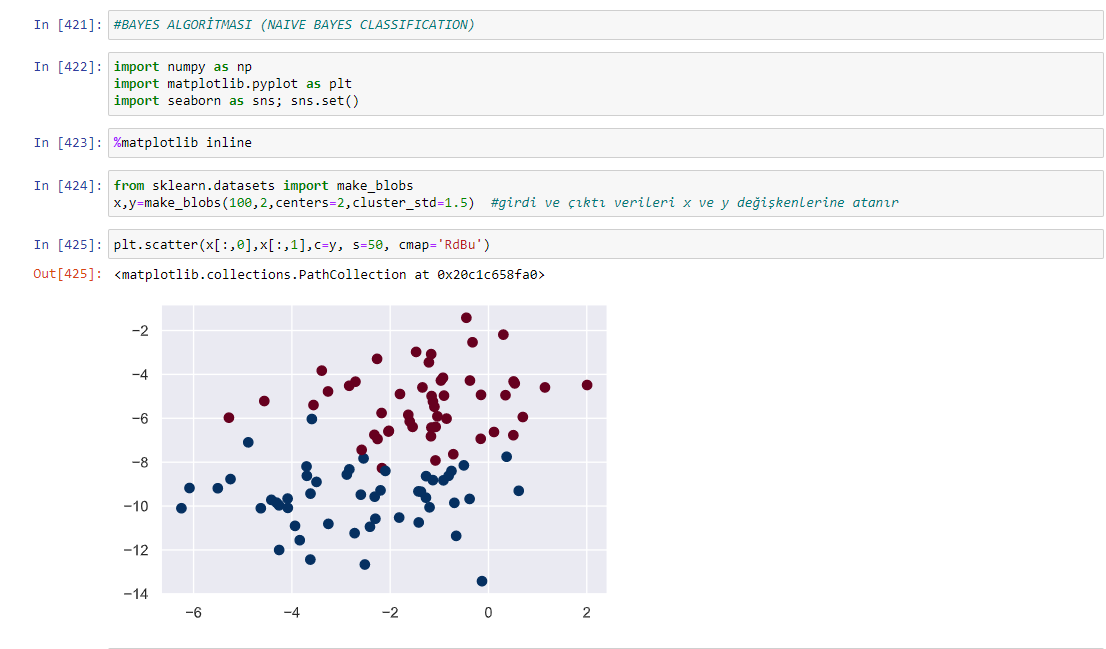
Burada modelimizi test etmek için yeni\_veri adında bir veri belirliyoruz ve sistemden bu değerli bir hasta değerleriyle kişinin hasta olup olmadığının tahminin yapmasını istiyoruz.İlgili komutlar sonucunda sistem array([1]) sonucu veriyor. Veri setinden alınan örnek sonunda da eşleşmesi tahminin başarılı bir şekilde yapıldığını göstermektedir. Modelimiz yüksek doğruluk oranı ile tahminler yapabiliyor.

|  |
| --- |
| #MODEL DEĞERLENDİRME  #modelin sınıflarının ne kadar doğru tahmin ettiğini bulmak için confusion kullanalım  #test verisindeki değerleri bulalım  from sklearn.metrics import confusion\_matrix  tahmin=lg\_model.predict(x\_test)  confusion\_matrix(y\_test,tahmin)  --out: array([[55, 1], [ 2, 36]], dtype=int64)  # 55 tane true pozitif, 36 tane true negatif tahmin yapılmış 1 tahmin pozitif iken yanlış tahminde bulunmuş  #2 pozitif değer de negatif bulunmuş  veri.head() |

|  |
| --- |
| #eğitilmiş sistemin doğru tahmin yapıp yapmadıgının kontrolü  import numpy as np  yeni\_veri=np.array([[80,12,0.5,1698,0.701,1.034,0,1]])  lg\_model.predict(yeni\_veri)  #sistem 1 değeri verdi bu demek ki bu kişi demented olabilir  --out array([1], dtype=uint8)  lg\_model.coef\_ #modeldeki özniteliklerin tahmin edilen katsayıların görüntülenmesi  --out : array([[-1.18534845e-02, -1.42147641e-01, 1.51854240e+00, 1.47436898e-03, 6.44653900e-03, 1.62621339e-01, -1.68342173e+00, 4.10250486e-01]]) |

**3.4) BAYES ALGORİTMASI (NAIVE BAYES ALGORİTHM)**

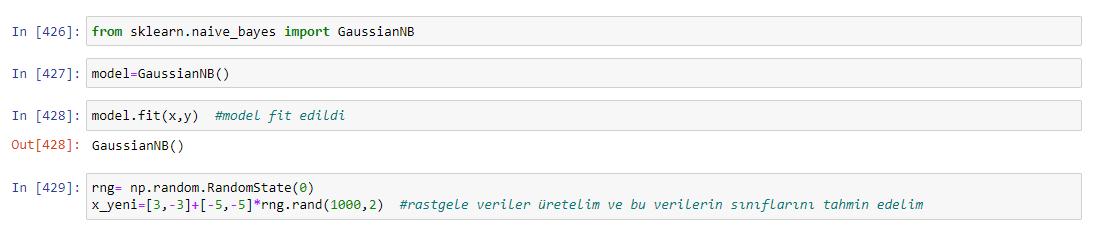
Bu sınıflandırma algoritması ile aynı veri setini bir de bu yöntem ile tahmin etmesi için ilgili işlemleri yaptık.



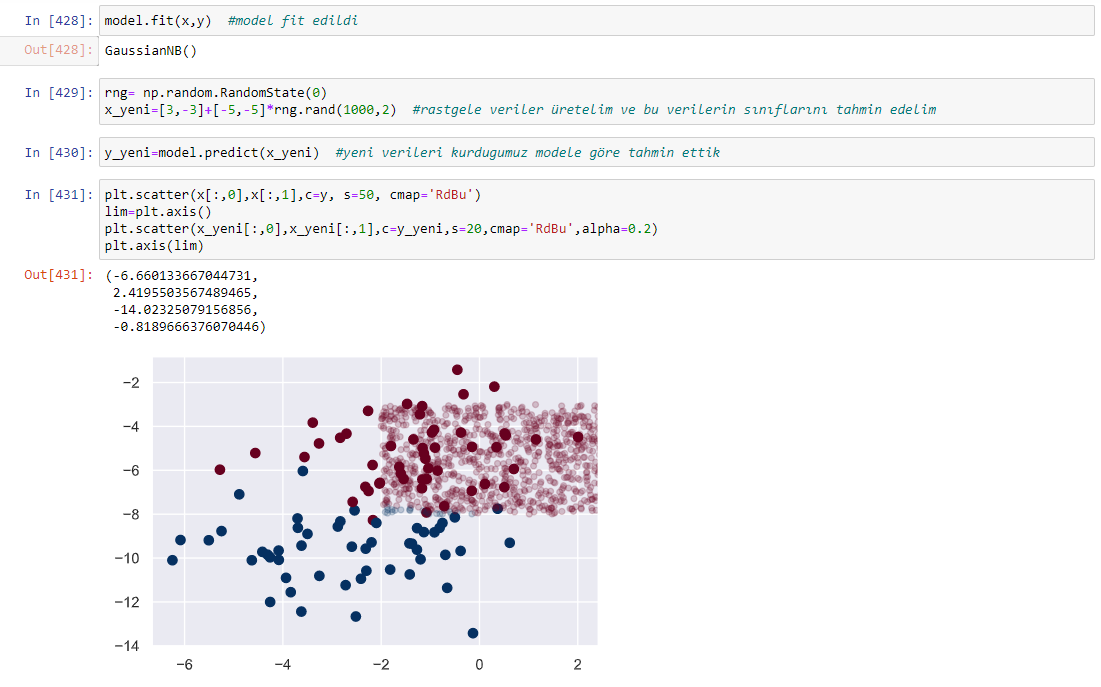
**demended ve nondemended dağılım grafiği**

Tahmin işleminden önce demented ve nondemented dağılımı görmek için grafik kullandık.

Bayes algoritması kullanarak tahmin işlemleri için uygun komutları yazıyoruz.



Modelimizi fit ettik.Ve sistemin yüzde kaç başarı oranı ile tahminleri gerçekleştirebildiğini görmek için x\_yeni adında rastgele bir veri giriş yaptık ve sisteme ekledik.



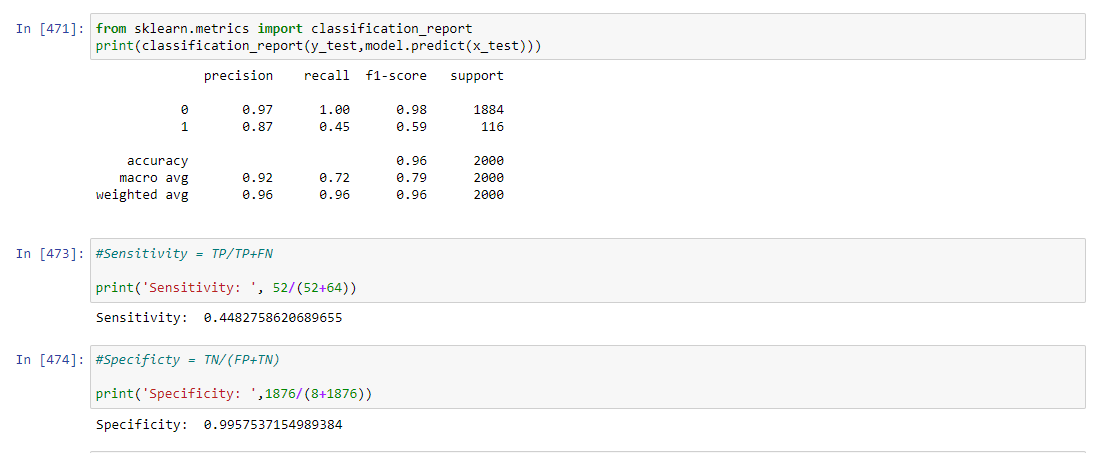
Y\_yeni model adında bir değişken oluşturduk.Bu değer tahmin işlemini sonucunu veren değerdir.

Ardından sistemin tahmin sonucunu grafik üzerinde görmek için uygun komutları yazdık. Sistem sonucunda bulanık kırmızı görülen alanlar sistemin tahmin ettiği aralığı göstermektedir.

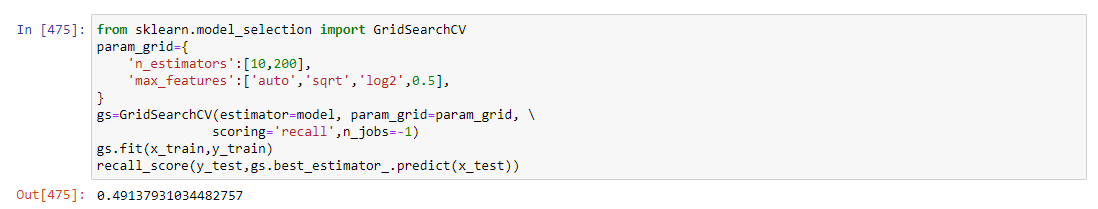
**3.5) Değerlendirme Metrikleri (Evaluation Metrics) -Naive Bayes**

Bayes algoritması kullanarak oluşturduğumuz bu modelin testlerini yaptık. Accuracy,recall,f1-score,sensitivity,specifity ve diğer başka ölçüm değerlerini bulduk.





Modelin recall\_score ölçüm değeri



**3.6) Değerlendirme Metrikleri (Evaluation Metrics)– Lojistik Regresyon**

Bayes algoritması kullanarak oluşturduğumuz bu modelin testlerini yaptık.Accuracy,recall, f1-score,sensitivity,specifity ve diğer başka ölçüm değerlerini bulduk.

#Accuracy = Number of correct predictions\Total number of predictions

from sklearn.naive\_bayes import GaussianNB

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test = train\_test\_split(x,y,test\_size=0.2, \

                                                 random\_state=42)

model=GaussianNB().fit(x\_train,y\_train)

model.score(x\_test,y\_test)

0.95

#DummyClassifier on the same dataset:

from sklearn.dummy import DummyClassifier

dummy = DummyClassifier(strategy='most\_frequent',random\_state=42)

dummy.fit(x\_train,y\_train)

dummy.score(x\_test,y\_test)

**0.4**

from sklearn.datasets import make\_classification

x,y = make\_classification(n\_samples=10000, n\_classes=2, \

                         weights=[0.95,0.05], random\_state=42) from sklearn.ensemble import RandomForestClassifier

model=RandomForestClassifier(random\_state=42).fit(x\_train,y\_train)

model.score(x\_test,y\_test)

0.95

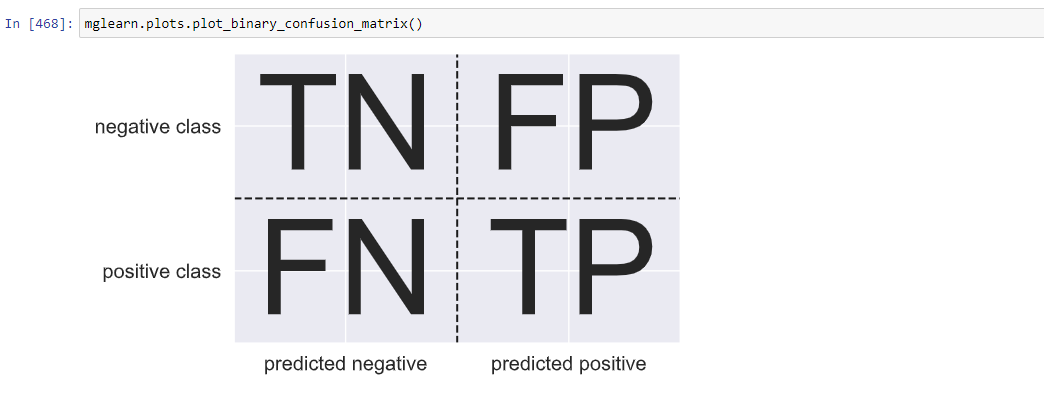
Burada lojistik regrasyon ile eğitilmiş modelimizin accuracy,dummy score ve model score değerlerini elde ettik. Değerlerin yüksek başarı oranı vermesi modelin bu özellikler bakımından başarılı olduğunu göstermektedir.

Modelin confusion\_matrix değerlerini ilgili komutlar yazarak sonuçlarını aldık. Bu değerleri diğer metrikleri toplamak için kullandık

from sklearn.metrics import classification\_report

print(classification\_report(y\_test,model.predict(x\_test)))

array([[11, 1],[0,8]], dtype=int64



Bu değerler bulduğumuz matrix değerleri ile eşleşmektedir. Ardından modelimizin recall f1-score vb. diğer performans değerlerini bulmak için ilgili komutları yazdık.



Modelin sensitivity ve specificity değerleri için fomüllerden yola çıkarak kalan değerleri de hesapladık.

**4)DENEYSEL SONUÇ VE TARTIŞMA**

Sınıflandırma yaklaşımını kullanarak 2 farklı algoritma kullanarak modelin hangi algoritmada daha başarılı olduğunu tespit etmek için çeşitli metriklerle test ettik. Alınan sonuçlara göre karşılaştırıldığında lojistik regrasyon modeliyle eğitilmiş sistemin başarı sonuçları daha yüksek olduğu tespit edilmiştir. Modellerin specificity, sensitivity vb diğer metrik ölçümlerine kıyasla Bayes algoritmasındaki bulduğumuz değerlere göre daha başarılı sonuçlar vermiştir. İki modelde başarı olarak yüksek yüzdeliklerde tahmin değerleri verebiliyor. Ancak yaptığımız test ölçümlerine göre bu durum değişmiştir. Bu değerler yukarıdaki sonuç ve grafiklerle de desteklenmektedir.

**5)SONUÇ**

Sonuç olarak alzaymır hastalık tespit veri setini kullanarak bir kişide bu hastalığın bulunup bulunmadğının tahminin yapılmasını hedefleyen bu projede, çeşitli algoritma kullanarak ve ve bu modellerin başarım oranlarını kıyaslayabilmek için yaptığımız test sonuçları başarılı bir şekilde gerçeklemiştir.

**Yazarların Katkıları**

Veri seti normalizasyon, Future selection, Bayes veri işlemlerini Hasan SOFİOĞLU raporlamıştır ve yukarıda işlemleri adım adım anlatmıştır.

Veri seti temel istatiksel bilgilerin raporlanması, lojistik regrasyon analizi ve ölçüm değerlerini İnci KÜÇÜK tarafından yapılmıştır. Ve yukarıda detaylı şekilde anlatımı mevcuttur.

Kullanılan bilgisayarın özellikleri:

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Kaynaklar**

Kullanılan Veri seti:

https://www.kaggle.com/brsdincer/alzheimer-features

Terimler için Yararlanılan Kaynaklar:

<https://www.doktorfizik.com/norolojik/>

<https://pubmed.ncbi.nlm.nih.gov/15488422/>

Diğer

https://www.youtube.com/watch?v=3y9XVlk9cDA&t=86s

<https://www.youtube.com/watch?v=Cb77_3veTTU>

<https://www.youtube.com/watch?v=5U8hc4irdbQ&t=950s>

https://www.youtube.com/watch?v=vP06aMoz4v8