

İş Zekası ve Veri Madenciliği Dersi Final Ödevi

AD SOYAD: SEDANUR GEBİÇ NUMARA: 368702

1. Veri setinizi kullanarak derste anlatılan sınıflandırma algoritmalarıyla bir örnek uygulama gerçekleştiriniz. Farklı sınıflandırma algoritmaları ile elde ettiğiniz sonuçları karşılaştırarak yorumlayınız. Karşılaştırmadan kastedilen hangi algoritmanın daha doğru sonuç ürettiğidir.

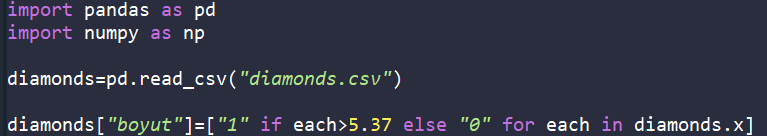
**LOGİSTİC REGRESSİON**

**Örnek Uygulamanın Kısa Açıklaması :**

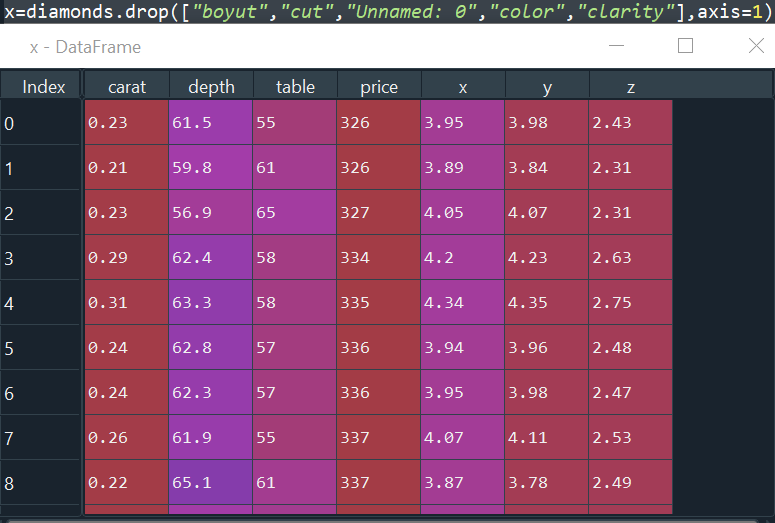
Makine öğrenmesinde sayısal verilerin tahmin edilmesi için tahmin algoritmaları kullanılır. sayısal olmayan yani kategorik verilerin tahmini için ise sınıflandırma kullanılır. Logistic Regression sınıflandırma işlemi yapmaya yarayan bir regresyon yöntemidir. Kategorik veya sayısal verilerin sınıflandırılmasında kullanılır. Bağımlı değişkenin yani sonucun sadece 2 farklı değer alabilmesi durumda çalışır. Veri setimde sınıflandırma işlemlerimde şartı sağlaması için elmasın uzunluğunun orta değerini alarak değerden büyük olanlara 1, değerden küçük olanlara 0 yazacak şekilde boyut adında iki değişken içeren yeni bir sütun oluşturdum.

**Kodlar ve Yorumlama :**

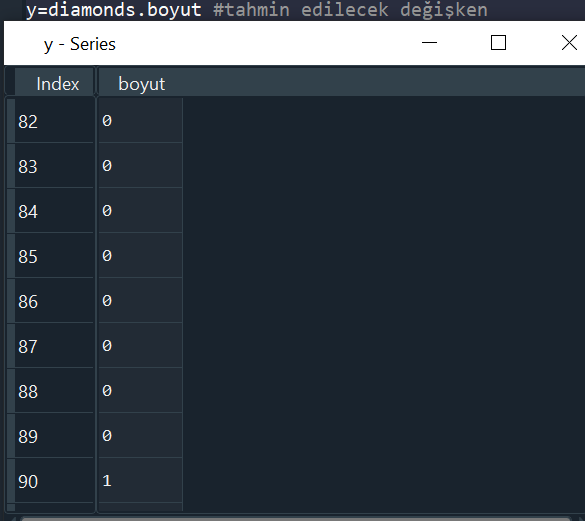
Kütüphaneleri import ederek veri setini okutup sınıflandırma işlemlerimde kolaylık sağlaması için elmasın uzunluğunun orta değerini alarak değerden büyük olanlara 1, değerden küçük olanlara 0 yazacak şekilde boyut adında iki değişken içeren yeni bir sütun oluşturdum.



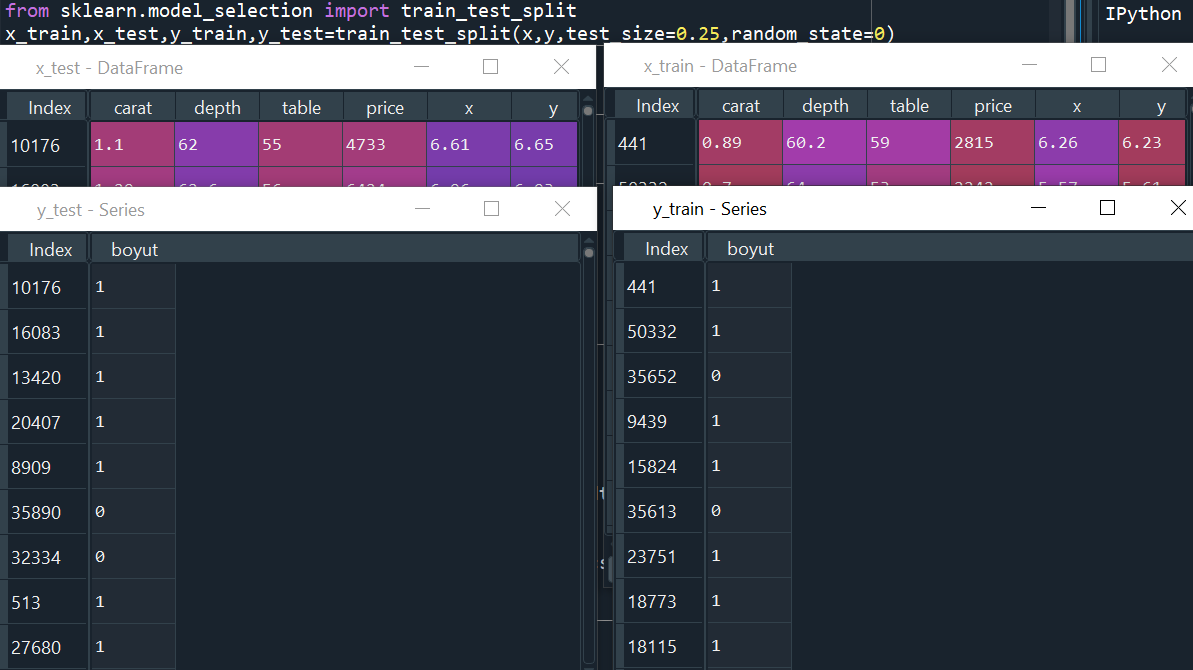
Kategori değişkeni olan boyutu ve gereksiz olan diğer alanları çıkararak y’yi tahmin edecek değişkenleri oluşturdum. x teste bağlı olan y değerleri;



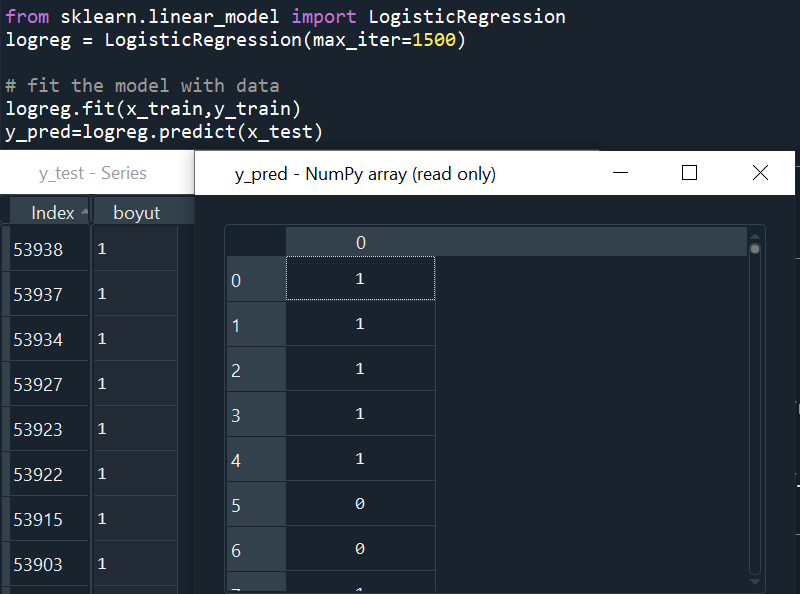
Tahmin edilecek değişkeni y olarak tanımladım.



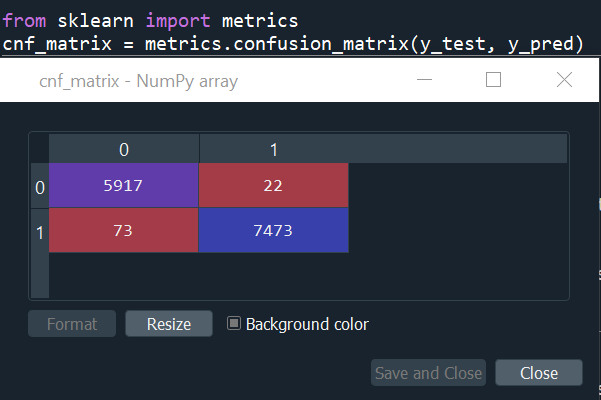
Yukarıda oluşturduğumuz x ve y ile birlikte test size ile değerlerin %25 ini alıyoruz ve aynı yerden bölmesi için random state kullanıyoruz. Böylelikle x\_test, y\_tes, x\_train, y\_train oluşturuyoruz.



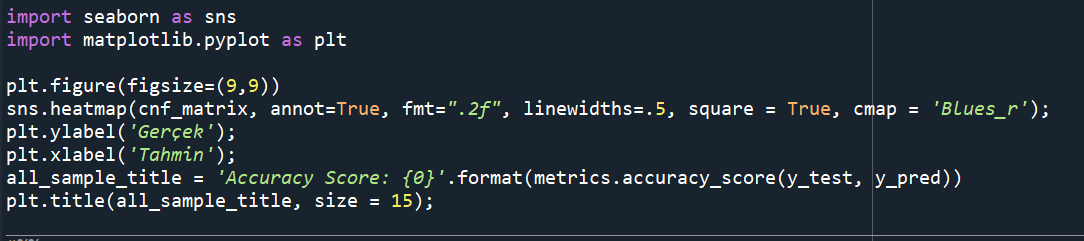
Logistic Regressionu fit ederek x değerlerini kullanarak y değerlerini tahmin etmesini isteyerek ortaya y\_pred diye bir set oluşturuyoruz. Y\_pred, x\_teste bağlı olan gerçek y\_test değerlerini kullanarak train verisinde oluşturduğumuz modeli kullanarak tahmin ettiğimiz y değerlerini ortaya koyuyoruz.

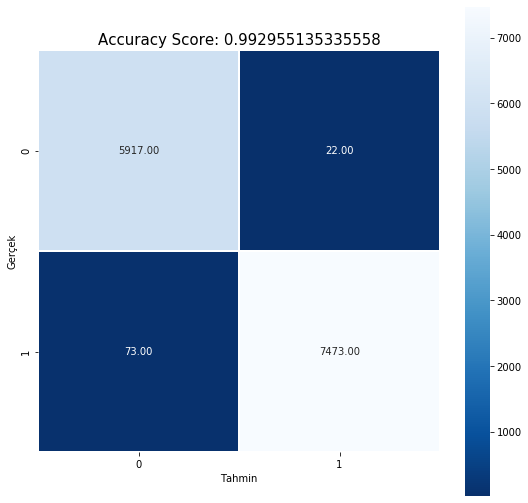


Tahmin ettiğimiz y değerleri ile gerçek ye değerlerinin ne kadar doğru olduğunu görmek için matrix kullanıyoruz. Boyut sütununda 0 küçükleri, 1 büyük göstermektedir. 0 olan ve bizim 0 olarak tahmin ettiğimiz doğru olan değer sayısı 5917, 0 olan ve bizim 1 olarak yanlış tahmin ettiğimiz değer sayısı 22, 1 olan ve bizim 1 olarak tahmin ettiğimiz doğru olan değer sayısı 7473, 1 olan ve bizim 0 olarak yanlış tahmin ettiğimiz değer sayısı 73’tür. Yani toplam 95 değeri yanlış olarak tahmin etmişiz.

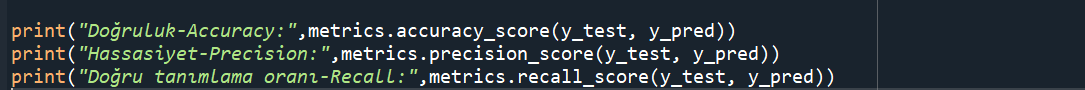


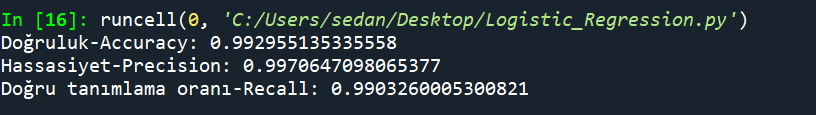
Aynı işlemi heatmap grafi ile çizdiriyoruz. Bu kez ne kadar doğru tahmin ettiğini yani güvenilirlik değerinide üstüne yazdırıyoruz. Yani oluşturduğumuz matrix setini bir grafiğe dönüştürüyoruz. Güvenilirlik değeri ne kadar 1’e yakınsa o kadar doğru olduğunu anlıyoruz. Güvenilirlik değerimizinde 0.99 olduğunu yani çok güvenilir olduğunu görmekteyiz.





Doğruluk il metodun doğruluk değerini, hassasiyet ile bu işlemi ne kadar hassas bir şekilde yaptığını ve doğru tanımla oranı ile de tahminimizin ne kadar doğru olduğunu print ile yazdırıyoruz.





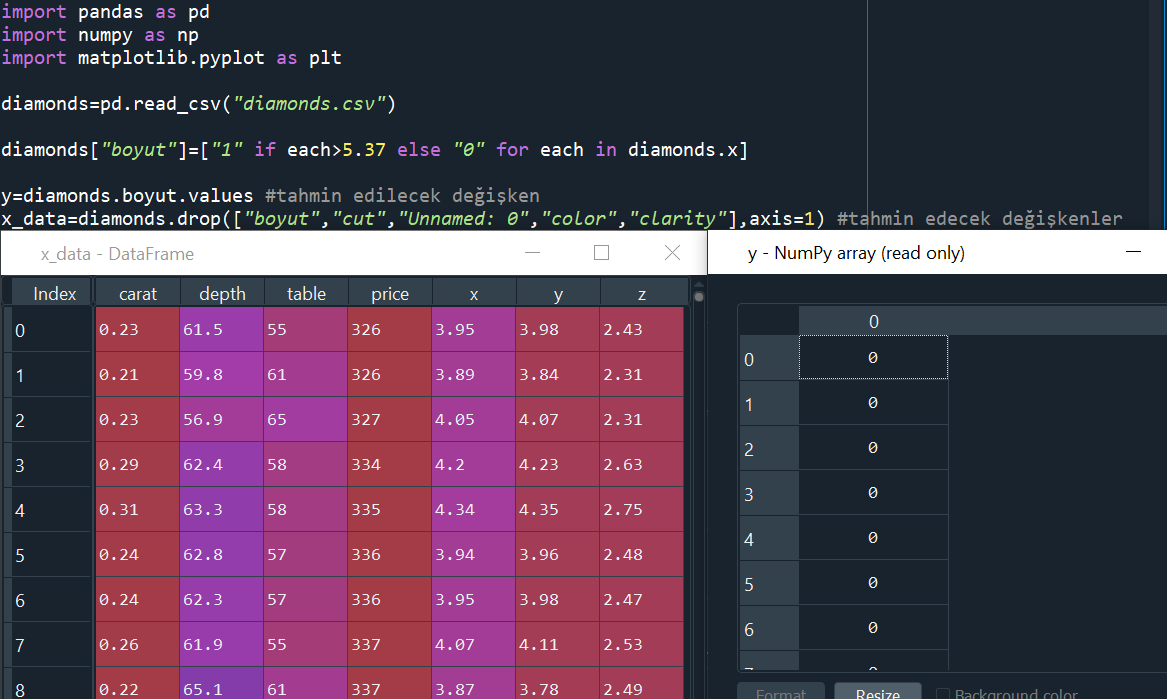
**K- NEAREST NEİGHBORS**

**Örnek Uygulamanın Kısa Açıklaması:**

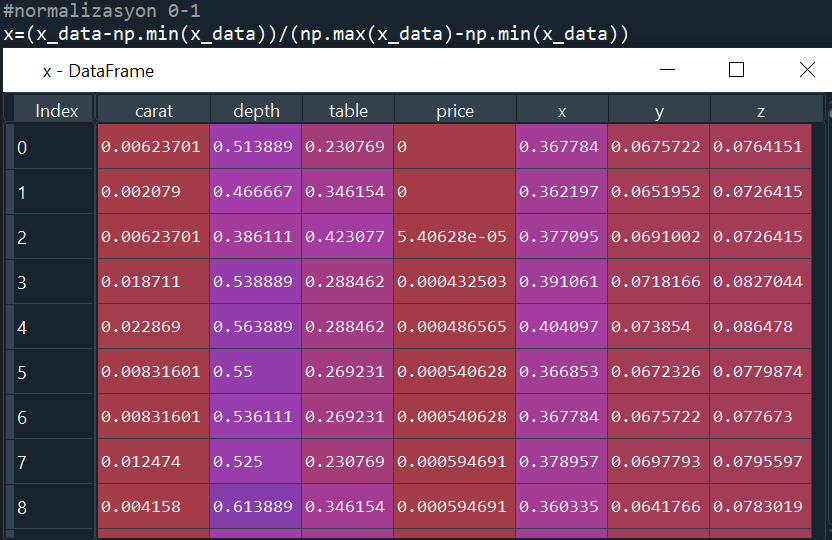
K-Nearest Neighbor algoritması en basit ve en çok kullanılan sınıflandırma algoritmasından biridir. K-NN  parametrik olmayan ,  tembel bir öğrenme algoritmasıdır. Veri setimde sınıflandırma işlemlerimde kolaylık sağlaması için elmasın uzunluğunun orta değerini alarak değerden büyük olanlara 1, değerden küçük olanlara 0 yazacak şekilde boyut adında iki değişken içeren yeni bir sütun oluşturdum.

**Kodlar ve Yorumlama :**

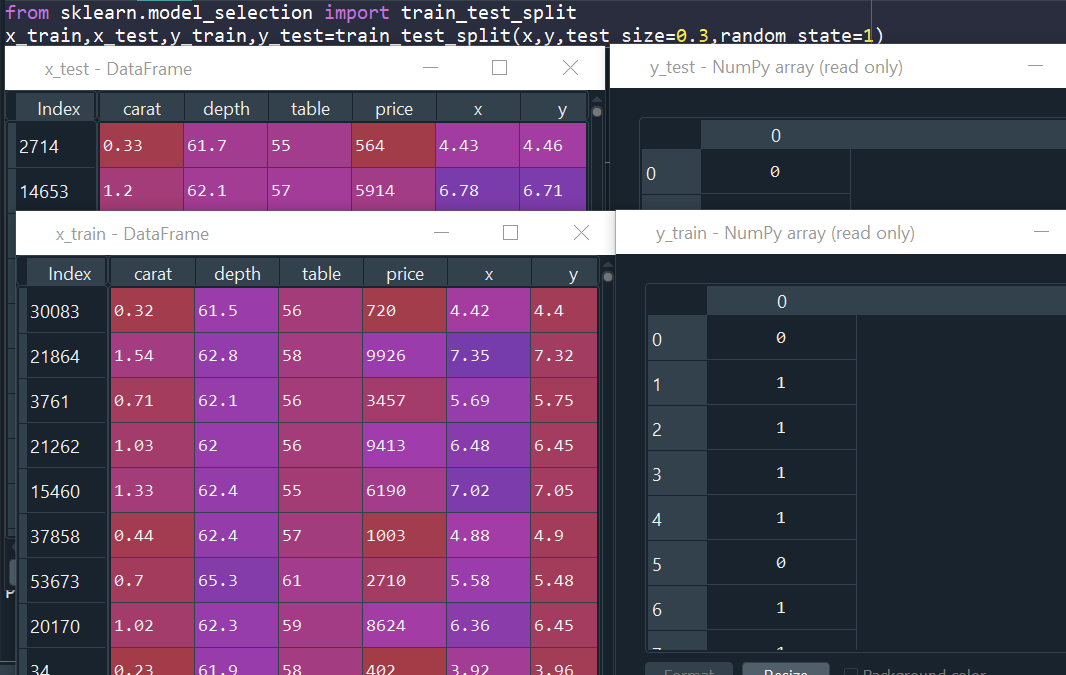
Kütüphaneleri import ederek veri setini yükledim ve boyut sütunumu oluşturdum. Kategori değişkeni olan boyutu ve gereksiz olan diğer alanları çıkararak x\_dataya bağlı olan y’yi tahmin edecek değişkenleri oluşturdum. Tahmin edilecek değişkenide y olarak tanımladım.

****

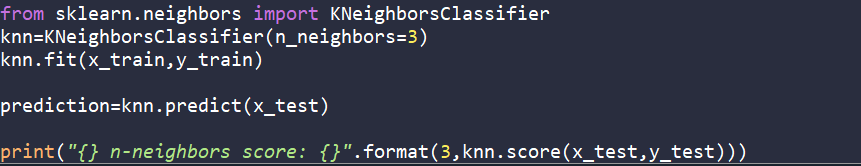
Normalizasyon yapıyoruz. Yani Bizim veri setimizden numpy kütüphanesindeki bir veri setinin içindeki en küçük ve en yüksek değerleri bulmamızı sağlıyor. Formülü de veriden minimum değeri çıkarıyoruz numpy bölü veri setindeki maksimum değerden minimum değeri çıkarınca bizim değerlerimizi 0 ile 1 arasında setlemiş oluyoruz**.**

****

Yukarıda oluşturduğumuz x ve y ile birlikte test size ile değerlerin %30’unu alıyoruz ve aynı yerden bölmesi için random state kullanıyoruz. Böylelikle x\_test, y\_tes, x\_train, y\_train oluşturuyoruz.

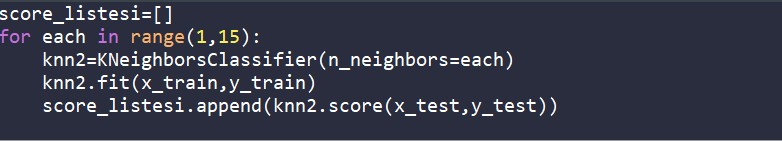


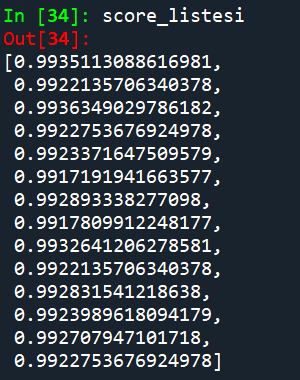
Knn diye değişken oluşturup en yakın üç değeri aldırıyoruz. Knn.fit diyerek test verimizi kullanarak modeli fit ediyoruz. Ardından prediction değişkeni ile tahmin etmeye çalışıyoruz ve skorunu hesaplıyoruz ve %99 oranında bu modelle tahminde bulunduğumuzu görüyoruz.



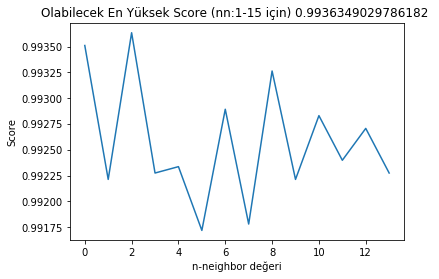


K değerini en ideal hangisi olarak seçmemiz gerektiğini bulmak için score listesi diye bir dizi oluşturarak 1 ile 15 arasında döngüye sokarak döngüdeki k sayıların knn değerini bulacağız ve bunu score listesine atıyoruz. Ne kadar 1 ‘e yakın olduğunda o kadar doğru tahminde bulunduğu için tüm değerler %99 değerinde doğruluk sağladığını görüyoruz ve tahminlerimizin doğruluğunun çok yüksek olduğunu görüyoruz.



****

Bunu aynı zamanda bir plot grafik olarak çizdirerek en yüksek skoru grafiğin üstüne yazdırıyoruz ve k değerinin üç seçtiğimizde en ideal seçenek olduğunu görüyoruz.



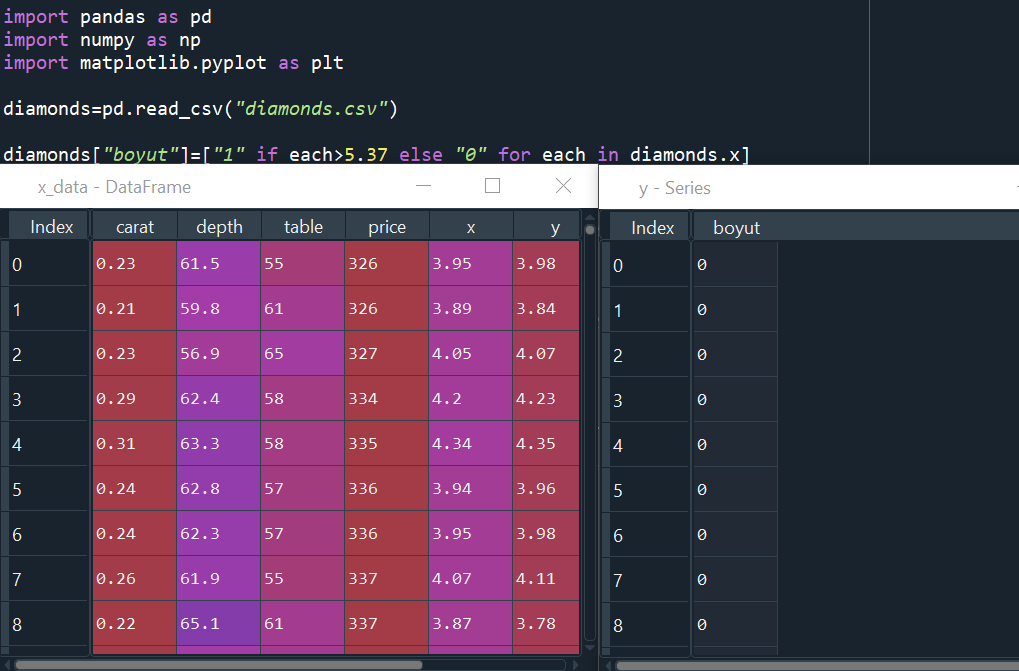
**SUPPORT VECTOR MACHİNE**

**Örnek Uygulamanın Kısa Açıklaması:**

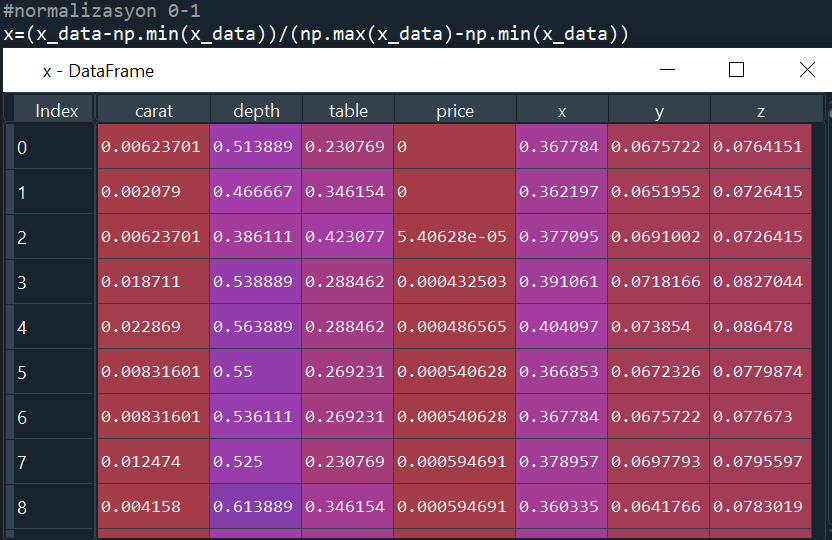
**Support Vector Machine** [Logistic Regression](https://medium.com/@ekrem.hatipoglu/machine-learning-classification-logistic-regression-part-8-b77d2a61aae1) ile benzer bir sınıflandırma algoritmasıdır. Her ikisi de iki sınıfı ayıran en iyi çizgiyi bulmaya çalışırlar. Algoritma çizilecek doğrunun iki sınıfında elemanlarına en uzak yerden geçecek şekilde ayarlanmasını sağlar. Hiçbir parametre almayan bir sınıflayıcıdır. SVM aynı zamanda doğrusal ve doğrusal olmayan verileri de sınıflandırabilir ancak genellikle verileri doğrusal olarak sınıflandırmaya çalışır. Sınıflandırma işlemlerimde şartı sağlaması için elmasın uzunluğunun orta değerini alarak değerden büyük olanlara 1, değerden küçük olanlara 0 yazacak şekilde boyut adında iki değişken içeren yeni bir sütun oluşturdum.

**Kodlar ve Yorumlama :**

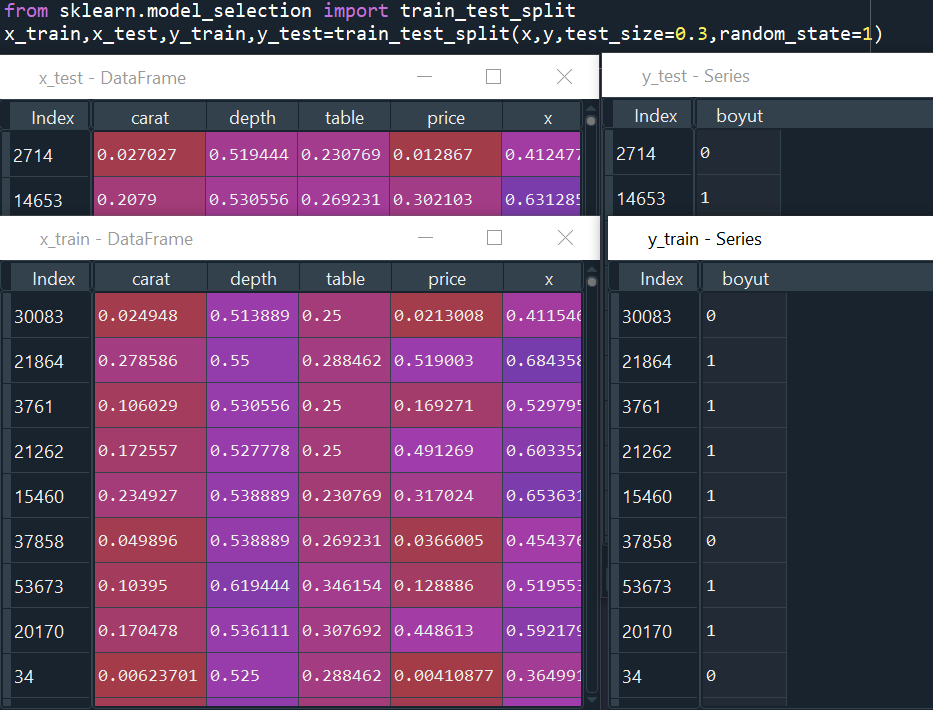
Kütüphaneleri import ederek veri setini yükleyerek boyut sütunumu ekledim. Kategori değişkeni olan boyutu ve gereksiz olan diğer alanları çıkararak x\_dataya bağlı olan y’yi tahmin edecek değişkenleri oluşturdum. Tahmin edilecek değişkenide y olarak tanımladım.

****

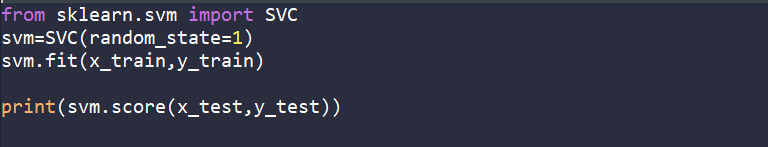
Normalizasyon yapıyoruz. Yani Bizim veri setimizden numpy kütüphanesindeki bir veri setinin içindeki en küçük ve en yüksek değerleri bulmamızı sağlıyor. Formülü de veriden minimum değeri çıkarıyoruz numpy bölü veri setindeki maksimum değerden minimum değeri çıkarınca bizim değerlerimizi 0 ile 1 arasında setlemiş oluyoruz**.**

****

Yukarıda oluşturduğumuz x ve y ile birlikte test size ile değerlerin %30’unu alıyoruz ve aynı yerden bölmesi için random state kullanıyoruz. Böylelikle x\_test, y\_tes, x\_train, y\_train oluşturuyoruz.



SVC için kütüphaneyi import ederek random\_state ile sabit nokta vererek çizgi çizdiriyoruz ardından fit ediyoruz. Ardından svm\_score ile test verisi modelimizi değerlendirmiş oluyoruz. Modelimiz %99 doğru tahminde bulunmaktadır. 1’e yakın olduğu için tahmin doğruluğu çok yüksektir.

****



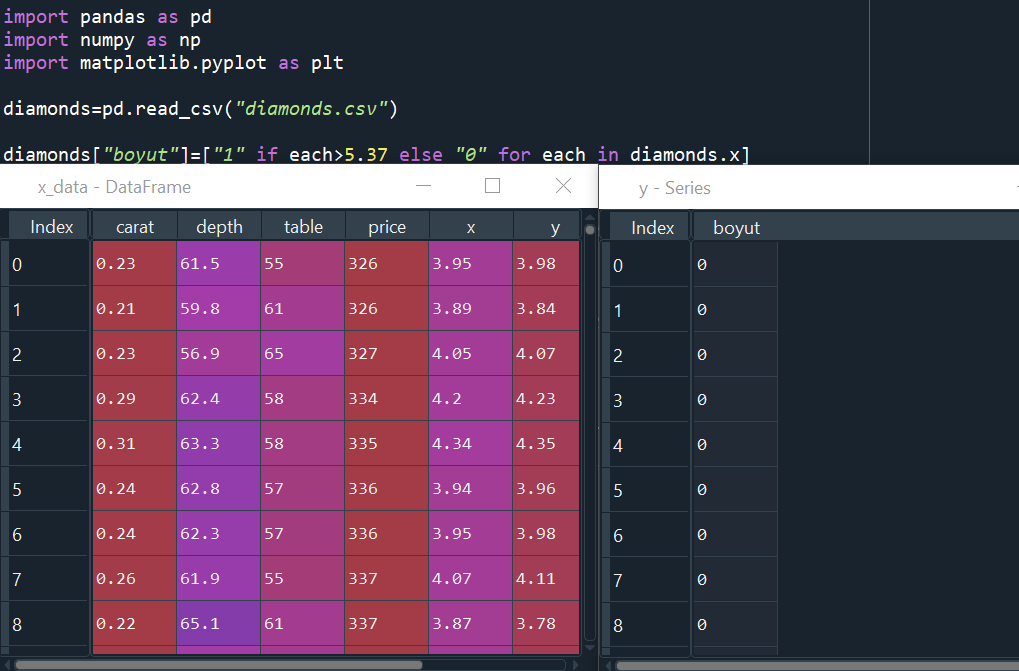
**NAİVE BAYES**

**Örnek Uygulamanın Kısa Açıklaması:**

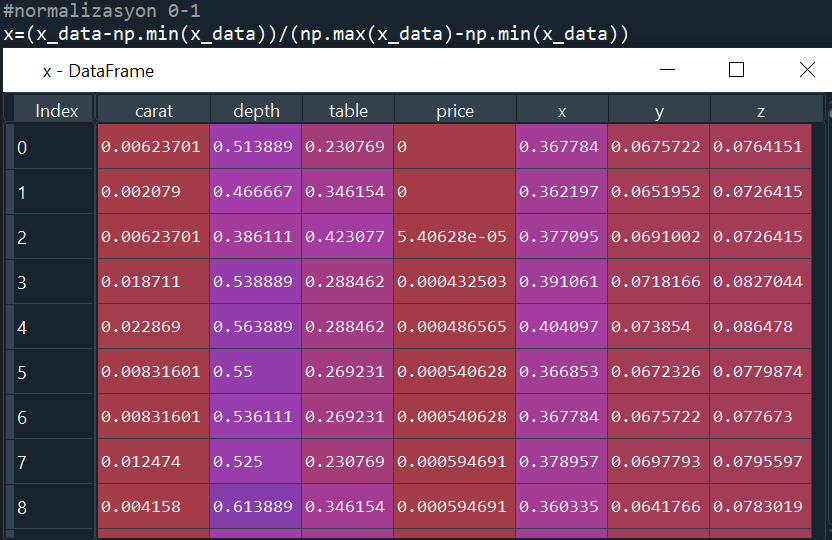
Naive Bayes sınıflandırıcısının temeli Bayes teoremine dayanır.  Tembel bir öğrenme algoritmasıdır aynı zamanda dengesiz veri kümelerinde de çalışabilir. Algoritmanın çalışma şekli bir eleman için her durumun olasılığını hesaplar ve olasılık değeri en yüksek olana göre sınıflandırır. Sınıflandırma işlemlerimde kolaylık sağlaması için elmasın uzunluğunun orta değerini alarak değerden büyük olanlara 1, değerden küçük olanlara 0 yazacak şekilde boyut adında iki değişken içeren yeni bir sütun oluşturdum.

**Kodlar ve Yorumlama :**

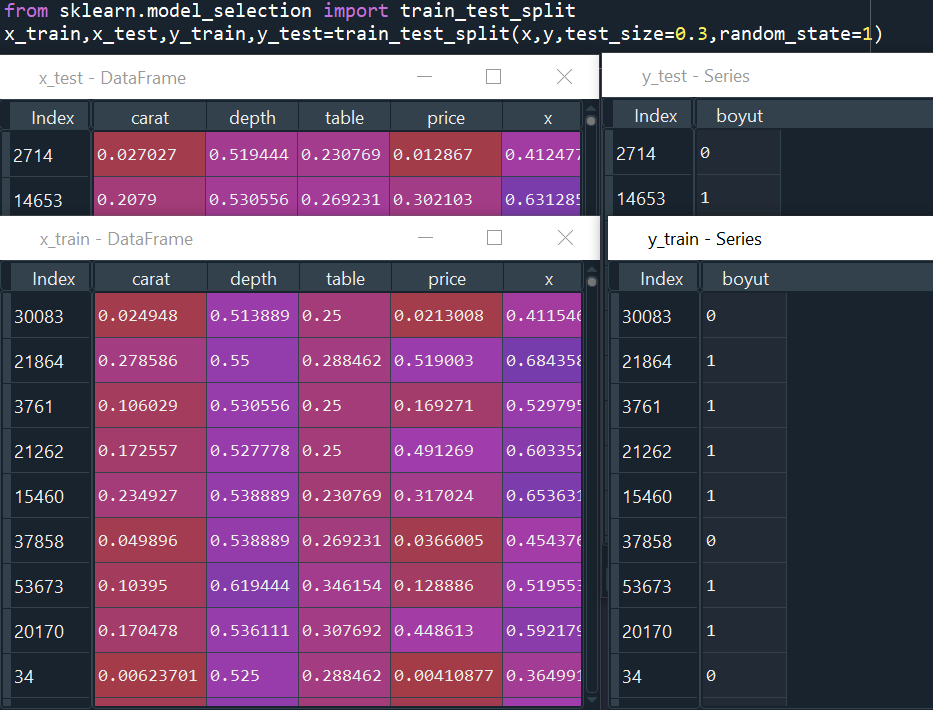
Kütüphaneleri import ederek veri setini yükleyerek boyut sütunumu ekledim. Kategori değişkeni olan boyutu ve gereksiz olan diğer alanları çıkararak x\_dataya bağlı olan y’yi tahmin edecek değişkenleri oluşturdum. Tahmin edilecek değişkenide y olarak tanımladım.

****

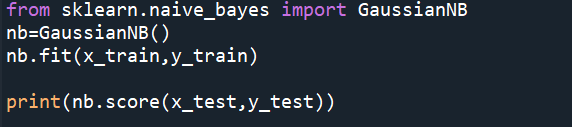
Normalizasyon yapıyoruz. Yani Bizim veri setimizden numpy kütüphanesindeki bir veri setinin içindeki en küçük ve en yüksek değerleri bulmamızı sağlıyor. Formülü de veriden minimum değeri çıkarıyoruz numpy bölü veri setindeki maksimum değerden minimum değeri çıkarınca bizim değerlerimizi 0 ile 1 arasında setlemiş oluyoruz**.**

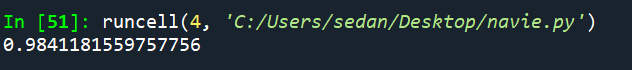
****

Yukarıda oluşturduğumuz x ve y ile birlikte test size ile değerlerin %30’unu alıyoruz ve aynı yerden bölmesi için random state kullanıyoruz. Böylelikle x\_test, y\_tes, x\_train, y\_train oluşturuyoruz.



Modelimiz %98 doğru tahminde bulunmaktadır. 1’e çok yakın olduğu için tahmin doğruluğu çok yüksektir.



****

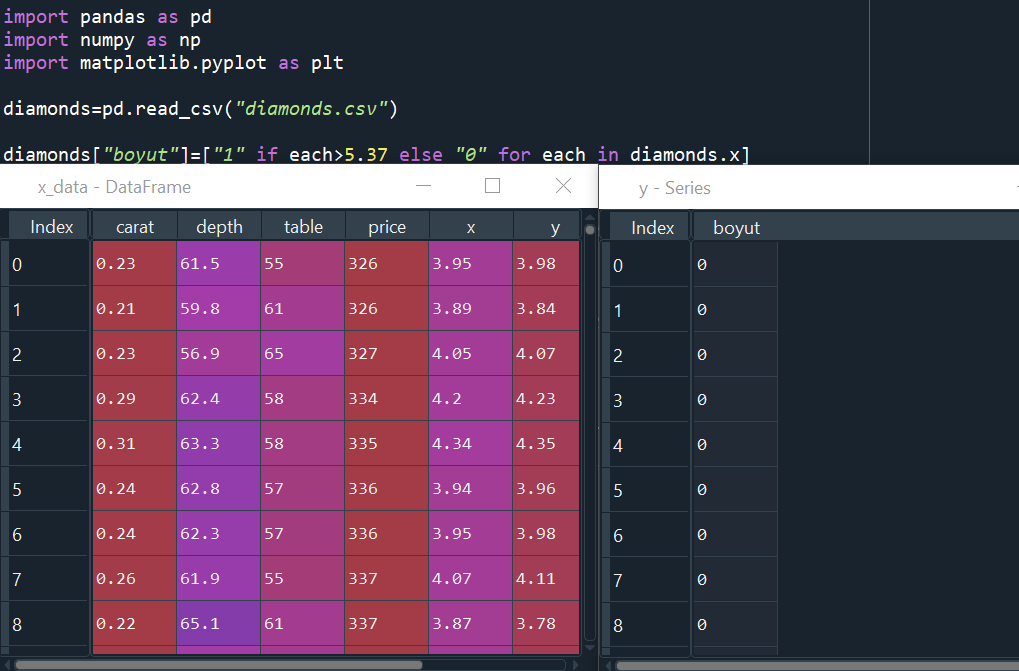
**RANDOM FOREST CLASSİFİCATİON**

**Örnek Uygulamanın Kısa Açıklaması:**

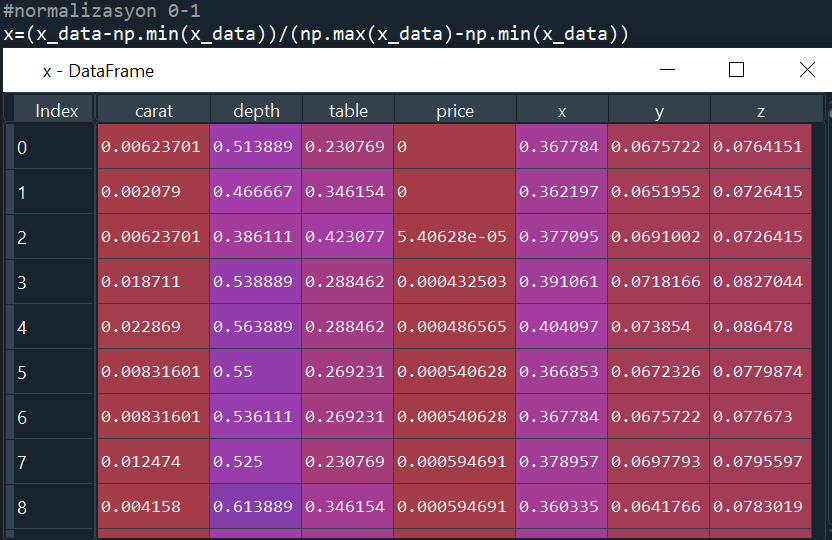
Random Forest algoritması Karar ağacı gibi hem sınıflandırma hem de regresyon problemlerinde kullanılabilir. Regresyon için çalışma mantığı veri seti küçük parçalara ayırarak karar ağaçları oluşturur. Daha sonra bir tahmin yapılacağı zaman bu karar ağaçlarındaki tahminlerin ortalaması alınır. Sınıflandırma işlemlerimde şartı sağlaması için elmasın uzunluğunun orta değerini alarak değerden büyük olanlara 1, değerden küçük olanlara 0 yazacak şekilde boyut adında iki değişken içeren yeni bir sütun oluşturdum.

**Kodlar ve Yorumlama :**

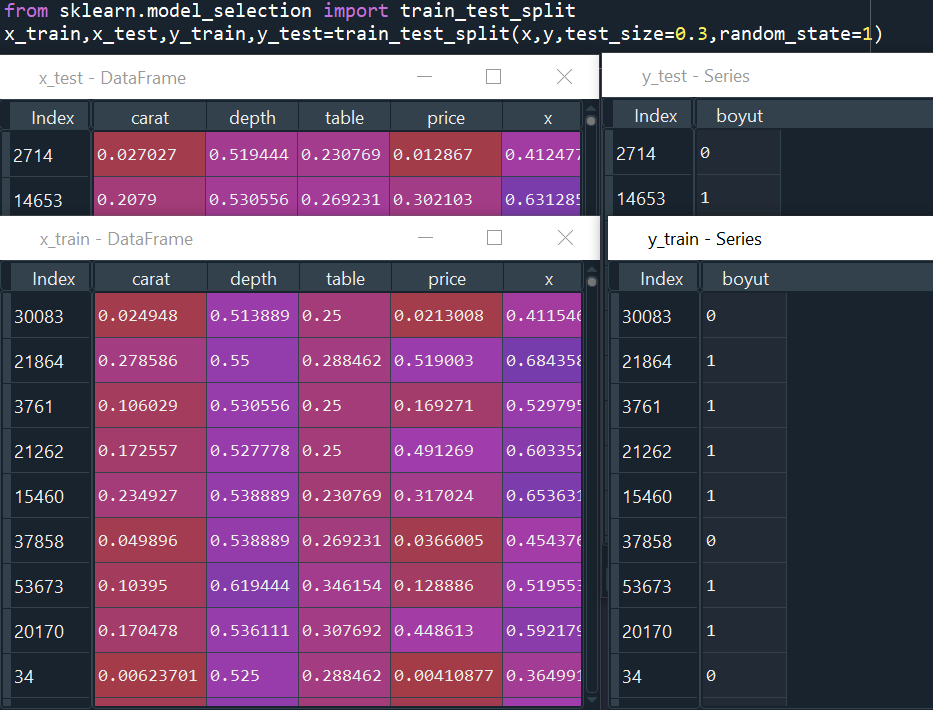
Kütüphaneleri import ederek veri setini yükleyerek boyut sütunumu ekledim. Kategori değişkeni olan boyutu ve gereksiz olan diğer alanları çıkararak x\_dataya bağlı olan y’yi tahmin edecek değişkenleri oluşturdum. Tahmin edilecek değişkeni de y olarak tanımladım.

****

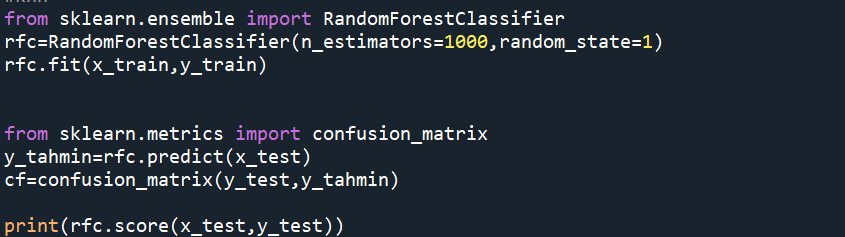
Normalizasyon yapıyoruz. Yani Bizim veri setimizden numpy kütüphanesindeki bir veri setinin içindeki en küçük ve en yüksek değerleri bulmamızı sağlıyor. Formülü de veriden minimum değeri çıkarıyoruz numpy bölü veri setindeki maksimum değerden minimum değeri çıkarınca bizim değerlerimizi 0 ile 1 arasında setlemiş oluyoruz**.**

****

Yukarıda oluşturduğumuz x ve y ile birlikte test size ile değerlerin %30’unu alıyoruz ve aynı yerden bölmesi için random state kullanıyoruz. Böylelikle x\_test, y\_tes, x\_train, y\_train oluşturuyoruz.



Modelimiz 1 değerinde doğru tahminde bulunmaktadır. 1 demek tamamen doğru demek olduğu için doğru tahminde bulunmaktadır.

****

****

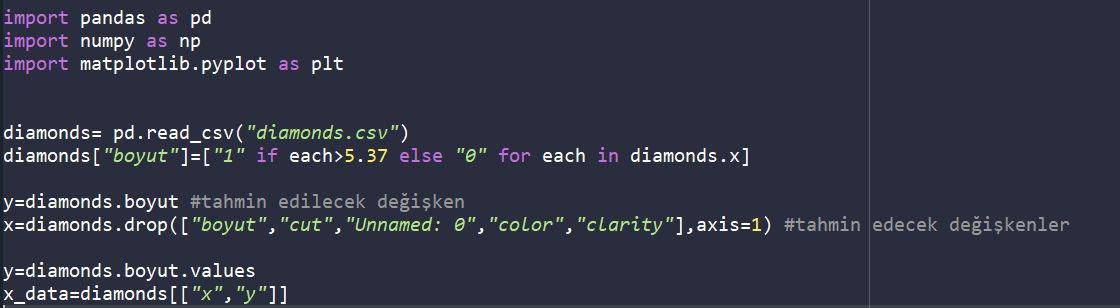
**DECİSİON TREE CLASSİFİCATİON**

**Örnek Uygulamanın Kısa Açıklaması:**

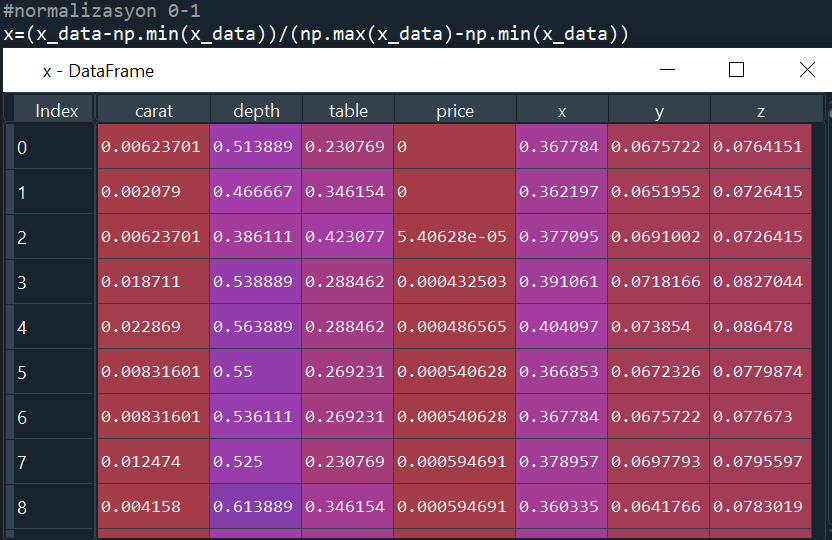
Decision Tree ( Karar ağacı ) yöntemi hem Sınıflandırma hem de regresyon problemlerinde kullanılan makine öğrenmesinin en popüler algoritmalarından biridir. Aynı zamanda veri madenciliği alanında da sıkça kullanılır. Karar ağaçları genellikle insan seviyesinde düşünülebilecek düzeydedir, böylece verileri anlamak ve bazı iyi yorumlar yapmak ve görselleştirmek çok basittir. Sınıflandırma işlemlerimde şartı sağlaması için elmasın uzunluğunun orta değerini alarak değerden büyük olanlara 1, değerden küçük olanlara 0 yazacak şekilde boyut adında iki değişken içeren yeni bir sütun oluşturdum.

**Kodlar ve Yorumlama :**

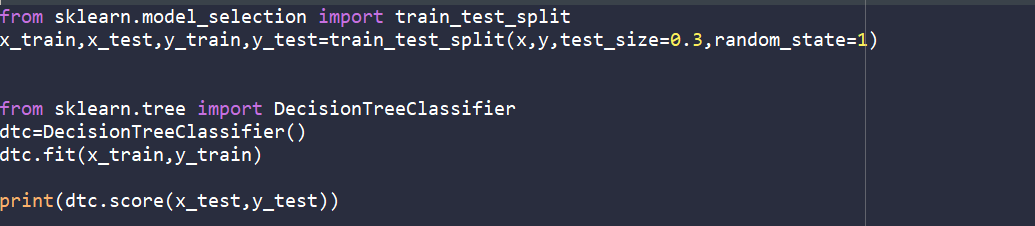
Kütüphaneleri import ederek veri setini okutup Kategori değişkeni olan boyutu ve gereksiz olan diğer alanları çıkararak x\_dataya bağlı olan y’yi tahmin edecek değişkenleri oluşturdum. Tahmin edilecek değişkeni de y olarak tanımladım. Kullanacağım sayısal değişkenleride x\_data olarak tanımladım.

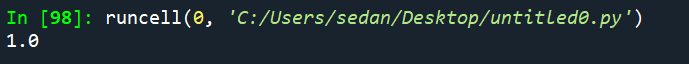
****

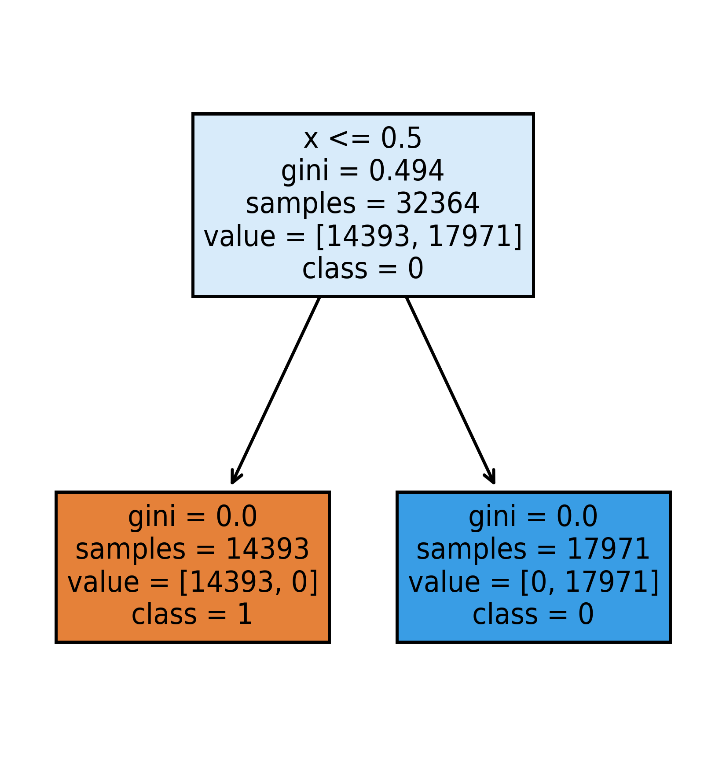
Normalizasyon yapıyoruz. Yani Bizim veri setimizden numpy kütüphanesindeki bir veri setinin içindeki en küçük ve en yüksek değerleri bulmamızı sağlıyor. Formülü de veriden minimum değeri çıkarıyoruz numpy bölü veri setindeki maksimum değerden minimum değeri çıkarınca bizim değerlerimizi 0 ile 1 arasında setlemiş oluyoruz**.**

****

Modelimiz 1 değerinde doğru tahminde bulunmaktadır. 1 demek tamamen doğru demek olduğu için doğru tahminde bulunmaktadır.

****

****

****

**SONUÇ**

Sınıflandırma algoritmalarında Logistic Regression yaptığımız zaman matrix ile tahmin ettiğimiz değerlerin ne kadar doğru ne kadar yanlış olduğunu görüyoruz. 13390 değer doğru, 95 değer yanlış tahminde bulunmuş. Doğruluk değeri ise %99 dur. Bu da neredeyse 1 olduğu için tahmin sonuçlarımızın güvenirliliğinin çok olduğunu ve yanılma payının çok düşük olduğunu söyleyebiliriz. K -Nearest yaptığımız zaman score listesi oluşturarak K değerinin en ideal sayısını buluyoruz. K değerinin en ideal 3 olduğunu görüyoruz. K değerini üç aldığımız zaman doğruluk değerinin%99 olduğunu görüyoruz. Aynı şekilde bu da neredeyse 1 olduğu için tahminlerin güvenilirliğinin çok yüksek olduğunu ve yanılma payının çok düşük olduğunu söyleyebiliriz. Support Vector Machine kullandığımız zaman yine doğruluk değeri %99 olarak çıkıyor. Aynı şekilde bu da neredeyse bir olduğu için güvenilirliğinin çok yüksek olduğunu ve yanılma payının çok düşük olduğunu söyleyebiliriz. Naive kullandığımız zaman tahminlerimizin doğruluk değerinin %98 olduğunu görüyoruz. Diğer üç algoritmaya göre %1 daha az doğru sonuç vermektedir. Genel olarak Naive sınıflandırmasına baktığımız zamanda yine 1’ e çok yakın olduğu için güvenilirliğinin çok iyi olduğunu ve yanılma payının çok düşük olduğunu söyleyebiliriz. Random Forest Classification kullandığımız zaman doğruluk değerini 1 olduğunu görüyoruz. Yani tüm tahminlerin gerçeklerle aynı olduğunu anlıyoruz. Aynı şekilde Desicion Tree Classification kullandığımız zamanda doğruluk değerinin 1 olduğunu ve onun da tahminlerle gerçeklerin uyuştuğunu görüyoruz. Yani genel olarak tüm sınıflandırma algoritmalarına baktığımız zaman hepsi 1’e çok yakın sonuçlar vermektedir.Random Forest Classification ve Decision Tree Classification tam 1 değerini göstermektedir. Yani ikisi %100 doğru sonuç göstermektedir ve onları kullanmamız daha doğru olur. Diğer algoritmalarda hep 98 ve üzeri olduğu için onların da yanılma payı çok düşüktür.

1. Veri setinizi kullanarak örnek bir kümeleme analizi (Kmeans ve Hierarchical tekniklerini kullanarak) gerçekleştiriniz. Analiz sonuçlarını yorumlayınız.

**KÜMELEME ANALİZİ**

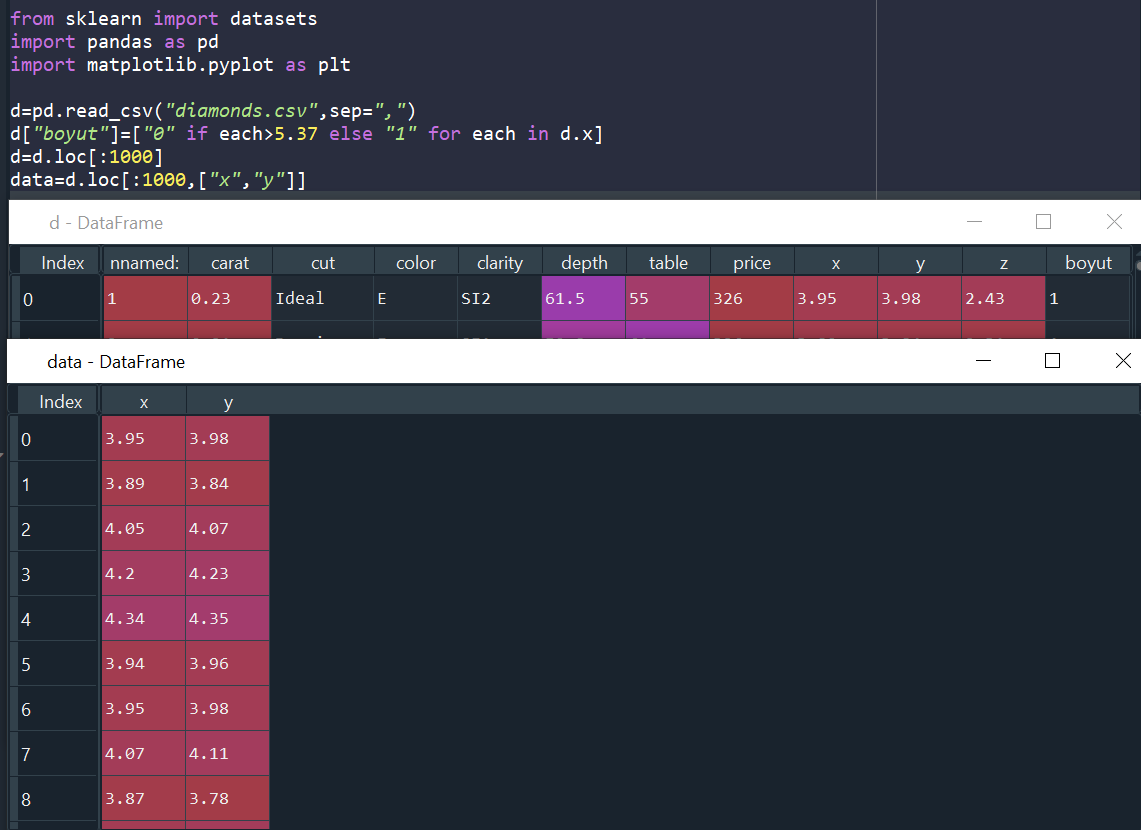
**K MEANS**

**Örnek Uygulamanın Kısa Açıklaması:**

K-Means algoritması bir unsupervised learning ve kümeleme algoritmasıdır. K-Means’ teki K değeri küme sayısını belirler ve bu değeri parametre olarak alması gerekir. Bu durum aslında bir dezavantajdır. K değerini kendi hepsalayan X-Means adında başka bir algoritmada vardır. Algoritmanın basit bir çalışma şekli vardır. Kümeleme işlemlerimde kolaylık sağlaması için elmasın uzunluğunun orta değerini alarak değerden büyük olanlara 1, değerden küçük olanlara 0 yazacak şekilde boyut adında iki değişken içeren yeni bir sütun oluşturdum.

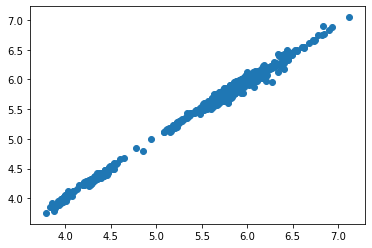
**Kodlar ve Yorumlama:**

Kütüphaneleri import ederek veri setini yükleyerek boyut sütunumu ekledim. Veri setimde veriler çok olduğu için hata veriyor o yüzden veri setimi 1000 satırla sınırlandırıyorum.

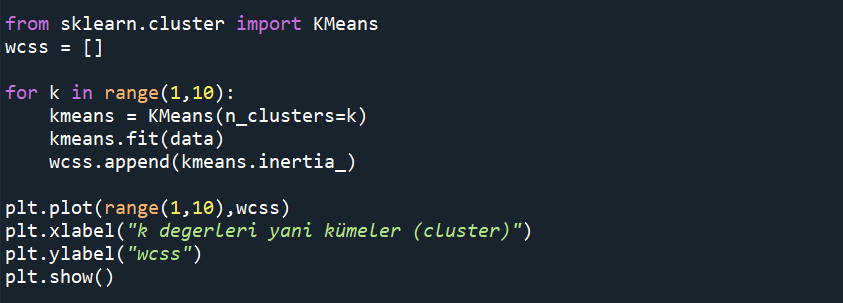
****

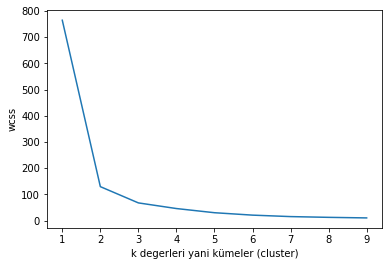
Plot grafiğini çizdiriyorum. Anlamlı olarak iki kümeye ayrıldıklarını görüyorum. O yüzden değerimi 2 olarak alıyorum.

****

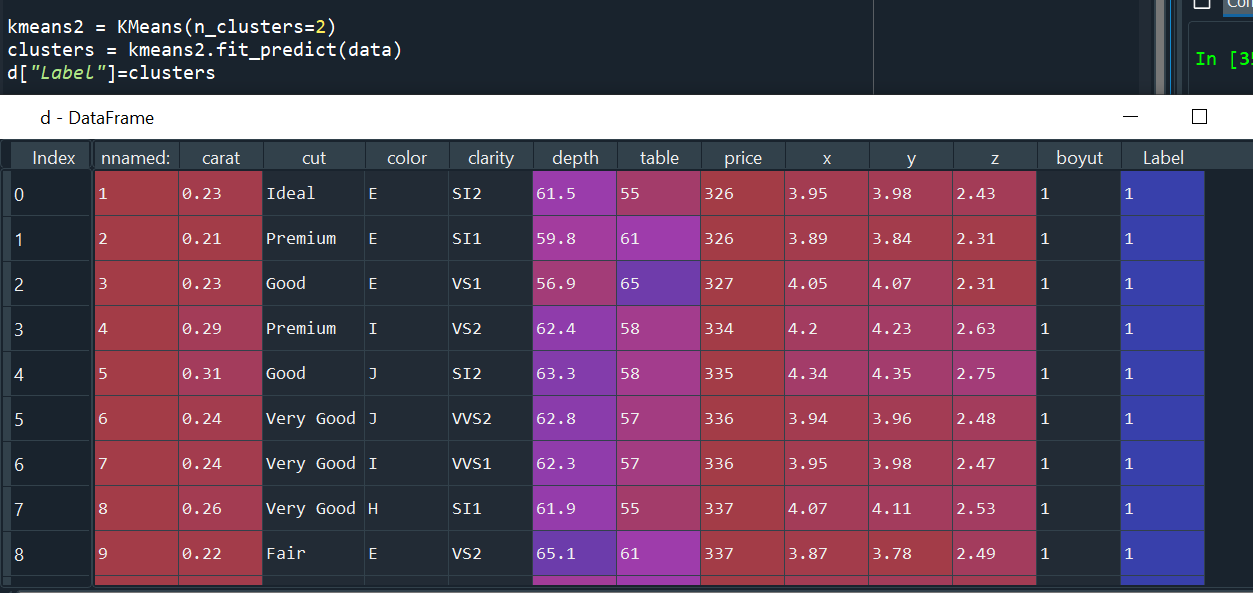
****

Hangi değişkeni alırsak daha doğrusunu ortaya koyar diye bakıyoruz. Bunun için wcss yani noktaların n ’e kadar gittikçe her birinin merkeze uzaklığının karelerini ayrı ayrı alarak toplar. Belli bir noktada keskin kırım olur ve düzleşme başlar. Kırımları dikkate alarak oluşacak küme sayısına karar veririz. Döngü oluştururuz ve fit ile n değeri ve dataya yani iki değişkenimiz olan x ve y ‘ye göre modeli oluştururuz. Diziye append ile değer ataması yaparız. İnertia yani wcss ‘nin kütüphane anlamı olan değişken ile tanımlarız. 10 değerden oluşan diziyi grafik haline çeviririz. Oluşan grafikte keskin kırım var sonra yavaş yavaş düzleşiyor. Tam kırılma kısmını göze alırsak n ’i 2 olarak alırız.

****



Son olarak değerimizi iki alarak label olarak veri setimizin yanına yeni bir alan oluşturuyoruz. Boyut değerimiz ile labelin aynı değerlerini verdiğini görüyoruz.



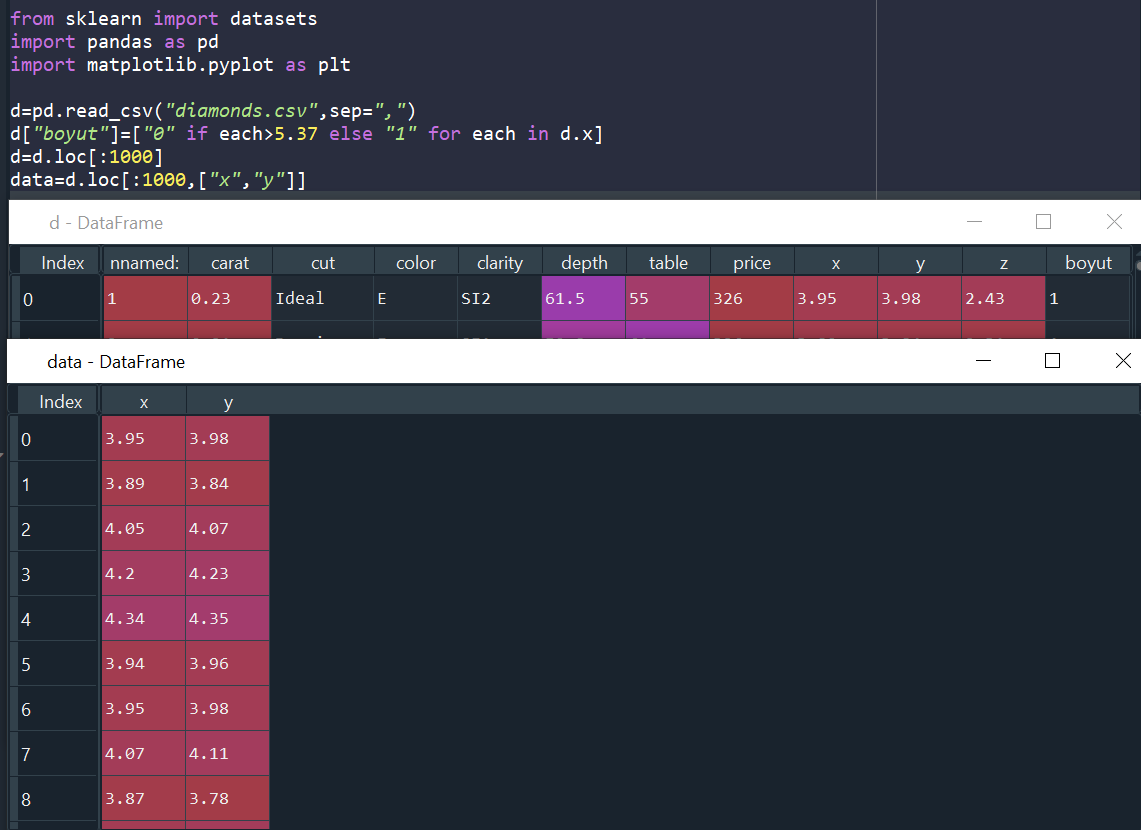
**HİERARCHİCAL**

**Örnek Uygulamanın Kısa Açıklaması:**

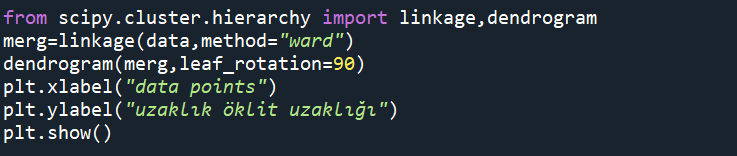
Hiyerarşik kümeleme adından da anlaşılacağı üzere bir kümeleme algoritmasıdır. Parçadan bütüne ve bütünden parçaya olarak iki farklı varyasyonu vardır. Agglomerative varyasyonunu anlatmaya çalışırsak ilk önce tüm veriler bir küme haline getirilir yani N tane eleman varsa N tane küme oluşur. Daha sonra birbirine mesafe olarak yakın olan kümeler birleşerek yeni bir küme oluşturur. Bu durum sistem kararlı oluncaya kadar devam eder. Kümeleme işlemlerimde şartı sağlaması için elmasın uzunluğunun orta değerini alarak değerden büyük olanlara 1, değerden küçük olanlara 0 yazacak şekilde boyut adında iki değişken içeren yeni bir sütun oluşturdum.

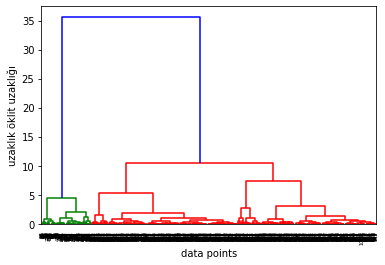
**Kodlama ve Yorumlama :**

Kütüphaneleri import ederek veri setini yükleyerek boyut sütunumu oluşturdum. Veri setimde veriler çok olduğu için hata veriyor o yüzden veri setimi 1000 satırla sınırlandırıyorum.

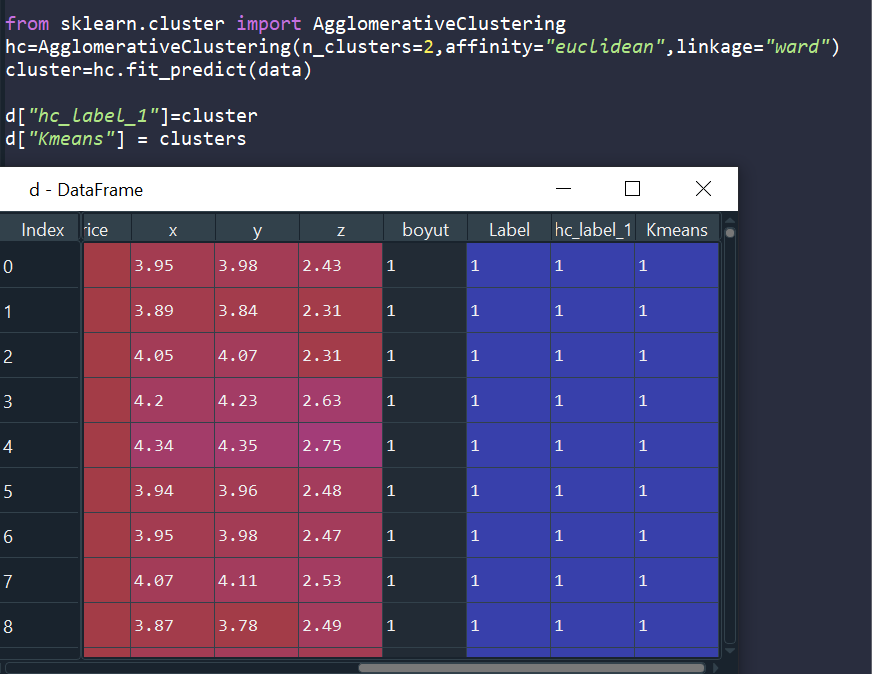
****

Dendogramı merg diye değişken oluşturuplinkage fonksiyou oluşturuyoruz. Datadaki iki değişkeni alıp ward metodunu kullanarak mergi oluşturuyoruz. Dendograma mergi veririz ve leaf\_rotation ile çizgileri tam 90 derece çizmesini sağlarız. Dendogramdan en uzun yer alınır ve yatay kesmeyecek şekilde seçeriz. O çizgiyi kaç noktadan kesiyor ise ona göre küme alırız. Görünüşe göre en üsten çizgi çekince iki nokta olur yani iki küme oluştururuz.

****

****

Kümeyi iki olarak tanımlayarak veri setimize ekleriz. İki kümeleme analizini karşılaştırdığımızda tamamen aynı tahminlerde olduğunu görürüz.

****

1. Veri madenciliği dersinde öğrendiğiniz bilgilerden hareketle gerçek bir dünya problemi belirleyip bu problemi hangi veri madenciliği tekniği veya teknikleri ile çözebileceğinizi gerekçeleri ile tartışınız. Bu soruda sizden gerçek bir analiz yapmanız beklenmemektedir. Sadece çevrenizde veya dünyada var olan bir probleme odaklanmanız, bu problemin çözümünde veri madenciliğini nasıl kullanabileceğinizi açıklamanız beklenmektedir. Hangi veri madenciliği tekniğini kullanacağınızı gerekçelendirerek açıklamalısınız.

**Problem**

Böbrek yetmezliği kronik böbrek hastalığının son aşamasıdır. Son dönem böbrek yetmezliği olarak da adlandırılır. Böbreklerin işlevlerini diyaliz veya böbrek nakli olmadan hayatta kalmayı mümkün kılacak kadar sürdürememesi durumudur. Örneğin küçük bir ameliyatta bile alınan ağrı kesici ilaçlar vücuttan çıkamayınca beyne pıhtı ve kanama olarak sonuçlar doğurabilir. Böbrek yetmezliği tehlikesinde olan insanların ne kadar erken tanı konulursa o kadar daha ilerlemeden belki de hiç olmamasını sağlatarak hayatının ilerleyişini değiştirebiliriz.

**Çözümü için veri madenciliği nasıl kullanılabilir? Neden?**

Veri madenciliği büyük ölçekli işlenmemiş ham verileri kullanarak tahminde bulunmamızı ve bilgiye ulaşmamızı sağlar. Veri madenciliği sayesinde gelecek ile ilgili bağıntıları bilgisayar programlarında matematiksel ve istatiksel alt yapıya sahip olan modellemeler kullanılır. Veri madenciliğinin birçok yöntemi vardır. Bu yöntemlerin tahmin doğruluğu ve güvenilirliği bizim veri setimize göre değişiklik gösterir. Bulduğumuz doğruluk değerlerinde bire en yakın olan en doğru sonucu verdiği için o yöntemi seçeriz. Yani istenen sonucu başarılı bir şekilde sağlamak amacıyla farklı amaçlara sahip birçok yöntem birlikte uygulanabilir.

Böbrek yetmezliği olarak seçtiğimiz problemi araştırarak yani nedenleri ve etken olan değerleri bulup ve bunu bir excele tanımlarız. Böbrek hastalığının öncelikle tıp terimi dışında olan belirtileri ve nedenlerini ele alabiliriz. Örneğin böbrek yetmezliğinin en önemli belirtilerinden düşük idrar üretimini, ellerde ve bacaklarda şişkinliği, nefes darlığını gibi belirtileri veri setimize belirti varsa 1 yoksa 0 olarak veya belirti varsa E yoksa H olarak tanımlayabiliriz. Aynı şekilde böbrek yetmezliğinin en önemli nedenlerinden olan yüksek tansiyon yani kişinin tansiyon değeri, diyabet hastalığı olup olmadığı, 60 yaş üzerinde olmak, ailede genetiklik durumu gibi nedenleri de veri setimize doğruluk derecesini arttırması için ekleyebiliriz. Neden ve belirtilerinden sonra kişinin idrar örneği alınarak bakılan protein veya şeker gibi anormallikleri, idrar sedimantasyon testi sonucunu, alyuvar ve akyuvarların miktarını, bakteri varlığını, üresel atık parçacıkları ve vücuttaki damarların sertliği, kan basıncı gibi böbrek yetmezliğiyle ilgili olan tıptaki gerekli verilerin değerlerini de sayısal olarak veri setimize kaydederiz.

Veri setinde böbrek hastası olan ve olmayan olarak bir alanda eklenir. Artık böbrek hastası olup olmadığına dair test edebileceğimiz bilgiler vardır. Bunlardan yola çıkarak ortaya çıkan veri seti bize çok karmaşık ve anlamsız gelse bile veri madenciliğinde matematiksel ve istatiksel alt yapıya sahip olan modellemeler ile bu veri setini anlamlı hale getirebiliriz. Kronik böbrek hastalığı erken dönemde saptanır ve uygun şekilde tedavi edilirse önlenebilir, ilerlemesi engellenebilir veya yavaşlatılabilir ve yol açtığı yüksek ölüm riski azaltılabilir. Tedaviye ne kadar erken başlanırsa yararı o kadar fazla olmaktadır. Kişinin diyaliz veya böbrek nakli evresine gelmemesi için bu şekilde veri seti oluşturup tahminle daha erken bir sonuca varabiliriz. Böbrek yetmezliğinin bir de evreleri vardır. Bunlar çok hafif olarak 1. Evre ile tam böbrek yetmezliği olarak 5. evre arasında değişmektedir. Aşamalar ilerledikçe belirtiler ve komplikasyonlar da artar. Veri setimizde bulunan değerlerin çoğuna saklama Aynı zamanda kişinin böbrek yetmezliği tehlikesi olduğuna baktığımızda eğer böbrek yetmezliği tehlikesiyle karşı karşıya ise kişinin değerlerinin çokluğuna ve yüksekliğine de bakarak hastalık evresine de bakabiliriz. Böylelikle nelerin gerekli olduğunu, böbreklerin ne durumda olduğunu ve hangi tedaviyi uygulayabileceğimizi daha kısa sürede tahmin ederek ona göre hazırlıklı olmuş oluruz.

Kişinin böbrek yetmezliğinin olup olmadığını anlamak için veri madenciliğinin sınıflandırma algoritmasını kullanırım. Sınıflandırma kavramı, bir veri kümesi üzerinde tanımlı olan çeşitli sınıflar arasında veriyi dağıtmaktır. Sınıflandırma algoritmaları, verilen eğitim kümesinden bu dağılım şeklini öğrenirler ve daha sonra sınıfının belirli olmadığı test verileri geldiğinde doğru şekilde sınıflandırmaya çalışırlar. Sınıfı bilinmeyen kayıtların kanser olup olmadığını gösteren sütun hariç diğerlerini daha önceden tanımlanmış kategorilerden birine atanmasına sınıflandırma adı verilir. Sınıflandırma işlemi genellikle bir sınıflandırma modeline uygun olarak yerine getirilir. Tahmin değişkenleri yardımıyla, hedef değişkeninin hesap edildiği bir fonksiyon olarak görülebilir. Tahmin değişkenleri ile hedef değişkenleri arasındaki ilişki veri seti üzerinde yapılan deneylerle bulunur. Bir sınıflandırma işleminde kullanılan veri temel olarak ikiye ayrılır. Verilerin ilk bölümü eğitim verisi diğeri ise test verisidir. Sınıflandırma modeli eğitim verisi ile eğitilir test verisi ile de değerlendirilir. Test verilerinde elde edilen sınıflandırma başarısı modelin doğruluğunu verir. Değer 1 ‘e çok yakınsa güvenilirliği sağlamış olur ve tahminlerimiz doğruya çok yakın olur. Oran olarak analiz edilen verilerin %70 civarındaki oranı eğitim amacıyla %30’luk kısmı ise test amacıyla kullanılırız. Her yeni girilen değerleri oluşturduğumuz eğitim setine göre bakarak tahminde bulunuruz.

Kişinin böbrek yetmezliğinin evresini öğrenmek için de veri madenciliğinin kümeleme yöntemini kullanırım. Kümeleme analizi, sınıflama analizinden farklı olarak denetimsizdir. Öngörülecek alanların belirlenmesini ve birbirine benzeyen verilerin alt kümelere ayrılmasını hedefler. Kümeleme analizinin hedefi, veri setinde doğal olarak meydana gelen altsınıfları bulmaktır. Kendi içinde çok çeşitli açılardan benzer özellikler, benzer tutum ve değer gösteren grupların benzer tepkiler göstermesi beklenmektedir. Yani çok yüksek değerler bir araya, orta ve düşük değerler bir araya gelerek kümelenirler.

Sınıflandırma yöntemi olarak karar ağacı seçilmiştir;

Veri setimden evreleri gösteren kategoriyi baz aldım. Bir karar ağacının amacı her düğümün sonunda en uygun seçimi yapması olduğundan, tam da bunu yapabilen bir algoritmaya ihtiyaç var. Her adımda en uygun kararı ve özyinelemeli anlamı daha büyük soruyu daha küçük sorulara böler ve aynı şekilde çözer. Her düğüme ayrılma kararı, çağrılan metriğe göre verilir. Ele aldığım veri setinde birden fazla feature vardır. Bu özellikler belirli kategorilerle örneğin; 1.Evre, 2.Evre, 3.Evre, 4.Evre ve 5.Evre olmak üzere sınıflandırılmıştır. Seçtiğim bu sınıflandırma modeli benim özelliklerimi ele alarak evrelere göre bir karar modeli oluşturacaktır. Oluşturulan karar modelinde hangi şartlar sağlandığında kaçıncı evre olduğu gösterilecektir. Bu bağlamda veri setine eklenecek olan herhangi veri karar ağacına bakılarak hangi evreye dâhil olduğu gözlemlenebilecektir.

Kümeleme yöntemlerinden hiyerarşik model seçilmiştir;

Veri setimden evreleri gösteren kategoriyi baz aldım. Veri setim birbirini tekrar eden değerler içermediği için karmaşık bir yapıya sahip, aykırı veri bulunmayan ve aynı zamanda orta hacimli bir veri setidir. Karmaşık yapıya sahip olduğu için oluşturulacak kümeler kestirilmemektedir. Bu özellikte veri setine uygun olan hiyerarşik yöntem seçilmiştir. Veri seti çok büyük hacimli olduğu durumda hiyerarşik yöntem yerine k mean yöntemini tercih edecektim. Hiyerarşik yöntem, büyük hacimli veriler üzerinde zor çalışma ve aykırı verilerden etkilenme gibi dezavantajları bulunmaktadır. Aynı zamanda küme sayısı kestirilemeyen veri setlerinde hiyerarşik kümeleme yöntem ideal bir yöntemdir.

Sınıflandırma yöntemi olarak Navie Bayes seçilmiştir;

Veri setimizden hasta olup olmadığını gösteren kategoriyi baza alarak veri madenciliği sınıflandırma modellerinden Naive Bayes metodunu seçtim çünkü Naive Bayes sınıflandırıcısının temeli tembel bir öğrenme algoritmasıdır aynı zamanda dengesiz veri kümelerinde de çalışabilir. Algoritmanın çalışma şekli bir eleman için her durumun olasılığını hesaplar ve olasılık değeri en yüksek olana göre sınıflandırır. Az bir eğitim verisiyle çok başarılı işler çıkartabilir. Öğrettiğimiz böbrek yetmezliği verileri üzerinde yapılan olasılık işlemleri ile, sisteme sunulan yeni test verileri, daha önce elde edilmiş olasılık değerlerine göre işletilir ve verilen test verisinin hangi kategoride olduğu tespit edilmeye çalışılır. Elbette öğretilmiş veri sayısı ne kadar çok ise, test verisinin gerçek kategorisini tespit etmek o kadar kesin olabilmektedir. Bizim de eğitim için veri setimizde birçok sütunumuz vardır. Naive Bayes sınıflandırma yönteminin birçok kullanım alanı bulunabilir fakat, burada neyin sınıflandırıldığından çok nasıl sınıflandırıldığı önemli. Yani öğretilecek veriler ikili veya metin halinde veriler olabilir, burada veri tipinden ve ne olduğundan ziyade, bu veriler arasında nasıl bir oransal ilişki kurduğumuz önem kazanıyor.