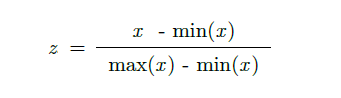
**Verilerin Ölçeklenmesi (Standart/Normalleştirilmesi)**

Gelen bir soru üzerine, farklı istatistiksel özelliklere sahip verilerin nasıl ön işlemeye tabi tutulacağını anlatmaya çalışacağız (bu içerik daha sonra video haline de getirilecektir, sorunun cevaplanması için önceden yazılı içerik halinde sunulmuştur).

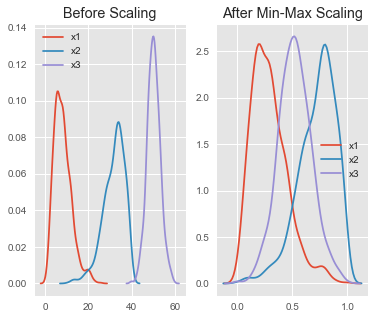
Python ile makine öğrenmesi dünyasında (scikit-learn) temel olarak 4 farklı sınıftan bu işlemler yapılabilir.

MinMaxScaler

Bu sınıfın amacı, verilerin 0 - 1 aralığına indirilmesi ve bu sırada verilerin birbirine göre görece olarak mesafe oranlarını korumaktır. Örneğin 100 - 200 aralığında dağılan verilerden iki tane sayıyı ele alalım. Bu sayılar arasındaki mesafe 30 ise, 0-1 aralığına inince 0.3 olacaktır (100 - 200 aralığı 100 birimlik min - Max aralığına sahiptir ve yeni min 0 ve yeni maks 1 olacak şekilde yeniden ölçeklendirilirler).



Yukarıdaki formülde de gösterildiği gibi, herhangi bir verinin ölçeklenmiş hali, o verinin bulunduğu serideki minimum değere uzaklığının, o serideki maksimum ve minimum değerler arasındaki farka bölümüdür.



Yukarıda soldaki dağılık 3 farklı veri kolonu için farklı aralıklarda dağılan verileri ifade etmektedir (x eksenindeki değerler ilk kolon (x1) için 0 - 30 aralığında, ikinci kolon (x2) için 5 - 45 aralığında ve son kolon (x3) için 35-65 aralığındadır). Bu verilerin tamamı 0 - 1 aralığına normalleştirilmiştir (sağdaki resim) ve artık birbiri ile karşılaştırılabilir bir değer kazanmıştır.

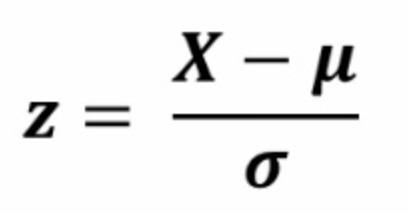
Python ile kodlarken aşağıdaki kod satırları kullanılabilir (df isimli bir dataframe'in daha önceden tanımlı olduğu kabul edilmiştir).

1. from sklearn import preprocessing
3. scaler = preprocessing.MinMaxScaler()
4. scaled\_df = scaler.fit\_transform(df)

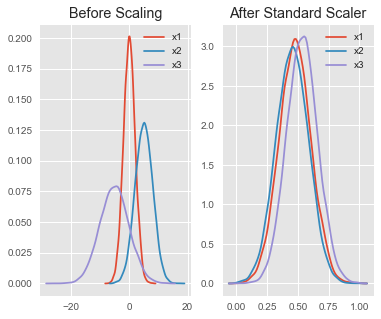
scikit-learn kütüphanesi linki : preprocessing.MinMaxScaler

Standard Scaler

Bu ölçekleme (scaler) yöntemi için verilerin normal dağılıma uygun dağıldığı kabulü yapılır (aksi durumlarda kullanımı risk içermektedir). Verilerin ortalama değeri 0 olacak ve standart sapma 1 olacak şekilde yeniden veriler ölçeklenir.



Yukarıda da formülü verildiği üzere, herhangi bir verinin ölçeklenmiş hali, o kolonun ortalama değerinin standart sapmasına bölümü ile bulunur.



Yukarıdaki şekilde de gösterildiği gibi, bütün veriler aynı x aralığına getirilmenin yanında ortalama değerleri ve standart sapmaları da sabitlenmiştir (solda ölçekleme öncesi ve sağda ölçekleme sonrası gösterilmektedir).

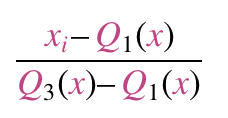
scikit-learn kütüphanesi ile kodlama için aşağıdaki komutlar kullanılabilir.

1. from sklearn import preprocessing
3. scaler = preprocessing.StandardScaler()
4. scaled\_df = scaler.fit\_transform(df)

scikit-learn kütüphanesine link : http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

RobustScaler

bu ölçeklemede ise bütün veriler, orijinal veri kümesindeki sırası korunarak ölçeklenir. Min-Max ölçeklemesinden en büyük farkı, aykırı / marjinal (outlier) verilere karşı dayanıklı (robust) olmasıdır.



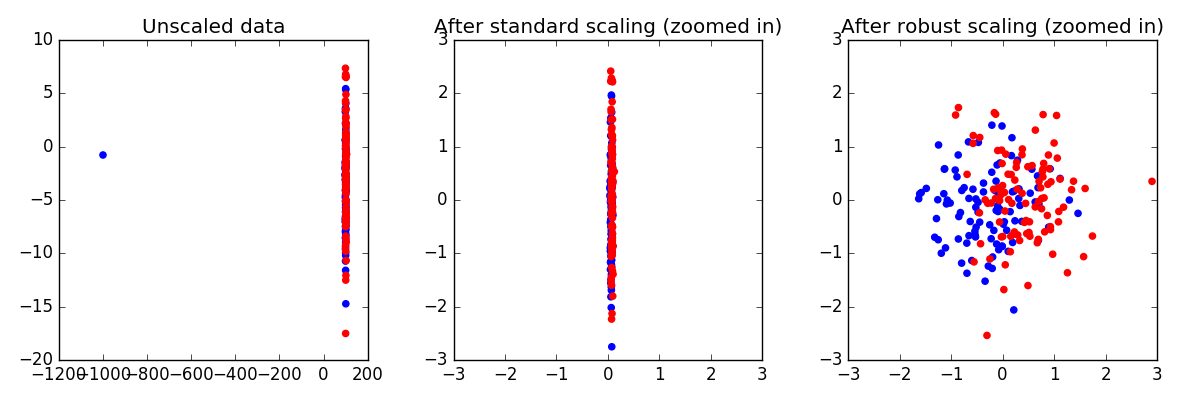
Python kodu aşağıdaki şekilde yazılabilir:

1. scaler = preprocessing.RobustScaler()
2. robust\_scaled\_df = scaler.fit\_transform(x)

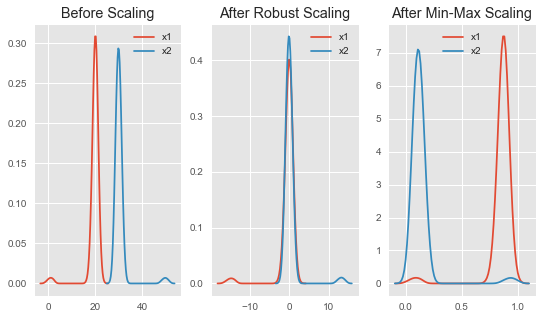
scikit-learn kütüphanesi linki : http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html

Örnek Oyuncak verisi üzerinde uygulanmış hali : http://scikit-learn.org/0.18/auto\_examples/preprocessing/plot\_robust\_scaling.html

Yukarıdaki son bağlantıda verilen uygulama incelenirse, standart skorlama ile de farkı net şekilde görülecektir. Bu durum, aşağıdaki şekilde görselleştirilmiştir



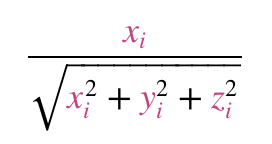
Solda verinin ölçeklenmemiş hali, ortada standard skor ile ölçeklenmiş hali ve en sağda ise robustscaling ile ölçeklenmiş hali gösterilmektedir. Görüleceği üzere ölçekleme sonrası iki boyutta da verilerin marjinal verilerden etkilenmemiş hali ölçeklendirilmiştir (verilerin ölçeklenmemiş halindeki tek başına duran mavi noktaya dikkat ediniz ve standart ölçeklemenin bu basit tek bir nokta yüzünden bir doğru üzerinde ölçekleme sonucu çıkardığını gözlemleyiniz).



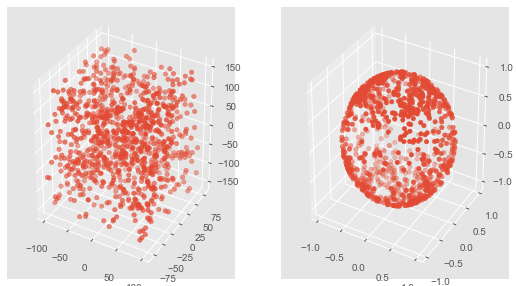
Orijinal verilere (solda) Robust Scaling uygulandıktan sonra (ortada) min-max ölçeklemesine göre (en sağda) marjinal verilerin etkilerinden arındırılmış sonuçlar üretildiği görülebilir.

Normalizer

Veriler her zaman iki boyutlu veya tek boyutlu uzayda ifade edilmez. Verilerin n boyutlu bir uzayda (n tane farklı kolon / öznitelik içermesi durumu) ifade edilmesi halinde, her boyut için farklı bir işleme tabi tutulması yerine, bütün boyutların tek bir formülde ölçekleme için kullanıldığı yöntemdir.



Örneğin yukarıdaki formülde verilen 3 boyutlu (boyutların x, y ve z olduğu kabul edilmiştir) uzaydaki bir veri noktasının normalleştirilme sonrası alacağı değer gösterilmiştir. Buradaki normalizer yöntemi aslında bir noktanın her üç boyutta da aldığı uzaklık değerlerine kartezyen mesafe ile ölçülmektedir.



Örneğin yukarıdaki şekilde, 3 farklı boyut (her boyutun aralığı farklı verilmiştir) sağdaki ölçeklenmiş haliyle 3 boyutun da -1 , 1 aralığına ölçeklenmesi ile sonuçlandırılmıştır.

Python kodu aşağıdaki şekilde yazılabilir

1. scaler = preprocessing.Normalizer()
2. scaled\_df = scaler.fit\_transform(df)

Devamı

Yukarıda anlatılan veri üzerinde ölçekleme işlemleri ile ilgili yazının devamı olabilecek aşağıdaki yazıyı da okumanızda fayda olabilir

http://scikit-learn.org/stable/auto\_examples/preprocessing/plot\_all\_scaling.html#sphx-glr-auto-examples-preprocessing-plot-all-scaling-py