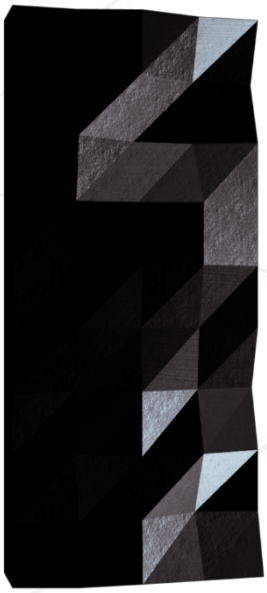


Veri Bilimi X Lojistik Regresyon

Harry Potter ve Bir Veri Bilimci

Özet:Bir sınıflandırıcı yaz ve Hogwarts'ı kurtar!

Sürüm: 2

İçindekiler

I Önsöz 2

II Giriş 3

III Hedefler 4

IV Genel talimatlar 5

V Zorunlu kısım 6

V.1 Veri Analizi 6

V.2 Veri Görselleştirme 7

V.2.1 Histogram 7

V.2.2 Dağılım grafiği 7

V.2.3 Çift çizimi 7

V.3 Lojistik Regresyon 8

VI Bonus Bölüm 9

VII Sunum ve akran değerlendirmesi 10

VIII Ekler 11

VIII.1 Matematik 11

VIII.2 Veri görselleştirme örnekleri 12

Bölüm I Önsöz

Wikipedia, yapay zekanın kurucu babalarından biri olan Yann Le Cun hakkında böyle diyor:

Yann Le Cun 1960 yılında Fransa'nın Paris kenti yakınlarında doğdu. Yapay zeka ve bilgisayarla görme (robotik) alanlarında araştırmacıdır. Derin öğrenmenin mucitlerinden biri olarak kabul edilmektedir.

1983'te ESIEE Paris'ten lisans derecesini aldıktan sonra Pierre ve Marie Curie Üniversitesi'nden mezun oldu ve 1987'de doktora derecesini aldı. Bu sırada, kayıp fonksiyonunun gradyanını hesaplayarak nöronların ağırlığını ayarlamak için gradyan iniş optimizasyon algoritmaları tarafından yaygın olarak kullanılan geri yayılım öğrenme algoritmasının erken bir formunu önerdi. 1987-1988 yılları arasında Toronto Üniversitesi'nde Geoffrey Hinton'ın laboratuvarında doktora sonrası araştırma görevlisi olarak çalışmıştır.

Yann Le Cun 1980'lerden beri makine öğrenimi, bilgisayarla görme ve derin öğrenme üzerinde çalışmaktadır: bir bilgisayarın temsilleri (görüntüler, metinler, videolar, sesler) tekrar tekrar eğitim örneklerine maruz bırakarak tanıma yeteneği.

Yann Le Cun 1987'de Toronto Üniversitesi'ne ve 1988'de AT&T re- search tesislerine katıldı ve burada denetimli öğrenme yöntemlerini geliştirdi.

Yann Le Cun aynı zamanda DjVu görüntü sıkıştırma teknolojisinin (Léon Bottou ve Patrick Haffner ile birlikte) ana yaratıcılarından biridir.

Yann Le Cun, Veri Bilimi Merkezi'ni kurduğu New York Üniversitesi'nde profesördür. Özellikle otonom araçların teknolojik gelişimi üzerine çalışmaktadır.

9 Aralık 2013'te Yann Le Cun, Mark Zuckerberg tarafından New York, Menlo Park ve 2015'ten beri Paris'te bulunan FAIR ("Facebook Yapay Zeka Araştırmaları") yapay zeka laboratuvarını tasarlamak ve görüntü tanıma üzerine çalışmak üzere Facebook'a katılmaya davet edildi. Daha önce Google'dan gelen benzer bir teklifi reddetmişti.

2016 yılında Paris'teki Collège de France'da "Chaire Annuelle Informatique et Sciences Numériques "de bilgisayar bilimleri alanında misafir profesör olarak bulunmuştur.

Bölüm II Giriş

Hayır! Kuruluşundan bu yana, ünlü büyücüler okulu Hogwarts böyle bir suçla hiç karşılaşmamıştı. Kötülüğün güçleri Seçmen Şapka'yı büyüledi. Artık tepki vermiyor ve öğrencileri evlere ayırma görevini yerine getiremiyor.

Yeni akademik yıl yaklaşıyor. Ne mutlu ki Profesör McGonagall böylesine stresli bir durumda harekete geçebildi, çünkü Hogwarts'ın yeni öğrencileri karşılamaması mümkün değil. . . Tüm muggle'ların nasıl kullanılacağını bildiği bir araç olan "bilgisayar" ile mucizeler yaratabilen bir muggle "veri bilimcisi" olan sizi çağırmaya karar verdi.

Birçok büyücünün içten gelen isteksizliğine rağmen, okul müdürü durumu açıklamak için sizi ofisine davet ediyor. Buradasınız çünkü muhbiri muggle aletlerinizi kullanarak sihirli bir Seçmen Şapka yaratabildiğinizi keşfetti. Ona "muggle" araçlarınızın çalışabilmesi için öğrenci verilerine ihtiyacınız olduğunu açıklıyorsunuz. Profesör McGonagall tereddütle size tozlu bir büyü kitabı veriyor. Neyse ki basit bir "Digitalis!" ile kitap bir USB belleğe dönüşüyor.

Bölüm III

Hedefler

Bu *DataScience x Logistic Regression* projesinde, aşağıdaki konuları keşfetmeye devam edeceksiniz Farklı araçları keşfederek *Makine Öğrenimi*.

Başlıkta *DataScience* teriminin kullanılması, bazıları tarafından açıkça küfürlü olarak değerlendirilecektir. Bu doğru. Bu konuda size *VeriBilimi*'nin tüm temellerini vermeyi iddia etmiyoruz. Konu çok geniş. Burada sadece *makine öğrenimi* algoritmasına göndermeden önce veri keşfi için bize yararlı görünen bazı temelleri göreceğiz.

Lineer *regresyon* konusunun devamı olarak bir lineer sınıflandırma modeli uygulayacaksınız: bir *lojistik regresyon*. Ayrıca, dal boyunca ilerlerken bir *makine öğrenimi* araç seti oluşturmanız için sizi çok teşvik ediyoruz.

Özetliyorum:

- Bir veri setini nasıl okuyacağınızı, farklı şekillerde nasıl görselleştireceğinizi, verilerinizden gereksiz bilgileri nasıl seçip temizleyeceğinizi öğreneceksiniz.
- Sınıflandırma problemini çözecek bir lojistik regresyon eğiteceksiniz.

Bölüm IV

Genel talimatlar

İstediğiniz dili kullanabilirsiniz. Ancak, bir veri kümesinin istatistiksel özelliklerinin çizilmesini ve hesaplanmasını kolaylaştıran bir kütüphaneye sahip bir dil seçmenizi öneririz.

Tüm ağır işleri sizin yerinize yapacak herhangi bir fonksiyon (örneğin *Pandas*) kütüphanesinin tanımlama işlevi hile olarak kabul edilecektir.

Bölüm V

Zorunlu kısım

Adımların aşağıdaki sırayla gerçekleştirilmesi önemle tavsiye edilir.

V.1 Veri Analizi

Veri keşfinin bazı temel adımlarını göreceğiz. Elbette bunlar mevcut tek teknik ya da izlenecek tek adım değildir. Her veri setine ve soruna benzersiz bir şekilde yaklaşılmalıdır. Gelecekte verilerinizi analiz etmenin başka yollarını da mutlaka bulacaksınız.

Her şeyden önce, mevcut verilere bir göz atın. hangi formatta sunulduğuna, çeşitli veri türleri olup olmadığına, farklı aralıklara vb. bakın. Başlamadan önce hammaddeniz hakkında bir fikir edinmek önemlidir. Veri üzerinde ne kadar çok çalışırsanız, onu nasıl kullanabileceğiniz konusunda o kadar çok sezgi geliştirirsiniz.

Bu bölümde Profesör McGonagall sizden describe[extension] adlı bir program üretmenizi istiyor. Bu program parametre olarak bir *veri kümesi* alacaktır. Tek yapması gereken örnekteki gibi tüm sayısal *özellikler* için bilgi görüntülemektir:

```
$> describe.[extension] dataset_train.csv
           Özellik      1Özellik      2Özellik      Özellik
Saymak     149.000000    149.000000    149.000000    149.000000
Ortalama     5.848322      3.051007      3.774497      1.205369
Std          5.906338      3.081445      4.162021      1.424286
Min          4.300000      2.000000      1.000000      0.100000
25%          5.100000      2.800000      1.600000      0.300000
50%          5.800000      3.000000      4.400000      1.300000
75%          6.400000      3.300000      5.100000      1.800000
Mak          7.900000      4.400000      6.900000      2.500000
```

Kullandığınız dil ne olursa olsun, sayım, ortalama, std, min, maks, yüzdelik dilim vb. gibi işi sizin için yapan herhangi bir işlevi kullanmak yasaktır. Tabii ki, describe kütüphanesini veya başka bir kütüphaneden ona benzeyen (az ya da çok) herhangi bir işlevi kullanmak da yasaktır.

V.2 Veri Görselleştirme

Veri görselleştirme, bir veri bilimcisi için güçlü bir araçtır. Verilerinizin neye benzediğine dair içgörü oluşturmanıza ve bir sezgi geliştirmenize olanak tanır. Verilerinizi görselleştirmek ayrıca kusurları veya anormallikleri tespit etmenizi sağlar.

Bu bölümde, bir soruyu yanıtlamak için her biri belirli bir görselleştirme yöntemi kullanan bir dizi komut dosyası oluşturmanız istenmektedir. Sorunun tek bir cevabı olması gerekmez.

V.2.1 Histogram

Bir sonraki soruyu yanıtlayan bir histogram görüntüleyen *histogram*.[extension] adlı bir komut dosyası oluşturun:

Hangi Hogwarts kursu dört ev arasında homojen bir puan dağılımına sahiptir?

V.2.2 Dağılım grafiği

Bir sonraki soruyu yanıtlayan bir *dağılım grafiği* görüntüleyen scatter_plot.[extension] adlı bir komut dosyası oluşturun:

Benzer olan iki *özellik* nedir?

V.2.3 Çift çizimi

pair_plot.[extension] adında, bir *çift grafiği* veya *dağılım grafiği matrisi* (kullandığınız kütüphaneye göre) görüntüleyen bir komut dosyası oluşturun.

Bu görselleştirmeden, lojistik regresyonunuz için hangi özellikleri kullanacaksınız?

V.3 Lojistik Regresyon

Son kısma geldiniz: Sihirli Şapkanızı kodlayın. Bunu yapmak için, lojistik regresyon *bire karşı bir* kullanarak çoklu sınıflandırıcı gerçekleştirmeniz gerekir.

İki program yapmanız gerekecektir:

- Birincisi modellerinizi *eğitecektir*, logreg_train.[extension] olarak adlandırılır. Parametre olarak dataset_train.csv. alır. Zorunlu kısım için, hatayı en aza indirmek için *gradyan inişi* tekniğini kullanmalısınız. Program, tahmin için kullanılacak ağırlıkları içeren bir dosya oluşturur.
- İkincisi logreg_predict.[extension] olarak adlandırılmalıdır. Parametre olarak şunları alır dataset_test.csv ve önceki program tarafından eğitilen ağırlıkları içeren bir dosya.

Sınıflandırıcınızın performansını değerlendirmek için bu ikinci programın tam olarak aşağıdaki gibi biçimlendirilmiş bir tahmin dosyası houses.csv oluşturması gerekecektir:

```
$> cat houses.csv
Dizin,Hogwarts Evi
0,Gryffindor
1,Hufflepuff
2,Ravenclaw
3,Hufflepuff
4,Slytherin
5,Ravenclaw
6,Hufflepuff
      [...]
```

Bölüm VI Bonus Kısım

Bu konu için pek çok ilginç bonus yapmak mümkün. İşte bazı öneriler:

- Açıklama için daha fazla alan ekleyin.[extension]
- *Stokastik gradyan inişi* uygulayın
- Diğer optimizasyon algoritmalarının uygulanması (Batch GD/mini-batch GD/ adını siz koyun)

Bonus kısım sadece zorunlu kısım MÜKEMMEL ise değerlendirilecektir. Mükemmel, zorunlu kısmın bütünsel olarak yapıldığı ve hatasız çalıştığı anlamına gelir. TÜM zorunlu gereklilikleri geçmediyseniz, bonus bölümünüz hiç değerlendirilmeyecektir.

Bölüm VII

Sunum ve akran değerlendirmesi

Ödevinizi her zamanki gibi Git deponuzda teslim edin. Savunma sırasında yalnızca deponuzdaki çalışmalar değerlendirilecektir. Doğru olduklarından emin olmak için klasörlerinizin ve dosyalarınızın adlarını iki kez kontrol etmekten çekinmeyin.

Düzeltme sırasında, seçimlerinizi sunma, açıklama ve gerekçelendirme yeteneğinizin yanı sıra tesliminiz (sizin için tüm ağır işleri yapan işlevler yok) üzerinden değerlendirileceksiniz.

Sınıflandırıcınız, dataset_test.csv dosyasında bulunan veriler üzerinde değerlendirilecektir. Cevaplarınız *Scikit-Learn* kütüphanesinin doğruluk puanı kullanılarak değerlendirilecektir. Profesör McGonagall, algoritmanızın yalnızca minimum **%98** hassasiyete sahip olması durumunda Sıralama Şapkası ile karşılaştırılabilir olduğunu kabul eder.

Kullanılan *makine öğrenimi* algoritmalarının işleyişini açıklayabilmek de önemli olacaktır.

Bölüm VIII

Ekler

VIII.1 Matematik

Lojistik regresyon neredeyse doğrusal regresyon gibi çalışır. İşte bir maliyet (k*ayıp*) fonksiyonu:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m} y^i \log(h_\theta(x^i)) + (1 - y^i)\log(1 - h_\theta(x^i))$$

Burada $h_\theta(x)$ aşağıdaki şekilde tanımlanır:

$$h_\theta(x) = g(\theta^T x)$$

ile :

$$g(z) = \frac{1}{1 + e^{-z}}$$

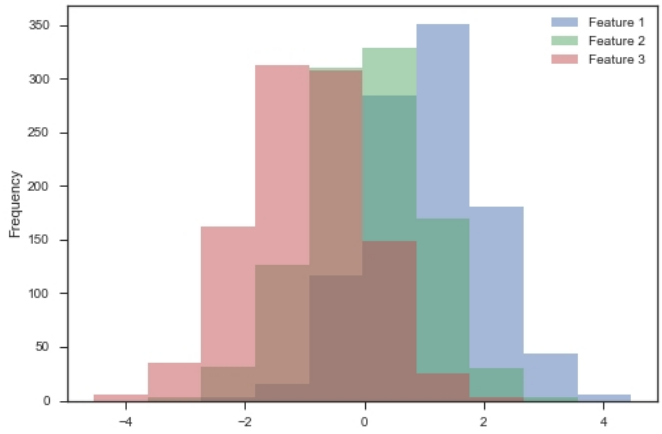
Kayıp fonksiyonu bize aşağıdaki kısmi türevi verir:

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{m}\sum_{i=1}^{n} (h_\theta(x^i) - y^i) x_j^i$$

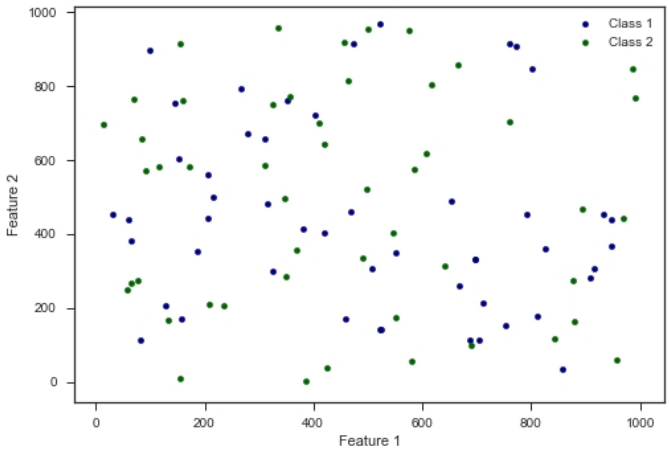
VIII.2 Veri görselleştirme örnekleri

İşte bazı veri görselleştirme örnekleri:

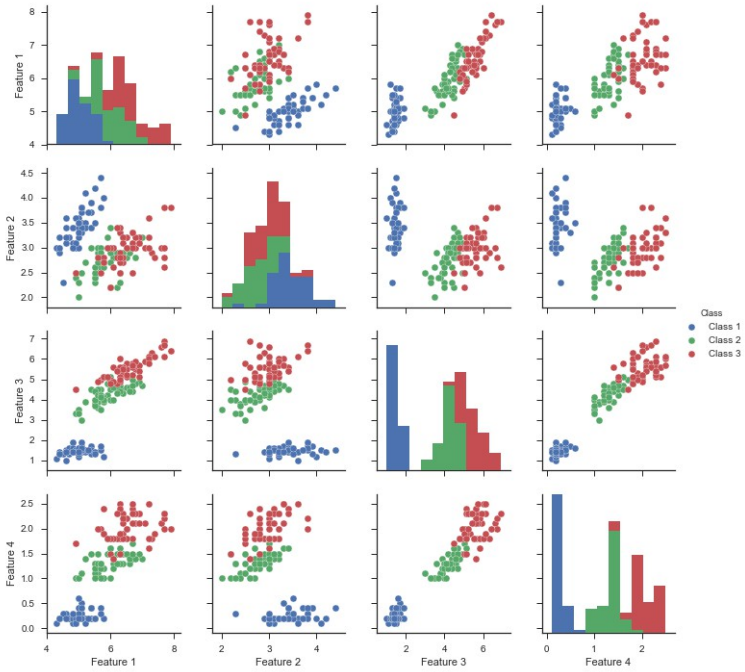
- Histogram



- Dağılım grafiği



- Çift çizimi


Feature 1
Feature 2
Feature 3
Feature 4
Class
Class 1
Class 2
Class 3