

Log Lineer Modeller ve Kosulsal Rasgele Alanlar (Log Linear Models and Conditional Random Fields -CRF-)

Ders 2

Charles Elkan ders notlari

Elkan'ın makine ogrenimi konusuna bakisi ilginc, ona gore makine ogrenimi bir bilgisayar bilim (computer science) konusudur, mesela altta islenen maksimum olurluk bilgisayar bilimdir.

Kosulsal Olurluk (Conditional Likelihood)

Diyelim ki elimizde egitim verisi olarak ikili $\langle x, y \rangle$ veri noktaları var. O zaman y 'nin x 'e kosulsal olarak bagli (conditional on) bir dagilimi oldugunu soyleyebiliriz.

$$y \sim f(x; \theta)$$

Yani her x icin farkli bir y dagilimi ortaya cikabilir. Ve tum bu farkli dagilimlerin ortak noktasi θ parametresidir. Kosulsal olasilik yani soyle yazilabilir,

$$P(Y = y|X = x; \theta)$$

Usttekiler Y icin bir model ortaya koydu, peki elimizde X 'in dagilimi icin bir olasilik modelimiz var mi? Cevap hayir. Niye? Dusunelim, $p(y, x)$ nedir ?

$$p(x, y) = p(x)p(y|x)$$

Ustte $p(y|x)$ 'i tanimlayacak (θ uzerinden) bir olasilik demeti / ailesi tanimladik, fakat elimizde $p(x)$ dagilimini verecek bir model yok, o zaman $p(x, y)$ 'yi tanimlayacak bir model de yok.

Fakat bu dunyanin sonu degil. Belki de Makine Ogrenimi bransinin bir slogani su ol-mali: “Ogrenmen gerekmeden seyi ogrenme”. Ustteki ornekte $p(y|x)$ 'i ogrenebiliriz, ama $p(x)$ 'i illa ogrenmemiz gerekir mi?

Siniflayici (classifier) ve takip edilen (supervised) ogrenim durumunu dusunursek, bize egitim amaclı olarak $\langle x, y \rangle$ ikili veri noktaları saglanacak. x kaynak veri, y tahmin edilecek (ya da basta egitim hedefi olan) etiket olacak. y icin bir model ortaya cikartiyoruz, cunku test zamanında y olmayacak, fakat x hep olacak. Yani y 'nin modellenmesi mecburi, cunku “genelleyerek” onun ne oldugunu bulacagiz, ama x hep verili.

Kosulsal Olurluk Maksimum Olurluk Prensibi

Egitim verisi $\langle x_1, y_1 \rangle, \dots, \langle x_n, y_n \rangle$ icin, θ 'yi soyle sec

$$\hat{\theta} = \arg \max_{\theta} \prod_{i=1}^n p(y_i|x_i; \theta)$$

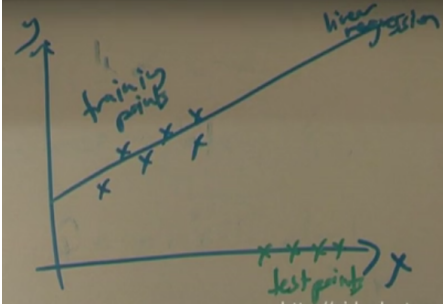
Normal maksimum olurlukta bilindigi gibi olasiliklari carpimi maksimize edilir, burada maksimize ettigimiz “kosulsal” olasiliklari carpimi.

Burada onemli bir soru su: bildigimiz gibi maksimum olurluk hesabi her veri noktasinin bir diglerinden bagimsiz oldugunu farzeder [cunku her olurluk hesabini bir digeri ile carpiyoruz, baska ek carpim, toplama, vs yapmıyoruz], bu faraziye dogru bir faraziye midir? Bu soru ve ona verilecek cevap cok onemli. Evet, eger eğitim noktaları birbirinden bagimsiz degilse maksimum olurluk kullanmamalıyız. Bagimsizligi da iyi tanımlamak gerekiyor tabii, eger üstteki durumda x_i verildikten sonra y_i ’lerin birbirinden bagimsiz olması yeterli.

Bu model klasik İstatistik’te cokca kullanılan bir yaklaşımdır, hatta lineer regresyon’un temeli üstteki faraziyedir.

$$y = \alpha + \beta \bar{x} + N(0, \sigma^2)$$

Bu standart lineer regresyon modeli, ve bu modelde her y ona tekabül eden x ’e bağlı, bu sayede x ’ler biliniyorsa y ’ler birbirinden kosulsal olarak bagimsiz hale geliyor, böylece x ’ler birbirine bagimli olsa bile α ve β ’nin bulunması mumkun oluyor.



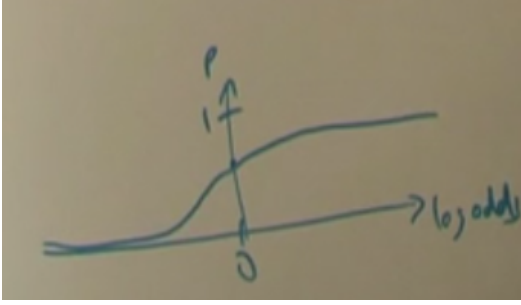
Üstteki resimde eğitim noktaları (training points) mavi olsun, test noktaları yeşil olsun (hemen altında). Bazı Yapay Öğrenim yaklaşımları diyebilir ki eğitim x ’lerinin dağılımı test x ’lerinin dağılımından farklı, bu veri seti öğrenilemez (yani genellenemez, modellenemez). Fakat klasik İstatistik buna bakar ve der ki x ’lerin verildiği durumda y ’ler bagimsizdir, bu şekilde bir kosulsal model öğrenilebilir.

Lojistik Regresyon aynı şekilde işler (lojistik regresyon, log lineer modellerin özel bir halidir, CRF’ler aynı şekilde). Burada da öğrenilen bir

$$p = p(y|x; \alpha, \beta)$$

modeli vardır ve y değerleri sadece 0 ve 1 olabilir. Tahmin edilen olasılık ise y ’nin 1 olma olasılığıdır. Bu model Rasgele Gradyan Çikisi ile eğitilir [detaylar için *Lojistik Regresyon* notlarımıza bakabilirsiniz].

$$\log \frac{p}{1-p} = \alpha + \sum_j \beta_j x_j$$



p log sansinin monotonik bir fonksiyonudur, ve ters yonden bakarsak, log sans p 'nin monotonik bir fonksiyonudur. Yani lineer bir fonksiyon (sag taraf) ne kadar buyurse, olasilik / log sans o kadar buyuyecektir. Bu buyume durumu mesela β_j katsayisini veri analizi baglaminda yorumlanabilir hale getirir. Diyelim ki β_4 katsayisi pozitif, o zaman diger tum sartlarin esit oldugu durumda (with all else being equal) x_4 ne kadar buyurse 1 olma olasiligi o kadar artar.

Lojistik modellerin onemli bazi avantajlari var, ki bu avantajlar log lineer modellere de sirayet ediyor (bu iyi).

1) Degiskenler arasi ilinti (correlation) probleme yol acmaz: Bu fayda aslinda daha once belirttigimiz x 'lerin birbirine bagimli olabilmesi ile alakali. Bagimsizlik onarti aranmadigi icin istedigimiz kadar x 'i problemin uzerine atabiliriz, egitici algoritma bunlardan cikartabildigi kadar iyi bir model bulacaktır.

Kiyasla mesela Naive Bayes boyle degildir, eger bir NB siniflayicisini egitiyorsak, ve ogelerin (feature) arasinda ilinti var ise, siniflayicinin dogrulugu (accuracy) azalabilir.

2) LR ile “1 olma olasiligini”, yani “bir sayisal skoru”, elde ediyoruz, bu sadece 1/0 degerinden daha fazla bir bilgi demektir.

3) Bu skor, anlami olan bir olasiliksal degerdir: Sonucta SVM siniflayicilari da $-\infty$ ve $+\infty$ arasinda degerler dondururler, ve bu degerler siralama (ranking) amaclı kullanilabilir, fakat olasilik matematigi acisindan anlami olan bir degerin olmasi bundan bile iyidir. Naive Bayes 0 ve 1 arasinda deger dondurebilir, fakat bu degerlerin de olasiliksal olarak aslinda anlami yoktur, pratikte goruldu ki bu degerler cok uc noktalarda, ya sifira cok yakin, ya bire cok yakin. Literaturde NB skorlarinin “iyi kalibre edilmiş olmadığı” soyleneir.

X_1, \dots, X_n test ornekleri ve tahmin edilen olasiliklar $P(Y = 1|x_i) = v_i$ olsun. Diyelim ki $s = \sum_i v_i$ ve t sayisi $1, \dots, n$ tane ogenin icinden $y = 1$ degerini tasiyan ogelerin sayisi olsun. Ornek, elimizde 100 tane egitim noktası var, bunlari 60'i 1 degerinde. Bu durumda s yaklasik 60 olacaktir (rasgele gurultuyu hesaba katarsak tabii), yani $E[t] = s$ denebilecektir ve bu sadece eger olasiliklar iyi kalibre edilmissse soylenebilir.

4) Dengesiz egitim verisi kullanilabilir: pek cok egitim setinde mesela 1 degeri tasiyan

degerleri 0 degeri tasiyanlardan cok daha fazla. Lojistik regresyon bu tur veriyle rahatca calisabilir.

Ders 3

Lojistik regresyon icin log olurlugun (LCL) turevini almak lazim. Once basitlestirme amacli $\alpha = \beta_o$, ve $x_0 = 1$. O zaman log sansin eski hali (altta esitligin sol tarafi) soyle yazilabilir (sag taraf), daha derli toplu bir formül olur,

$$\alpha + \sum_j \beta_j x_j = \sum_{j=0}^d \beta_j x_j$$

Bulmak istedigim her j icin $\frac{d}{d\beta_j} LCL$ lazim

$$\frac{d}{d\beta_j} LCL = \sum_{i:y_i=1} \frac{d}{d\beta_j} \log p(1|..) + \sum_{i:y_i=0} \frac{d}{d\beta_j} \log p(0|..) \quad (3)$$

Eger ustteki bir bolumu p digerine $1 - p$ dersem, yani soyle

$$= \sum_{i:y_i=1} \frac{d}{d\beta_j} \underbrace{\log p(1|..)}_p + \sum_{i:y_i=0} \frac{d}{d\beta_j} \underbrace{\log p(0|..)}_{1-p}$$

O zaman

$$= \sum_{i:y_i=1} \frac{d}{d\beta_j} \log p + \sum_{i:y_i=0} \frac{d}{d\beta_j} \log(1 - p)$$

Biliyoruz ki

$$\frac{d}{d\beta_j} \log p = \frac{1}{p} \frac{d}{d\beta_j} p \quad (1)$$

$$\frac{d}{d\beta_j} \log(1 - p) = \frac{1}{1 - p} (-1) \frac{d}{d\beta_j} p \quad (2)$$

Ustteki son iki formülün her ikisinde de $d/d\beta_j p$ kısmi olduguna dikkat.

Notasyon

$$e = \exp \left[- \sum_{j=0}^n \beta_j x_j \right]$$

$$p = \frac{1}{1 + e}$$

$$1 - p = \frac{1 + e - 1}{1 + e} = \frac{e}{1 + e}$$

Simdi $d/d\beta_j p$ 'e donelim, ve p 'nin ustteki gibi oldugundan hareketle,

$$\begin{aligned} \frac{d}{d\beta_j} p &= (-1)(1 + e)^{-2} \frac{d}{d\beta_j} e \\ &= (-1)(1 + e)^{-2} (e) \frac{d}{d\beta_j} (x_j) \\ &= \frac{1}{1 + e} \frac{e}{1 + e} x_j = p(1 - p)x_j \end{aligned}$$

Son ifade kodlama icin oldukca uygun, $d/d\beta_j p$ hesabini yine icinde p iceren bir ifadeye bagladik, ayrica turev x_j ile orantili.

Bu hesaplama aslinda (1) icindeki $d/d\beta_j p$ kismini hesaplamis olduk. Eger yerine koyarsak,

$$\frac{d}{d\beta_j} \log p = \frac{1}{p} p(1 - p)x_j$$

p 'ler iptal olur

$$= (1 - p)x_j$$

Ayni sekilde (2) icin

$$\begin{aligned} \frac{d}{d\beta_j} \log(1 - p) &= \frac{1}{1 - p} (-1)p(1 - p)x_j \\ &= -px_j \end{aligned}$$

Ustteki turevler tek bir egitim veri noktası icin. Tum egitim veri setinin turevi her noktanin turevlerinin toplami olacak, (3)'de goruldugu gibi.

$$\frac{d}{d\beta_j} LCL = \sum_{i:y_i=1} (1 - p_i)x_{ij} + \sum_{i:y_i=0} -p_i x_{ij} \quad (4)$$

x_{ij} notasyonunda j , j^{inci} oge / ozellik anlamina geliyor. Simdi notasyonel bir numara kullanacagim,

$$= \sum_{tum\ i} (y_i - p_i)x_{ij}$$

Bunu niye yaptım? (4) formülünde eşitliğin sağ tarafı, birinci terim içinde 1 sayısı var, sonraki terimde 1 yok. Eğer 1 olup olmaması yerine y_i kullanırsam, ki zaten 1'in olup olmaması y_i 'nin 1 olup olmamasına bağlı, tek bir terimde işi halledebilirim. $y_i = 1$ olduğu zaman üstteki ifade $1 - p_i$ olacaktır, olmadığı zaman $-p_i$ olacaktır.

Eristigimiz sonucu analiz etmemiz gerekirse, nihai formül gayet basit ve temiz çıktı.

[24:10] kalibrasyonla alakalı bir yorum

Rasgele Gradyan Çikisi (Stochastic Gradient Ascent)

Fikir: türevi eğitim noktası başına hesapla, ve modeli hemen güncelle.

Eğitim noktaları $\langle x, y \rangle$ olarak gelsinler. Her nokta için, ve her β_j için

$$\frac{d}{d\beta_j} p(y|x; \beta) = g_j$$

hesapla.

$$\beta_j := \beta_j + \alpha g_j$$

Gradyanın ne olduğunu hatırlayalım, bir fonksiyonun maksimumuna “doğru” olan bir gidis yönünü gösterir, ve bu gidis yönü o fonksiyonu oluşturan değişkenlerin (parçalı türevleri) üzerinden belirtilir. O zaman elimizdeki gradyan o iç değişkenlerin maksimum yandaki değişim şeklini bize tarif eder.

Algoritmanın tamamı: alttaki formül için

$$\frac{d}{d\beta_j} p(y|\bar{x}; \bar{\beta}) = (y - p)x_j$$

Her x için

- O anki modele göre p 'yi hesapla

- Her $j = 0, \dots, d$ için

- $\beta_j := \beta_j + \alpha \underbrace{(y - p)x_j}_{\text{kısmi türev}}$ hesapla

Peki metotun ismindeki “rasgele (stochastic)” tanımı nereden geliyor? İyi bir soru bu çünkü metotta rasgele sayı üretimi gibi şeyler görmüyoruz. Cevap, metot yine de rasgele, çünkü her noktayı ayrı ayrı işliyoruz, ve bu noktaların eğitim algoritmasını gelişi bir nevi “veriye örneklemek” gibi sanki, ek olarak veriyi eğitime almadan önce rasgele şekilde karıştırmak da iyi olabilir.

Bazı Tavsiyeler (Heuristics)

1) Her özellik (feature) x_j 'i ölçeklemek, yani aynı ortalama (mean) ve varyansa sahip olacak şekilde tekrar ayarlamak. Yani mesela 0 ile 100 arasında olabilecek “yas” gibi

bir özelliği, 0 ve 1 arasında değişen özellikler ile aynı ortalama ve varyansa sahip olacak şekilde ayarlamak. Bunun sebebi güncelleme hesabındaki λ 'nin tek bir sabit olması, ve bu sabit her j için aynıdır, o sebeple λ 'nin her öğeye “aynı şekilde” uygulanabilmesi için öğelerin birbirine yakın olması iyidir. Ek olarak, genellikle eğitim verisinde 0 ile 1 arasında ikisel türden öğeler vardır, o sebeple bu şekilde olmayan diğer öğeleri 0 ve 1 arasında çekmek daha uygun ve kolay olur.

2) Veriyi rasgele şekilde sıralamak. Terminoloji: eğitim veri seti üzerinden bir geçiş yapmak bir “cag” (epoch) olarak bilinir.

3) λ 'yi deneme / yanılma yöntemi ile bulun (bu sabiti bulmanın sistemik bir yöntemi yok). Belki verinin içinden alınan daha ufak bir örneklem üzerinde bu deneme / yanılma işlemi yapılabilir.

4) Deneme yanılma işlemi şöyle yapabilirsiniz: büyük bir λ ile işe başlarsınız, ve her cagda λ değerini azaltabilirsiniz (mesela her cag sonunda $1/2$ ile çarparak).

Ders 4

Log Lineer Modeller

Bu modeller lojistik regresyonun yapıya sahip (structured) girdiler ve çıktılar için genellenmiş halidir. Lojistik regresyonda girdi $\bar{x} \in \mathbb{R}^d$ ve çıktı $y \in 0, 1$ idi, yani çıktı ikiseldi. Fakat biz bundan daha genel makine öğrenimi problemlerini çözmek istiyoruz, yani istediğimiz $x \in \mathbb{X}$, ki \mathbb{X} herhangi bir uzay olabilmeli, ve $y \in \mathbb{Y}$ ki \mathbb{Y} aynı şekilde herhangi bir uzay olabilmeli.

Mesela x bir cümle olabilmeli, diyelim ki $x = \text{“he sat on the mat”}$, tercümesi “adam paspasın üzerinde oturdu”. Buna karşılık olan y ise mesela şöyle olabilmeli, $y = \text{“pronoun verb article noun”}$, yani her kelimenin hangi gramer ögesi olduğunu gösteren bir ibare. Mesela “sat” yani oturmak, bir fiil (verb), “mat” paspas, bir isim (noun), ve y içinde gelen eğitim verisinde bunlar olabilmeli (üstteki örnekte ikinci öge), sadece 0/1 değerleri değil.

Bu tabii ki takip edilen (supervised) bir eğitim şekli olacak. Fakat dikkat bazı makine öğrenimi uygulamalarında “çok sınıftan gelen” ama tek bir değer vardır, mesela $y \in 1, 2, 3$ olabilir, 3 sınıflı bir çıktı yani. Bazen çıktı gerçek sayı (real number) olabilir, ama yine de tek bir y değeri vardır. Üstteki durum böyle değildir. Potansiyel olarak y 'nin büyüklüğü x ile birebir aynı bile olmayabilir. Bu tür bir karışık eslemeden bahsediyoruz. Tek sınırlamamız \mathbb{Y} 'nin sonlu (finite) olması.

Model şöyle (notasyonu biraz değiştirdik, β yerine w kullanıyoruz mesela, w modelin “ağırlıklarını (weights)” temsil ediyor.

$$p(y|x; w) = \frac{\exp[\sum_j w_j F_j(x, y)]}{Z(x, w)}$$

Yakından bakarsak model LR modeline benziyor. Bir lineer fonksiyonun \exp 'si alınıyor ve bu değer olasılık hesabında kullanılıyor. İleride zaten göreceğiz ki LR üstteki yaklaşımın bir “özel durumu”, yani üstteki model daha genel bir tanım.

Aklımıza birçok soru geliyor herhalde, mesela “ Z nedir?” ya da “ F_j nasıl hesaplanır?” gibi. Z şöyle tanımlanır

$$Z(x, w) = \sum_{y'} \exp \left[\sum_j w_j F_j(x, y') \right]$$

Tüm y' 'lere bakılıyor, yani tüm mümkün \mathbb{Y} değerleri teker teker y' üzerinden toplamda kullanılıyor. \mathbb{Y} 'nin sonlu olma faraziyesi burada önemli hale geliyor, toplamı sonsuz bir kume üzerinden yapamayız.

Z normalizasyon için kullanılıyor, çünkü olasılık teorisinde eğer elimizde çoklu bir hedef var ise, bu hedeflere olan olasılık değerlerinin toplamı 1 olmalıdır. Z iste bunu garantiler, bu sebeple bölen (denominator) bölümün (nominator) toplamı olmalıdır.

Her $F_j(x, y)$ bir özellik fonksiyonudur (feature function). Niye? Çünkü elimdeki x 'ler illa bir vektor olmayabilir, yani x_j “vektörünü” alıp w_j “vektörü” ile carpamam, bu sebeple önce bir fonksiyon ile bir numerik değer üretmem gerekiyor. Kume olarak

$$F_j : \mathbb{X} \times \mathbb{Y} \rightarrow \mathbb{R}$$

Eğer $F_j(x, y) > 0$ ve $w_i > 0$ ise, o zaman $F_j(x, y) = 0$ 'a kıyasla $p(y|x; w)$ artar. Sezgisel olarak tarif edersek özellik fonksiyonun (OF) soylediği sudur, eğer ağırlık pozitif ise OF'in değeri ne kadar büyürse elimizdeki y , x ile o kadar “uyumludur” (tabii ki belli bir özellik yani j için). Negatif ilinti bunun tam tersi olurdu.

Eğitim w_j ağırlıklarını bulmamızı sağlar. F önceden tanımlıdır (yani eğitime bile başlamadan önce), bu fonksiyonun ne olacağı “secilir”. Secilirken tabii ki x, y arasındaki ilintiye göre fazla / az sonuç geri getirebilecek şekilde secilmelidir.

Kelime örneğine geri dönersek, bir F şöyle olabilir,

$F_{15}(x, y) =$ “eğer ikinci kelimenin baş harfi büyük ve ikinci etiket isim (noun)”. OF'ler reel değerlidir. Bunun özel durumu 0/1 değeri veren OF'lerdir. Biraz önceki örnek mesela 0/1 donduruyor.

Ya da $F_{14}(x, y)$ diyelim ki şöyle “ilk kelimenin baş harfi büyük, ve ilk etiket bir isim”. Tahmin edebiliriz ki eğitim setimizde ilk kelimesinin baş harfi büyük *olan* ama o kelimesi isim olmayan pek çok örnek olacaktır. Bu durumda w_{14} küçük olur.

Dedğimiz gibi F reel değeri olabilir, mesela

$$F_{16}(x, y) = \text{length}(y) - \text{length}(x)$$

yani bu fonksiyonda x 'nin uzunluğunu y 'nin uzunluğundan çıkartıyoruz. Bu ne işe yarar? Diyelim ki otomatik tercüme yapması için bir yapay öğrenim programı yazıyoruz, x, y eğitim noktaları birbirinin tercümesi olan İngilizce/Fransızca cümleler. Cogunlukla Fransızca cümleler tekabül ettikleri İngilizce cümlelerden çok daha uzun oluyorlar, yani üstteki çıkarma cogunlukla pozitif sonuç verecek. Değişik bir

acidan bakarsak, pozitif bir sonuc, bir tercumenin dogru oldugu yonunde bir isaret olarak kabul edilebilir, ve ustteki OF uzerinden egitim algoritmasi bunu kullanir. Egitim sonrasi w_{16} pozitif bir agirlik alacaktır.

Bir log lineer modelde (buna CRF'ler de dahil) ilk yapilan is probleminiz icin onemli olan OF'leri ortaya cikartmak.

F tanimlamanin degisik bir baska yolu:

$a(x)$ bir fonksiyon olsun. Her $v \in \mathbb{Y}$ icin

$$F_j(x, y) = a(x)I(y = v)$$

tanimlayalim.

$$p(y|x; w) = \frac{\exp \sum_j w_j F_j(x, y)}{Z}$$

Simdi lineer zincirli CRF konusuna bakalim. Yine $x \in \mathbb{X}$ ve $y \in \mathbb{Y}$. x bir girdi zinciri, y bir cikti zinciri ve en basit durumda x ile ayni uzunlukta. Konusma bolumlerini etiketlemek bu kategoriye dahil, ama bir diger uygulama kelimeyi arasina eksi isaretleri koyarak bolme (hyphenation).

Mesela girdi $x = \text{"beloved"}$, cikti $y = \text{"00100000"}$ cunku bu kelime "be-loved" olarak bolunur.

Bu uygulama icin bir OF

$$F_j(x, y) = \frac{\text{kac tane 1 var}}{x \text{ uzunlugu}}$$

$x = \text{"beloved"}$, cikti $y = \text{"00100000"}$ icin sonuc $1/7$ olurdu.

Lineer zincir CRF icin hangi OF'lerin bazi sinirlari var.

$$F_j(\bar{x}, \bar{y}) = \sum_i f_j(y_{i-1}y_i\bar{x}i)$$

ki sembol uzeri duz cizgiyi (\bar{x} gibi) bu sefer bir sirali veri temsil etmek icin kullaniyoruz)

Mesela

$$f_{18} = f_j(y_{i-1}y_i\bar{x}i) = \text{"}i = 2, y_{i-1} = 0, y_i = 1, x_1x_2 = \text{"as"}}$$

Mesela "async" kelimesi "a-sync" olarak bolumelir, ve egitim setinde "async" ile " $y = 01...$ " gelirse ustteki OF bu bolunmeyi odullendirir / ogrenir.

Simdi CRF olmayan bir Lineer Model'e bakalim,

Mesela cok etiketli takip edilen ogrenim. "Cok etiketli" ne demektir? Dikkat, "cok sinifli (multi label)" degil, yani tek ogenin iki veya daha fazla deger arasindan birini secmesinden bahsetmiyoruz. Birde fazla etiket alabilmekten bahsediyoruz, mesela bir Internet sayfasi, bir veya daha fazla kategoriye ayni anda ait olabilir, mesela hem Spor, hem Is Dunyasi. Diyelim ki 10 mumkun etiket var, bir dokuman kac degisik sekilde etiketlenebilir?

$2^{10} = 1024$ sekilde (bu sayi, hesap bir kumenin kac degisik sekilde alt kumesi olabilir hesabini yansitiyor ayni zamanda, yani siralama onemli olmadan belli sayida ogenin kac degisik sekilde alt kumeleri olabilir sorusu). Bu buyuk bir rakam. Ve bu kadar cok olasilik var ise, egitim verisi tum kombinasyonlar icin ornek veri icermeyebilir. Fakat muhakkak algoritmamizin bu kombinasyonlari tahmin edebilmesini tercih ederiz.

Cozum? 10 degisik siniflayici kurarak bu problemi cozebiliriz (ayri ayri, tek basina tek sinifa bakilince yeterli veri cikar herhalde), fakat bu sekilde "siniflararasi" iliskileri yakalayamayiz. Log lineer model yaklasiminda oyle bir ikisel (binary) OF yaratirsiniz ki, mesela,

$$F_{19}(x, y) = "Spor \in y, Is Dunyasi \in y"$$

Dikkat edersek OF sadece y 'ye bakiyor. Bu OF'yi iceren algoritma egitilince ustteki OF icin bir pozitif agirlik ogrenilebilecektir.

Soru: bir anlamda problemin yerini degistirmis olmuyor muyuz? Mesela ustteki sekilde bu sefer her turlu kombinasyon icin OF'mi yaratacagiz? Cevap: eger sadece ikili eslere bakiyorsak, kombinasyon hesabi $C(10, 2) = 45$ sonucunu verir. Bu fena bir sayi degil.

Ayrica verinin seyrekligi bize hangi kombinasyonlarin dahil edilip edilmeyecegi yonunde yardimci olabilir.

Soru: cok sinifli problemler[lojistik regresyonu gelistirerek cozulemez mi? Cevap: boyle bir yaklasim var, buna multinom lojistik regresyon deniyor. Fakat bu yaklasimin log lineer modellerin ozel bir hali oldugunu belirtmek isterim, yani makine ogrenimi dunyasinin aktif olarak arastirdigi alan artik burasi, multinom lojistik regresyon asildi. Zaten log lineer modeller ile cok etiketli problemleri de cozebiliyor-sunuz.

Ders 5

Soru: biraz once sadece y 'ler arasinda bir OF tanimlayabildigimizi gorduk. Peki sadece x 'ler arasinda OF tanimlamak faydali olur muydu? Cevap: Formulu tekrar hatirlayalim,

$$p(y|x; w) = \frac{\exp \sum_j w_j F_j(x, y)}{Z(x, w)}$$

OF'nin gorevi hangi y 'lerin daha yuksek olasiligi oldugunu belirtmek. Eger sadece x var ise, bu durumda bolum ve bolendeki degerler birbirini iptal ederdi. Her y icin ayni x "katkisi" olurdu ve bunun siniflayiciya hicbir faydasi olmazdi.

[8:00-18:00 atlandi]

Cozdugumuz problemler su formatta

$$p(\bar{y}|\bar{x}; w) = \frac{\exp \sum_j w_j F_j(\bar{x}, y)}{Z(\bar{x}, w)}$$

Tahmin etmek icin

$$\hat{y} = \arg \max_y \exp \sum_j w_j F_j(\bar{x}, y)$$

Bir \bar{y} tahmin etmek icin bu modellerden birini kullanacaksak, $p(\bar{y}|\bar{x}; w)$ formulune \bar{x} 'i koyariz, ve elde edilen dagilimda hangi \bar{y} 'nin olasiligi daha yuksekse onu seceriz. Daha yuksek olasiliga sahip olan \bar{y} , $p(\bar{y}|\bar{x}; w)$ formulunde bolumu daha yuksek olendir. Bolen her \bar{y} icin sabit / ayni.

Aslinda exp'ye ihtiyac yok, cunku exp monotonik bir fonksiyon, yani sadece su kullanilabilir,

$$\hat{y} = \arg \max_y \sum_j w_j F_j(\bar{x}, y)$$

En olasi y 'yi bulmak icin Z 'nin gerekmedigine de dikkat, cunku bu sabit tum secenekler icin ayni.

Burada tahmin etmek baglaminda zor olan sey, en yuksek y 'yi bulmak icin tum y 'lere teker teker bakmaya mecbur olmamiz. Bu bakma islemi cok zaman alabilir, o zaman bu problemi bir sekilde cozmek lazim.

Diger problem, tum olasiliklarin 1'e toplanabilmesini saglayan normalize sabitinin hesabi, yani $Z(\bar{x}, w) = \sum_{y'} \exp[\sum_j w_j F_j(\bar{x}, \bar{y})]$, ki eger olasilik degeri hesaplayacaksak bu sabit gerekli.

Yani iki ana problem var, bir de egitim algoritmasi var, ki bu aynen lojistik regresyon orneginde oldugu gibi rasgele gradyan cikisi uzerinden olacak, bu 3 algoritmayi simdi sunacagiz.

Algoritma 1

Once,

$$\hat{y} = \arg \max_{\bar{y}} \sum_j w_j F_j(\bar{x}, \bar{y})$$

Bu hesabi polinom zamanda (polynomial time) yapmak istiyoruz. Tanımı biraz degistirelim,

$$= \arg \max_{\bar{y}} \sum_j w_j \sum_i f_j(y_{i-1}y_i\bar{x}i)$$

j tum ozellikler, i x, y “boyunca” ilerleyen indisler. Ustteki ibare tek bir egitim veri noktası için yapıyor, yani i degisik veri noktalarını indislemiyor (genellikle öyle olur, o yüzden belirtmek istedik).

Toplam işlemlerinin sırasını degistirelim,

$$= \arg \max_{\bar{y}} \sum_i \sum_j w_j f_j(y_{i-1}y_i\bar{x}i)$$

İçerideki toplama $g_i(y_{i-1}y_i)$ ismi verelim, böylece her i için degisik bir g fonksiyonuna sahip oluyorum.

$$= \arg \max_{\bar{y}} \sum_i g_i(y_{i-1}y_i)$$

y_{i-1}, y_i kelime bolme probleminde iki degerden birini alabilir. Cumle etiketleme probleminde belki 20 degerden birini alabilirler. \bar{x}, \bar{w} zaten sabit (egitim verisi içindir, ya da sabit olarak goruluyorlar). Bu durumda g ’yi temsil etmek için nasıl bir veri yapısı kullanmalıyım? Çünkü bilgisayar bilim yapıyoruz, ve bilgisayar biliminde veri yapıları vardır. Bize gereken belli y_{i-1}, y_i kombinasyonu için bir g degeri dondurulmesi, ve bu sonucu bir yerde depolayabilmek.

Gereken yapı basit bir matris olabilir. Diyelim ki m farklı y degeri var ise, m^2 hücresi olan bir matris isimizi görür. Her g_i için ayrı bir m^2 matrisi olacak tabii ki. n tane matris, d deger var ise işlem zamanı $O(m^2nd)$.

Algoritmamda ilk yapacağım iş mümkün g degerlerini önceden hesaplayıp (precompute) bir yerde depolu olarak tutmak / hazır etmek.

Tanım

$$Skor(y_1, \dots, y_k) = \sum_{i=1}^k g_i(y_{i-1}y_i)$$

Amacımız öyle bir y sıralaması (sequence) bulmak ki bu sıranın skoru en yüksek olsun.

$U(k)$ = en iyi sıralama y_1, \dots, y_k ’nin skoru

$U(k, v)$ = $y_k = v$ olma şartıyla en iyi sıralama y_1, \dots, y_k ’nin skoru

Amacım $U(n+1, BITIS)$ ’i bulmak. Mümkün etiketlere BASLA, BITIS adlı iki yeni deger ekledik, bu bazı formülleri kolaslastırarak. Bu tanım aslında $\arg \max$ ile bulmaya çalıştığım şeyin bir bölümü aslında, sadece amacımı bu şekilde tekrar tanımladım. Tekrar belirtmek gerekirse,

$$U(k, v) = \max_{y_1, y_{k-1}} \left[\sum_{i=1}^{k-1} g_i(y_{i-1}y_i) + g_k(y_k, v) \right]$$

Ilginç boluma geldik. Ustteki tanımı ozyineli olarak tanımlarsak,

$$U(k, v) = \max_{y_{k-1}} [U(k-1, y_{k-1}) + g_k(y_{k-1}, v)]$$

Bu ozyineli fonksiyonun avantajı nedir? Aslında bir önceki formüle göre çok daha cıtrefil duruyor. Avantaj surada, dinamik programlama (dynamic programming) tekniklerini kullanarak bir dongu icinde ustteki ozyineli hesabi yapmak mumkun. Simdi teker teker bakalim,

$$y_0 = BASLA$$

$$U(1, v) = \max_{y_0} [U(0, y_0) + g_1(y_0, v)]$$

Bu ilk basamakta aslında bir maksimizasyon yok, o zaman

$$= g(BASLA, v)$$

yeterli.

Ama ikinci basamakta isler zorlasiyor,

$$U(2, v) = \max_{y_1} [U(1, y_1) + g_2(y_1, v)]$$

Fakat esitligin sag tarafındaki U hesabini bir önceki basamakta hesapladiim ve depoladim, onu hemen kullanabilirim. Bu hesabin yuku nedir? Her mumkun v degerine (m tane) bakmam lazim, ve bu islem sirasinda her y_1 mumkun degeri (yine m tane) irdelemem lazim. Yani $O(m^2)$.

Bu islemi $U(n+1, BITIS)$ 'e kadar yapmam lazim. Toplam yuk $O(nm^2)$.

g matrislerini hesaplamak icin $O(nm^2d)$ demistik, bu $O(nm^2)$ 'ten daha buyuktur / ona baskindir, ve O aritmetigine gore daha buyuk olan kullanilir.

Bu algoritma dinamik programlamanin ozel bir halidir, bazen ona Viterbi algoritmasi ismi de verilir. Bilindigi gibi Viterbi algoritmasi Gizli Markov Modelleri (Hidden Markov Models) yapisini dekode etmek icin kullaniliyor. CRF'lerin HMM'e kismen baglantisi oldugu dusunulurse, Viterbi algoritmasinin burada da ortaya cikmasi sasirtici degil.

Algoritma 2

Sunu hesapla

$$Z(\bar{x}, w) = \sum_{y'} \exp \underbrace{\sum_j w_j F_j(\bar{x}, \bar{y})}_g$$

Icerideki toplama g_i demistik,

$$g = \sum_i g_i(y_{i-1}y_i)$$

Yani

$$Z(\bar{x}, w) = \sum_{\bar{y}} \exp \sum_i g_i(y_{i-1}y_i)$$

Bir toplamın exp'si, exp'lerin carpimi haline donusur, yani exp toplamdan “iceri” nufuz eder,

$$= \sum_{\bar{y}} \prod_i \exp g_i(y_{i-1}y_i) \quad (5)$$

$t = 1, \dots, n + 1$ icin sunu tanimlayalim,

$$M_t(u, v) = \exp g_t(u, v)$$

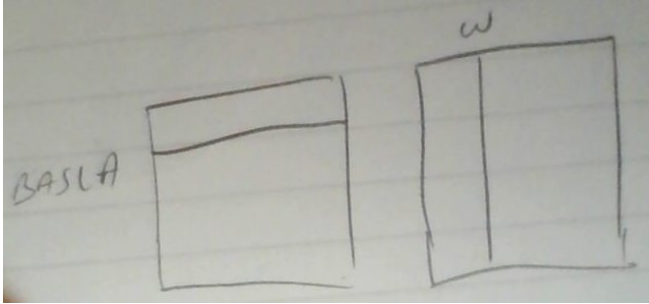
M_t asagi yukari g ile ayni sey, her degisik g fonsiyonu icin degisik bir matris var, bu matris hucrelerinin exp'sinin alinmis hali M_t matrisi.

$M_1(u, v)$ sadece $u = BASLA$ icin gecerli.

$M_{n+1}(u, v)$ sadece $v = BITIS$ icin gecerli.

$M_{12} = M_1 M_2$ yani matris carpimi.

$M_{12}(BASLA, w)$ 'yu dusunelim (ki bu tek bir hucre degeri)



Bu ifadenin sol taraftaki M_1 icinde $BASLA$ satirini sag taraftaki M_2 w kolonu ile carptigini dusunebiliriz.

$$\begin{aligned} M_{12}(BASLA, w) &= \sum_v M_1(BASLA, v) M_2(v, w) \\ &= \sum_v \exp[g_1(BASLA, v) + g_2(v, w)] \end{aligned}$$

Bu istedigimiz gibi bir ifadeye donusmeye basladi, cunku hatirlarsak, (5)'e benzeyen bir seyleri elde etmeye ugrasiyoruz. Gerci ustteki ifade tum y degerleri icin degil, tek bir v icin, ama yine de uygun, ustteki v yerine y_1 dersek belki daha uygun olur,

$$= \sum_{y_1} \exp[g_1(BASLA, y_1) + g_2(y_1, w)]$$

Uclu bir carpma gorelim: M_{123} .

$$\begin{aligned} M_{123} &= \sum_{y_2} M_{12}(BASLA, y_2) M_3(y_2, w) \\ &= \sum_{y_2} [\sum_{y_1} \exp[g_1(BASLA, y_1) + g_2(y_1 y_2)] \exp g_3(y_2, w)] \\ &\quad \sum_{y_1, y_2} \exp[g_1(BASLA, y_1) + g_2(y_1 y_2) + g_3(y_2, v)] \end{aligned}$$

Yani uc matrisi birbiriyle carparak y_1, y_2 uzerinden toplam almis oluyorum. Ve boyle devam edersem, yani tum matrisleri birbiriyle carparsam ve $BASLA, BITIS$ degerlerine bakarsam,

$$\begin{aligned} M_{123\dots n+1}(BASLA, BITIS) &= \\ \sum_{y_1, \dots, y_n} \exp[(g_1(BASLA, y_1) + g_2(y_1, y_2) + g_3(y_2, y_3) + \dots + g_{n+1}(y_n, BITIS))] \end{aligned}$$

Bu ifade parcalara ayirma (partition) fonsiyonu icin tam ihtiyacim olan sey. Daha once Viterbi algoritmasindan bahsettik, hatta bu algoritma dinamik programlama kategorisine girer dedik, ustteki algoritma dinamik programlama bile sayilmaz, aslinda bir matris carpimi sadece. Daha genel olarak ustteki algoritma ileri-geri (forward-backward) algoritmasinin bir turevi, bu algoritmalar bildigimiz gibi HMM'lerde sikca kullaniliyorlar.

Bu iki algoritma CRF'ler icin gerekli. Simdi CRF'leri nasil egitecegimizi gorelim.

Egitim

Maksimizasyon icin rasgele gradyan cikisi kullanacagiz.

$$p(y|x; w) = \frac{\exp \sum_j w_j F_j(x, y)}{Z(x; w)}$$

adyan cikisi icin ustteki formulun turevini alabilmeliyiz. Once log'unu almak lazim, cunku $\partial/\partial w_j \log p$ hesabi gerekli, usttekinin log'u ise bolumun log'u eksi bolenin log'u.

$$\frac{\partial}{\partial w_j} \log p = \frac{\partial}{\partial w_j} [\sum_j w_j F_j(x, y)] - \frac{\partial}{\partial w_j} \log Z(x; w)$$

Turevin eksi oncesi ilk bolumu cok basit, w_j ve toplam yokolacak (tum j 'lerin toplami yokoldu, cunku turev “tek” bir j degeri ile ilgileniyor, digerleri sifir oluyor)

$$= F_j(x, y) - \frac{\partial}{\partial w_j} \log Z(x; w)$$

Eksiden sonraki kisim cok zarif bir sonuca donusecek, birazdan gorecegiz.

$$\frac{\partial}{\partial w_j} \log Z(x; w) = \frac{1}{Z} \frac{\partial}{\partial w_j} Z$$

Turevi toplam icine tasiyoruz,

$$\begin{aligned} &= \frac{1}{Z} \sum_{y'} \frac{\partial}{\partial w_j} [\exp[\sum_{j'} w_{j'} F_{j'}(x, y')]] \\ &= \frac{1}{Z} \sum_{y'} \left[\exp[\sum_{j'} w_{j'} F_{j'}(x, y')] F_j(x, y') \right] \\ &= \sum_{y'} F_j(x, y') \frac{\exp[\sum_{j'} w_{j'} F_{j'}(x, y')]}{Z} \end{aligned}$$

Simdi ilginç kisma geldik, ustteki kesirli kisim $p(y'|x; w)$ degerine esittir.

$$= \sum_{y'} F_j(x, y') p(y'|x; w)$$

Ilginç durum burada ortaya cikiyor, cunku ustteki aynı zamanda bir beklenti (expectation) tanimi degil mi? Tum F_j degerlerini o degerlerin olasiliklari ile carpip toplarsak bir beklenti elde etmez miyiz? Evet. O zaman beklenti tanimini kullanabiliriz,

$$\frac{\partial}{\partial w_j} \log p = F_j(x, y) - E[F_j(x, y')]$$

ki y' soyle bir dagilimi takip ediyor,

$$y' \sim p(y'|x; w)$$

Soru: $\frac{\partial}{\partial w_j} \log p = ?$ Yani bu turev ne zaman sifra esittir?

Cevap: Turevin acilimina bakınca mesela, $F_j = 0$ olunca mı? Hayır, çünkü OF sıfır olsa bile beklenti kısmı sıfır olmayabilir. O zaman şöyle söylemek gerekir, eğer tüm y' için $F_j(x, y') = 0$ ise, o zaman türev sıfır olur.

Cogunlukla $F_j(x, y) = a(x)I(y = v)$. Hatırlarsak bu yöntem bir özelliği (ki $a(x)$ ile temsil ediliyor), her mümkün v değeri için bir OF'ye çevirmenin yolu idi (tek x 'e bağlı OF olamaz).

O zaman sunu da söyleyebiliriz, eğer $a(x) = 0$ ise, her y için $F_j(x, y) = 0$ demektir.

Bu bilginin faydası sudur, veride seyreklik var ise, lojistik regresyon bunları atlamayı bilir. Demek ki aynı şekilde kosulsal lineer modeller de bu özellikleri atlayabilir. Eğer bir özellik $a(x) = 0$ ise, o özellik ağırlık güncellemesi (weight update) sırasında atlanır.

Algoritma

Her eğitim noktası x, y için

j için

$E[F_j(x, y)]$ hesapla (buna sadece E diyelim)

Güncelle: $w_j := w_j + \lambda[F_j(x, y) - E]$

Bu hesabın en pahalı kısmı neresi? Beklenti hesabı. Bu beklentileri hesaplanması için daha önce verdiğimiz matris çarpımı yöntemine benzer bir yöntem kullanmak gerekiyor (burada vermeyeceğiz, arama motorunda Rahul Gupta üzerinden arayabilirsiniz, bu kişi bu konuyu anlatıyor).

Kaynak

[1] http://videlectures.net/cikm08_elkan_llmacrf