Makine Öğrenmesine Giriş – 2. Gün

Global AI Hub

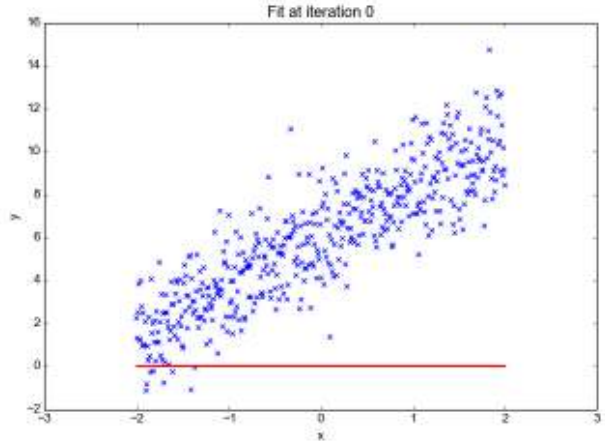
ALP SARICA
alp.sarica@globalaihub.com
linkedin.com/in/alpsarica

Regresyon Nedir?

Regresyon bir bağımlı değişken ile diğer bir veya birkaç bağımsız değişken arasındaki ilişkiyi belirler.

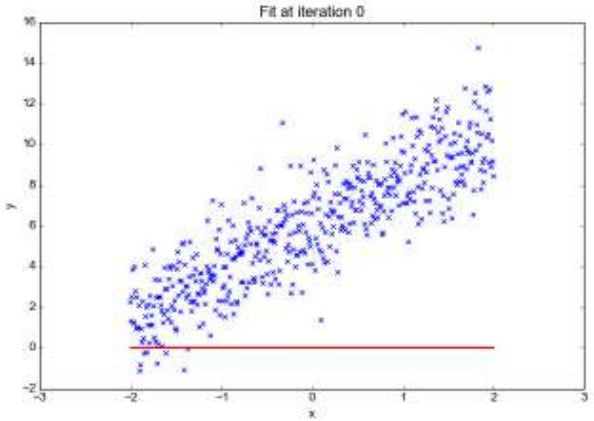
Modelin tahmin edeceği değer **sürekli** bir değerse bu tip problem **Regresyon** problemidir.

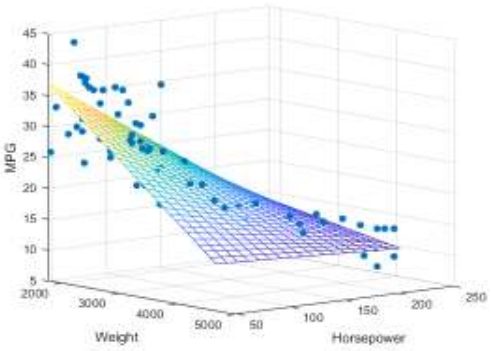
Problemin tanımına göre 3 farklı Regresyon çeşidi vardır.

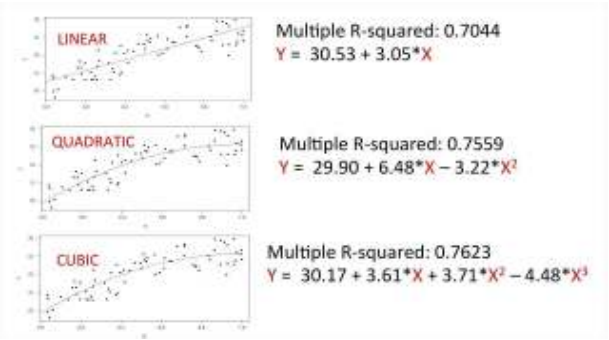


Regresyon Çeşitleri?

1. **Basit Lineer Regresyon:** Bir tane bağımsız değişkenin olduğu durumda kullanılan Regresyon çeşididir
2. **Çoklu Lineer Regresyon:** Birden fazla bağımsız değişkenin olduğu durumda kullanılan Regresyon çeşididir
3. **Polinomal Regresyon:** Bağımsız değişken x ile bağımlı değişken y arasındaki ilişkinin x'te n'inci dereceden bir polinom olarak modellendiği bir Regresyon çeşididir



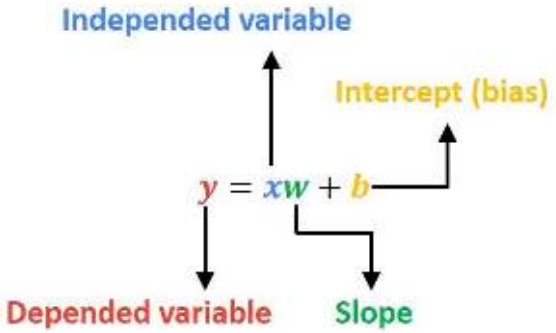


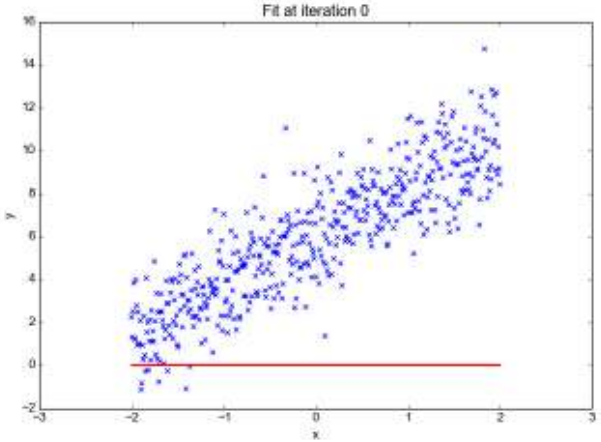


Basit Lineer Regresyon

Basit Lineer Regresyon en uygun düz çizgi (regresyon çizgisi olarak da bilinir) kullanarak bağımlı değişken (Y) ile bir bağımsız değişken (X) arasında bir ilişki kurar.

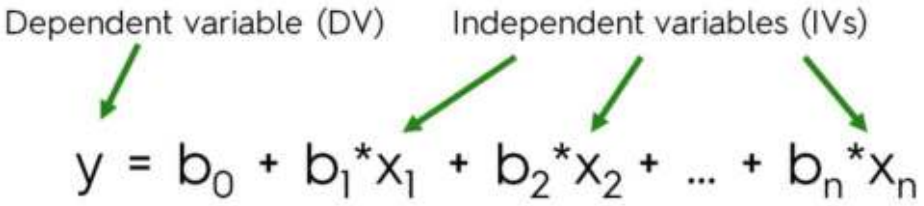
Regresyon, tüm noktalara oturmaya çalışacak en optimum doğru denklemini bulmaya çalışır.

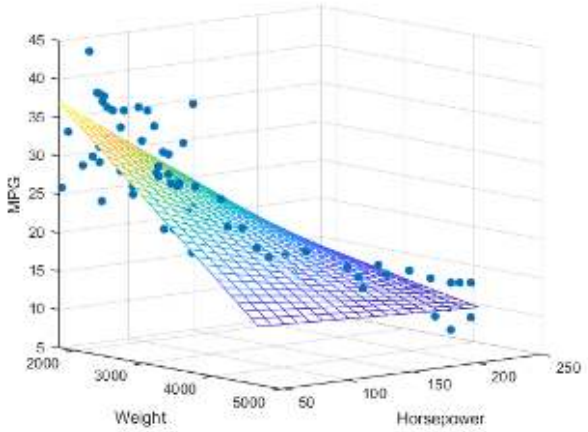




Çoklu Lineer Regresyon

Çoklu Lineer Regresyon en uygun düz çizgi (regresyon çizgisi olarak da bilinir) kullanarak bağımlı değişken (Y) ile birden fazla bağımsız değişken (X) arasında bir ilişki kurar.

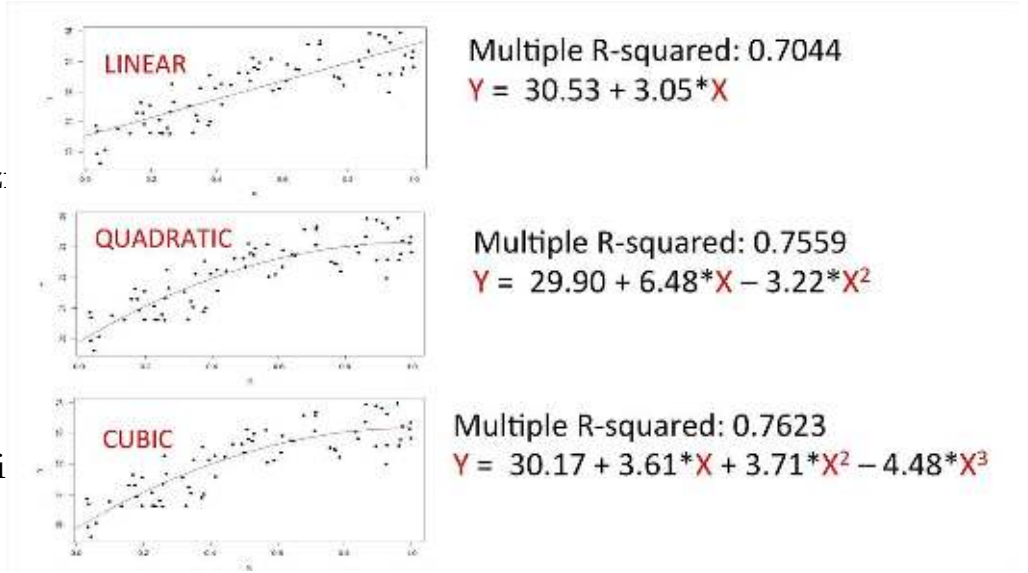




Polynomial Regresyon

Polinom regresyon, bir veya birden fazla bağımsız değişken x ile bağımlı değişken y arasındaki ilişkinin x'te n'inci dereceden bir polinom olarak modellendiği bir regresyon analiz biçimidir.

- Basit ve çoklu lineer regresyon modellerimizin tahminde yetersiz kaldığı durumda (verimiz nonlineer olduğunda) polinomel regresyon yardımıyla modelimizi verimize daha iyi oturtabiliriz. Bunun için modele verdiğimiz featureları dereceli (üslü) vermemiz gerekmektedir



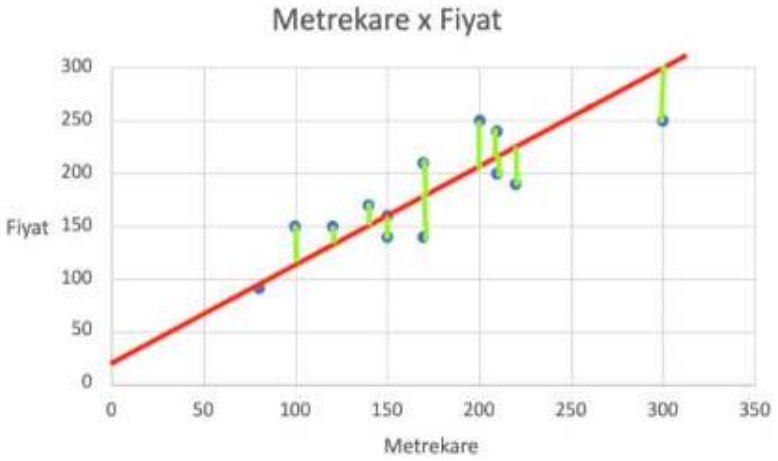
Regresyon Modelinin Performansını Ölçmek

Regresyon Modelinin Performansını Ölçmek - Hata Kavramı

Regresyon modelimizde hata, verinin gerçek değeri ile tahmin edilen değeri arasındaki uzaklıktır. Amacımız bu uzaklığı her veri noktamız için en aza indirmektir.

Negatif hatalarla pozitif hatalar birbirini götüreceğinden standart toplama işlemi yapmak yerine hatayı hesaplamak için belli istatistiksel denklemler kullanırız.

Metrekare	Fiyat
100	150
150	160
200	250
170	140
210	240
300	250
140	170
220	190
80	90
120	150
150	140
210	200
170	210



R-squared (R^2)

R^2, verilerin yerleştirilmiş regresyon hattına ne kadar yakın olduğunun istatistiksel bir ölçüsüdür. Yani doğrusal regresyon modelleri için uygunluk ölçüsüdür.

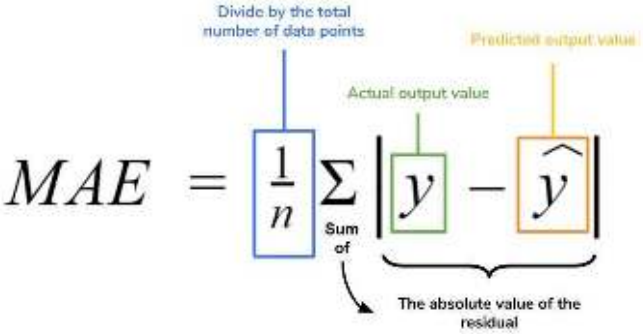
$$R^2 = 1 - \frac{SS_{Regression}}{SS_{Total}}$$

⭐ Yüksek R^2 değeri her zaman model iyidir demek değildir. Overfitting durumunda da R^2 skorumuz yüksek çıkacaktır.

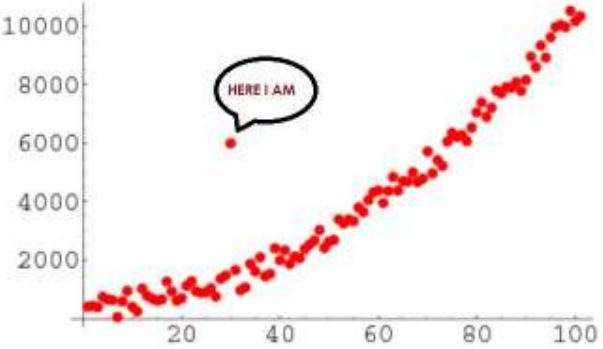
Mean Absolute Error (MAE)

MAE, tahmin edilen değerler ile gerçek değerler arasındaki farkların mutlak değerlerinin ortalamasıdır.

- MAE yöntemi, eğitim verilerindeki aykırı değerlerden aşırı etkilenmeme avantajına sahiptir. MEA yaklaşımıyla eğitilmiş bir model, 5 birimlik 1 hataya ve 1 birimlik 5 hataya eşit önem verecektir
- MAE ile ilgili ana sorun, minimumda türevlenebilir olmamasıdır. Bu türevlenebilirlik eksikliği,



ınsama sc



Mean Squared Error (MSE)

MSE, en sık kullanılan regresyon hata fonksiyonudur. Kare alma işlemi nedeniyle, büyük hataların (gerçek değer ile tahmin değeri arasındaki farkın) MSE üzerinde küçük hatalardan daha fazla etkisi vardır. Aykırı değerlerin tahminlerinizi fazlasıyla etkileyip etkilemediğini görmek istediğinizde gerçekten yararlıdır.

- MSE fonksiyonu basit, sürekli ve türevlenebilir olduğu için yaygın olarak kullanılmaktadır
- MSE ile eğitilmiş bir model, 1 birimlik 25 hataya karşılık 5 birimlik tek bir hataya aynı önemi verecektir. Model en büyük hataları azaltmak için önyargılı olacaktır

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Mean Squared Error (MSE)

- Ortalama kare hata fonksiyonu basit, sürekli ve türevlenebilir olduğu için yaygın olarak kullanılmaktadır
- Önemli bir MSE özelliği, küçük hatalara kıyasla büyük hatalara karşı orantısız duyarlılığıdır. MSE ile eğitilmiş bir model, 1 birimlik 25 hataya karşılık 5 birimlik tek bir hataya aynı önemi verecektir. Başka bir deyişle, birçok ortak koşulun tahminlerini cezalandırsa bile, model en büyük hataları azaltmak için önyargılı olacaktır

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Model Eğitiminde Karşılaşabileceğiniz Hatalar

Underfitting & Overfitting

Bias/Variance Tradeoff

Bias

Modelin eğitim veri seti üzerindeki hatası olarak kabul edilebilir. Bu underfit olarak adlandırılabilir

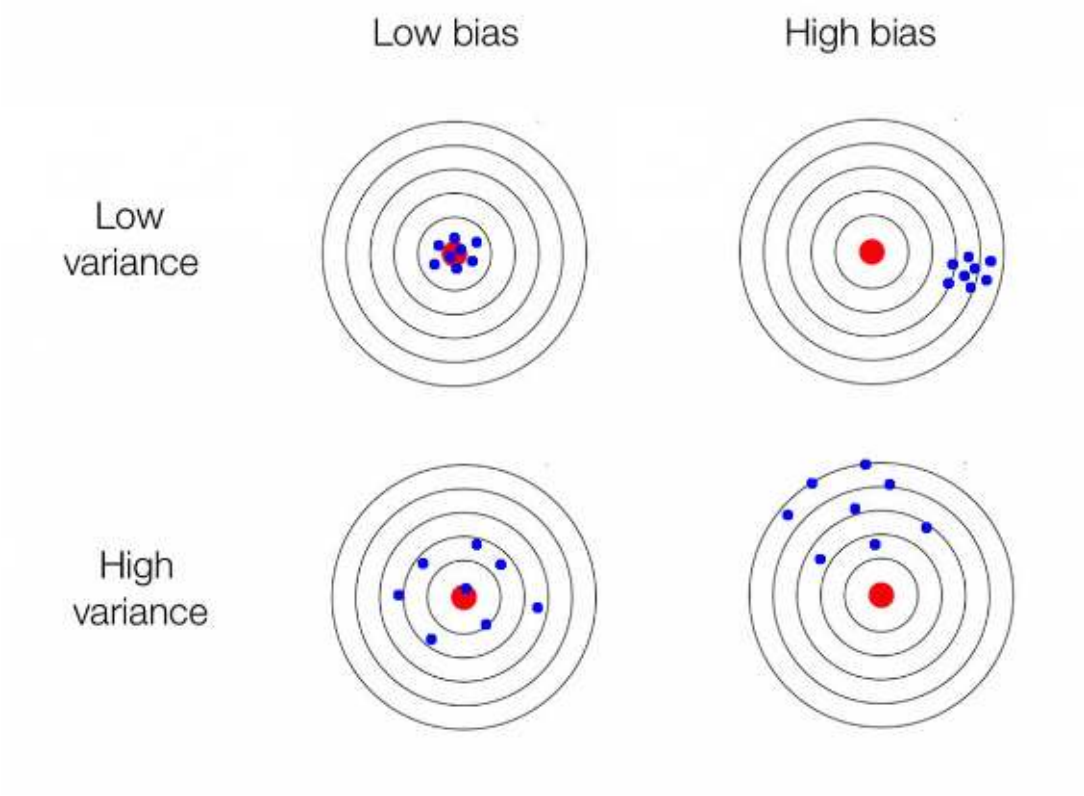
✯ Bias arttıkça varyans azalmaktadır.

Variance

Varyans, belirli bir veri noktası için model tahmininin değişkenliğidir. Modelin test setindeki performansının, eğitim setindekine göre ne kadar değiştiğini, kötüleştiğini ifade eder

✯ Varyans arttıkça bias azalmaktadır.

Bias/Variance Tradeoff



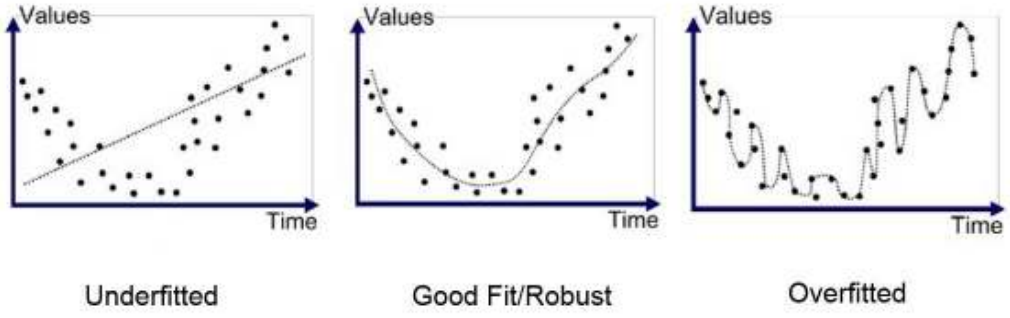
Underfitting & Overfitting

Underfitting

Makine Öğrenmesi modelinin verimizin trendini yakalayamadığında ortaya çıkan bir problemdir. Bir başka deyişle, modelimizin veriyi yeterince öğrenememesi veya veriden yeterli anlamı çıkarmamasıdır.

Overfitting

Makine Öğrenmesi modelinin, verimizin trendini yakalaması gerekenden daha çok yakaladığında ortaya çıkan bir problemdir. Bir başka deyişle, modelimiz ona verdiğimiz veriyi öğrenmekten çok ezberler ve daha önce görmediği veriler üzerinde başarılı bir çıkarım



Underfitting & Overfitting

- İstatistik ve Makine Öğreniminin önemli bir teorik sonucu, bir modelin genelleme (görmediği verileri kullanarak tahmin gerçekleştirebilme yeteneği) hatasının çok farklı üç hatanın toplamı olarak ifade edilebilmesidir:
  - Bias
  - Variance
  - İndirgenemez Hatalar
- Bias'ın fazla olması varyansın az olması anlamına gelmektedir ve bu da underfit'e örnektir
- Varyansın fazla olması bias'ın az olması anlamına gelmektedir ve bu da overfit'e örnektir
- İndirgenemez hatalar verideki noise ile alakalıdır ve çözmenin tek yolu veriyi temizlemektir

✯ Underfit olan bir model **düşük varyans, yüksek bias** değerlerine sahiptir.

✯ Overfit olan bir **model yüksek varyans, düşük bias** değerlerine sahiptir.

Engellemek İçin Neler Yapılabilir?

Underfitting

1. Yeni feature değerleri veya katmanlar eklenerek modelin karmaşıklığı arttırılabilir
2. Veriden gürültü temizlenebilir
3. Eğitim sırasında epoch sayısı arttırılabilir
4. Daha iyi feature değerleri verilebilir (Feature Engineering)
5. Modeli sınırlandıran değerler azaltılabilir (örneğin regülarizasyon parametresi)

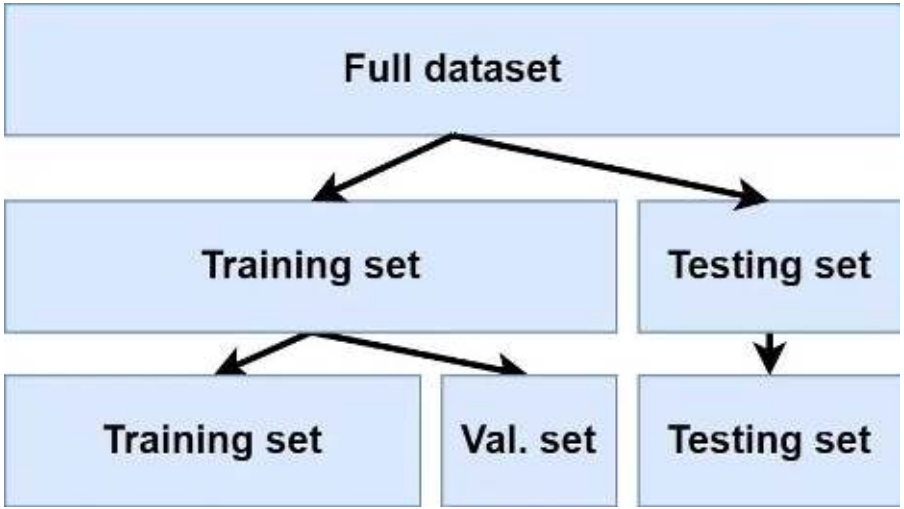
Overfitting

1. Eğitim verisi arttırılabilir
2. Modelin karmaşıklığı azaltılabilir
3. Eğitim sırasında erken durdurma (early stopping) yapılabilir
4. Regularizasyon uygulanabilir

Hata Azaltma Yöntemleri

Verimizi Eğitim-Test-Validasyon Olarak Ayırmak

1. Elimizdeki veri modelimize eğitim için gönderilir. Modelimiz elindeki veriyi öğrenir ve öğrendiği verinin üstünden tekrar tekrar geçerek kendisini geliştirir *(forward-backward prop.)*
2. Eğitim tamamen bittikten sonra model kendini skorlayabilmek için öğrenimde kullandığı veri ile kendini ölçer
3. Eğer skor yüksek çıkarsa modelimiz başarılı demektir
4. Modele eğitim için verilen veri arasında bulunmayan bir veri verildiğinde modelin aslında iddia ettiği kadar başarılı tahminle yapamadığını görülebilir



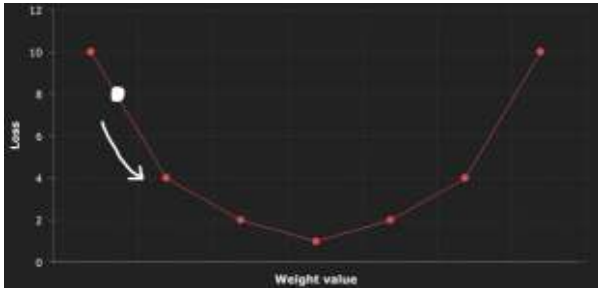
Erken Durdurma (Early Stopping)

1. Gradient Descent gibi yinelemeli öğrenme algoritmalarını düzenli hale getirmenin çok farklı bir yolu, doğrulama hatası minimuma ulaşır ulaşmaz eğitimi durdurmaktır
2. 2018 Turing Ödülü'nü Yoshua Bengio ve Yann LeCun'la birlikte almaya hak kazanmış Geoffrey Hinton'ın "No Free Lunch Theorem" dediği basit ve etkili bir düzenleme tekniğidir
3. Minimum bir hata değeri belirlenir ve validasyon seti hatası bu sınırın altına indiğinde veya sınıra eşit olduğunda n kadar daha iterasyon gerçekleşir. Eğer performansımız iyileşmezse hata değerimiz en iyi noktaya geri döner ve eğitim durur *(callback)*

Gradient Descent Algoritması

Gradient Descent'in genel fikri, bir maliyet fonksiyonunun sonucunu en aza indirmek için ağırlıkları yinelemeli olarak ayarlamaktır. Bu işlemi hata değeri lokal minimuma yakınsayana kadar yapar.

Gradient Descent tam olarak bunu yapar: θ parametre vektörüne göre hata fonksiyonunun yerel gradyanını ölçer ve azalan gradyan yönünde gider. Gradyan sıfır olduğunda, en optimum ağırlık değeri bulunmuş olur

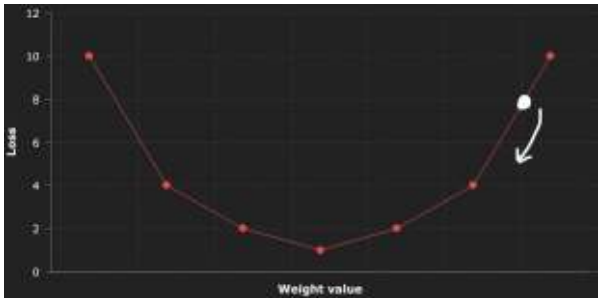


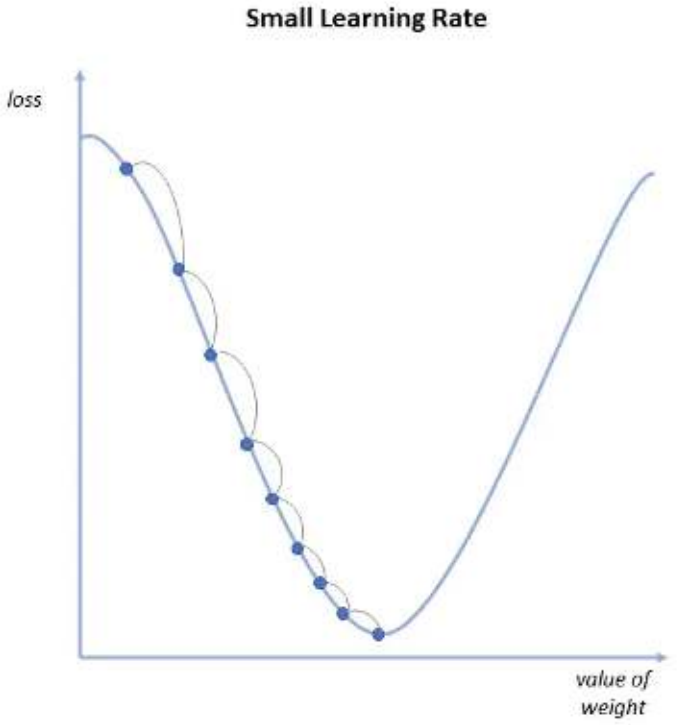
Repeat until convergence

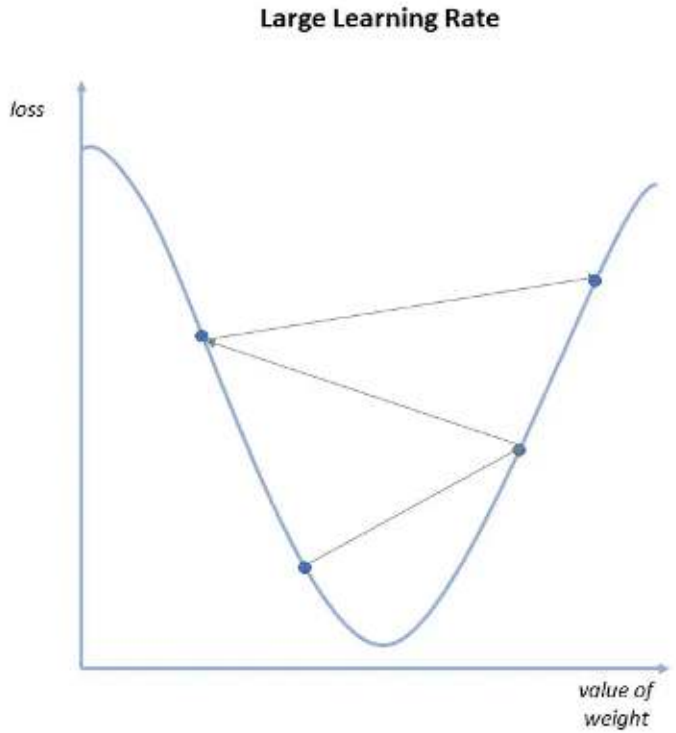
{

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

}

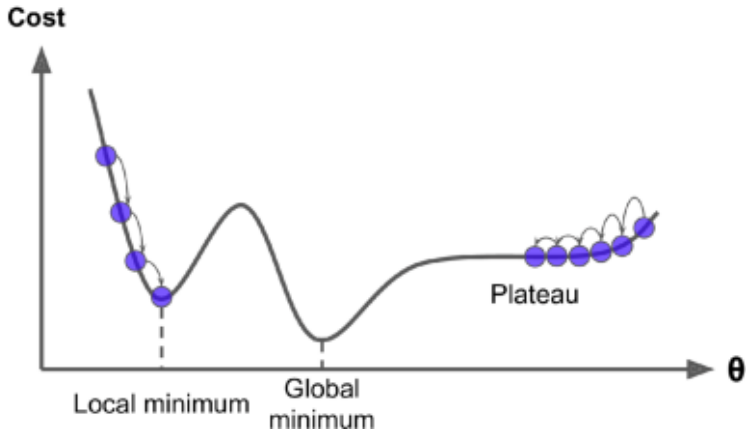



Small Learning Rate
loss
value of
weight


Large Learning Rate
loss
value of
weight

Gradient Descent Algoritması

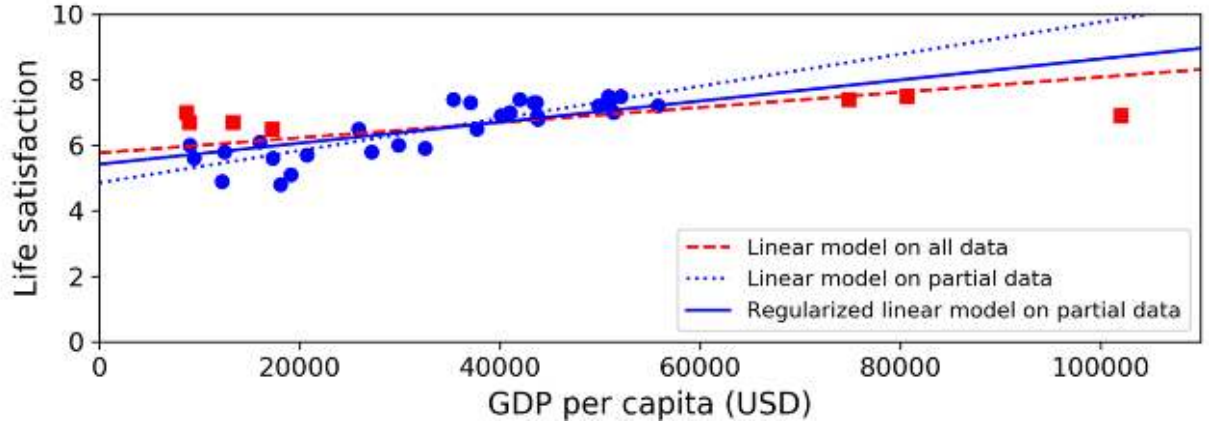
- Tüm cost functionlar convex şeklinde değildir, bu da algoritmanın lokal minimuma yakınsamasına sebep olur
- MSE ve MAE gibi cost functionlar convex şeklindedir. Bu sayede sadece bir global minimum olduğundan emin olabiliriz



✯ Gradient Descent kullanırken, tüm featureların benzer bir scale'e sahip olduğundan emin olmalısınız. (örneğin, Scikit-Learn'in StandartScaler sınıfını kullanarak) yoksa yakınsaması çok daha uzun sürer

Regülarizasyon

Regülarizasyon, bir modeli basitleştirmek ve buna bağlı olarak **overfit'i önlemek** amacıyla **kısıtlanma** işlemidir. Bu sayede modelin görünmeyen veriler üzerindeki performansı iyileştirir. Bu yöntemde cost functiondaki ağırlıklar, regülarizasyon terimi olarak bilinen başka bir terim eklenerek güncellenir yani cezalandırılır.



⭐ Öğrenme sırasında uygulanacak regülarizasyon miktarı bir **regularization term (regülarizasyon parametresi)** isimli hiper parametre ile kontrol edilebilir

Regülarizasyon

L1 Lasso

- L1 Regülarizasyonda hata fonksiyonu, ağırlıkların mutlak değeri ile cezalandırılır
- L1'in türevi k'dir (değeri ağırlıktan bağımsız olan bir sabit)
- L1'in türevini her seferinde ağırlıktan bir miktar sabit çıkaran bir kuvvettir

Repeat {

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

$$\theta_j := \theta_j - \alpha \left[\left(\frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} \right) + \frac{\lambda}{m} \theta_j \right]$$

}

L2 Ridge

- L2 Regülarizasyonda hata fonksiyonu, ağırlıkların karesi ile cezalandırılır
- L2'nin türevi 2 * ağırlıktır
- L2'nin türevini her seferinde ağırlığın %x'ini kaldıran bir kuvvettir

Repeat {

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

$$\theta_j := \theta_j - \alpha \left[\left(\frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} \right) + \frac{\lambda}{m} \theta_j^2 \right]$$

}

Regülarizasyon

L1 Lasso

- L1, katsayı normunu sınırlayarak ve bazı katsayı değerlerini 0'a sabitleyerek çoklu bağlantı(multicollinearity) sorununu çözebilir. Çoklu bağlantı, iki bağımsız değişkenin yüksek korelasyonlu olmasıdır
- Eğer gözlem sayınızdan (N) çok feature'ınız varsa, L1, en fazla N kadar katsayı tutar
- L1 bazen bir feature selection yöntemi olarak kullanılır. Kullanabileceğiniz özelliklerin sayısı konusunda bir tür sabit sınırınız varsa istediğiniz sayıda sıfır olmayan(non-zero) feature'a ulaşmak için L1 regülarizasyonu kulanmayı deneyebilirsiniz

L2 Ridge

- L2, katsayı normunu sınırlayarak ve tüm değişkenleri koruyarak çoklu bağlantı sorununu çözebilir
- Gözlem sayınızdan (N) çok feature'ınız olduğu regresyon problemini çözmenin "klasik" yöntemidir
- L2, gözlemlerden daha fazla özellik olsa bile her özellik için bir katsayı tahmin edebilir

Hiper Parametre Tanımları

Learning Rate (α): Gradient Descent'in çalışma sırasında hatanın optimum noktaya yakınsama hızını belirler. (https://developers.google.com/machine-learning/crash-course/fitter/graph)

Theta (θ): Feature değerimizin ağırlığıdır.

Regularization Term (λ): Ağırlığın ne kadar cezalandırılacağını belirleyen ceza terimi veya düzenleme parametresidir. Eğer çok fazla verilirse ağırlıklarımız 0'a yakın olur. Bu da modelimizin overfit olmasını engeller ama aynı zamanda oluşturduğu model çok verimli olmaz.

Repeat {

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

$$\theta_j := \theta_j - \alpha \left[\left(\frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} \right) + \frac{\lambda}{m} \theta_j^2 \right]$$

}

Colab Zamanı!