W6 - Promising Avenues Practical

→ Diyelim ki predicting housing prices örneğini yapıyoruz. Regularized linear regression uyguluyoruz:

- $J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h(x^{(i)}-y^{(i)})^2 + \lambda\sum_{j=1}^{m}\theta_j^2\right]$

→ Diyelim ki hipotezini new set of houses üzerinde test ediyorsun, ama bir bakıyorsun ki çok büyük hatalar veriyor. Ne yapacağız? Learning algorithm'i nasıl geliştireceğiz? Bazı yollar şunlar:

- Get more training examples (her zaman işe yaramaz)
- Try smaller sets of features ($x_1, x_2 \dots x_n$ çok fazla ise overfitting olabilir. Aralarından seçilebilir.)
- Try getting additional features (belki eldeki features yetersiz)
- Try adding polynomial features ($x_1^2, x_2^2, x_1 \cdot x_2$, etc.)
- Try decreasing λ
- Try increasing λ

- Peki hangisini yapmamız gerektiğini nasıl bileceğiz? Çoğu insan bunların arasından rastgele seçim yapıyor

- Fortunately, there is a technique (simple one) that can let you very quickly rule (elemek) out half of the things on this list as being potentially promising things to pursue

- Bu tekniği uygulayarak bu seçeneklerin bir çoğunu eleyebiliriz.

İlerleyen videolarda bu tekniklerden bahsedilecek, they are called as:

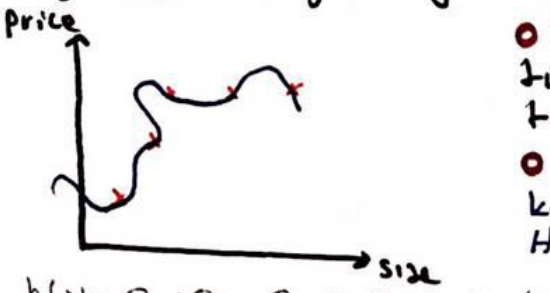
Machine Learning Diagnostics

Diagnostic: A test that you can run to gain insight what is/isn't working with a learning algorithm, and gain guidance as to how best to improve its performance.

Diagnostics can take time to implement, but doing so can be a very good use of your time.

W6 - Evaluating a Hypothesis

→ Bu videoda eğitilmiş hipotezimizi değerlendirmeyi göreceğiz. İlerleyen videolarda underfitting ve overfitting'i nasıl önleyebileceğimizi de göreceğiz.



$h(x) = \Theta_0 + \Theta_1 x + \Theta_2 x^2 + \Theta_3 x^3 + \Theta_4 x^4$

- Bizim şimdiye kadar odaklandığımız şey cost function'ı minimize eden Θ'lar ile hipotezi oluşturmaktı
- Şunu da biliyoruz ki Low Training Error demek kesinlikle iyi bir hipotez oluşturduk demek değil. Hipotez overfitting yapabilir.
- Fails to generalize to new examples not in training set.

Hipotezimin overfit edip etmediğini nasıl bileceğim? Yukarıdaki örnekte 2 feature olduğu için plot ile görebiliriz ama daha karmaşık problemlerde bunu visualize etmek mümkün değil.

→ O zaman hipotezi değerlendirmek için bize başka bir yol gerek.

Standard way to Evaluate a Learned Hypothesis is as follows:

Size	Price
2104	400
1600	330
2400	369
1416	232
3000	540
1985	300
1534	315
1427	199
1380	212
1494	243

70% Training Set — 30% Test Set

- Yandaki gibi bir training set olsun. Evaluation of hypothesis için bunu ikiye bölüyoruz.

70% → $(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})$

30% → $(x^{(1)}_{test}, y^{(1)}_{test}), \dots, (x^{(m_{test})}_{test}, y^{(m_{test})}_{test})$

- m_{test} = # of test examples

⚠ Eğer data ordered ise 70% - 30% split'i random yapmamız faydamıza olur!

Training/testing procedure for Linear Regression

- Learn parameter Θ from training data (minimizing training error $J(\Theta)$) → $J(\Theta)$ 70%'lik data kullanılarak oluşturuluyor!
- Compute test set error: (Eğitilmiş parametre kullanılıp test yapılıyor.)

$$J_{test}(\Theta) = \frac{1}{2m_{test}} \sum_{i=1}^{m_{test}} \left(h(x^{(i)}_{test}) - y^{(i)}_{test}\right)^2$$

Burada yapılan: Training set için $J(\Theta)$'yı minimize edilen yeni bir set için elde edilen en iyi Θ ile deneyip bir test cost'u $J_{test}(\Theta)$ olmaktır.

- Logistic Regression ile Classification Yapıyor Olsaydık:
  - Learn parameter Θ from training data
  - Compute test set Error:

$$J_{test}(\Theta) = -\frac{1}{m_{test}} \sum_{i=1}^{m_{test}} y^{(i)}_{test} \log h(x^{(i)}_{test}) + (1 - y^{(i)}_{test}) \log h(x^{(i)}_{test})$$

W6 - Evaluating a Hypothesis II

↳ Eğitilen parametrelerin hatasını anlayabilmek için $J_{test}(\Theta)$ yi test training ex.s kullanarak hesaplamak mantıklı bunun yanında alternatif yöntemler de kullanılabilir:

Misclassification Error (0/1 misclassification error):

$$err(h(x), y) = \begin{cases} 1 \text{ if } h(x) \geq 0.5, y = 0 \\ \quad \text{or if } h(x) < 0.5, y = 1 \\ 0 \quad \text{otherwise} \end{cases}$$

$err(h(x), y)$ bir tahminin hatasını temsil ediyor eğer tahmin ($h(x) \geq 0.5$) 1 ise ama gerçek sonuç 0 ise error 1 oluyor veya tahmin 0 ama gerçek sonuç 1 ise.

$$\text{Test error} = \frac{1}{m_{test}} \sum_{i=1}^{m_{test}} err(h(x_{test}^{(i)}, y^{(i)}))$$

Yani sonuçla hipotesin yanlış bildiklerinin sayısının tüm test examples'a oranı!

↳ Bunu sonra W4, W5 için pratikte nasıl kullanmışlık.

İlerleyen videolarda bu öğrenilenleri de kullanarak nasıl feature seçileceği, degree of polynomial'a ise with learning algorithm, nasıl regularization parameter λ seçileceğini göreceğiz...

W6 - Model Selection and Training / Validation / Test Sets

How to decide what degree of polynomial to fit a data set? What features to include? How to choose regularization parameter λ? These are called model selection problems.

→ Daha önce de gördük ki, just because the trained hypothesis fits the training set very well doesn't mean it is a good hypothesis çünkü overfitting olabilir.

- More generally, this is why training set's error ($J(\theta)$ for trained θ) is not a good predictor for how well the hypothesis do in new examples.
- Yani hipotezin eğitildiği setteki hatası genelde actual generalization error'dan daha düşüktür. Yeni tahminlerde daha başarısız demek.

Let's consider the Model Selection Problem

Let's say you're trying to choose what degree polynomial to fit to data.

1.) $h(x) = \underset{d=1}{\theta_0 + \theta_1 x}$ 2.) $h(x) = \underset{d=2}{\theta_0 + \theta_1 x + \theta_2 x^2}$ ... 10.) $h(x) = \underset{d=10}{\cdots}$

- θ parametresinin yanında bir parametreyi daha bulmaya çalışıyormuşsun gibi düşün: d = degree of polynomial.

Let's say: Yapmak istediğimiz modeli seçip, modeli eğitmek ve eğitilen modelin yeni örneklere ne kadar iyi oturduğunu tahmin etmek istiyoruz. Bunun için ne yapabiliriz?

$d=1, d=2, \ldots d=10$ olan hipotezler için aynı aynı training set ile eğitimi yaparım hem bini için aynı bir θ elde ederim. $\theta^{(1)}, \theta^{(2)} \ldots$ şeklinde. Daha sonra hangisinin daha iyi olduğunu anlamak için Test Seti kullanarak hepsinin performansını test ederim $J_{test}(\theta^{(1)}), \ldots J_{test}(\theta^{(10)})$ hesaplanır. En iyi olanı seçerim!

Mesela $J_{test}(\theta^{(5)})$ en düşük diyelim o zaman $d=5$ seçtim. Şimdi bu son hipotezin generalized error'unu tahmin etmek istiyorum, nasıl yapacağım? Eğer test seti kullanırsak saçma olacak çünkü $d=5$ parametresini zaten test set'e göre seçtik. Yeni training ex.s için iyi bir tahmin yapamıyor.

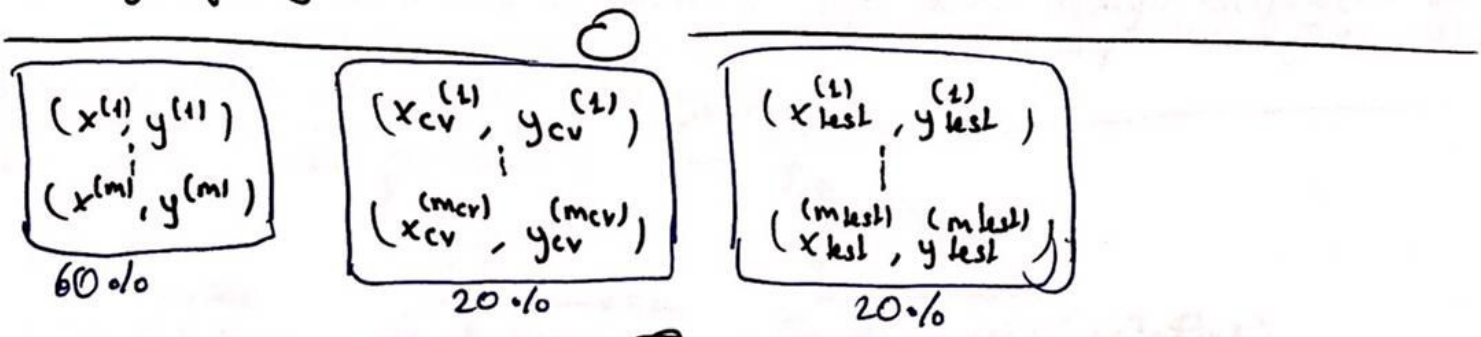
Bu yüzden training seti 2 yerine 3'e bölmeliyiz

Training Set / Cross Validation Set / Test Set

W5 - Model Selection Training / Validation / Test sets II

Sonuçta Training Set 3 parta bölünecek %60 - %20 - %20

1. part: Training Set, Θ ları eğitmek için kullanılacak
2. part: Cross Validation Set, bu set ile Θ'lar test edilecek ve sonucunda d parametresine karar verilecek.
3. part: Test Set, bu set ile karar verilen sonuç hipotezinin (Θ ve d'den oluşuyor) yeni training ex.s için nasıl bir performans sergileyeceği simüle edilmeye çalışılıyor



Train / Test / Validation Errors

Training Error:

$$J_{train}(\Theta) = \frac{1}{2m} \sum_{i=1}^{m} (h(x^{(i)}) - y^{(i)})^2$$

Cross Validation Error:

$$J_{cv}(\Theta) = \frac{1}{2m_{cv}} \sum_{i=1}^{m_{cv}} (h(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$

Test Error

$$J_{test}(\Theta) = \frac{1}{2m_{test}} \sum_{i=1}^{m_{test}} (h_\Theta(x_{test}^{(i)}) - y_{test}^{(i)})^2$$

Sonuçta şu yol izlenecek:

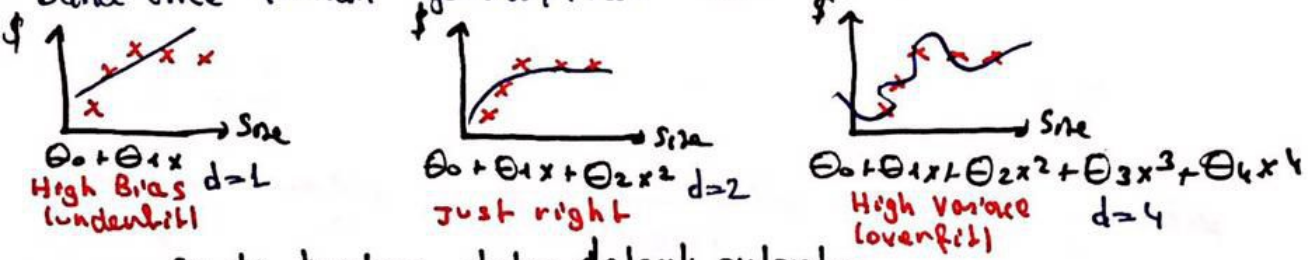
- Farklı hipotezler için Training Set kullanılarak Θ'lar bulunacak!
  1) $h(x) = \Theta_0 + \Theta_1 x \longrightarrow \min_\Theta J(\Theta) \rightarrow \Theta^{(1)} \rightarrow J_{cv}(\Theta^{(1)})$
  2) $h(x) = \Theta_0 + \Theta_1 x + \Theta_2 x^2 \longrightarrow \Theta^{(2)} \rightarrow J_{cv}(\Theta^{(2)})$
  ⋮
  10) $h(x) = \Theta_0 + \cdots + \Theta_{10} x^{10} \longrightarrow \Theta^{(10)} \rightarrow J_{cv}(\Theta^{(10)})$
- Her hipotez için J_{cv} hesaplanacak (CV set kullanılarak) min J_{cv} seç!
- Sonuçta seçilen hipotezin Θ ve d belli. generalized error estimation için $J_{test}(\Theta)$ hesaplanır. (Test Set kullanılarak)

~W6 - Diagnosing Bias vs Variance ~

→ If you run a learning algorithm and it doesn't do as well as you are hoping, almost EVERYTIME, it will be because you have either high bias problem <or> high variance problem. In other words either overfitting problem <or> underfitting problem

- Bu durumda hangisinin olduğunu veya ikisinden de az az mı olduğunu bilmek çok işe yarar çünkü bunlar algoritmayı nasıl geliştirebileceğimiz hakkında güçlü işaretçilerdir.
- Bu videoda overfitting ve underfitting'e daha detaylı değinecek ve bir learning algoritmasının ikisinden birine sahip olup olmadığını nasıl değerlendirebileceğimizi göreceğiz.

Daha önce şunları görmüştük:



$\theta_0 + \theta_1 x$ — High Bias (underfit) — d=1

$\theta_0 + \theta_1 x + \theta_2 x^2$ — Just right — d=2

$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$ — High Variance (overfit) — d=4

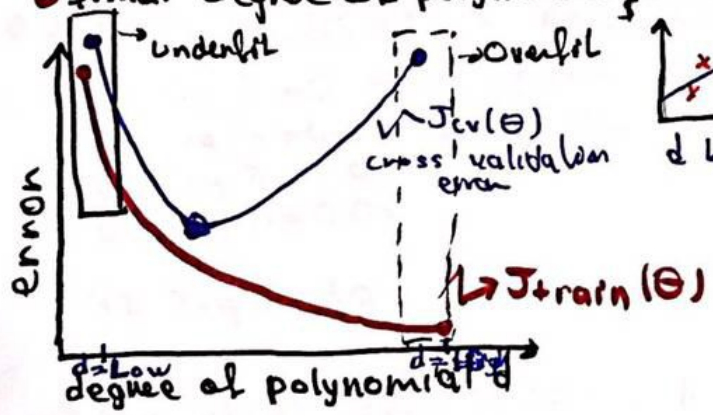
Şimdi bunları daha detaylı anlayalım.

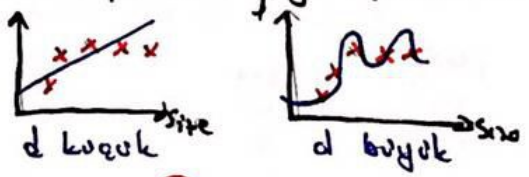
Bias / Variance: Let's assume Training Error and Validation Error is defined as:

Training Error: $J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h(x^{(i)}) - y^{(i)})^2$

Cross Validation Error: $J_{cv}(\theta) = \frac{1}{2m_{cv}} \sum_{i=1}^{m_{cv}} (h(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$

- Şimdi degree of polynomial'a karşı error grafiklerini çizelim:





d küçükken underfitting olur ve yüksek bir $J_{train}(\theta)$ bulunur aksine d yüksek iken ise $J_{train}(\theta)$ belki 0'a kadar düşebilir.

Ayrıca yeni datasetler üzerinde yapılan testlerde d'nin küçük olması kadar büyük olması da yüksek $J_{cv}(\theta)$ hatası doğurur!

Sonuçta assume your LA performing less well than you were hoping. ($J_{cv}(\theta)$ or $J_{test}(\theta)$ is high) Is it a bias problem or variance problem?

Bias (underfit)

$J_{train}(\theta)$ will be high

$J_{cv}(\theta) \approx J_{train}(\theta)$

Clue of that the problem MAY BE underfitting

High Variance (Overfit)

$J_{train}(\theta)$ will be low

$J_{cv}(\theta) \gg J_{train}(\theta)$

Clue of that problem MAY BE Overfitting

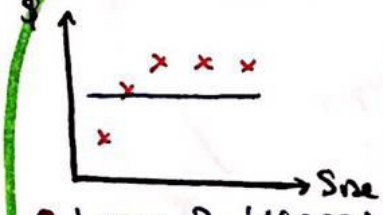
→ Regularization'ın overfitting'i önleyebileceğini görmüştük. How does it affect the bias and variances of a learning algorithm?

→ Suppose we're fitting a high order polynomial like below but to prevent overfitting we're gonna use regularization. By regularization we are trying to keep values of parameters small.

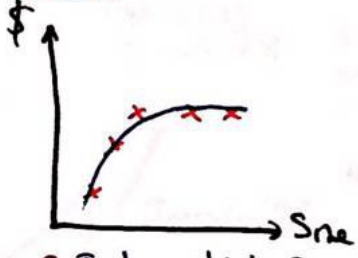
Model: $h(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m}\sum_{j=1}^{m}\theta_j^2$$

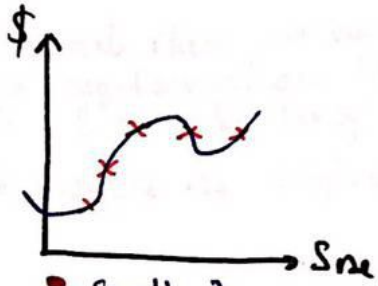
Let's consider 3 cases:



- Large λ (10000)
- High Bias (Underfit)
- $\theta_1 \approx 0$, $\theta_2 \approx 0$, ...
- $h(x) \approx \theta_0$



- Intermediate λ
- Just Right



- Small λ
- High Variance (overfit)
- or $\lambda = 0$

So how can we choose a good value for the λ?

→ Assume this is our model and $J(\theta)$ is defined as above:

- Denemek istediğimiz λ değerlerini alalım mesela:

1. Try $\lambda = 0$ → $\min_\theta J(\theta)$ gives $\theta^{(1)}$ → $J_{cv}(\theta^{(1)})$
2. Try $\lambda = 0.01$ → $\min_\theta J(\theta)$ gives $\theta^{(2)}$ → $J_{cv}(\theta^{(2)})$
3. Try $\lambda = 0.02$ → " $\theta^{(3)}$ → $J_{cv}(\theta^{(3)})$
4. Try $\lambda = 0.04$ → " $\theta^{(4)}$ → $J_{cv}(\theta^{(4)})$

⋮

12. Try $\lambda = 10$ → " $\theta^{(12)}$ → $J_{cv}(\theta^{(12)})$

En düşük J_{cv}'si olanı seçersin burada λ=... olsun yani 5. rivadaki

→ Son olarak $J_{test}(\theta^{(5)})$'i bularak $\lambda = 0.08$ için $h(x)$ hipotezinin generalized errorunu bulabiliriz.

Aslında bu 2 sayfa önce yaptığımız işlemle aynı sayılır $J(\theta)$ farklı!

$$J_{train}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2$$

? Bunu burada neden tanımladık anlamadım yukarıda $J(\theta)$ kullandık.

$$J_{cv}(\theta) = \frac{1}{2m_{cv}}\sum_{i=1}^{m_{cv}}(h(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$

$$J_{test}(\theta) = \frac{1}{2m_{test}}\sum_{i=1}^{m_{test}}(h(x_{test}^{(i)}) - y_{test}^{(i)})^2$$

-W6- Learning Curves -

→ Learning Curves diz often a very useful thing to plot if either you wanted to sanity check that your algorithm is working correctly or if you wat to improve the performance of the algorithm.

→ Learning Curves is a very often use tool to diagnose if a particular learning algorithm may be suffering from bias, or a variance problem or a bit of both.

Learning Curves dediğimiz $J_{train}(\theta)$ ve $J_{cv}(\theta)$'nın m'e yani # of training examples'a göre çizilmesi ile oluşur.

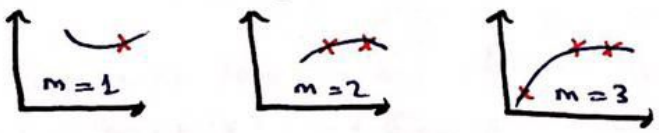
- Bu grafiği çizmek için m'i artificially düşürmek yani normalde Training Set size 100 ise bunu 10 için 20 için vb. çizebiliriz. Training Error ve Validation Error bu azaltılmış yeni set için çizilir

- $J_{train}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2$
- $J_{cv}(\theta) = \frac{1}{2m_{cv}}\sum_{i=1}^{m_{cv}}(h(x^{(i)}) - y_{cv}^{(i)})^2$

Plotting $J_{train}(\theta)$

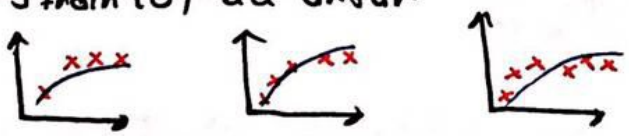
- Let's see what these plots may look like;

- 1) Let's say I have only 1 training ex. and I am fitting a quadratic function $h(x) = \theta_0 + \theta_1 x + \theta_2 x^2$ (Diğerleri için de bu h(x)!)
  - Training Ex sayısı 1, 2 ve 3 iken quadratic hipotez iyi oturur reg. yok ise tam oturur varsa da çok küçük hata.



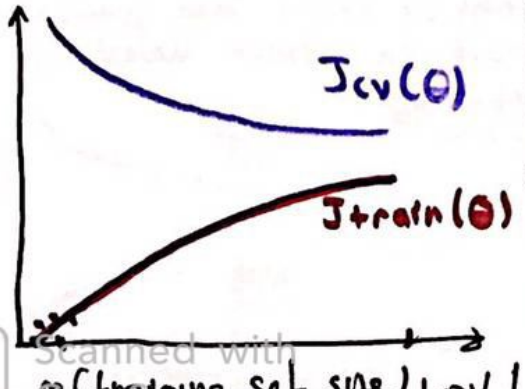
- $J_{train}(\theta) = 0$ if no reg. $J_{train} \cong 0$ if reg.

- 2) $m >= 4$ için quadratic function data set'e artık perfectly oturmaz. m arttıkça tüm datalara ~~tüm~~ oturması zorlaşır!
  - Kısaca, training set boyutu arttıkça average training error $J_{train}(\theta)$ da artar



> **Intuition:** when m is small it is easy to fit every single training ex.s perfectly. When m is getting larger it becomes harder

Jcv'yi kullanmış bir validation set için hesaplanıyor. Training Set yani m değişiyor sadece. Sadece eğitim ediliyor.



Val-set ve Test Set size önemli değil!

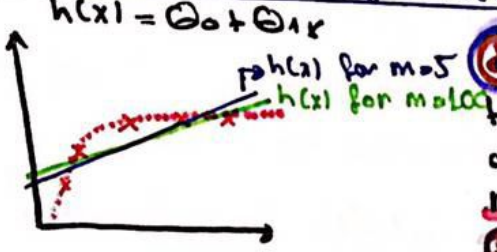
How about the $J_{cv}(\theta)$?

- Jcv is my error on cross validation set that I haven't seen.
- when I have very small training set I am not gonna generalize well.
- when I get a larger training set I am starting to get a hypothesis maybe fit to the data better
- The more data you have, the better you do at generalizing to new examples.

-W6- Learning Curves -

Let's look at what the learning curves may look like if we have either HIGH BIAS or HIGH VARIANCE problems:

1) Suppose our Hypothesis have HIGH BIAS: Plot $J_{cv}(\theta)$



Lets think what would happen if we were to increase the training set size. So instead of five ex. let's say we have 100. (Pembe noktalar)

Buyuk bir degisim olmuyor zaten belli bir data sayisinda sonra best hypothesis'e cok yaklasilir. m=1, 2 vb. gibi kucuk m'ler icin hata yuksek olabilir.

How about $J_{train}(\theta)$: It will start small and for HIGH BIAS case it will end up close to the $J_{cv}(\theta)$

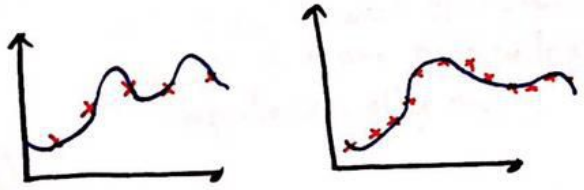
Because you have so few parameters and so much data, at least when m is large the performance on the training set and the cross validation set will be very similar.

The problem of high bias is reflected in the fact that both the J_{cv} and J_{train} are high (relatively)

Also Note That: If a learning algorithm is suffering from high bias, getting more training data will not (by itself) help much. Yani ile yesil lineler asagi yukari ayni!

2) Suppose our Hypothesis have HIGH VARIANCE:

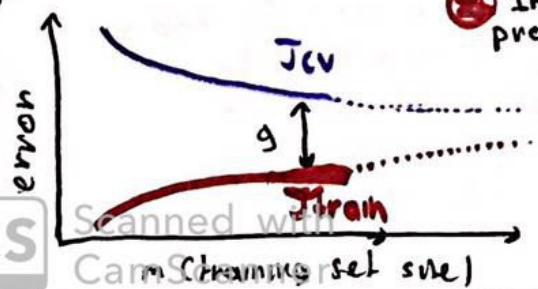
Assume $h(x) = \theta_0 + \theta_1 x + \dots + \theta_{100} \cdot x^{100}$ and λ = small value



- m kucuk iken hipotez gayet rahat oturtulabilir zaten d cok buyuk o zaman $J_{train}(\theta)$ kucuk olur.
- m buyuk iken hala hipotez gurel bir sekilde oturtulabilir cunki d buyuk ama m kucuk oldugu kadar rahat olmaz. Sonuçta biraz daha buyuk bir $J_{train}(\theta)$!

How about the $J_{cv}(\theta)$: In high variance setting, a hypothesis is overfitting and cross validation error will remain high even if we get moderate number of training examples.

Indicative diagnostic that we have a high variance problem is the large gap, g between J_{cv} and J_{train}



Adding more training data is likely to help for high variance problem.

Curveler her zaman boyle net olmasa amen yine de high bias mi high variance mi problem oldugunu anlayabiliriz. genelde

1. Sayfalı Örneğe Geri Dönelim: Suppose you have implemented regularized linear regression to predict housing prices. However, when you test your hypothesis in a new set of houses, you find that it makes unacceptably large errors in its prediction. What should you try next?

- Get more training examples → It fixes high variance
  Eğer high bias problem varsa bir işe yaramaz.
- Try smaller sets of features → It fixes high variance
  High bias için useless
- Try getting additional features → Usually a solution for fixing a high bias problem.
  Current hypothesis is too simple so we wanna get additional features to make our hypothesis better able to fit the training set
- Try adding polynomial features → Similarly fix a high bias problem.
  High var. problem için çok işlersel olmaz.
- Try decreasing λ → Fixes high bias
- Try increasing λ → Fixes high variance

Let's take everything we have learned and relate it back to Neural Networks:

1) "Small" Neural Networks: Relatively few hidden units, maybe only one hidden layer. • This kind of networks have relatively few parameters • More prone to underfitting • Small advantage is that they are computationally cheaper.

2) "Large" Neural Networks: Either more hidden units or more hidden layers • More parameters • More prone to overfitting • Computationally more expensive.

Genelde using a larger NN by using regularization is more effective than using a small NN.

Peki number of hidden layers kaç tane olmalı?

- Using a single hidden layer is reasonable default.
- But if you want to choose the # of hidden layers we can try → Find a training, cross-validation and test sets and try training NNs for different number of hidden layers and see which of those performs best on the cross validation set.

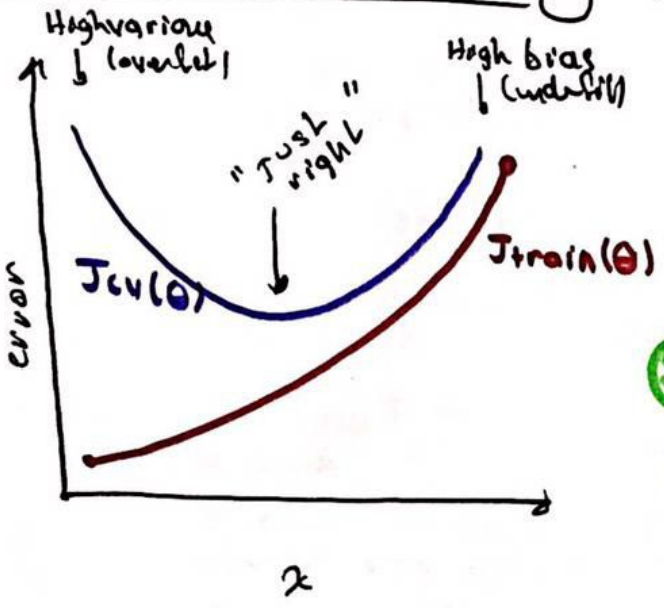
Let's look at how Cross Validation Error J_{cv}, and Training Error J_{train} vary as we vary the λ.

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$ → This is our original cost function

Plot them against λ

$$J_{train}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2$$ → Bunu yeni tanımladık grafik ve karşılaştırma için kullanılacak.

$$J_{cv}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$



- λ küçük iken we are not using much regularization then larger risk of overfitting
- λ büyük ise higher risk of underfit

Burada sanırım şöyle kabul ediyorsun λ için θ bulunuyor sonra bulunan θ için $J_{train}(\theta)$ ve $J_{cv}(\theta)$ hesaplanıyor bu grafik o şekilde çiziliyor

- Gerçekte elde edilen curves daha complex olur biraz daha gürültülü olur. Bazı data setleri için gerçekten böyle grafikler görebilirsin
- Bu grafiği elde edip λ'yı ona göre seçmek mantıklı olacaktır.

λ is only used during training. λ is used to train an algorithm to obtain a good θ which will balance the desire for low bias and low variance (no underfit and no overfit). Once training is complete, the training error and cross validation errors can be calculated to understand the quality of θ. These error metrics must include actual outputs and predicted outputs, and not the λ