Coursera

Week 6

Koray Poyraz

Table of Contents

[Evaluating a learning algorithm 2](#_Toc529692959)

[Deciding what to try next 2](#_Toc529692960)

[Machine learning diagnose 2](#_Toc529692961)

[Evaluating a hypothesis 3](#_Toc529692962)

[Training/testing procedure for linear regression 4](#_Toc529692963)

[Model selection and train/validation/test sets 6](#_Toc529692964)

[Bias vs. Variance 9](#_Toc529692965)

[Diagnosing Bias vs. Variance 9](#_Toc529692966)

[Regularization and Bias/Variance 11](#_Toc529692967)

[Learning curves 14](#_Toc529692968)

[Deciding What to Do Next Revisited 17](#_Toc529692969)

[Building a spam classifier 19](#_Toc529692970)

[Prioritizing What to Work On 19](#_Toc529692971)

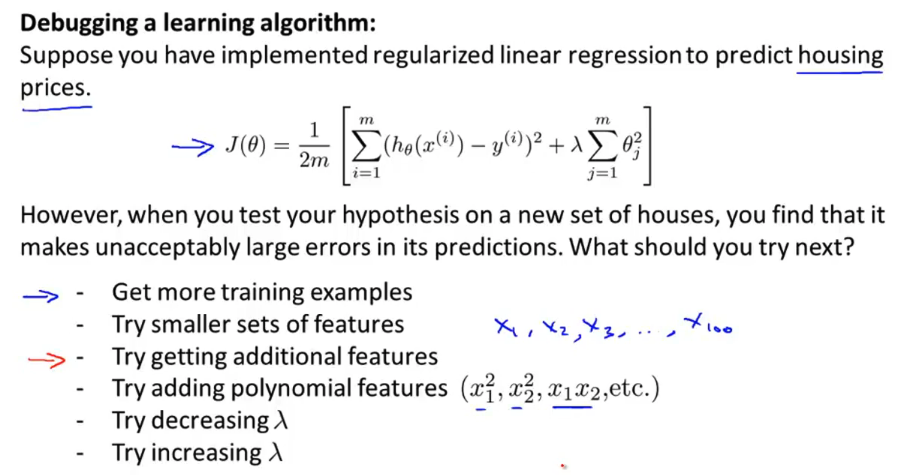
# Evaluating a learning algorithm

## Deciding what to try next

Voorbeeld je hebt een linear regressie geïmplementeerd alleen hij geeft ongewenste errors bij de test set als je wilt voorspellen. Hoe kun je dit probleem oplossen? Door te debuggen! Dus een linear regressie debuggen. Hoe kun je dat doen:

* Meer training voorbeelden krijgen
* Probeer kleine sets van features
* Probeer extra features te krijgen
* Probeer polynomial features toe te voegen
* Lambda waarde verlagen
* Lambda waarde verhogen

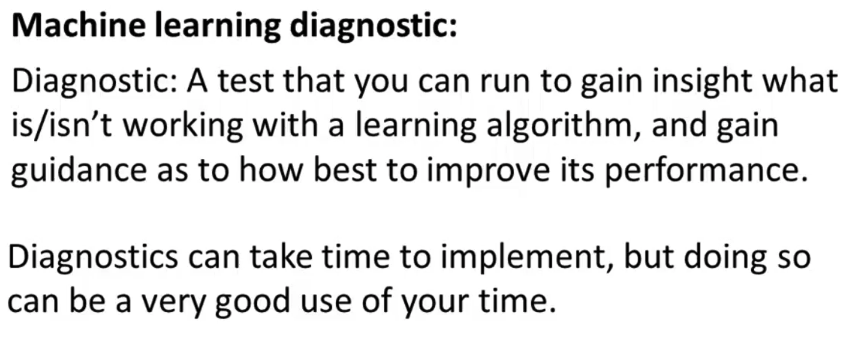
De bovenstaande mogelijkheden zijn dus opties die je kunt nemen om de linear regressie betere voorspellingen te laten doen. Vaak worden deze opties random geselecteerd. Helaas vergt dit veel tijd en komt men pas voorbeeld 6 maanden later erachter dat het niet goed is. Gelukkig is er een vrij eenvoudige techniek waarmee je heel snel de helft van de dingen op deze lijst kunt uitsluiten als potentieel veelbelovende dingen om na te streven. En er is een zeer eenvoudige techniek, die als je loopt, gemakkelijk veel van deze opties kan uitsluiten, en mogelijk bespaart men veel tijd op het nastreven van iets dat gewoon niet gaat werken. Deze technieken worden ook wel machine learning diagnose genoemd.

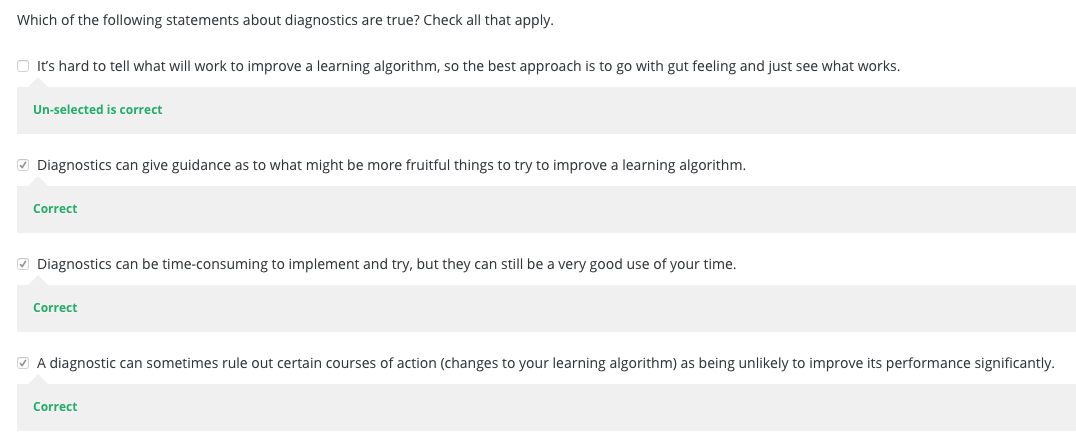


### Machine learning diagnose

Wat is een diagnose?

Is een test die je kunt uitvoeren om inzicht te krijgen in wat al dan niet werkt met een algoritme en die je vaak inzicht geeft in wat veelbelovende dingen zijn om te proberen de algoritmen van een lerend algoritme te verbeteren.



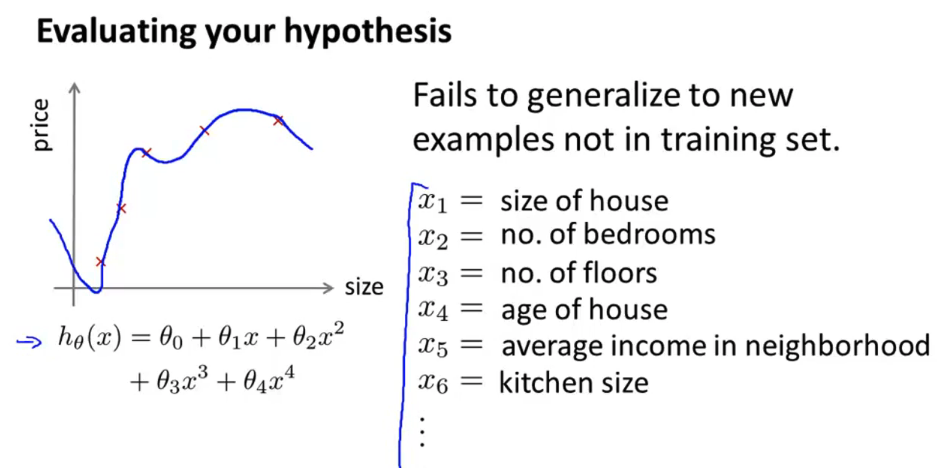


## Evaluating a hypothesis

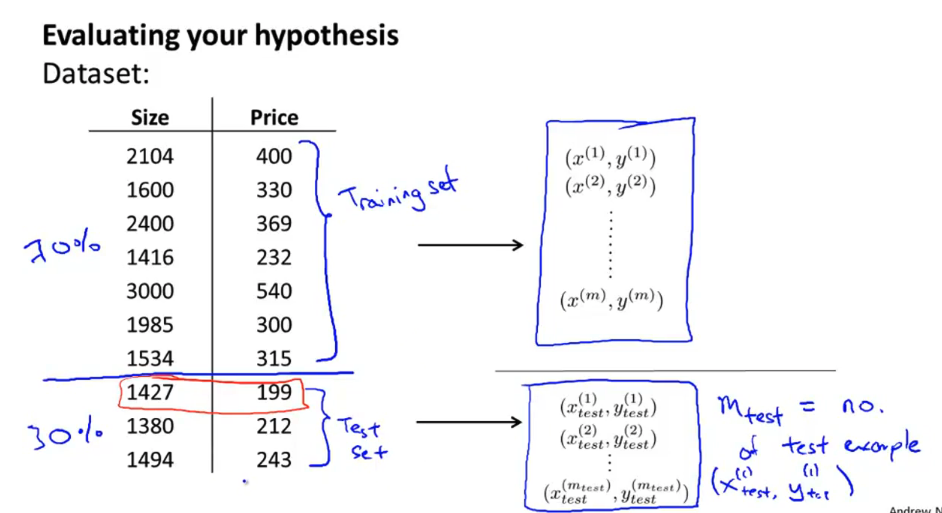
Hier wordt dus de hypothese geëvalueerd die is geleerd door de algoritme. Een hypothese met een lage error waarde hoeft niet per definitie te betekenen dat het een goede hypothese is, denk aan HIGH VARIANCE dus te veel golven.

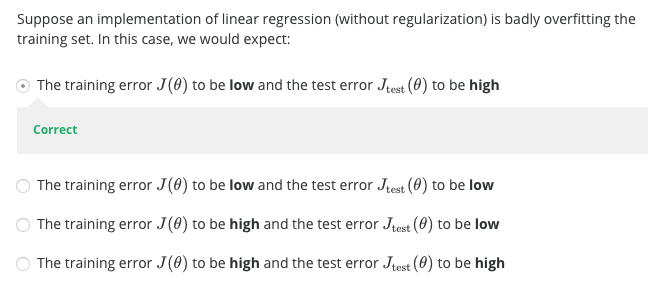
Hoe kun je het beste erachter komen of een hypothese overfit?

Door het te gaan plotten. Maar wanneer er heel veel features zijn dan is het wat moelijker om een hypothese te gaan plotten. Voor hypothese met veel features is er een ander oplossing om te gaan plotten.



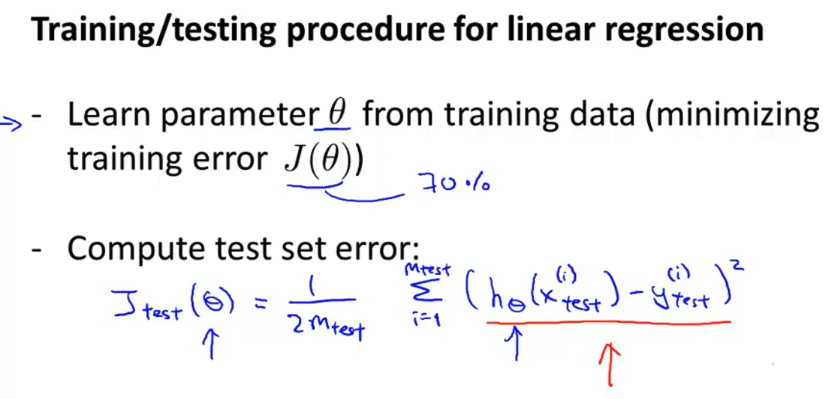
Om een hypothese te kunnen evalueren moet de dataset onderverdeeld worden in Training set 70% en Test set 30%.



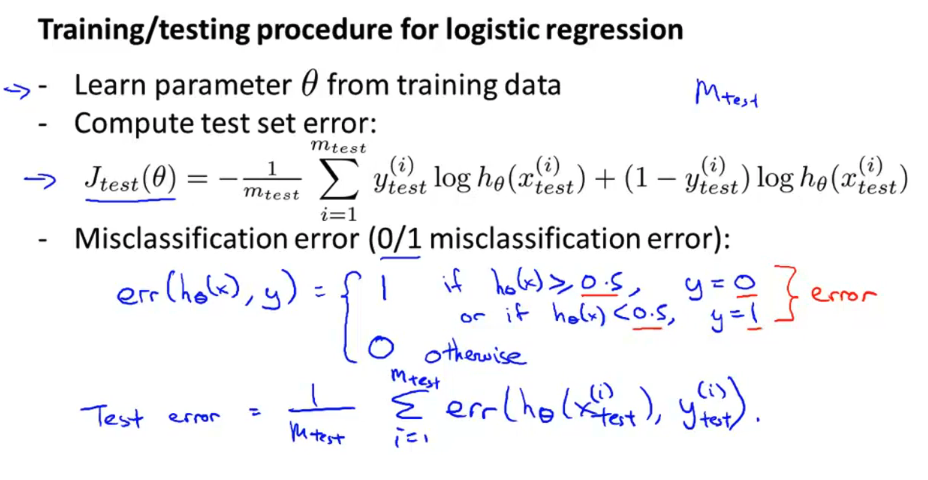


### Training/testing procedure for linear regression

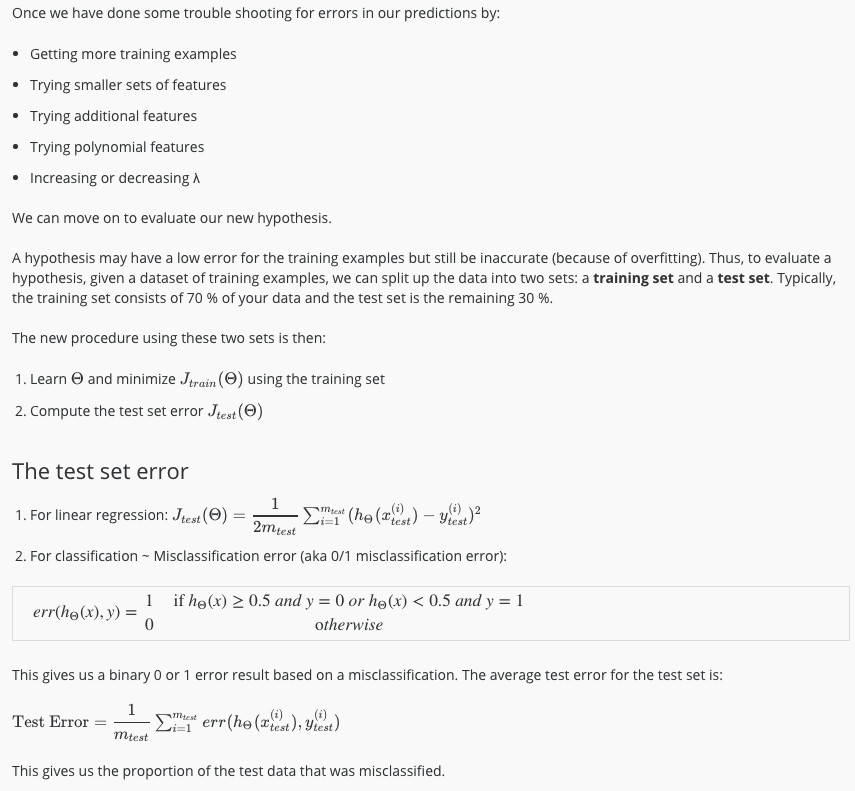
De verlaagde theta waarde van je training set geef je door aan je test set.

Voor logistic regressie is bijna gelijk als bij linear regressie.

Soms is er een alternatieve testset-metriek die misschien makkelijker te interpreteren is, en dat is de misclassification-fout. Het wordt ook wel de zero/one misclassificatie-fout genoemd, waarbij zero/ one aangeeft dat je een goed voorbeeld krijgt of dat je een verkeerd voorbeeld krijgt.



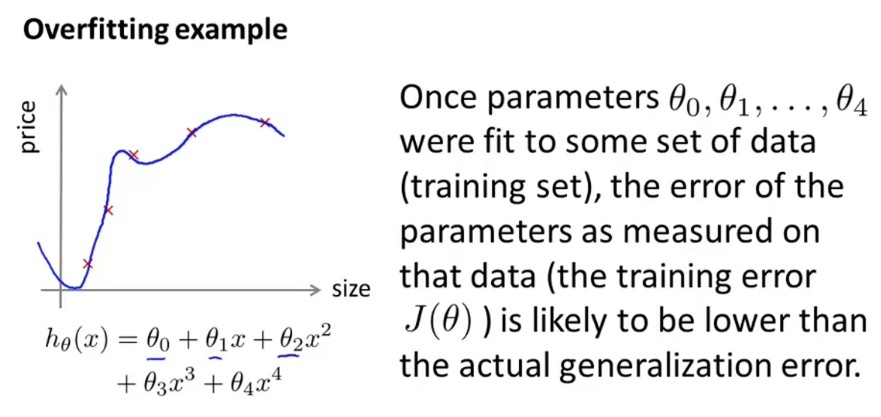
Dit is een standaard techniek voor het evalueren hoe goed een hypothese is geleerd.



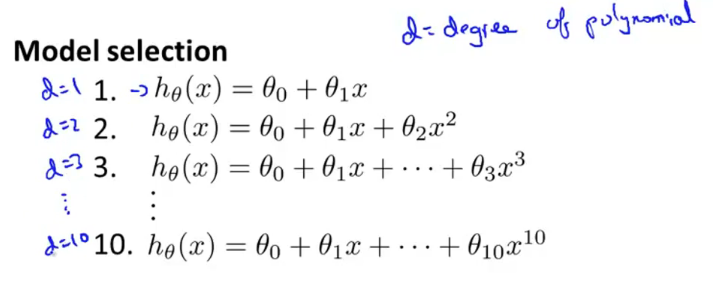
## Model selection and train/validation/test sets

Stel dat je wordt overgelaten om te bepalen in welke mate een polynomial moet worden gefit aan een dataset. Dus welke features je wilt opnemen om een learning algoritme te krijgen. Of stel dat je de regularisatie parameter langer wilt kiezen voor de learning algoritme. Hoe doe je dat? Dit wordt dus gedaan met model selectieproces.

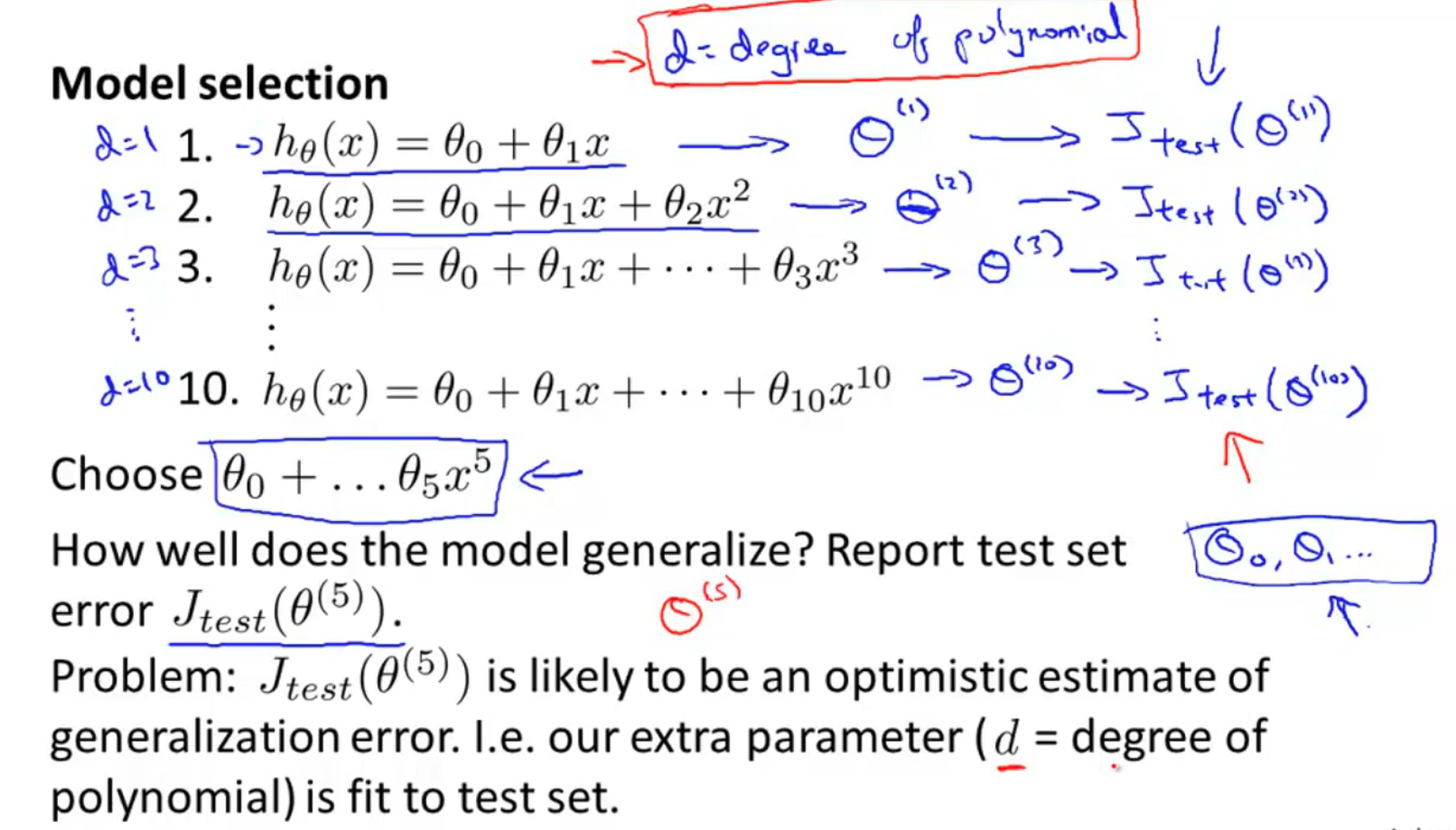
Ter herinnering, een learning algoritme die de dataset goed fit hoeft niet per definitie te betekenen dat de hypothese goed is. Daarom is de trainings error niet een goede voorspeller voor hoe goed een hypothese zal doen op nieuwe example datas.



Bij model selection kijk je eerst naar welke degree van polynomial je wilt gebruiken, dus een linear function, een quadratic function, een cubic function? Helemaal tot de 10e polynomial.

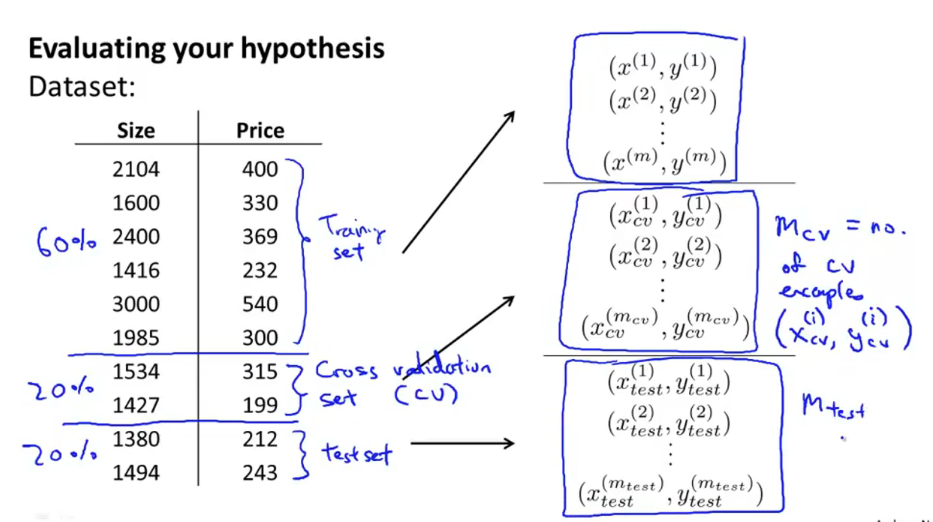


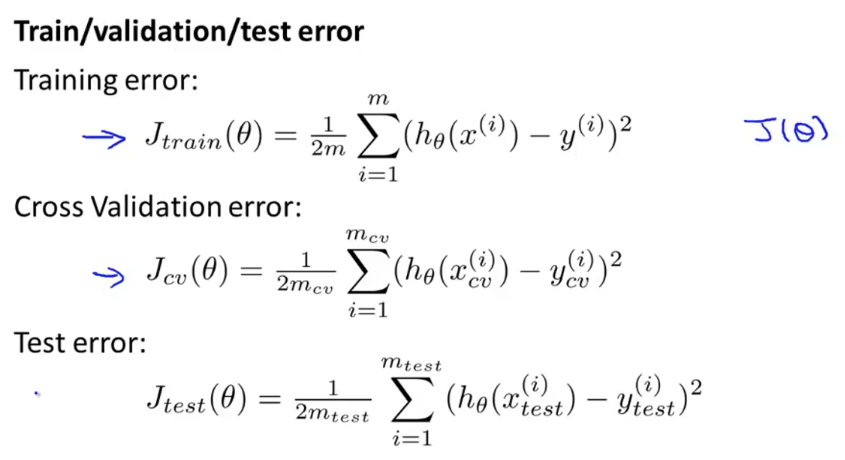
Dus je pakt een model, bijv. nummer 1 dus degree 1 en daarvan minimaliseer je de training error. Dit geeft je vervolgens een parameter vector theta. Dit pas je dan toe bij elke model. Dus elk model fit je bij je trainingsset, die geeft je een parameter vector theta terug. Vervolgens pak je alle parameters en kijk je naar de test set error. Daarna kun je kijken welk model de laagste testset error geeft.



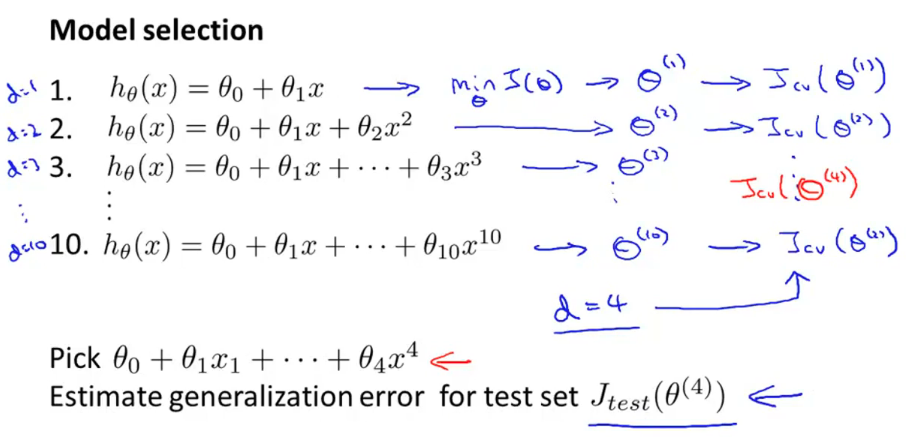
Maar het probleem is, het geeft geen redelijke schatting over hoe goed je hypothese generalized is.

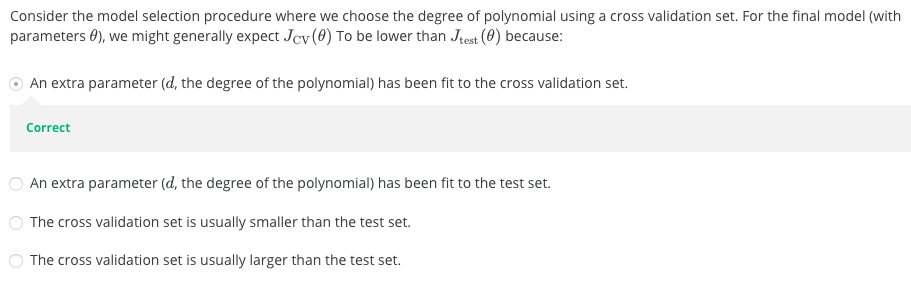
Voor evalueren van de hypothese wordt eigenlijk de techniek splitten in training, cross-validation en test toegepast. Dus niet onderverdelen in training en testset maar training, cross-validation en testset.

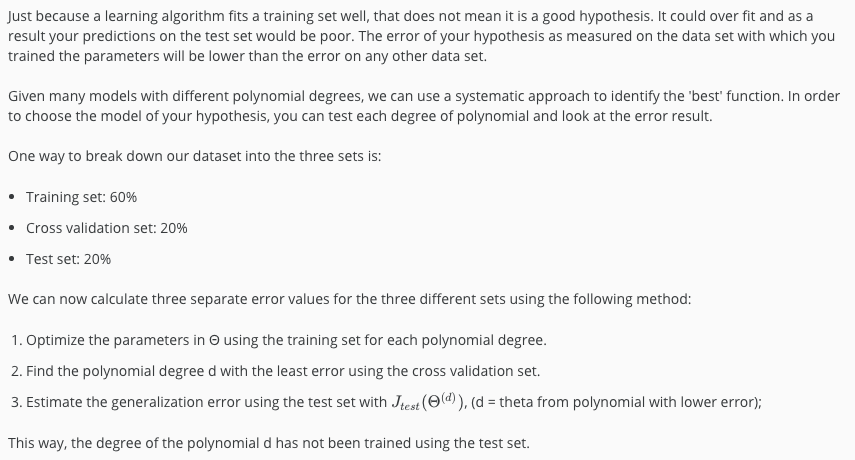




In plaats van een model selecteren aan de hand van de testset wordt aan de hand van cross-validation een model geselecteerd. Dus bij cross-validation model selection ziet het er als volgt uit:







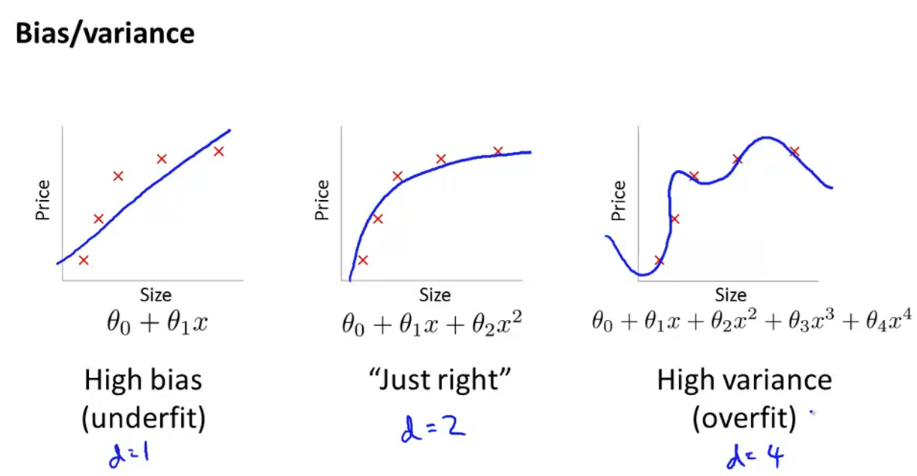
# Bias vs. Variance

## Diagnosing Bias vs. Variance

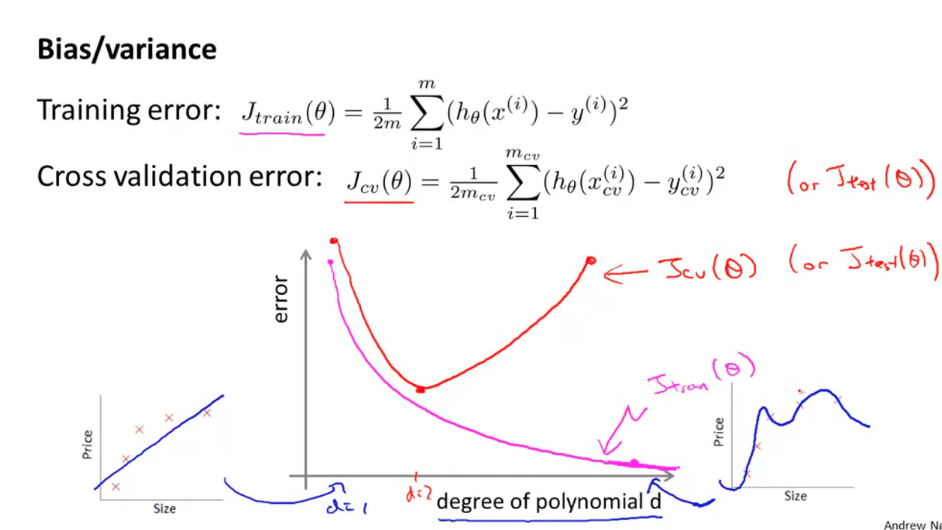
Ter herinnering:

High bias = underfitting = de lijn gaat niet goed door de data heen dus hij fit niet goed

High variance = overfitting = de lijn is in golvende vorm dus hij fit te goed



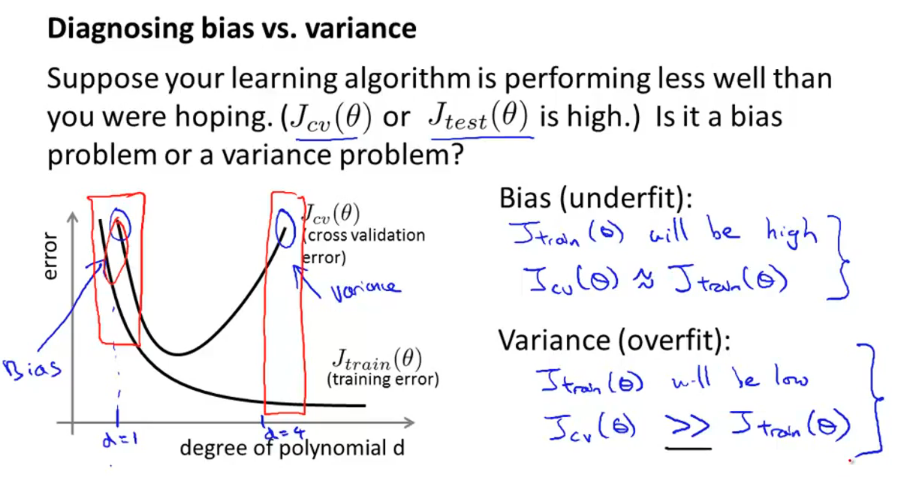
De volgende plot geeft een duidelijker beeld van Bias/Variance:

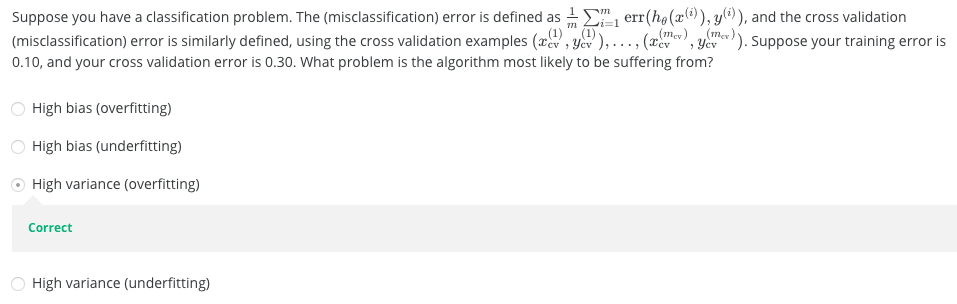


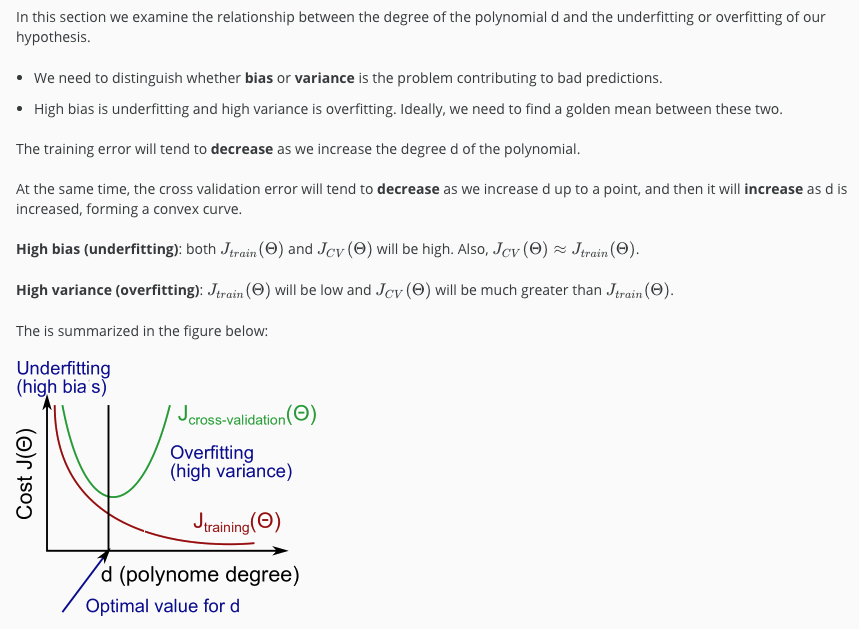
Bij Cross validation error lijn is het laagste punt goed en bij de Trainings error lijn vaak de middelste.

Wanneer weet je of een Cross validation error (HIGH) of test error (HIGH) een Bias of een Variance probleem is?

* Bias
  + Wanneer de Cross validation error HIGH is en de training error HIGH dus de lijnen staan dicht bij elkaar.
* Variance
  + wanneer de Cross validation error HIGH is en de training error LOW dus de lijnen staan ver van elkaar.

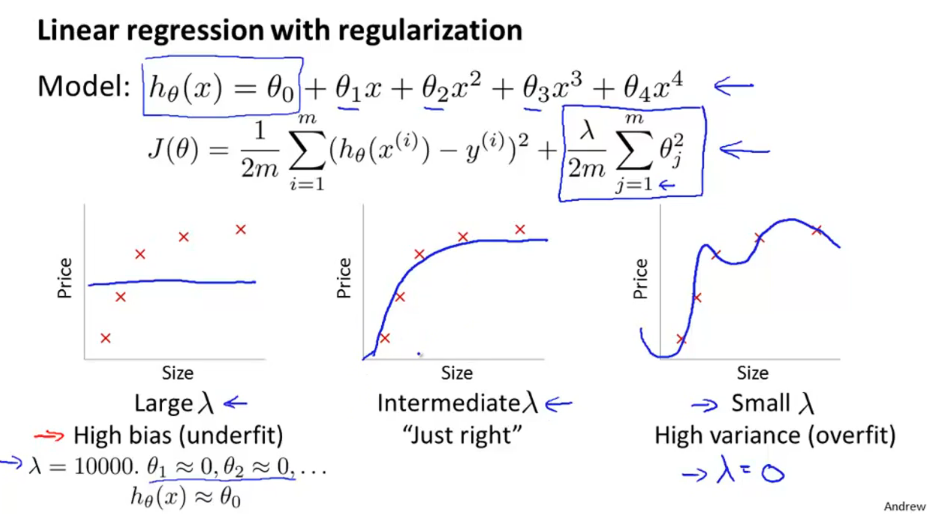


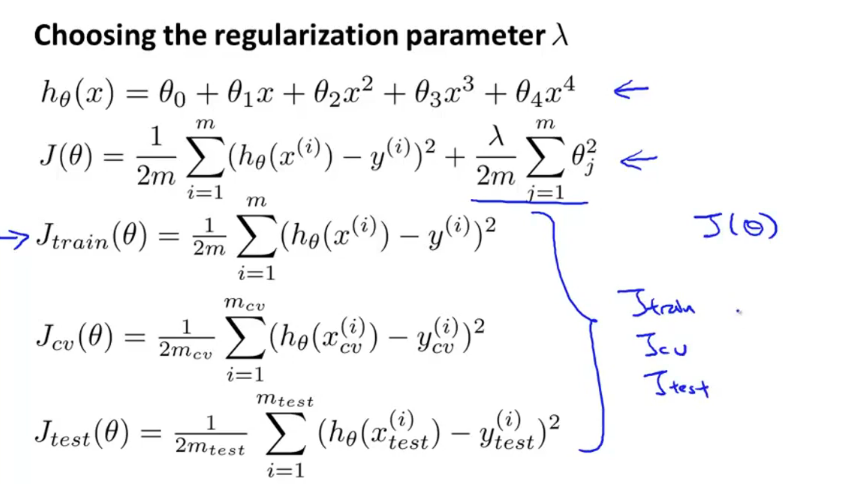




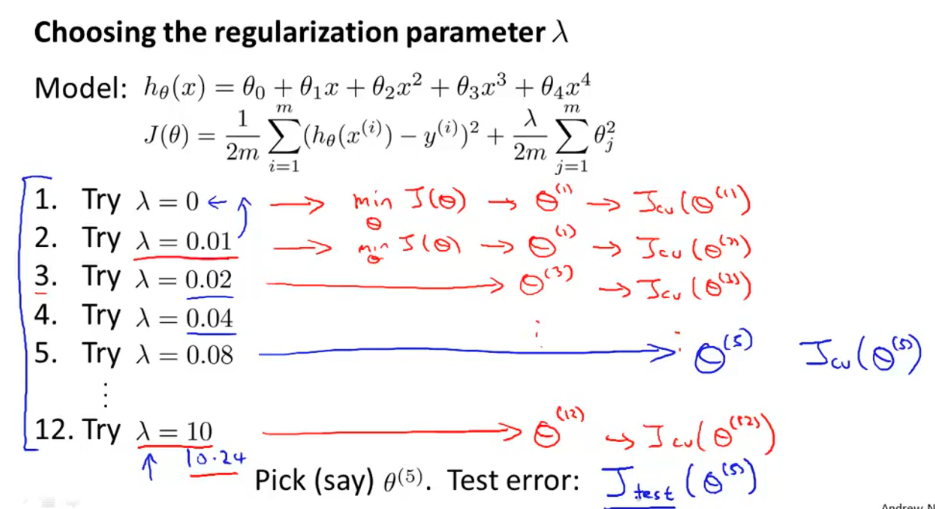
## Regularization and Bias/Variance

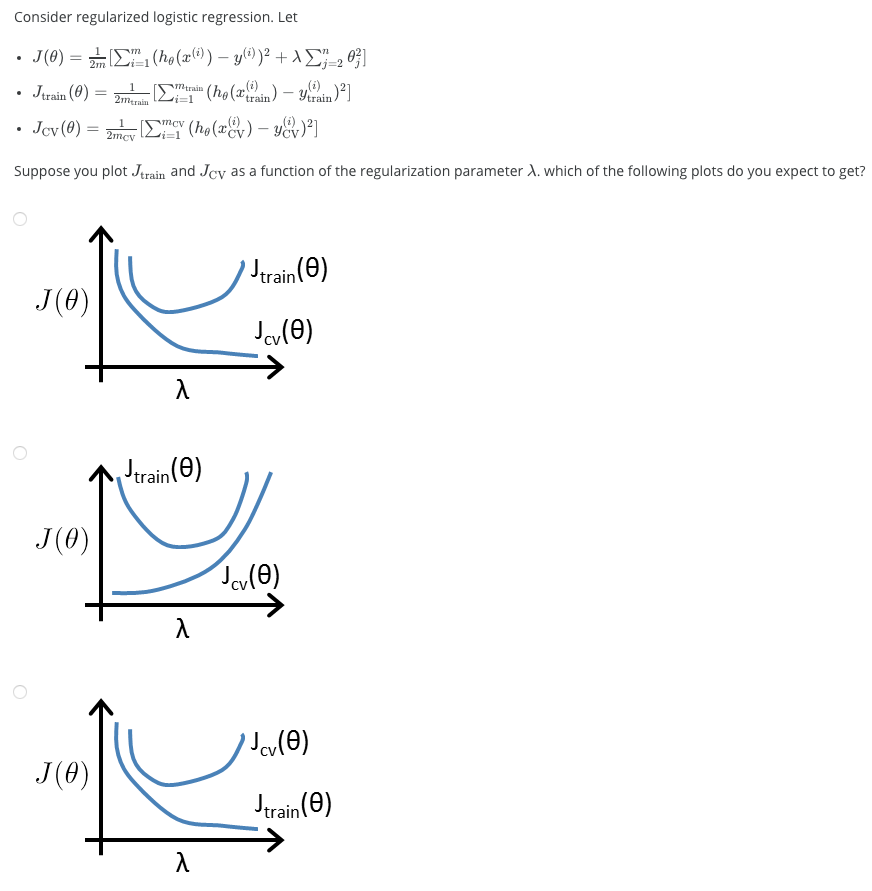
Je hebt gezien hoe regularisatie kan helpen overfitting te voorkomen. Maar hoe beïnvloedt dit de Bias en Variance van een learning algoritme? Hier gaat dit onderdeel over.

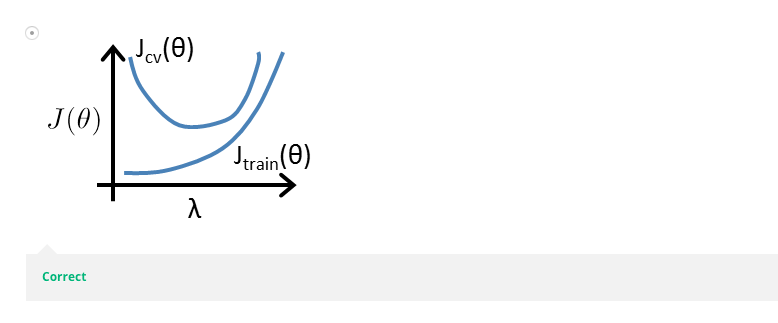




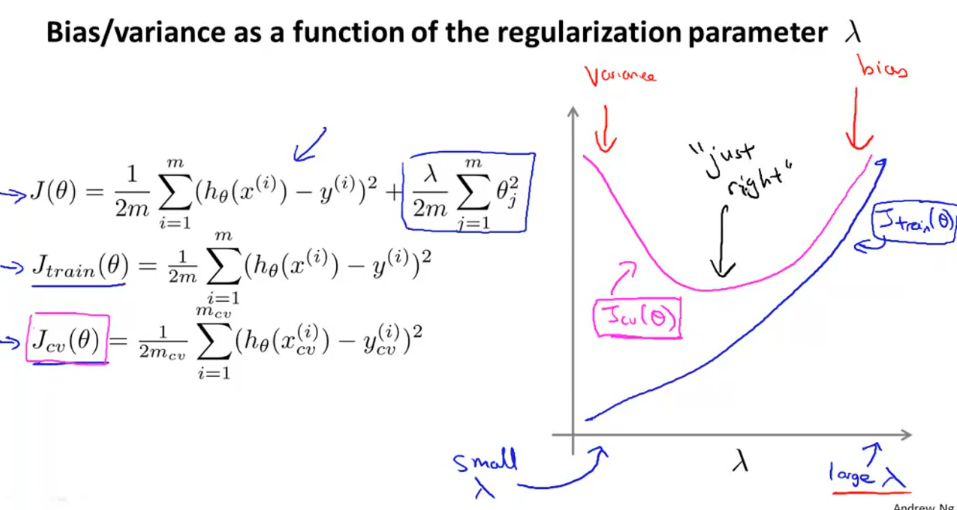
Voor het selecteren van de juiste waarde voor regularisatie voer je dezelfde techniek uit als bij model selection. Dus aantal modellen met verschillende regularisatie waarde uitproberen en de parameters vervolgens meegeven aan de Cross Validation error om naar zijn uitkomst te kijken. Daarna pak je het model met de laagste Cross Validation error. Vervolgens kijk je hoe goed het model doet met de Test error.

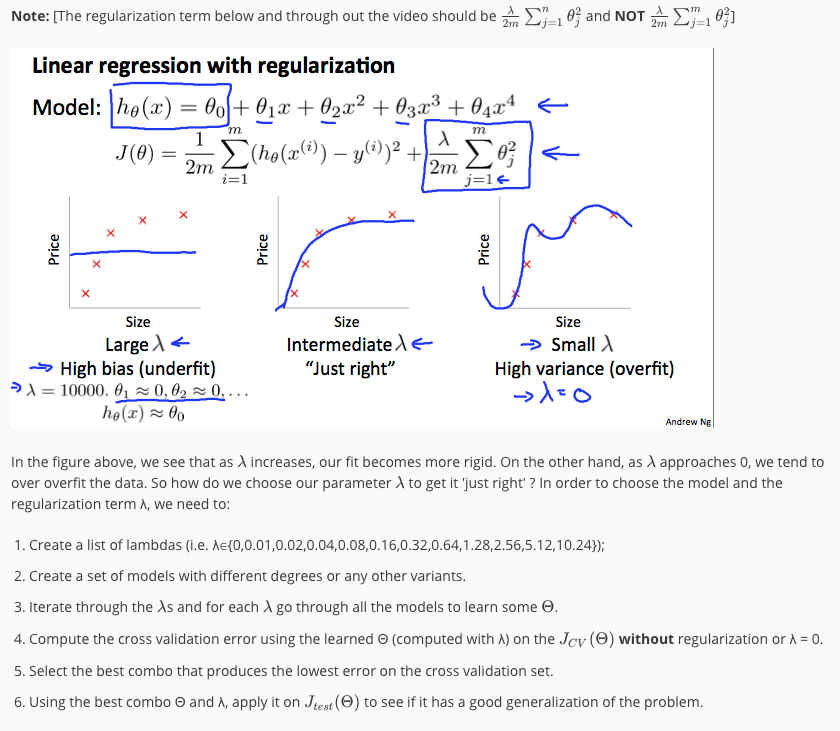






Vergeet bij zulke plaatjes niet naar de waarde van de lambda te kijken! Dus lage lambda waarde is HiGH VARIANCE en hoge lambda waarde is HIGH BIAS!

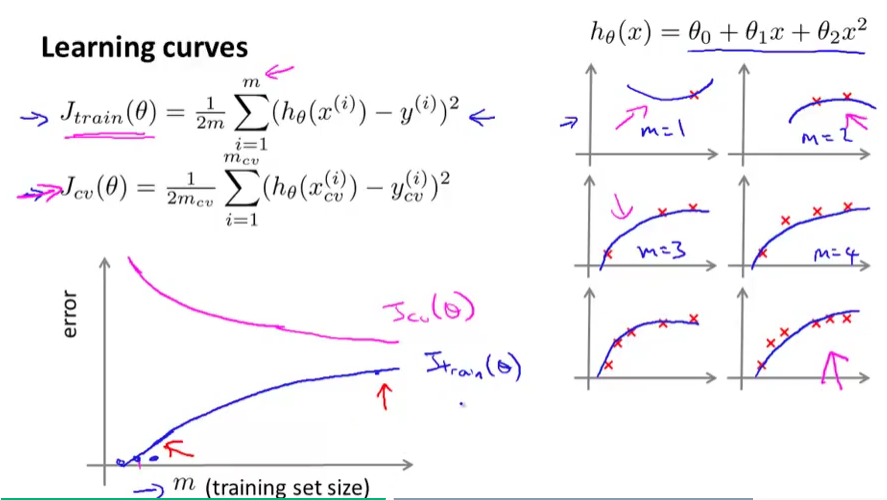




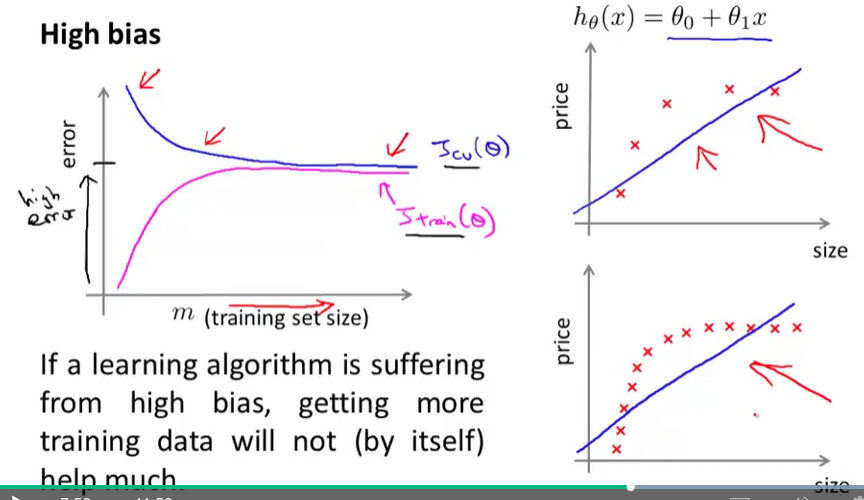
## Learning curves

Leerning curves is een tool die wordt gebruik om te proberen vast te stellen of een fysiek learning algoritme mogelijk lijdt onder **Bias**, een soort **Variance** probleem of een beetje van beide.

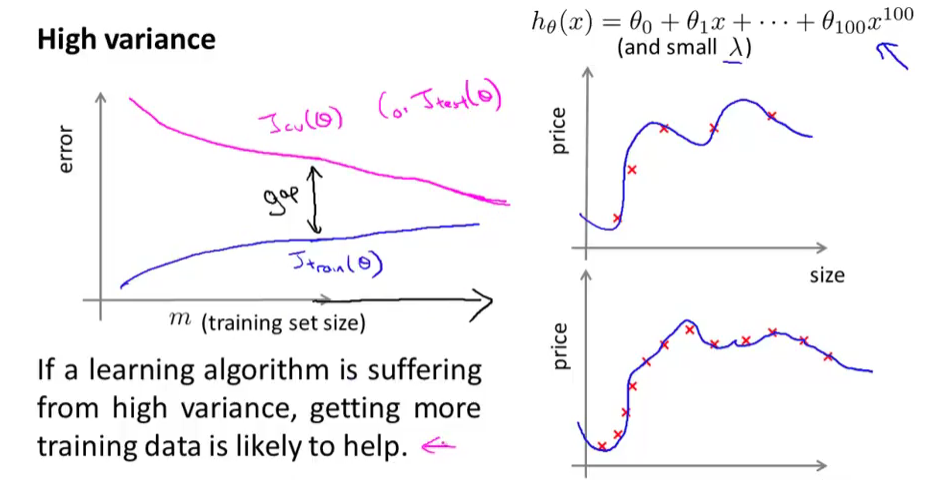
Als de training set klein is dan is de trainings error ook klein. Dit komt door omdat het makkelijk is om een kleine trainings set te fitten. Wanneer de trainings set groter wordt wordt het ook lastiger om alle training examples te fitten. De error wordt dan ook groter (Error, het gemiddelde error van je hypothese).

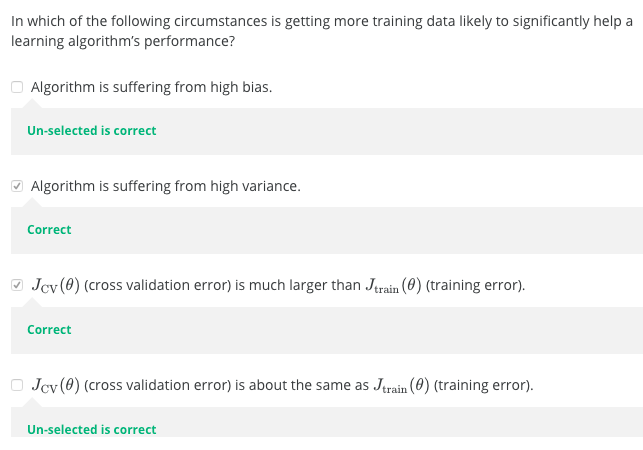


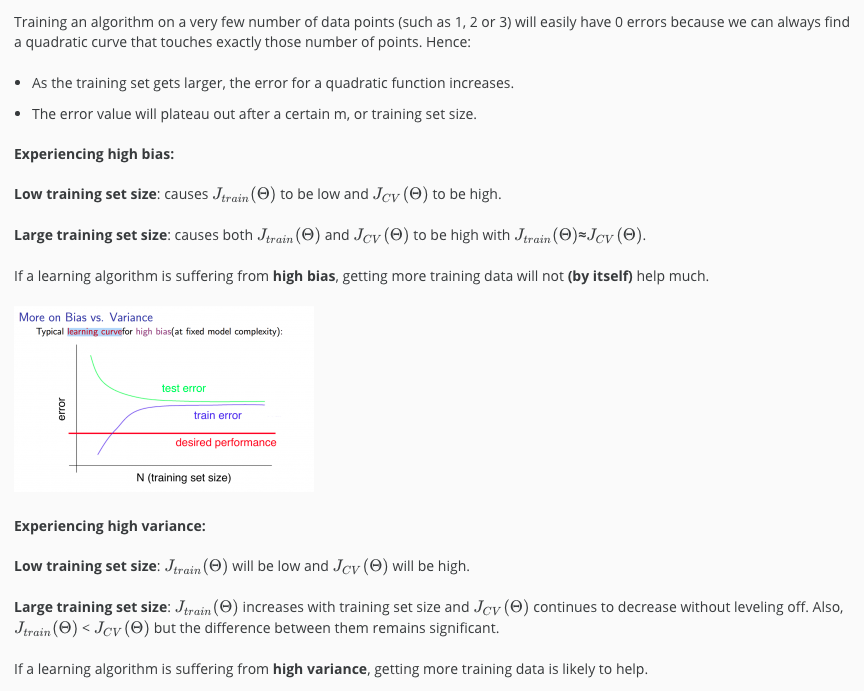
Bij HIGH BIAS helpt het meer toevoegen van trainings examples niet. Dit is handig om te weten om je tijd niet te verspillen.

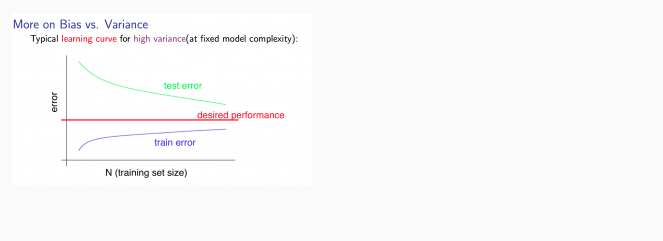


Bij het probleem HIGH VARIANCE is het meer toevoegen van trainings examples juist goed.



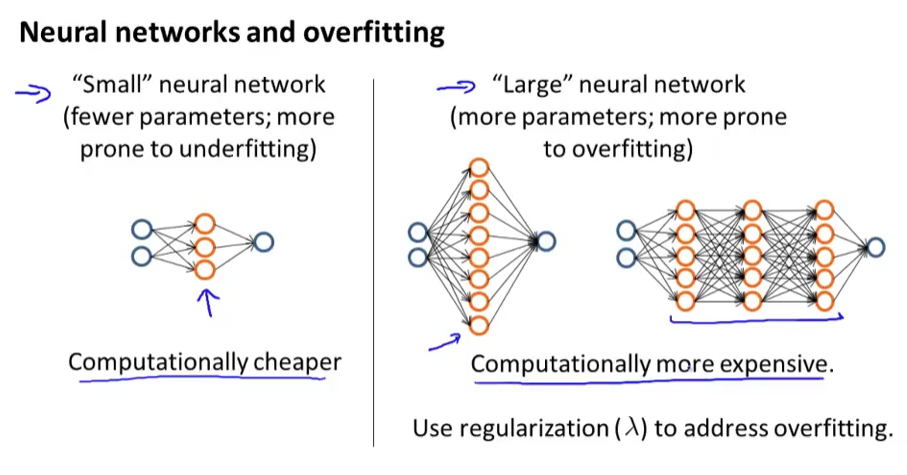




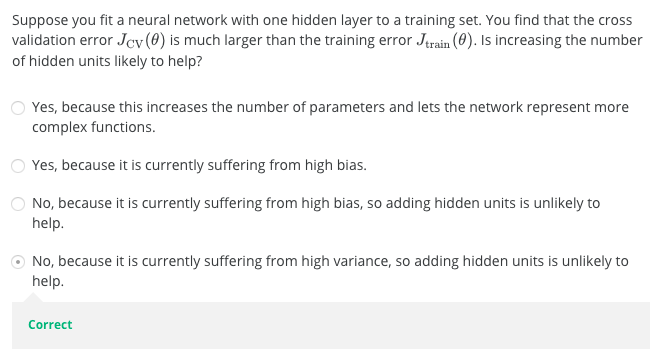


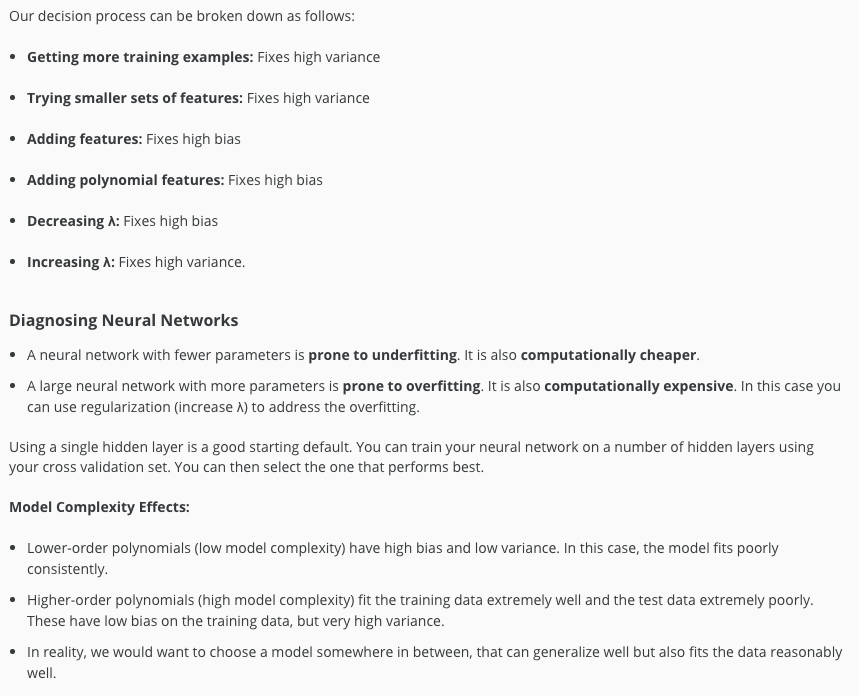
## Deciding What to Do Next Revisited





Je kunt een neural network ook testen met eerst 1 layer, daarna 2 layer en 3 layer etc om te zien welke het beste fit. Voor overfitten in neural network kun je ook gebruik maken van regularizatie.





# Building a spam classifier

## Prioritizing What to Work On