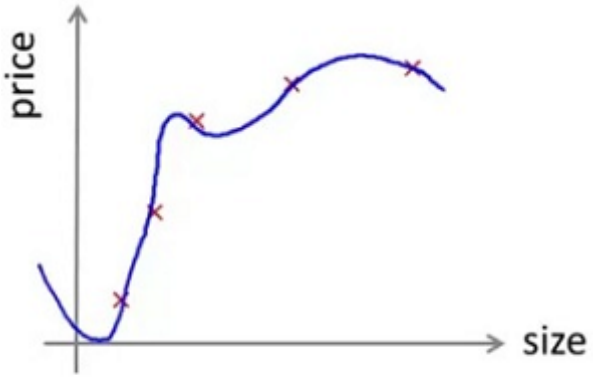
Model Selection and Train/Validation/Test Sets

Suppose we're left to decide what degree of polynomial to fit to a data set. So that what features to include that gives us a learning algorithm. Or suppose we'd like to choose the regularization parameter lambda for learning algorithm. How do we do that?

We've already seen a lot of times the problem of overfitting, in which just because a learning algorithm fits a training set well, that doesn't mean it's a good hypothesis. This is why the training set's error is not a good predictor for how well the hypothesis will do on new example.



$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$$

More general, once parameters θ_0, θ_1, ..., θ_4 were fit to some set of data (training set), the error of the parameters as measured on that data (the training error $J(\theta)$) is likely to be lower than the actual generalization error.

Model Selection

Now let's consider the model selection problem, let's say we're trying to choose what degree polynomial to fit to data. So, should we choose a linear function, a quadratic function, a cubic function? All the way up to a 10^{th}-order polynomial.

So it's as if there's one extra parameter in this algorithm, which we're going to denote d, which is, what degree of polynomial do we want to pick. So it's as if, in addition to the theta parameters, it's as if there's one more parameter d (degree of polinomial), that we're trying to determine using our data set.

$d=1$ 1. $\rightarrow h_\theta(x) = \theta_0 + \theta_1 x$
$d=2$ 2. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$
$d=3$ 3. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_3 x^3$
$\vdots$ $\vdots$
$d=10$ 10. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_{10} x^{10}$

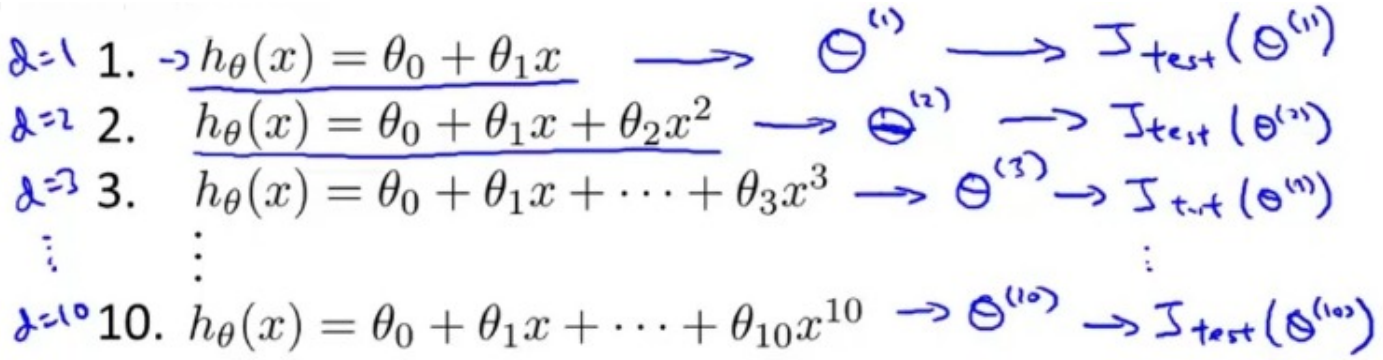
Concretely let's say that we want to choose a model, that is choose a degree of polynomial, choose one of the 10 models. And we fit that model and also get some estimate of how well our fitted hypothesis was generalize to new examples.

What we could do is:

- Take model 1, minimize with training data which generates a parameter vector $\theta^{(1)}$ (where $d = 1$)

- Take mode 2, we do the same, get a different $\theta^{(2)}$ (where $d = 2$)
- And so on until 10^{th} model.

Take these parameters and look at the test set error for each using the previous formula



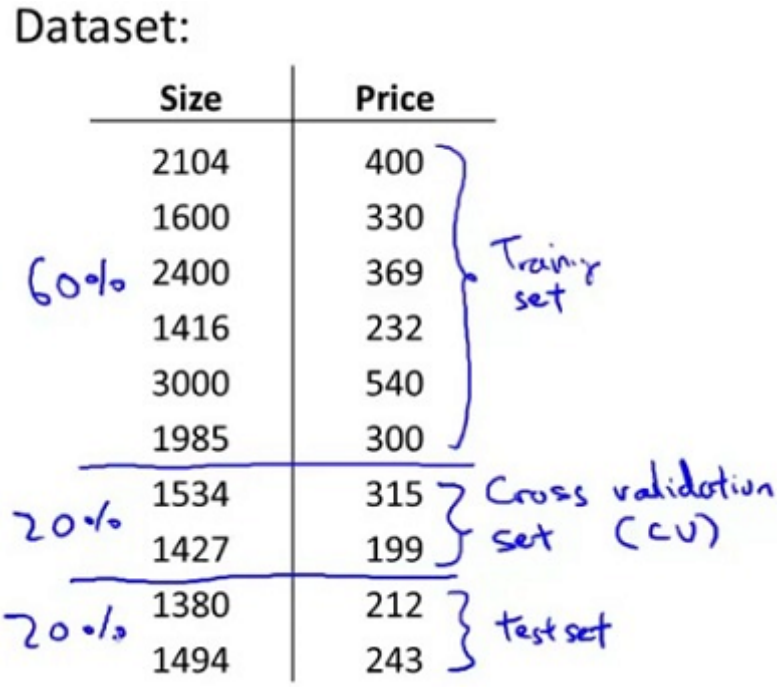
We could then see which model has the lowest test set error $J_{test}(\theta^{(d)})$, and let's just say for this example that we ended up choosing the 5^{th} order polynomial. Now we take the $d = 5$ model and we say, how well does the model generalize? One thing we could do is look at how well our 5^{th} order polynomial hypothesis had done on our test set. $J_{test}(\theta^{(5)}$. But the problem is this will not be a fair estimate of how well our hypothesis generalizes.

Problem: $J_{test}(\theta^{(5)})$ is likely to be an optimistic estimate of generalization error. Because our extra parameter ($d =$ degree of polynomial) is fit to that test set (i.e. specifically chose it because the test set error is small). So not a good way to evaluate if it will generalize.

Evaluating our hypothesis

To address this problem, in a model selection setting, if we want to evaluate a hypothesis, this is what we usually do instead.

Given the data set, instead of just splitting into a training test set, what we're going to do is then split it into three pieces, and the first piece is going to be called the **training set**, the second piece of the data, is going to be called **the Cross Validation set (CV)**, and the last part is going to be called **the test set**.



Concretely, Given a training set instead split into three pieces:

1. Training set (60%) - m values
2. Cross validation (CV) set (20%) m_{CV}
3. Test set (20%) m_{test}

Train/validation/test error

Now that we've defined the training validation or cross validation and test sets. We can also define **the training error**, **cross validation error**, and **test error**.

Training error:

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

Cross Validation error:

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

Test error:

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

Video Question: Consider the model selection procedure where we choose the degree of polynomial using a cross validation set. For the final model (with parameters θ), we might generally expect $J_{\text{CV}}(\theta)$ To be lower than $J_{\text{test}}(\theta)$ because:

> An extra parameter (d, the degree of the polynomial) has been fit to the cross validation set.

- An extra parameter (d, the degree of the polynomial) has been fit to the test set.
- The cross validation set is usually smaller than the test set.
- The cross validation set is usually larger than the test set.

So when faced with a model selection problem, what we're going to do is, instead of using the test set to select the model, we're instead going to use **the validation set**, or **the cross validation set**, to select the model.

Concretely, what we're going to do is:

- Minimize cost function for each of the model as before $min_\theta J(\theta)$.
- We test these hypotheses on the cross validation set to generate the cross validation error $J_{CV}(\theta)$.
- We pick the hypothesis with the lowest cross validation error (e.g. pick $\theta^{(4)}$).
- Finally, we estimate generalization error for test set $J_{test}(\theta)$

What we've done is we'll fit that parameter d and we'll say $d = 4$, and we did so using the cross-validation set. And so this degree of polynomial, so the parameter, is no longer fit to the test set, and so we've not saved away the test set, and we can use the test set to measure, or to estimate the generalization error of the model that was selected.

Summary

Just because a learning algorithm fits a training set well, that does not mean it is a good hypothesis. It could over fit and as a result your predictions on the test set would be poor. The error of your hypothesis as measured on the data set with which you trained the parameters will be lower than the error on any other data set.

Given many models with different polynomial degrees, we can use a systematic approach to identify the 'best' function. In order to choose the model of your hypothesis, you can test each degree of polynomial and look at the error result.

One way to break down our dataset into the three sets is:

- Training set: 60%
- Cross validation set: 20%
- Test set: 20%

We can now calculate three separate error values for the three different sets using the following method:

1. Optimize the parameters in Θ using the training set for each polynomial degree.
2. Find the polynomial degree d with the least error using the cross validation set.
3. Estimate the generalization error using the test set with $J_{test}(\Theta^{(d)})$, (d = theta from polynomial with lower error);

This way, the degree of the polynomial d has not been trained using the test set.