Machine Learning

By Ghazal Lalooha

Troubleshooting a machine learning algorithm

Troubleshooting

- Suppose you have implemented the adjusted linear regression algorithm in order to predict house prices:

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

- But when you test the obtained hypothesis on a new set of houses, you realize that this hypothesis contains large and unacceptable errors in its prediction.
- Question: How can this problem be solved?

Troubleshooting

- Possible solutions:
  - Use more training examples.
  - Use fewer features.
  - Try adding polynomial features to the feature set. (x_1^2, x_2^2, x_1x_2, …)
  - Reduce the regularization factor.(λ)
  - Increase the regularization factor. (λ)

Troubleshooting a machine learning system

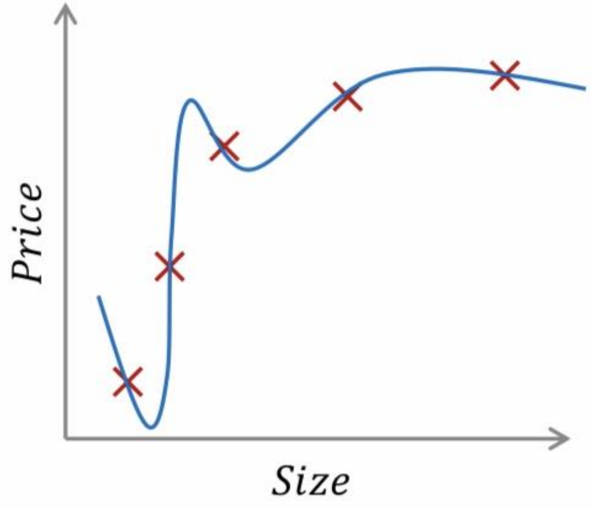
- diagnosis:
  - An experiment by which you can understand which aspects of a learning algorithm do not work properly and how to improve the performance of the learning algorithm in the best possible way.

- Although implementing troubleshooting methods may be time-consuming, using these methods will ultimately save you considerable time.

Hypothesis evaluation

Hypothesis evaluation

- Inability to generalize:
  - Inappropriate response for new exemplars not previously taught.
- Attributes:
- X_1: house size
- X_2: number of bedrooms
- X_3: number of levels
- X_4: age
- X_5: kitchen size
- …
- X_{100}: average income of neighbors



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

Hypothesis evaluation

- Data set

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

Training set (70 %)

$$\left(x^{(1)}, y^{(1)}\right)$$
$$\left(x^{(2)}, y^{(2)}\right)$$
$$\vdots$$
$$\left(x^{(m)}, y^{(m)}\right)$$

Test set (30%)

$$\left(x_{test}^{(1)}, y_{test}^{(1)}\right)$$
$$\left(x_{test}^{(2)}, y_{test}^{(2)}\right)$$
$$\vdots$$
$$\left(x_{test}^{(m_{test})}, y_{test}^{(m_{test})}\right)$$

Training and testing for linear regression

- Train:
  - Learning theta parameters using the training set by minimizing the J cost function

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

- **Test:**

Calculate the error for the test set

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

Training and testing for logistic regression

- Train:
  - Learning theta parameters using the training set by minimizing the J cost function

$$J(\theta) = -\sum_{i=1}^{m} y^{(i)} \log h_\theta\left(x^{(i)}\right) + \left(1 - y^{(i)}\right) \log\left(1 - h_\theta\left(x^{(i)}\right)\right)$$

- **Test:**

Calculate the error for the test set

$$J_{test}(\theta) = -\sum_{i=1}^{m_{test}} y_{test}^{(i)} \log h_\theta\left(x_{test}^{(i)}\right) + \left(1 - y_{test}^{(i)}\right) \log\left(1 - h_\theta\left(x_{test}^{(i)}\right)\right)$$

Training and testing for logistic regression

- Train:
  - Learning theta parameters using the training set by minimizing the J cost function

$$J(\theta) = -\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log\left(1 - h_\theta(x^{(i)})\right)$$

- **Test:**

Calculate the error of classification for the test set

$$J_{test}(\theta) = \sum_{i=1}^{m_{test}} err\left(h_\theta(x^{(i)}), y^{(i)}\right)$$

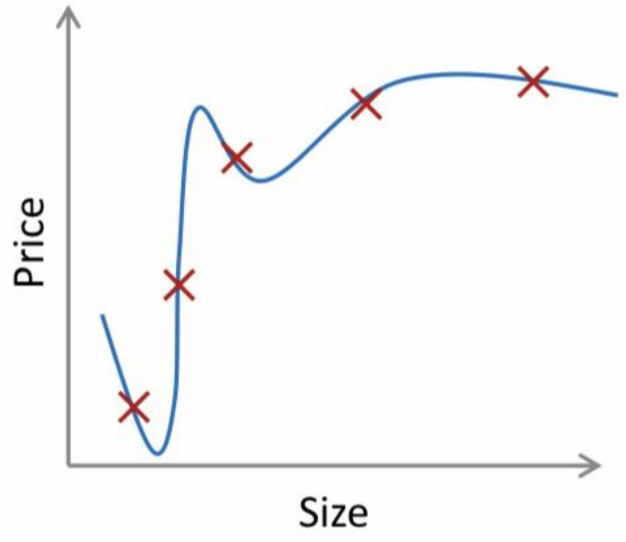
$$err(h_\theta(x), y) = \begin{cases} 1 & h_\theta(x) < 0.5, y = 1 \\ & h_\theta(x) \geq 0.5, y = 0 \\ 0 & otherwise \end{cases}$$

Model selection

- Training set
- Validation set
- Test set

Over fit example

- After learning the parameter values from a training set, the error calculated on the training set is usually lower than the actual generalization error.
- In other words, low training error does not necessarily mean that the hypothesis is appropriate.
- Question: How can generalization error be estimated?



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

Model selection

- Question: Which of the following models is best for a given set of data?

$$h_\theta(x) = \theta_0 + \theta_1 x \qquad \min_\theta J(\theta) \rightarrow \theta^{(1)} \rightarrow J_{test}(\theta^{(1)})$$

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 \qquad \min_\theta J(\theta) \rightarrow \theta^{(2)} \rightarrow J_{test}(\theta^{(2)})$$

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 \qquad \min_\theta J(\theta) \rightarrow \theta^{(3)} \rightarrow J_{test}(\theta^{(3)})$$

$$\vdots \qquad \vdots$$

$$h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_{10} x^{10} \qquad \min_\theta J(\theta) \rightarrow \theta^{(10)} \rightarrow J_{test}(\theta^{(10)})$$

$$\cancel{J_{test}(\theta^{(5)})}$$

Model Selection Select the model that has the lowest experimental error. Suppose the polynomial of degree 5 has the least experimental error.

Estimation of generalizability:

What is the generalizability of the selected model ? What is the generalizability of the selected model

Hypothesis evaluation

- **Data set**

	meterage	price
Training set (60%)	2104	400
	1600	330
	2400	369
	1416	232
	3000	540
	1985	300
Validation set (20%)	1534	315
	1427	199
Test set (20%)	1380	212
	1494	243

$$\left(x_{cv}^{(1)}, y_{cv}^{(1)}\right)$$
$$\left(x_{cv}^{(2)}, y_{cv}^{(2)}\right)$$
$$\vdots$$
$$\left(x_{cv}^{(m_{cv})}, y_{cv}^{(m_{cv})}\right)$$

Training/ Validation/ Test set

- Training set:
  - Learning the value of
  - model parameters

Validation set:

  - Select a model

  (specifying a

  Hyper parameter)

- Test set:
  - Estimating of the error of the

  selected model generalization

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

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

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

Bias and high variance detection

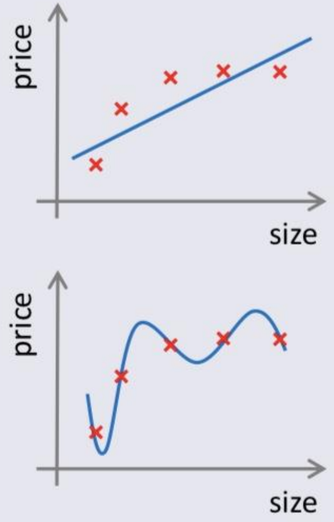
Bias and variance

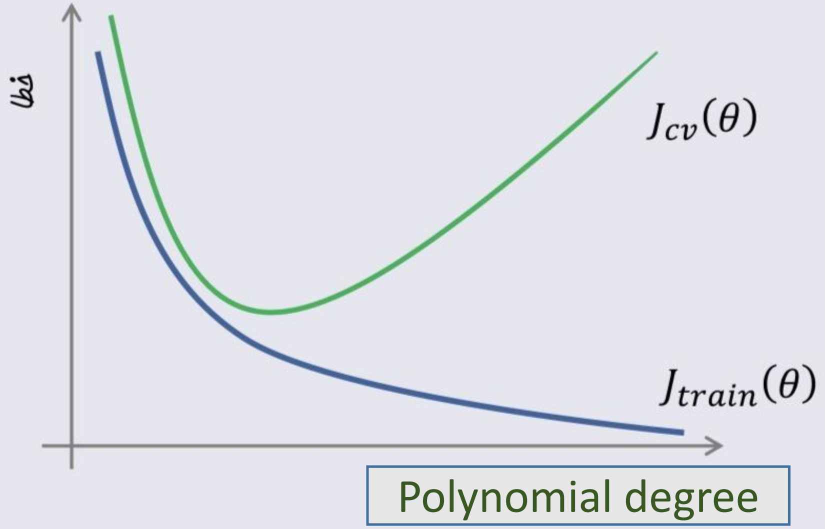
$$J_{train}(\theta) = \frac{1}{2}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

Training error

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

Validation error





Bias Vs variance detection

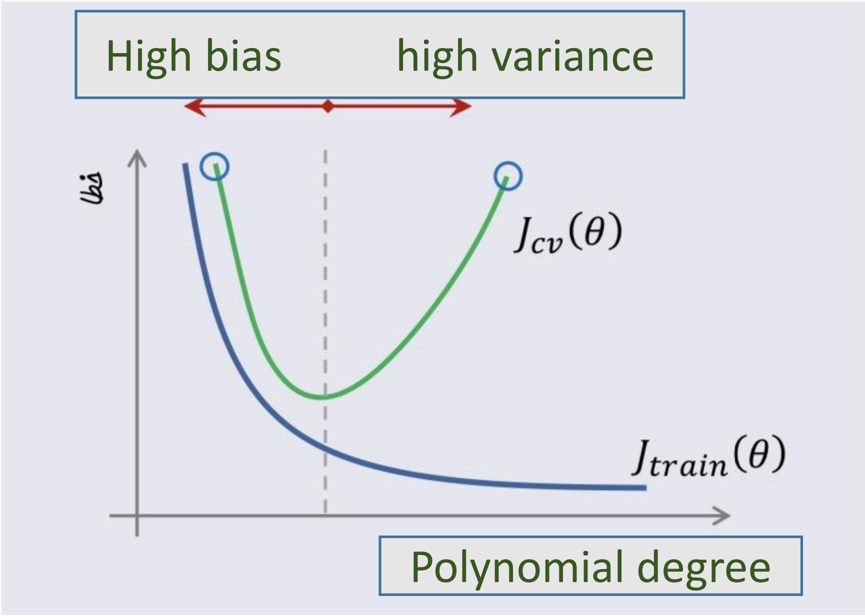
- Suppose the performance quality of your algorithm is lower than you expect! (validation error is high)
- Question: How can you determine if this problem is due to bias or variance?

High bias:

- High training error
- High validation error

High variance:

- low training error
- High validation error



Regularization and bias/ variance issue

Regularization

- Regularization: Using regularization can be effective in dealing with the problem of overfitting.

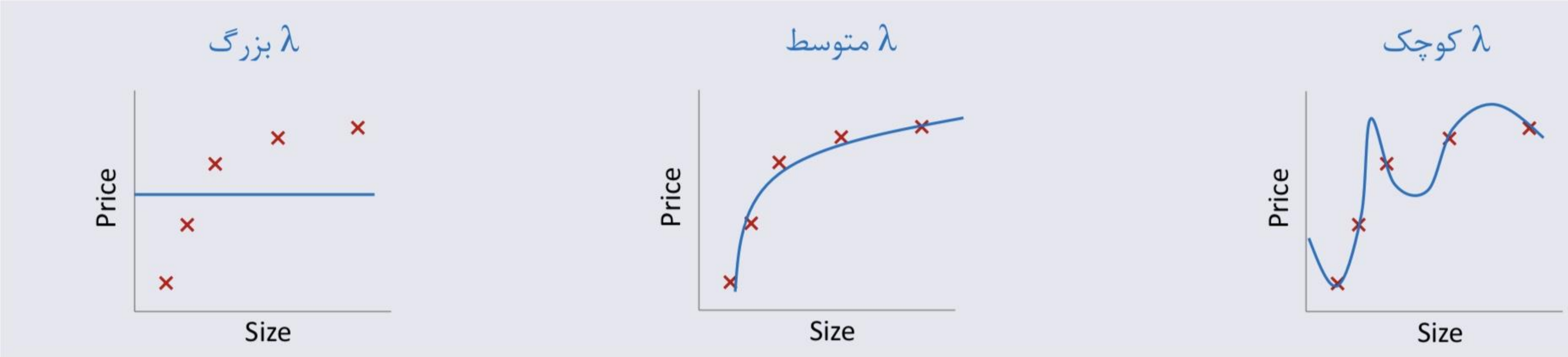
$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{n}\theta_j^2$$

- But how does regularization affect bias and variance?

Regularization factor selection

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

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



Under fit (high bias)

Correct model

over fit (high variance)

Regularization factor selection

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

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

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

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

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

Regularization factor selection

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

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

$$\left.\begin{array}{lll}
1. & \lambda = 0.00 & \min_\theta J(\theta) \rightarrow \theta^{(1)} \rightarrow J_{cv}\left(\theta^{(1)}\right) \\
2. & \lambda = 0.01 & \min_\theta J(\theta) \rightarrow \theta^{(2)} \rightarrow J_{cv}\left(\theta^{(2)}\right) \\
3. & \lambda = 0.02 & \min_\theta J(\theta) \rightarrow \theta^{(3)} \rightarrow J_{cv}\left(\theta^{(3)}\right) \\
4. & \lambda = 0.04 & \min_\theta J(\theta) \rightarrow \theta^{(4)} \rightarrow J_{cv}\left(\theta^{(4)}\right) \\
\vdots & & \\
12. & \lambda = 10.0 & \min_\theta J(\theta) \rightarrow \theta^{(12)} \rightarrow J_{cv}\left(\theta^{(12)}\right)
\end{array}\right\}$$

Choosing the model that has the lowest validation error.

Regularization factor selection

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

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

$$\left.\begin{array}{lll}
1.\ \lambda = 0.00 & \min_\theta J(\theta) \rightarrow \theta^{(1)} \rightarrow J_{cv}\left(\theta^{(1)}\right) \\
2.\ \lambda = 0.01 & \min_\theta J(\theta) \rightarrow \theta^{(2)} \rightarrow J_{cv}\left(\theta^{(2)}\right) \\
3.\ \lambda = 0.02 & \min_\theta J(\theta) \rightarrow \theta^{(3)} \rightarrow J_{cv}\left(\theta^{(3)}\right) \\
4.\ \lambda = 0.04 & \min_\theta J(\theta) \rightarrow \theta^{(4)} \rightarrow J_{cv}\left(\theta^{(4)}\right) \\
\vdots & \\
12.\ \lambda = 10.0 & \min_\theta J(\theta) \rightarrow \theta^{(12)} \rightarrow J_{cv}\left(\theta^{(12)}\right)
\end{array}\right\}$$

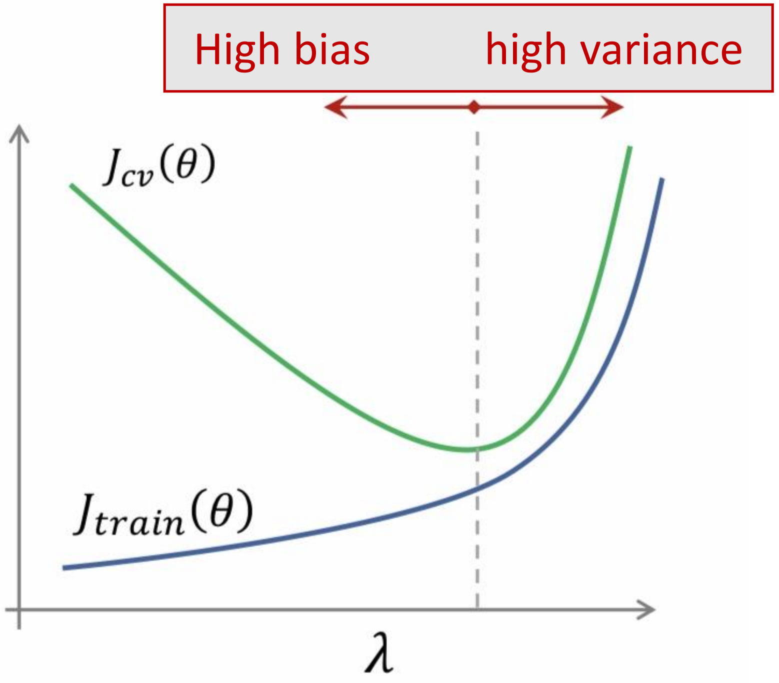
Choosing the model that has the lowest validation error.

Regularization factor selection

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

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

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

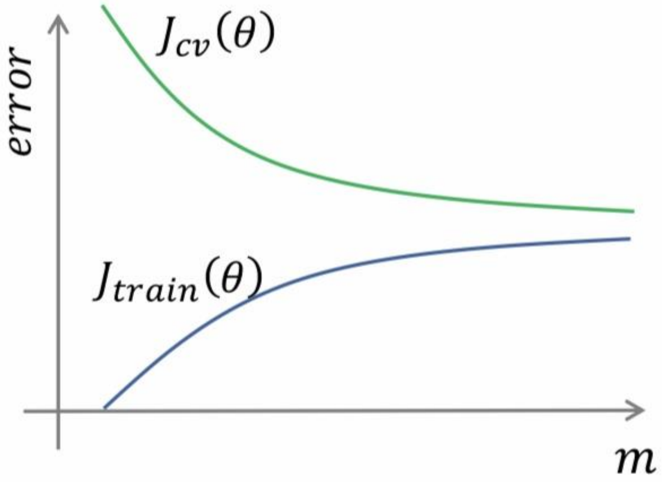


Learning curves

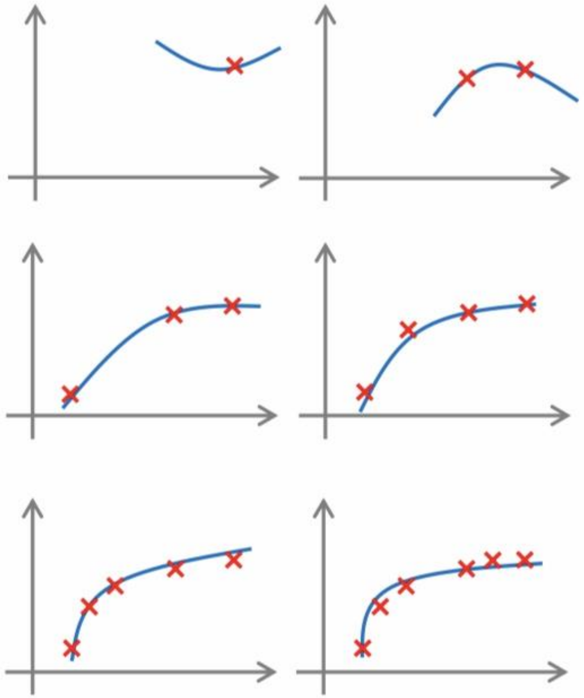
Learning curves

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

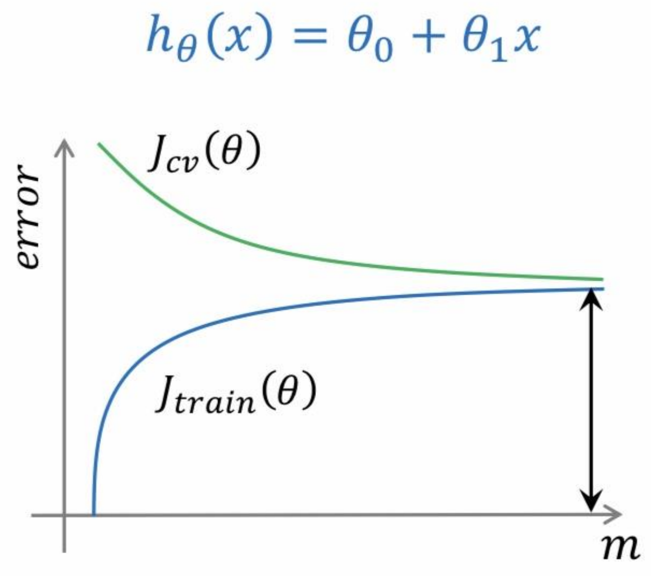
$$J_{cv}(\theta) = \frac{1}{2}\sum_{i=1}^{m_{cv}}\left(h_\theta\left(x_{cv}^{(i)}\right) - y_{cv}^{(i)}\right)^2$$

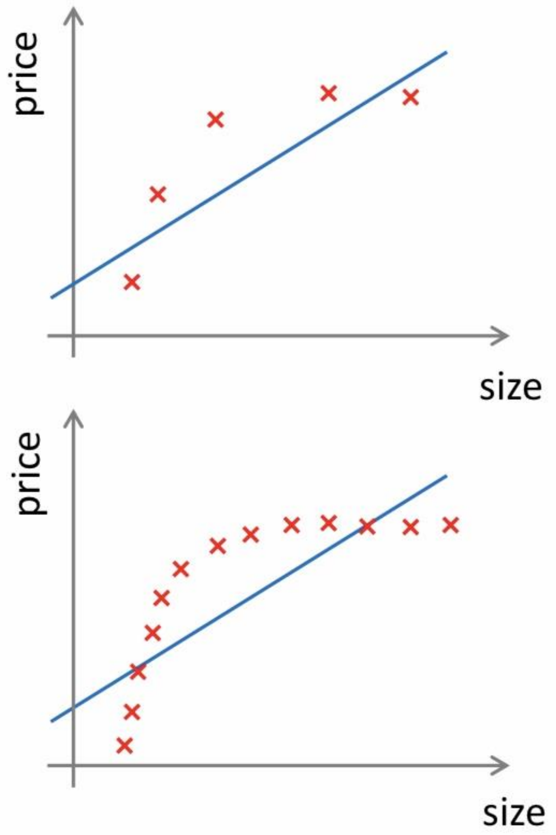


$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$$



High bias

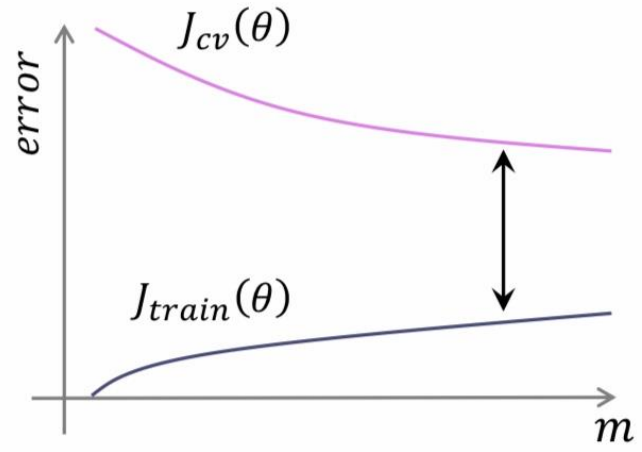




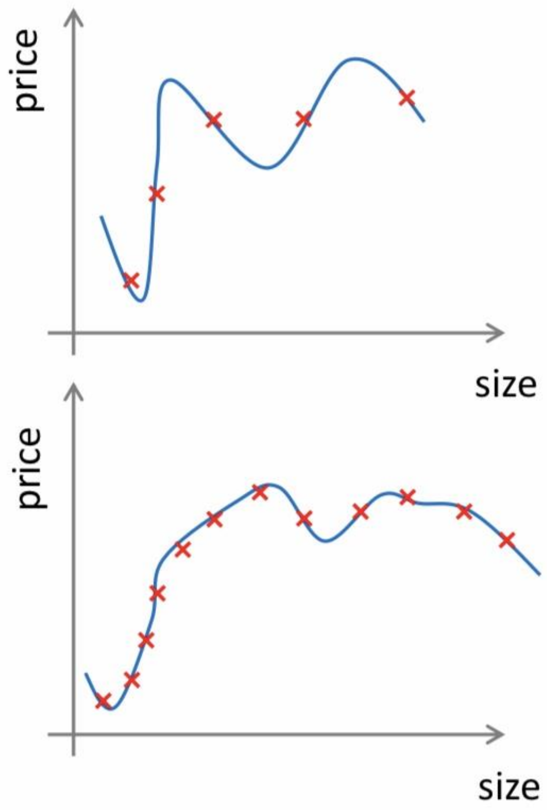
If a learning algorithm suffers from high bias, increasing the number of training samples will not help it much.

High variance

$$h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_{100} x^{100}$$

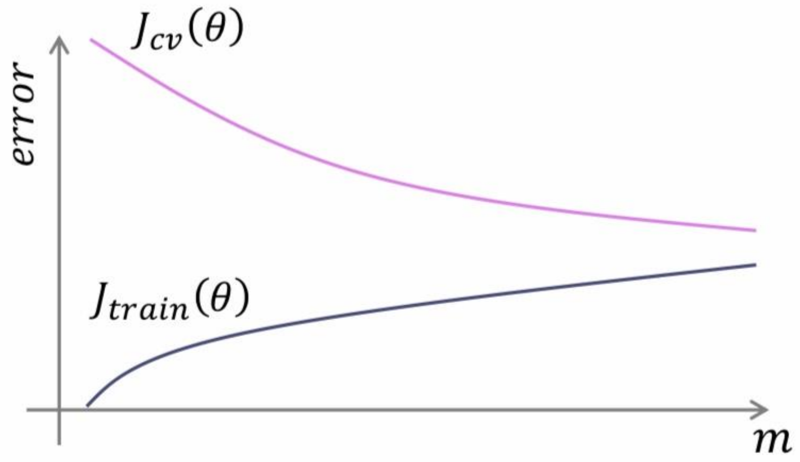


If a learning algorithm suffers from high variance, increasing the number of training samples will help it much.

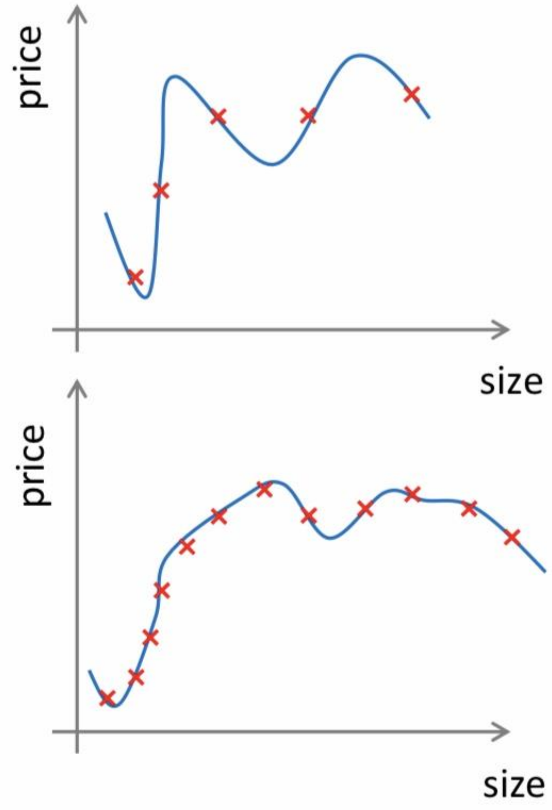


High variance

$$h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_{100} x^{100}$$



If a learning algorithm suffers from high variance, increasing the number of training samples will help it much.



Troubleshooting

Troubleshooting

- Suppose you have implemented the regularized linear regression algorithm in order to predict house prices:

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

- But when you test the obtained hypothesis on a new set of houses, you realize that this hypothesis contains large and unacceptable errors in its prediction.

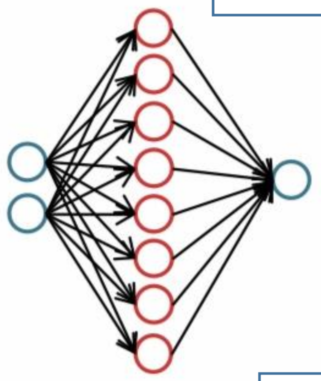
- Question: How can this problem be solved?

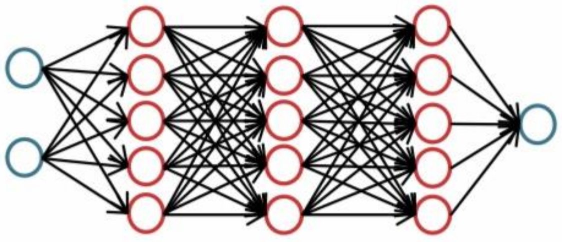
Troubleshooting

- Possible solutions:
  - Increasing the number of training samples → solving the high variance problem
  - Reducing the number of features → solving the high variance problem
  - Increase the number of features → solve the problem of high bias
  - Adding polynomial features → Solving the high bias problem
  - Reducing the regularization coefficient → solving the problem of high bias
  - Increasing the regularization coefficient → solving the problem of high variance

Neural Networks and Over fit

“Large" neural network (more parameters, more overfitting)

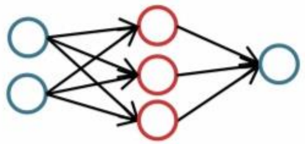




more computational cost

Using regularization to face with overfitting

"Small" neural network (fewer parameters, more underfitting)



Less computational cost