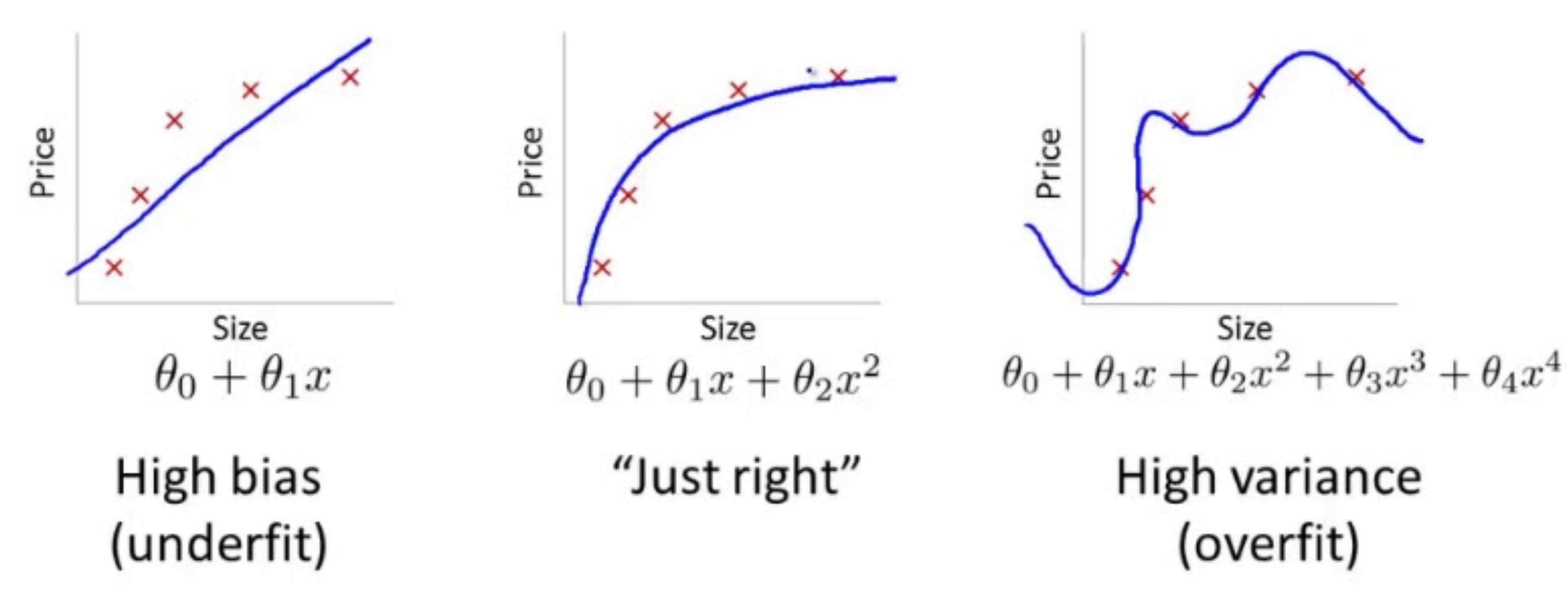
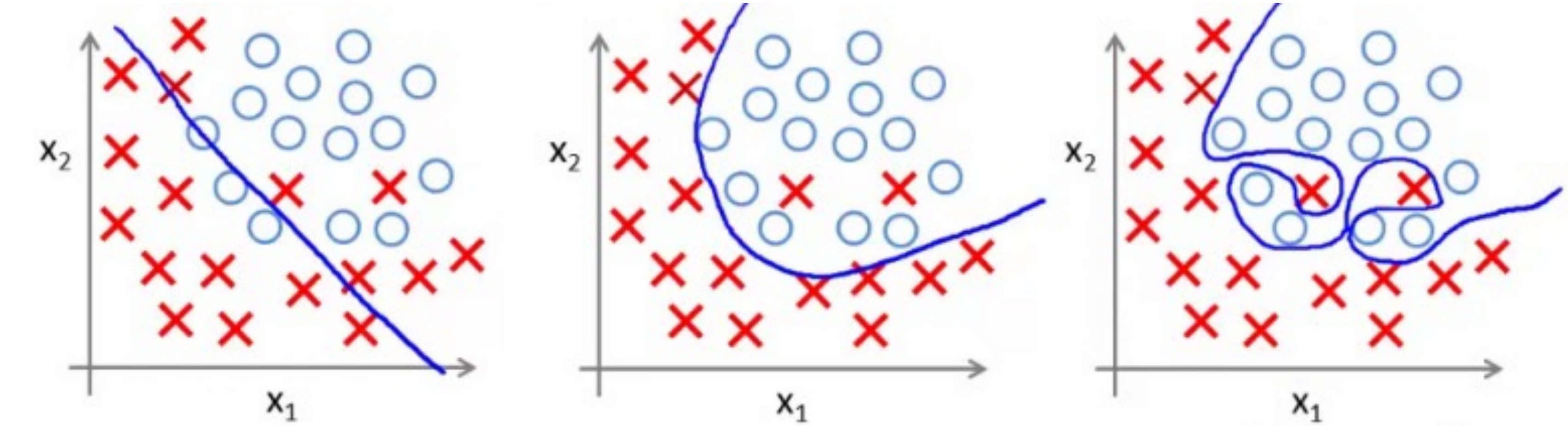
Fitting

Just Fit

Fitting - regression



Fitting - classification



$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

(g = sigmoid function)

UNDERFITTING
(high bias)

$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2)$

$g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \dots$

OVERFITTING
(high variance)

Overfitting and Underfitting

We want a model able to generalize from its experience (training example). Machine learning is concerned with perform accurately on new, unseen examples/tasks after having experienced a learning data set

Underfitting: the model does not fit the training data, because it is too simple and not able to learn. It performs bad both on training and test data.

Overfitting: the model performs well on the training data but not on unseen data, because it is too complex and not able to generalize. The model begins to "memorize" training data rather than "learning" to generalize from trend

Expected squared error and its relationship with expected value

- Expected value of a random variable X

$$E[X] = \sum_{i=1}^{m} p_i \cdot x_i$$

- Let X be the input and Y be the output in a problem we want to model

$$Y = f(X) + e$$

- We assume our hypothesis $h(X)$ to approximate $f(X)$. The expected squared error (a.k.a. **M**ean **S**quared **E**rror) is then

$$Err(f(X)) = E[(Y - h(X))^2]$$

Bias and Variance trade-off

We assume to have a number of $Dataset \quad D_i$, each of size m, sampled from the original data distribution. They are subsets of the corresponding random variable

We fix the model and we train it each time with a different D_i . We get a corresponding number of hypothesis $h^{(D_i)}(x)$.

Each training sample is in the form $< \mathbf{x^{(i)}}, y^{(i)} >$, where y is the sum of the ground truth and a gaussian error with zero mean and variance σ_e:

$$y^{(i)} = f(\mathbf{x^{(i)}}) + e^{(i)}$$

Each hypothesis $h^{(D_i)}(x)$ will be affected by an error in pradicting values w.r.t. the ground truth.

We can model this error in the form of a Mean Squared Error (MSE):

$$MSE\left(h^{(D_i)}(x)\right) = E_x\left[(h^{(D_i)}(x) - f(x))^2\right]$$

Bias and Variance trade-off (cont.d)

We want to estimate the expectation of the error w.r.t. all the possible D_i.

In details we want to compute the MSE of each hypothesis and then we compute the overall mean.

We can compute the expected value of the mean as the expectation of the MSE over all the datasets.

This expectation is named Generalization Error (GER):

$$
\begin{aligned}
GER &= E_D[MSE] \\
&= E_D\left[E_x\left[\left(h^{(D_i)}(x) - f(x)\right)^2\right]\right] \\
&= E_x\left[E_D\left[\left(h^{(D)}(x^{(i)}) - f(x^{(i)})\right)^2\right]\right]
\end{aligned}
$$

The last step can be performed because of the linearity of expectation operator

Bias and Variance trade-off (cont.d)

Focus on the term $E_D\left[\left(h^{(D)}(x^{(i)}) - f(x^{(i)})\right)^2\right]$

We can define a new quantity, the best estimation of $f(x^{(i)})$, by feeding each hypothesis with the same training sample $x^{(i)}$, and then we compute the mean of all the values:

$$\overline{h}(x^{(i)}) = E_D\left[h^{(D)}(x^{(i)})\right]$$

We can now perform a sum zero operation by adding and subtracting the latter from the initial term:

$$E_D\left[\left(h^{(D)}(x^{(i)}) - \overline{h}(x^{(i)}) + \overline{h}(x^{(i)}) - f(x^{(i)})\right)^2\right]$$

We can solve the power and exploit the linearity of E:

$$E_D\left[\left(h^{(D)}(x^{(i)}) - \overline{h}(x^{(i)})\right)^2\right] + E_D\left[\left(\overline{h}(x^{(i)}) - f(x^{(i)})\right)^2\right] + E_D\left[2\left(h^{(D)}(x^{(i)}) - \overline{h}(x^{(i)})\right)\left(\overline{h}(x^{(i)}) - f(x^{(i)})\right)\right]$$

Bias and Variance trade-off (cont.d)

$$E_D\left[\left(h^{(D)}(x^{(i)})-\overline{h}(x^{(i)})\right)^2\right]+E_D\left[\left(\overline{h}(x^{(i)})-f(x^{(i)})\right)^2\right]+E_D\left[2\left(h^{(D)}(x^{(i)})-\overline{h}(x^{(i)})\right)\left(\overline{h}(x^{(i)})-f(x^{(i)})\right)\right]$$

Let us focus on the first term:

$$E_D\left[\left(h^{(D)}(x^{(i)})-\overline{h}(x^{(i)})\right)^2\right]=Var\left(h^{(D)}(x^{(i)})\right)$$

It matches the variance definition.

The second term:

$$E_D\left[\left(\overline{h}(x^{(i)})-f(x^{(i)})\right)^2\right]=(\overline{h}(x^{(i)})-f(x^{(i)}))^2$$

Matches the squared bias

Bias and Variance trade-off (cont.d)

The third term:

$$
\begin{aligned}
&2\left(\overline{h}(x^{(i)}) - f(x^{(i)})\right) E_D\left[h^{(D)}(x^{(i)}) - \overline{h}(x^{(i)})\right] \\
=&2\left(\overline{h}(x^{(i)}) - f(x^{(i)})\right) \left(E_D\left[h^{(D)}(x^{(i)})\right] - E_D\left[\overline{h}(x^{(i)})\right]\right) \\
=&2\left(\overline{h}(x^{(i)}) - f(x^{(i)})\right) \left(\overline{h}(x^{(i)}) - \overline{h}(x^{(i)})\right) \\
=&0
\end{aligned}
$$

We can move back to the MSE:

$$
MSE(h^{(D)}(x^{(i)})) = Bias^2\left(h^{(D)}(x^{(i)})\right) + Var\left(h^{(D)}(x^{(i)})\right)
$$

Bias and Variance

Bias is the systematic deviation of the estimator. It represents the mean of the error of the predicted values:

$$Bias\big(h(X), f(X)\big) = \big(E[h(X)] - f(X)\big)^2$$

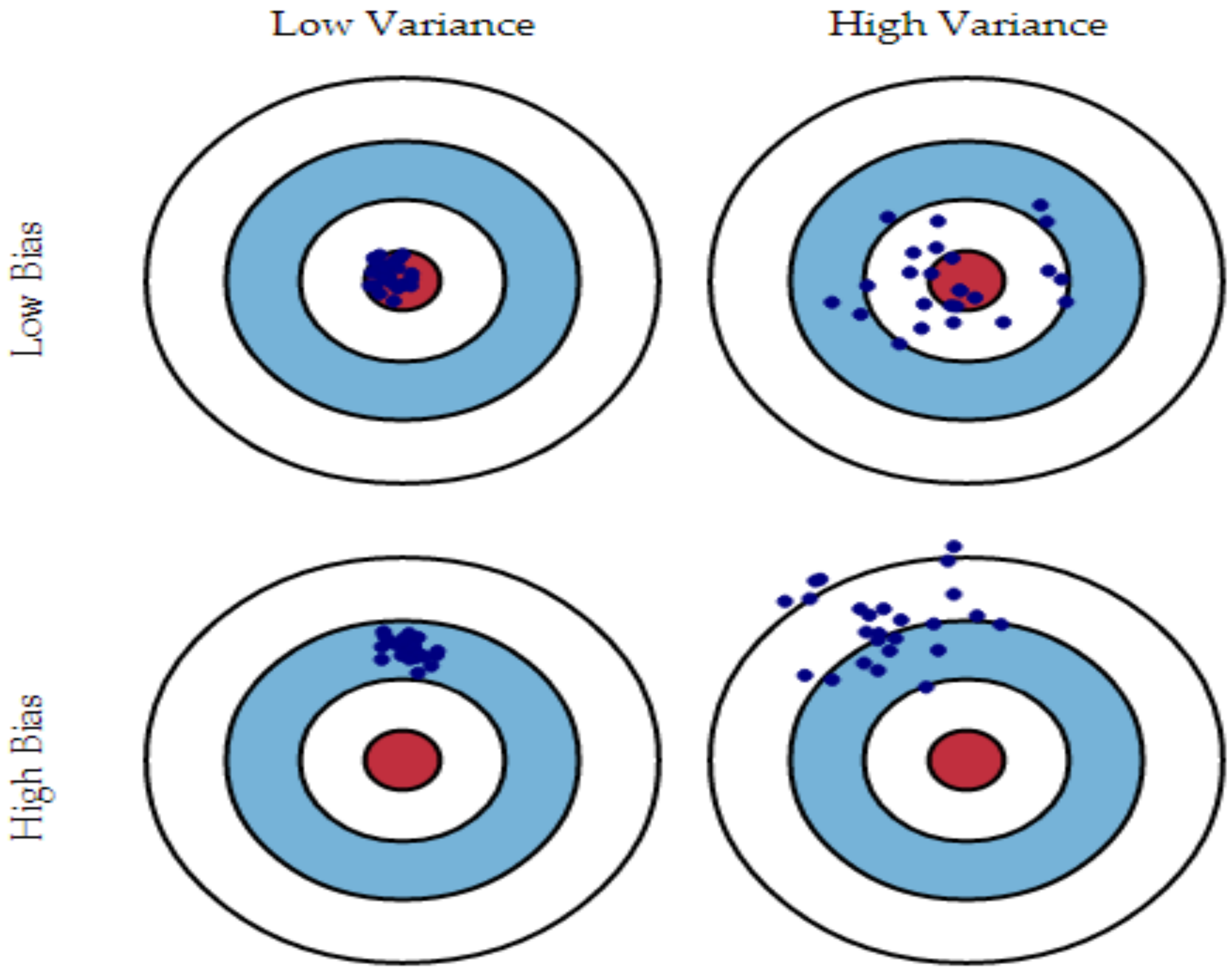
Variance represents the mean of the variations of the predicted values with respect to the mean of the predicted values.

The variance is about the stability of the model in response to new training examples.

It gives us an idea of the mean of the variations of the errors of the predicted values w.r.t. Y :

$$Var\big(h(X)\big) = E(h(X) - E[h(X)])^2$$

Bias and Variance - Intuition



Bias and Variance trade-off (cont.d)

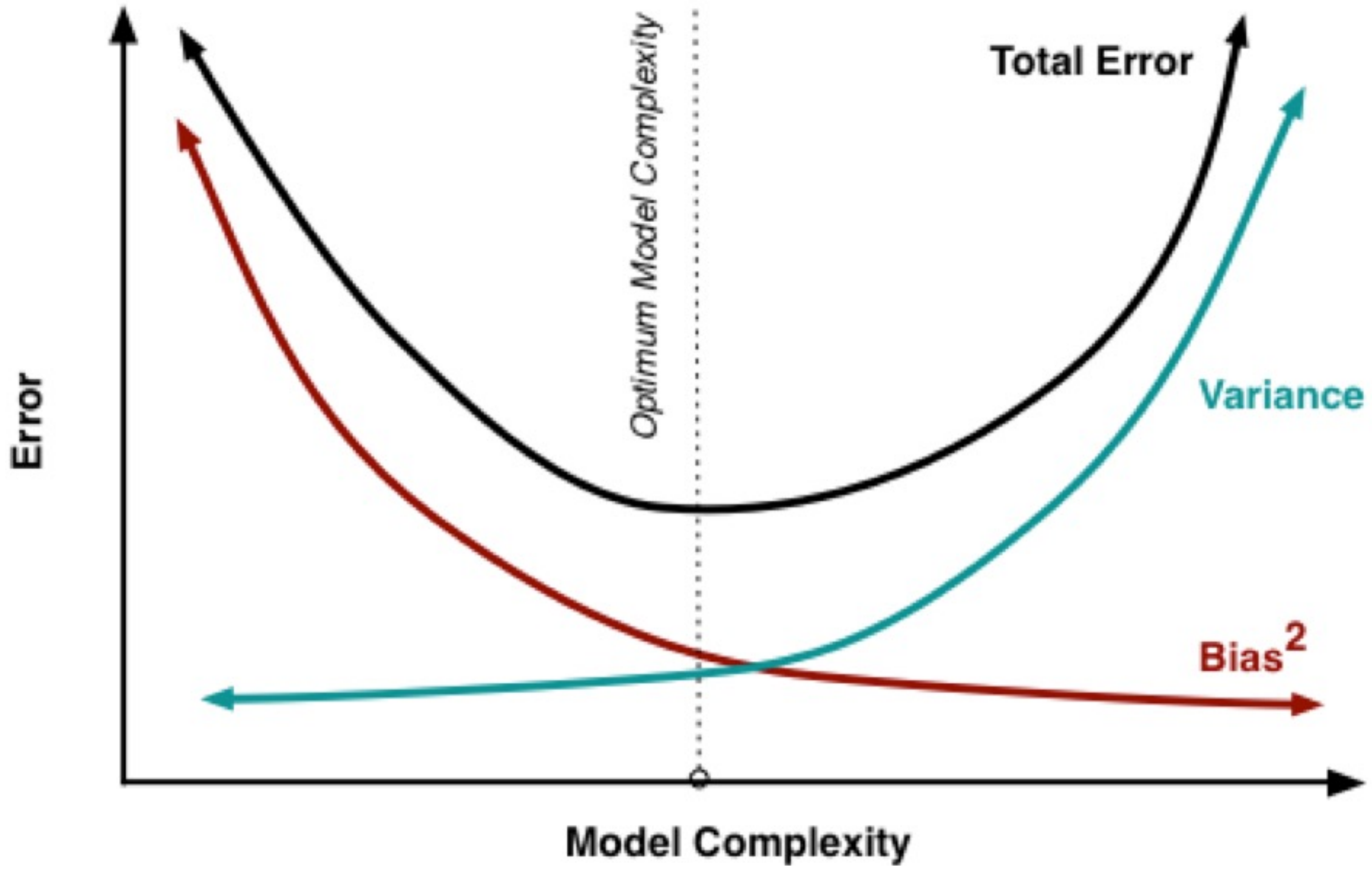
In order to reduce the Generalization Error, we should reduce the bias or the variance:

- Bias represents how much the best estimation is able to get close to the ground truth. The high bias for a specific model denotes a too-simple model that is not able to predict, whatever Dataset you are training on.

- Variance represents how much a single hypothesis can be different from the best estimation. The high variance means that the same model, trained with different datasets, generates very different hypotheses. When the hypotheses are very complex, they are prone to overfit and they are not able to generalize.

The optimum would be to maintain low bias and variance, but, a model given, the bias and the variance behave at the opposite w.r.t. the complexity of the model.

Bias and Variance w.r.t. complexity



The best trade-off is choosing the model complexity that show the minimum sum between bias and variance

Evaluating our hypothesis

Area	Price
31770	215000
11622	105000
14267	172000
11160	244000
13830	189900
9978	195500
4920	213500
5005	191500
5389	236500
7500	189000
10000	175900
7980	185000
8402	180400
10176	171500
6820	212000
53504	538000
12134	164000
11394	394432
19138	141000

80%

Training set

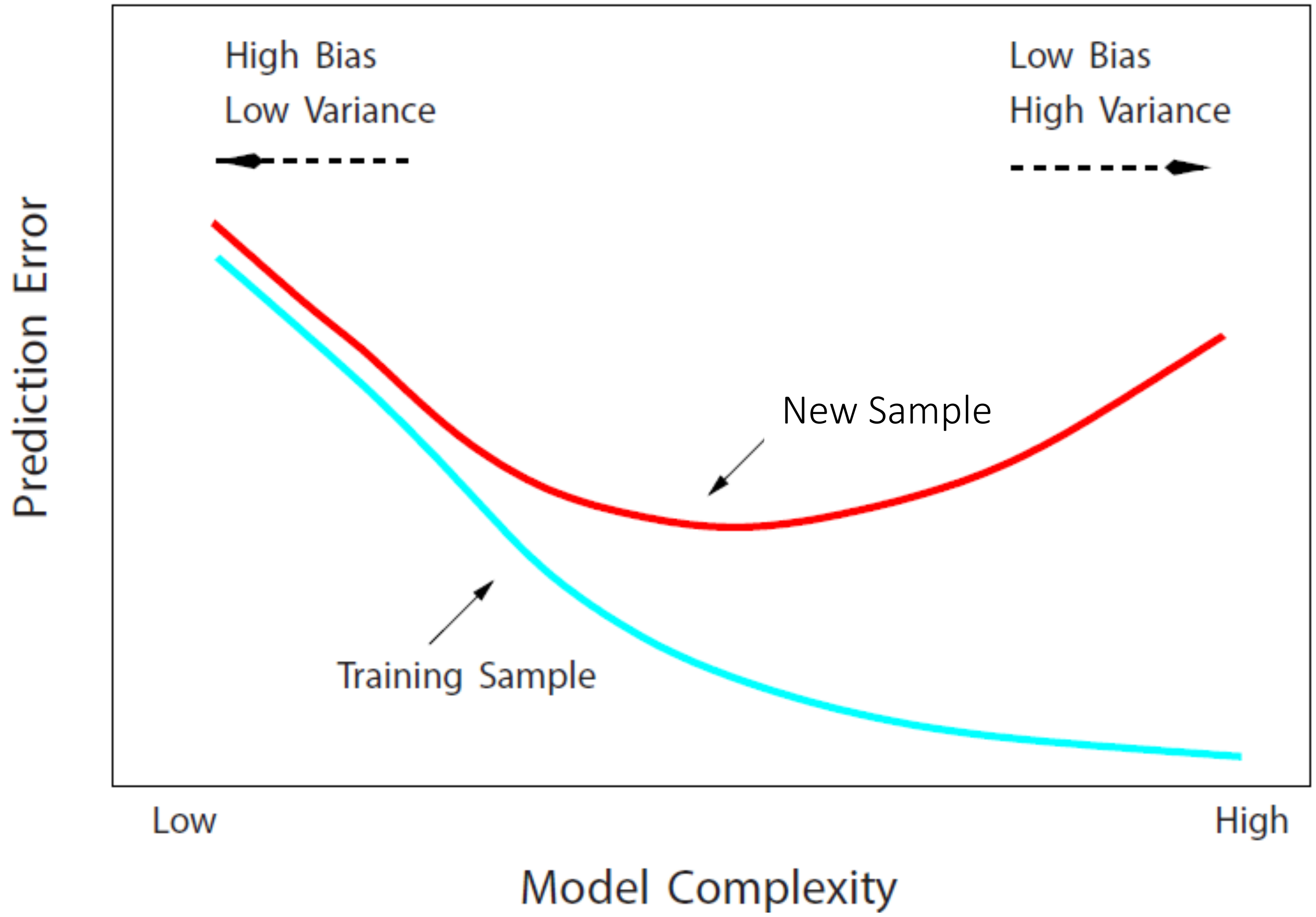
$$(x^{(1)}, y^{(1)})$$
$$(x^{(2)}, y^{(2)})$$
$$\vdots$$
$$\vdots$$
$$(x^{(m)}, y^{(m)})$$

20%

Test set

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

Training set and test set w.r.t. complexity



Fitting

Regularization

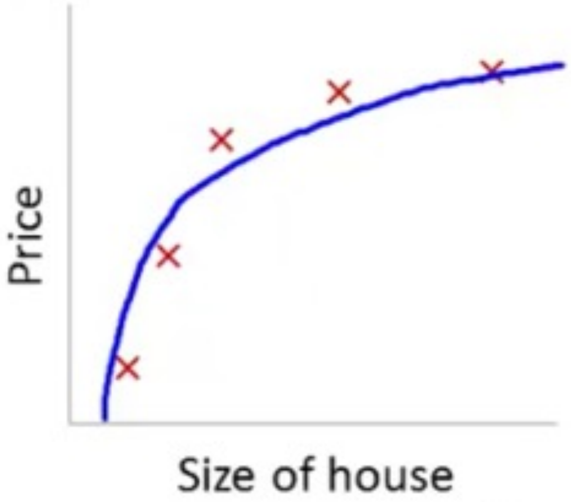
Addressing overfitting:

Options:

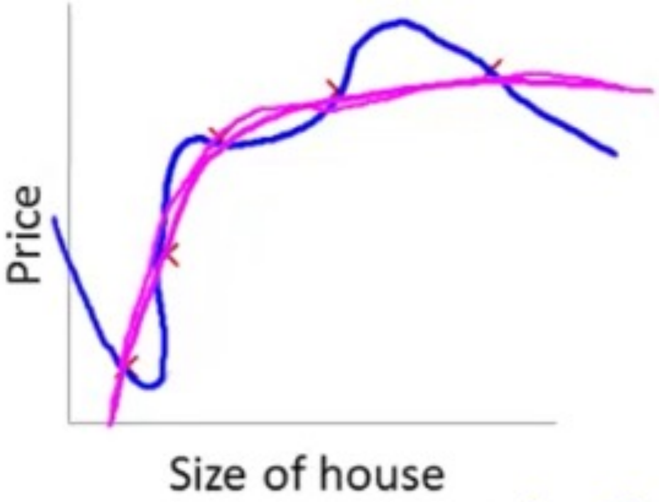
1. Reduce number of features
   - Manually select which features to keep
   - Model selection algorithm

2. Regularization
   - Keep all the features, but reduce magnitude/values of the parameters
   - Works well when we have a lot of features, each of which contributes a bit to predicting y

Regularization Intuition



$$\theta_0 + \theta_1 x + \theta_2 x^2$$



$$\theta_0 + \theta_1 x + \theta_2 x^2 + \cancel{\theta_3 x^3} + \cancel{\theta_4 x^4}$$

We want to reduce the contribution of the high degree terms of the polynomial to make the curve smoother. If we added an heavy term to the high degree parameters we are imposing the minimization function to make them really small:

$$\underset{\theta}{min} J(\theta) = \underset{\theta}{min} \frac{1}{2m} \sum_{i=1}^{m} (h(\mathbf{x}^{(\mathbf{i})}) - y^{(i)})^2 + 1000\theta_3 + 1000\theta_4$$

$$\theta_3 \approx 0 \quad \theta_4 \approx 0$$

Regularization

If we have small values for the parameters leads to

have Simple hypothesis that is less prone to overfitting

$$J(\theta) = \frac{1}{2m}\left(\sum_{i=1}^{m}(h(\mathbf{x}^{(\mathbf{i})}) - y^{(i)})^2\right)$$

$$J(\theta) = \frac{1}{2m}\left(\sum_{i=1}^{m}(h(\mathbf{x}^{(\mathbf{i})}) - y^{(i)})^2 + \lambda||\theta||_2^2\right)$$

Squared L2-Norm

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

$\theta_0 \quad not \quad included$

Lambda is the regularization parameter that controls the influence of regularization w.r.t. the hypothesis:

The bigger is, the smaller will be the theta and smoother will be the curve

Regularization for regression

In regularized linear regression, we choose to minimize

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

What if lambda is set to an extremely large value?

- Algorithm works fine; setting to be very large can’t hurt it.
- Algorithm fails to eliminate overfitting.
- Algorithm results in underfitting. (Fails to fit well even training data).
- Gradient descent will fail to converge.

Regularization for regression

In regularized linear regression, we choose to minimize

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

because the loss function empasize

What if lambda is set to an extremely large value?

- ○ Algorithm works fine; setting to be very large can't hurt it.
- ○ Algortihm fails to eliminate overfitting.
- ● Algorithm results in underfitting. (Fails to fit even training data well).
- ○ Gradient descent will fail to converge.

Fitting

Regularized linear regression

Regularization for linear regression

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

j=1 not from 0, because we dont want non associeted feature influencing the gradiant descend. in this case the bias

We want to derive the cost function to modify the update rule in the gradient descent:

Repeat until convergence {

how to generalize this?

$$\begin{cases} \theta_0 = \theta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(\mathbf{x}^{(\mathbf{i})}) - y^{(i)}\right)x_0^{(i)} \\ \theta_j = \theta_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(\mathbf{x}^{(\mathbf{i})}) - y^{(i)}\right)x_j^{(i)} + \frac{\lambda}{m}\theta_j\right] \quad j = 1, 2, \ldots, n \end{cases}$$

}

Regularization for linear regression

The update function can be re-written as

$$\theta_j = \theta_j \left(1 - \alpha \frac{\lambda}{m}\right) - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(\mathbf{x}^{(\mathbf{i})}) - y^{(i)}\right) x_j^{(i)} \quad j = 1, 2, \ldots, n$$

It is worth to note that the second part is same as before the regularization but we update the theta moving from a ratio of the old theta smaller than 1:

large

$$\underbrace{\left(1 - \underbrace{\alpha \frac{\lambda}{m}}_{0.1}\right)}_{0.9}$$

much smaller than 1

large

since we don't want to change the behavior of gradient descent we want that the 1-alpha lambda on m should be close to 1

Fitting

Regularized logistic regression

Regularization for logistic regression

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

We want to derive the cost function to modify the update rule in the gradient descent:

Repeat until convergence {

$$\begin{cases} \theta_0 = \theta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(\mathbf{x}^{(\mathbf{i})}) - y^{(i)}\right)x_0^{(i)} \\ \theta_j = \theta_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(\mathbf{x}^{(\mathbf{i})}) - y^{(i)}\right)x_j^{(i)} + \frac{\lambda}{m}\theta_j\right] \quad j = 1, 2, \ldots, n \end{cases}$$

}

Fitting

Regularization with L1 norm

Regularization with L1-norm

It is possible to regularize the cost function using L1 norm:

$$\ell^1 = \sum_{j=1}^{n} |\theta_j|$$

But L1-norm is no more derivable and standard gradient descent cannot be used.

Why should we consider (in some very specific cases) to use it?

Let us focus on the minimization of the norms:

$$\text{Norm } \ell^2 : \quad \min \sum_{i=1}^{m} (\dots)^2 \qquad \text{s.t.} \quad ||\theta||_2^2 < t$$

$$\text{Norm } \ell^1 : \quad \min \sum_{i=1}^{m} (\dots) \qquad \text{s.t.} \quad ||\theta||_1 < t$$

Limiting our analysis to two parameters

$$\text{Norm } \ell^2 : \quad \min \sum_{i=1}^{m} (\dots)^2 \qquad \text{s.t.} \quad \theta_1^2 + \theta_2^2 < t$$

$$\text{Norm } \ell^1 : \quad \min \sum_{i=1}^{m} (\dots) \qquad \text{s.t.} \quad |\theta_1| + |\theta_2| < t$$

we use L2 because we can
calculate the partial derivate

lagrangia multiplier
give a function f(x)
we give a function
g(x)<t
and we minize
f(x)+\lambda g(x)

Regularization with L1-norm

$\sum(\dots)^2$ denotes a parabolic curve equation;

$\theta_1^2 + \theta_2^2 < t$ denotes the internal area of a circle;

$|\theta_1| + |\theta_2| < t$ denotes the internal area of a rhombus.

During the minimization we want to minimize that parabolic curve and lie inside the circle or the rhombus.

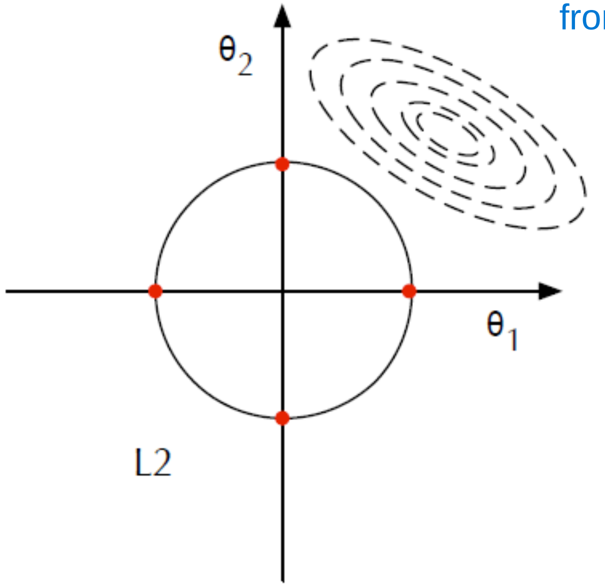
It can be proved that it is easier, in the L1 case, to reach the intercept with the axis and hence to set that parameter to zero.

This lead to a simple but effective feature selection.

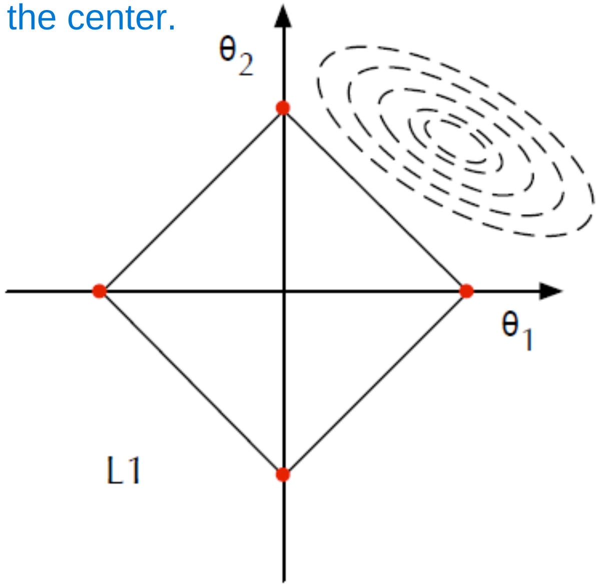
the dotted circle is the minimization function

we want to find the intersection between the two, preferably on the center of the minimization function.

in the l2 the probabilities of intercept the minimization function is equal between the point. in l1 instead not, because some point are farther from the center.



L2



L1

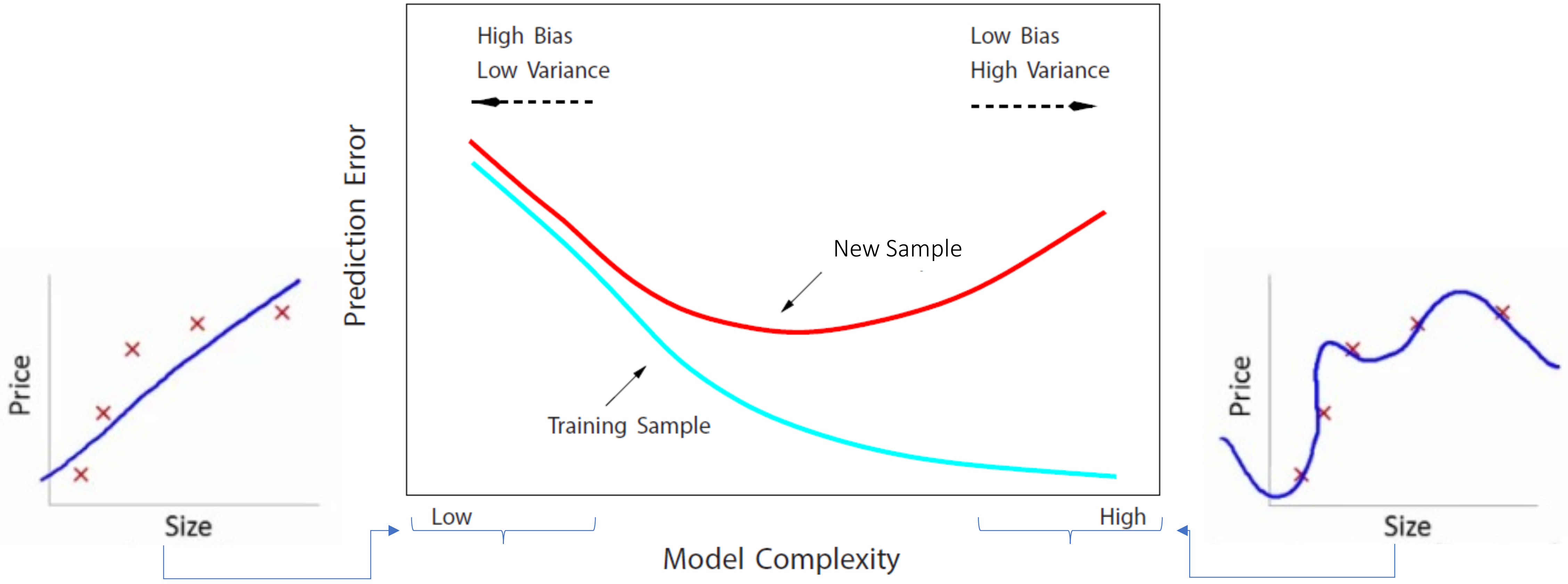
it is used the l1 norm for feature selection while computing the optimal model.

Fitting

Regularization vs Bias/Variance

Training set and test set w.r.t. complexity

Let us recap the Bias variance behavior w.r.t. the complexity of the model



Training set and new samples w.r.t. regularization