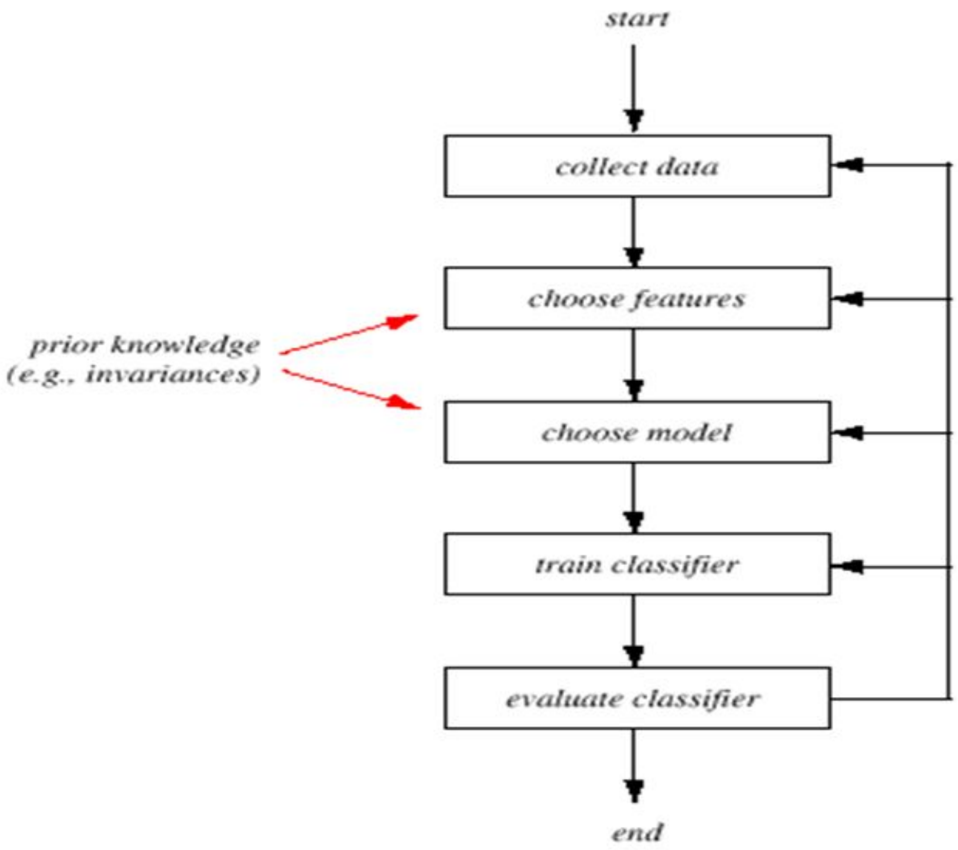
Machine Learning in Practise

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY
DELHI

The Design Cycle



Computational Complexity

- What is the trade-off between computational ease and performance?

- How an algorithm scales as a function of the number of features, patterns or categories?

Performance Evaluation of Learning Tasks

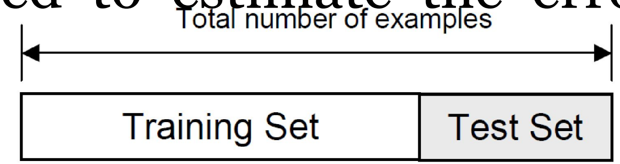
- Entire population is unavailable

- Finite set of training data, usually smaller than desired

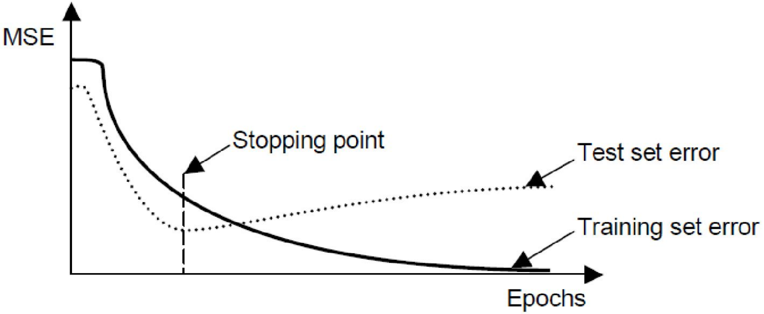
- Naïve approach: use all available data
  - The final model will typically **overfit** the training data
    - More pronounced with high-capacity models (e.g., neural nets)
  - The true error rate is **underestimated**
    - Not uncommon to have 100% accuracy on training data

Validation Method: Holdout

- Split dataset into two groups:
  - Training set: used to train the model
  - Test set: used to estimate the error rate of the trained model



- Typical application: early stopping

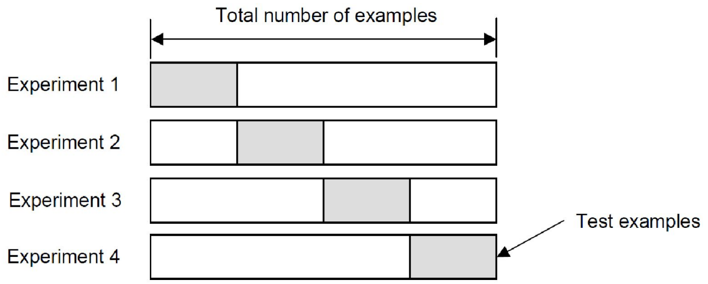


Holdout

- Drawbacks
  - For small training sets, setting aside a subset may be infeasible
    - Sample Size = 10 => Training set = 7, Testing set = 3
  - For a single train-and-test experiment, the holdout estimate of error rate will be misleading if we happen to get an '*unfortunate*' split
    - Training Set = 1,0,1,0,0,0,0, Testing set = 1,1,1
- Alternatives: a family of resampling methods: Cross Validation
    - Random Subsampling
    - Leave-one-out Cross-Validation
    - K-Fold Cross-Validation

Validation Method: K-Fold Cross-validation

- Create a K-fold partition of the dataset
  - For each of K experiments, use K-1 folds for training and the remaining one for testing



- True error is estimated as the average error rate

$$E = \frac{1}{K}\sum_{i=1}^{K} E_i$$

Validation Method: K-Fold Cross-validation

- Example: [0.1, 0.2, 0.3, 0.4, 0.5]
- K = 3

Fold	Training Set	Testing Set
1	[0.1, 0.2, 0.3]	[0.4, 0.5]
2	[0.1, 0.2, 0.5]	[0.3, 0.4]
3	[0.1, 0.4, 0.5]	[0.2, 0.3]

Validation Method: K-Fold Cross-validation

- Example: [0.1, 0.2, 0.3, 0.4, 0.5]
- K = 3

Fold	Training Set	Testing Set	Predictions
1	[0.1, 0.2, 0.3]	[0.4, 0.5]	[0.3, 0.5]
2	[0.1, 0.2, 0.5]	[0.3, 0.4]	[0.35, 0.35]
3	[0.1, 0.4, 0.5]	[0.2, 0.3]	[0.23, 0.33]

Calculate average error rate

Validation Method: K-Fold Cross-validation

Fold	Testing Set	Predictions	Error Rate
1	[0.4, 0.5]	[0.3, 0.5]	0.005
2	[0.3, 0.4]	[0.35, 0.35]	0.0025
3	[0.2, 0.3]	[0.23, 0.37]	0.0058

Average Error Rate= 0.0044

How many folds are needed?

- Large number of folds
  - + smaller bias of the true error rate estimator
  - - larger variance of the true error rate estimator
  - - higher computational time (many experiments)
- Small number of folds
  - + lower computation time
  - + smaller variance
  - - larger bias
- In practice, the choice of the number of folds depends on the size of the dataset
  - For large datasets, even 3-Fold Cross Validation is reasonable
  - For very sparse datasets, ‘*leave-one-out*’ is beneficial
- A common choice for K-Fold Cross Validation is K=10

Bias and Variance

Two ways to measure the “match” or “alignment” of the learning algorithm.

- Bias measures accuracy of the match: high => poor match
  - Bias arises when the classifier cannot represent the true function – that is, the classifier underfits the data
- Variance measures precision of match: high => weak match
  - Variance arises when the classifier overfits the data.
- There is often a tradeoff between bias and variance.

Bias and Variance

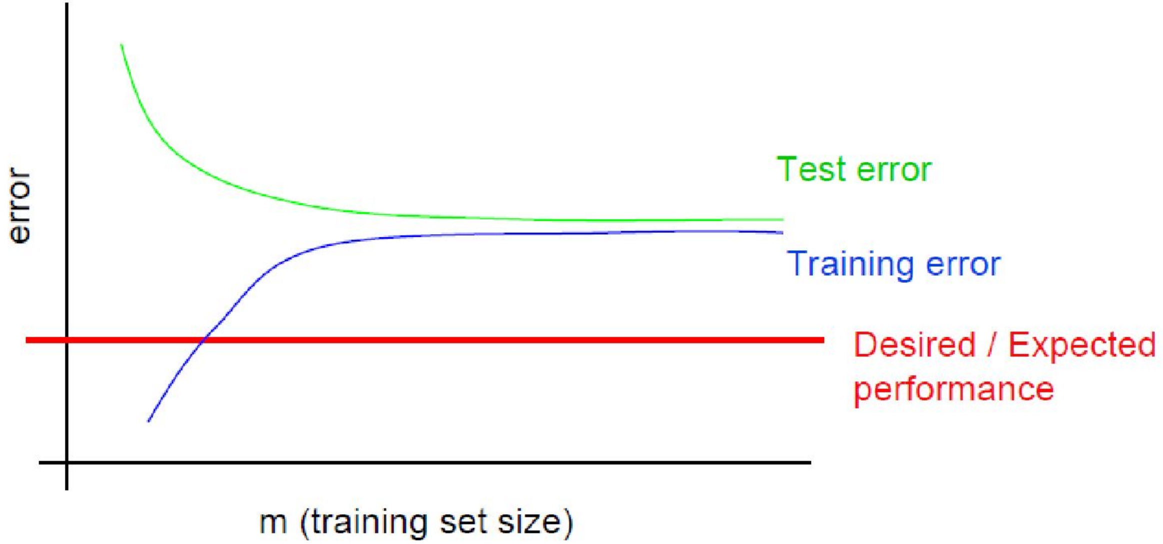
Calculate bias, variance and give conclusions.

Training error	Dev error	Bias	Variance	Conclusions
1%	11%			
15%	16%			
15%	30%			
0.5%	1%			

Bias and Variance

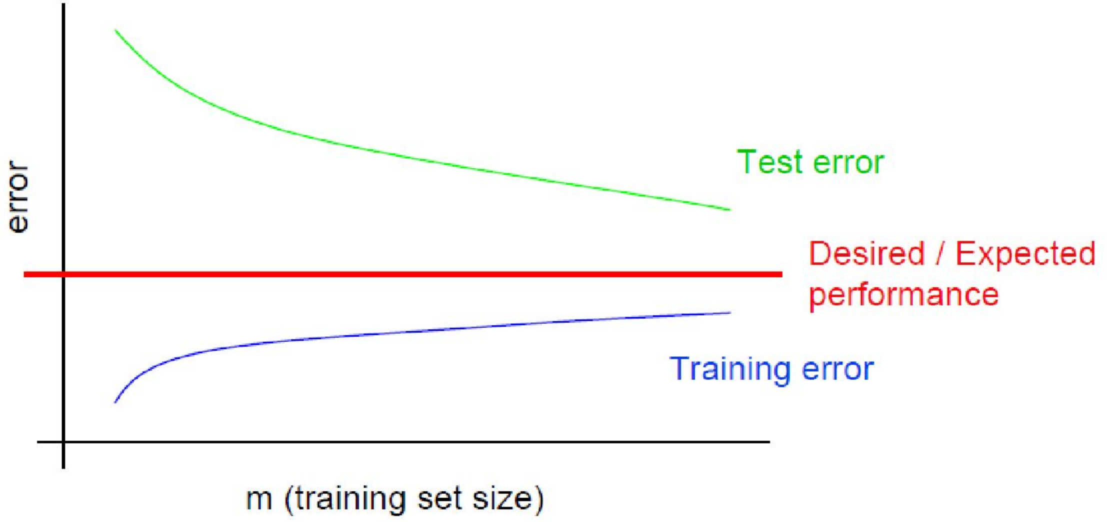
Training error	Dev error	Bias(Training error)	Variance(Dev error - Training error)	Conclusions
1%	11%	1%	10%	High variance, overfitting
15%	16%	15%	1%	High bias, underfitting
15%	30%	15%	15%	High bias and high variance, poor performance
0.5%	1%	0.5%	0.5%	Low bias and low variance, good performance

Bias vs. Variance Analysis: High Bias



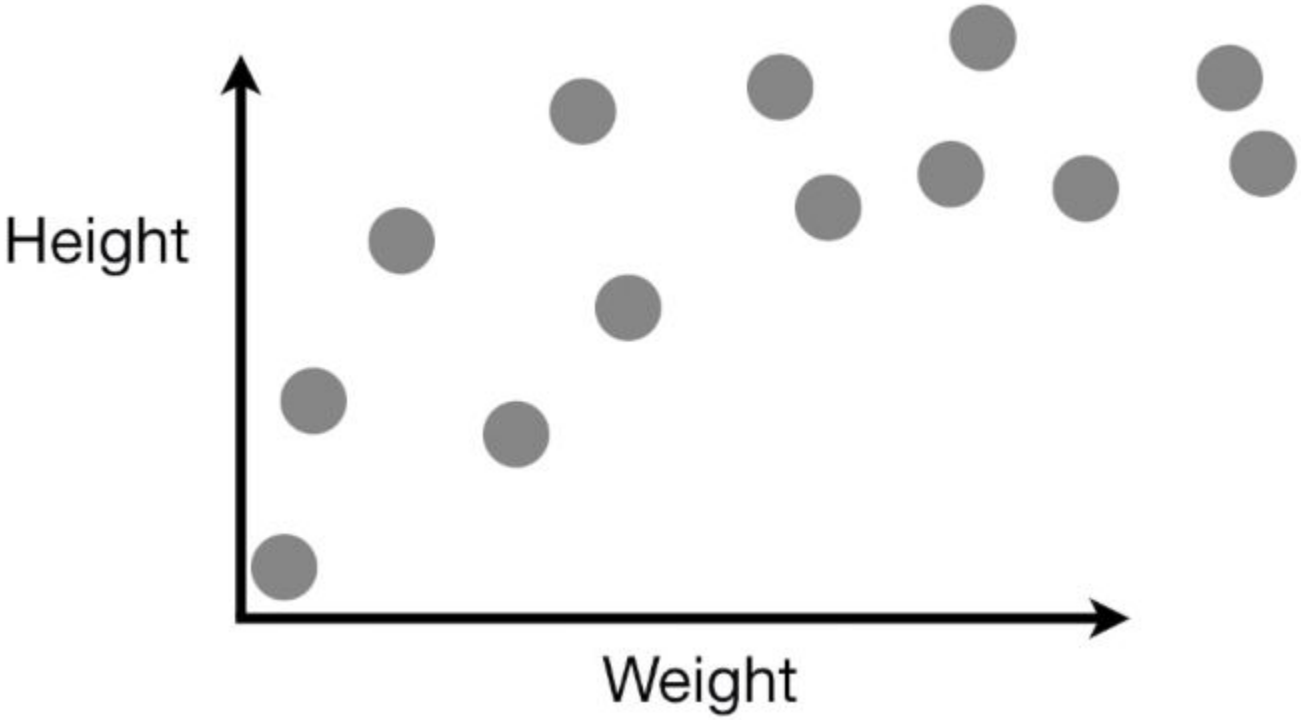
- Even training error is unacceptably high.
  - Features are not discriminative enough
- Small gap between training and test error.
  - Likely underfitting: a higher capacity model could be tried

Bias vs. Variance Analysis: High Variance

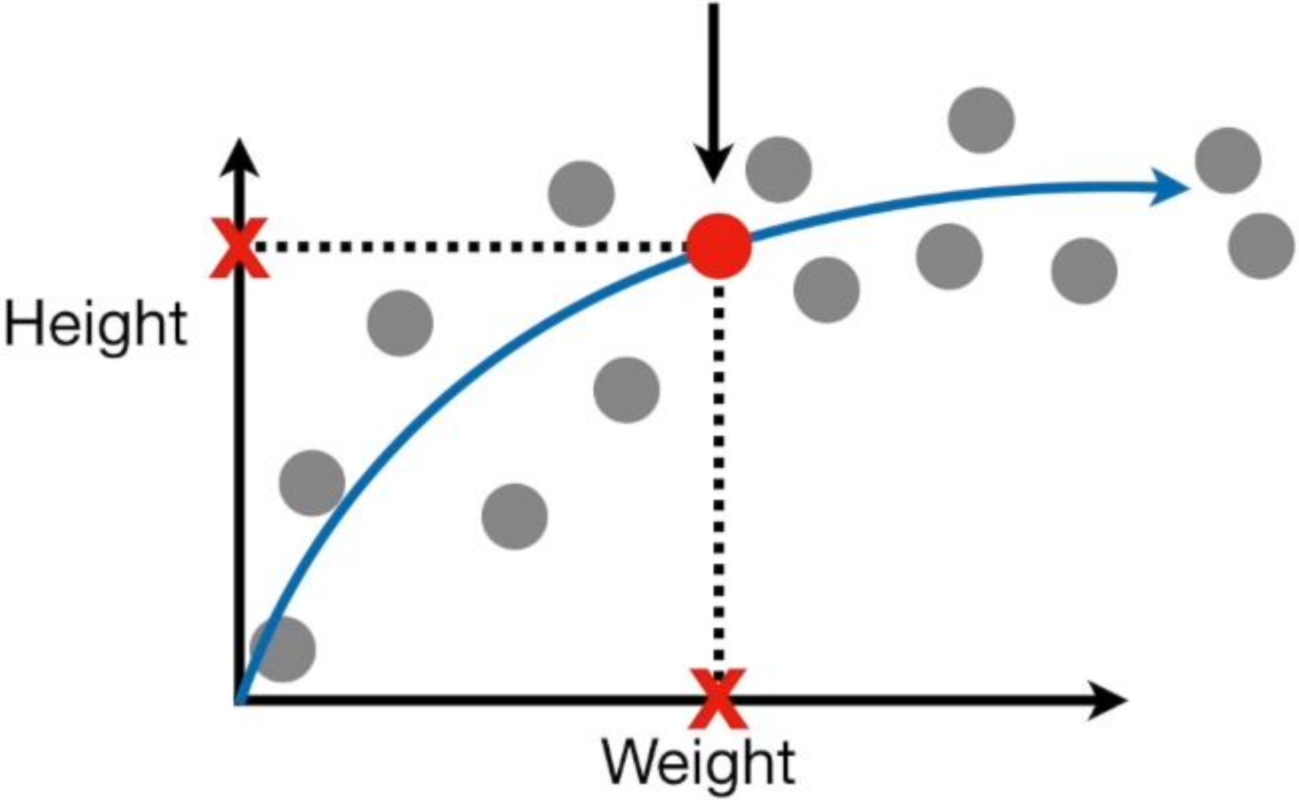


- Test error still decreasing as training set size increases.
    - Suggests a larger training set will help.
- Large gap between training and test error
    - Likely overfitting

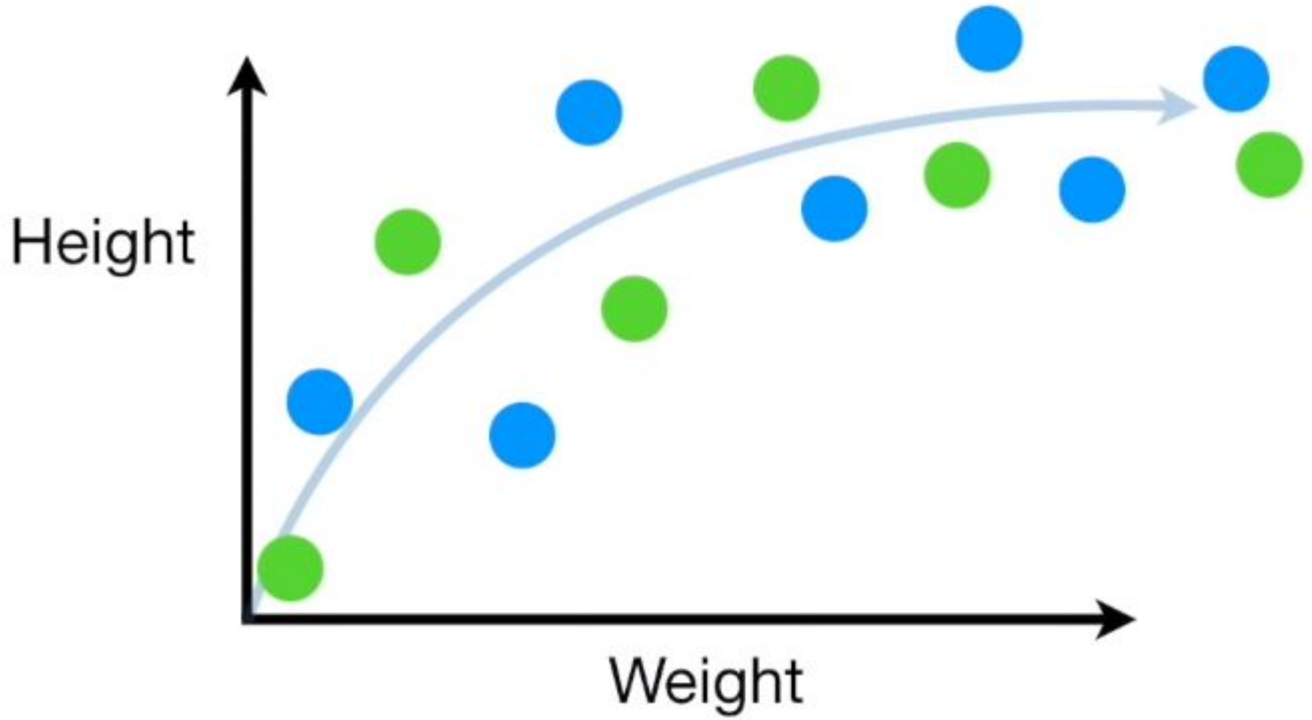
A sample data [Regression]



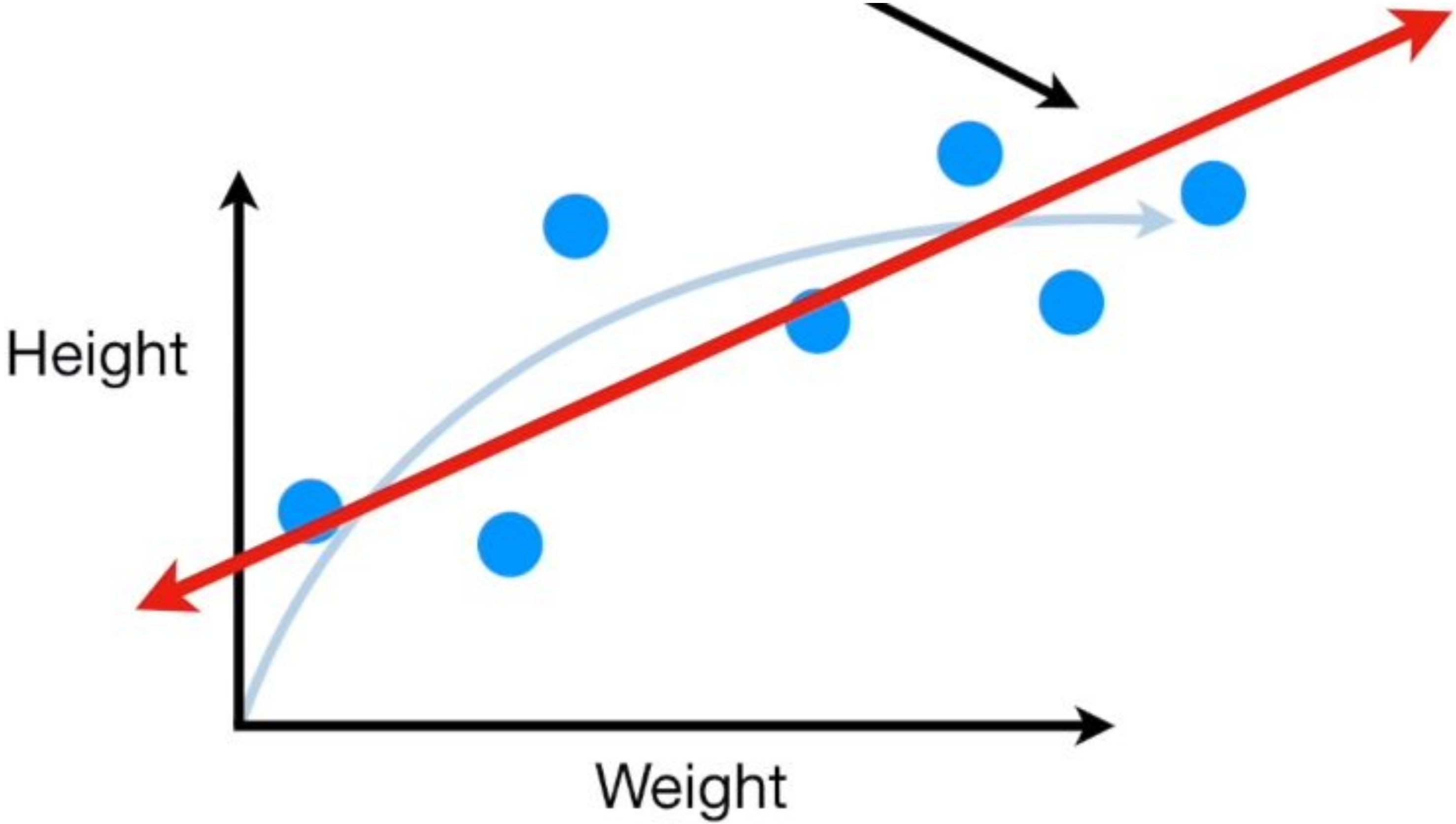
True Relationship



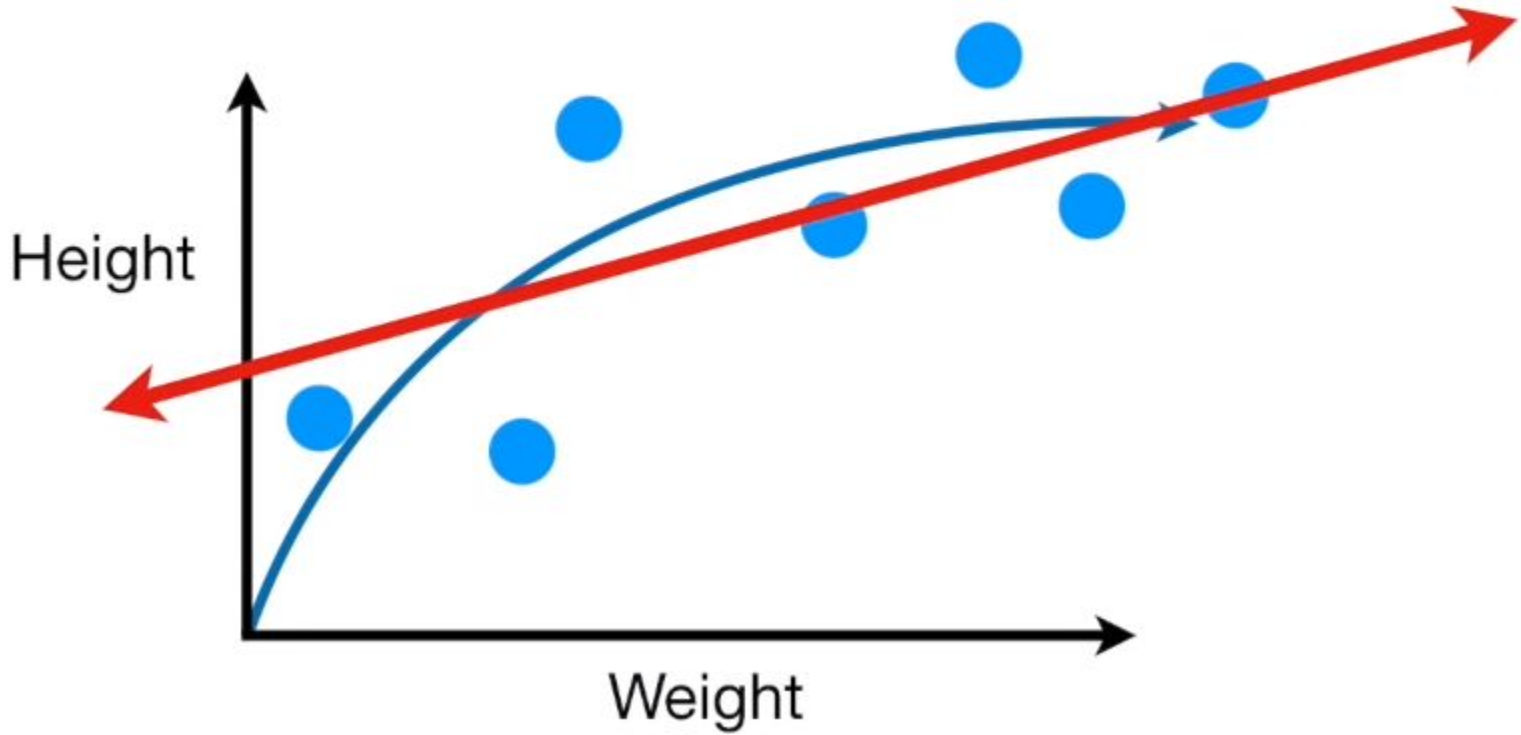
Training and Testing Data



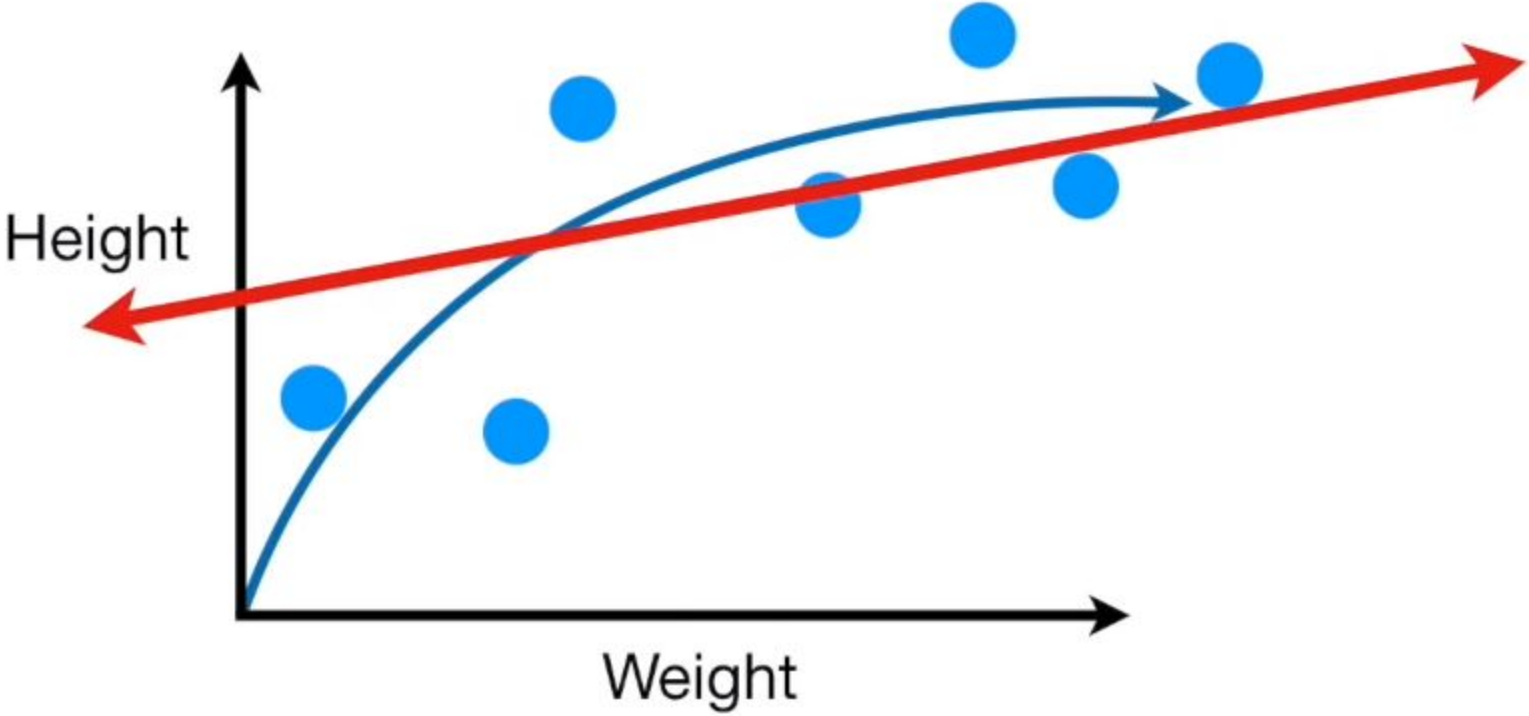
Linear Regression Fit - I



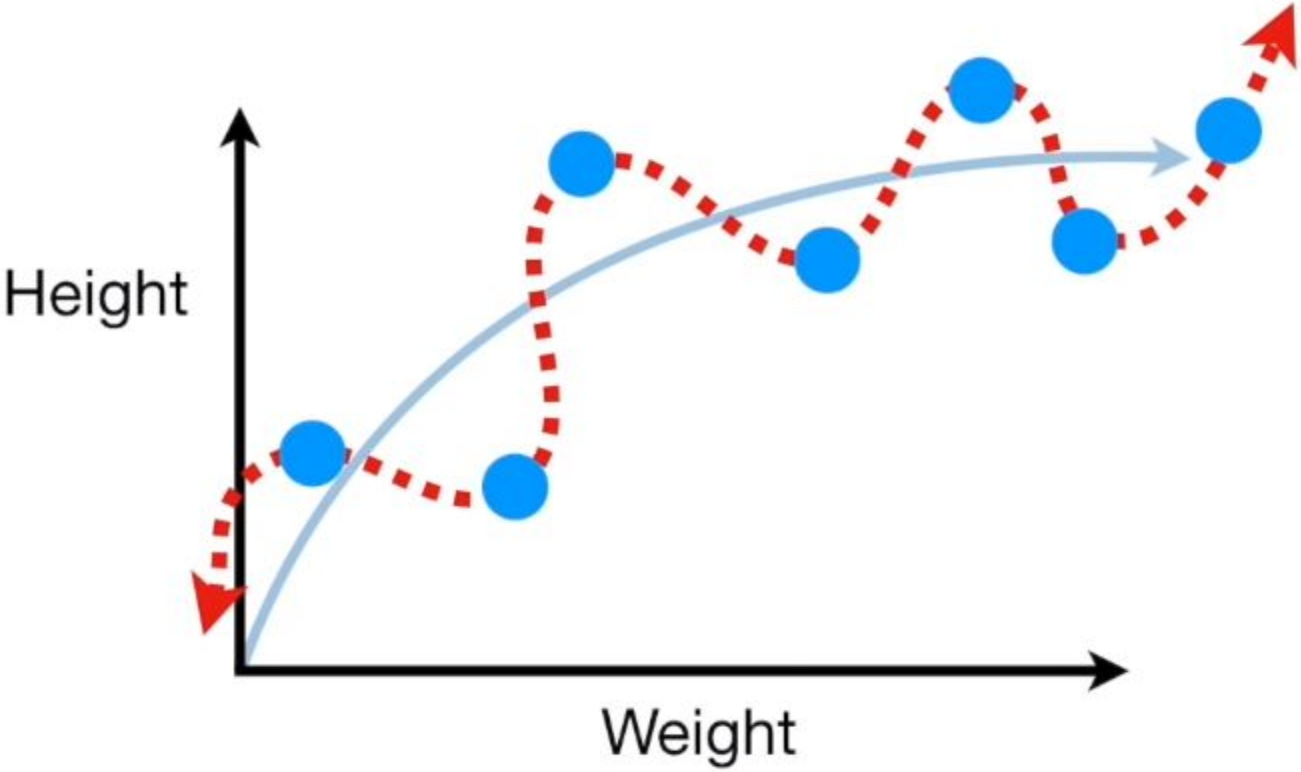
Linear Regression Fit - II



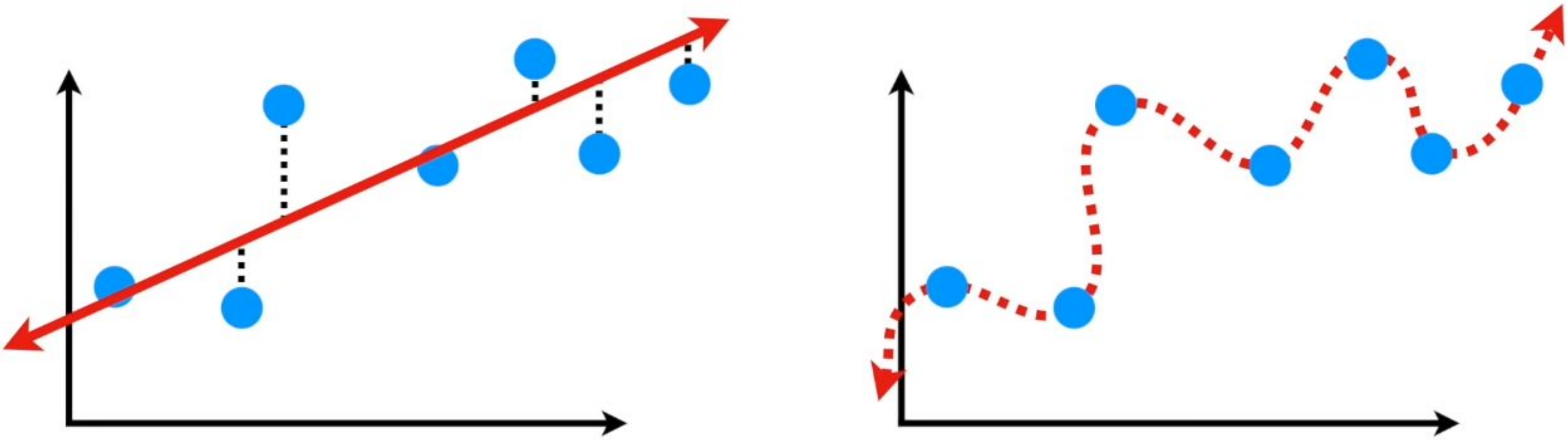
Linear Regression Fit - III



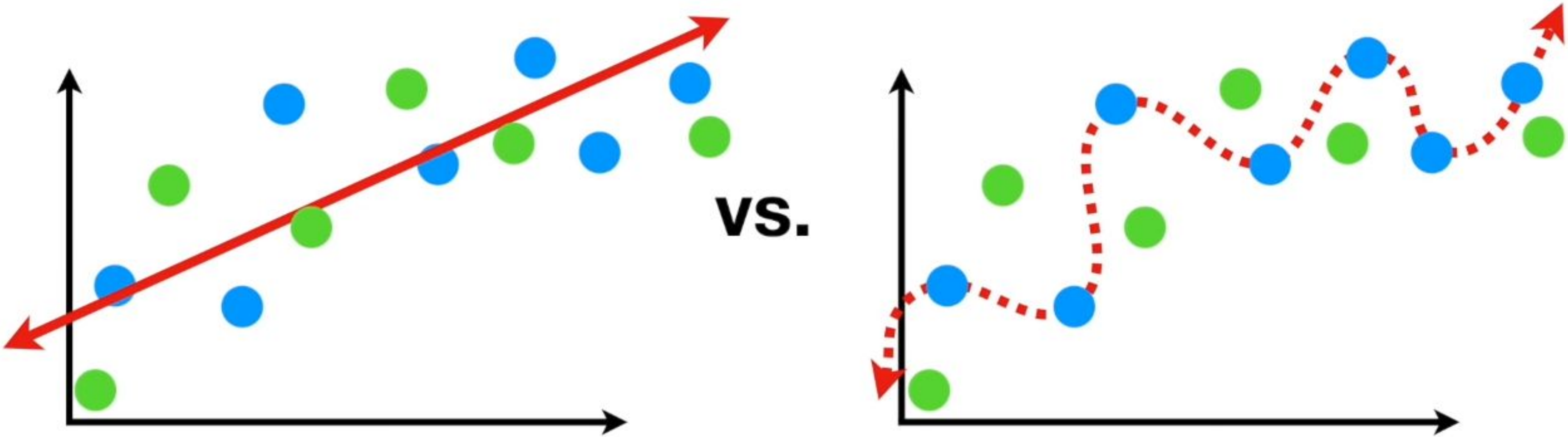
Polynomial Fit



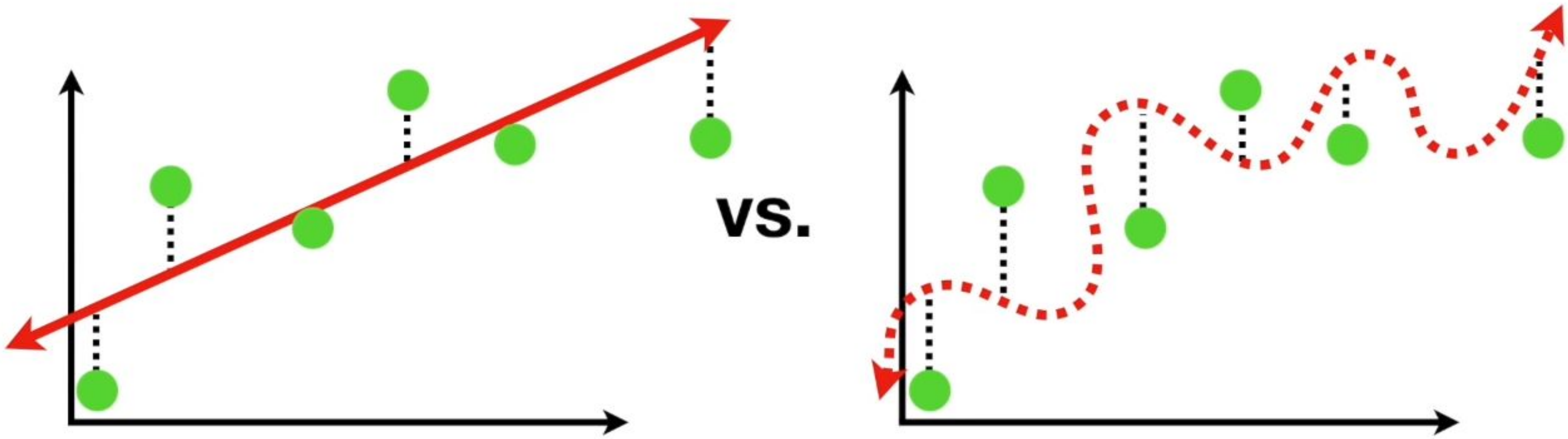
Bias



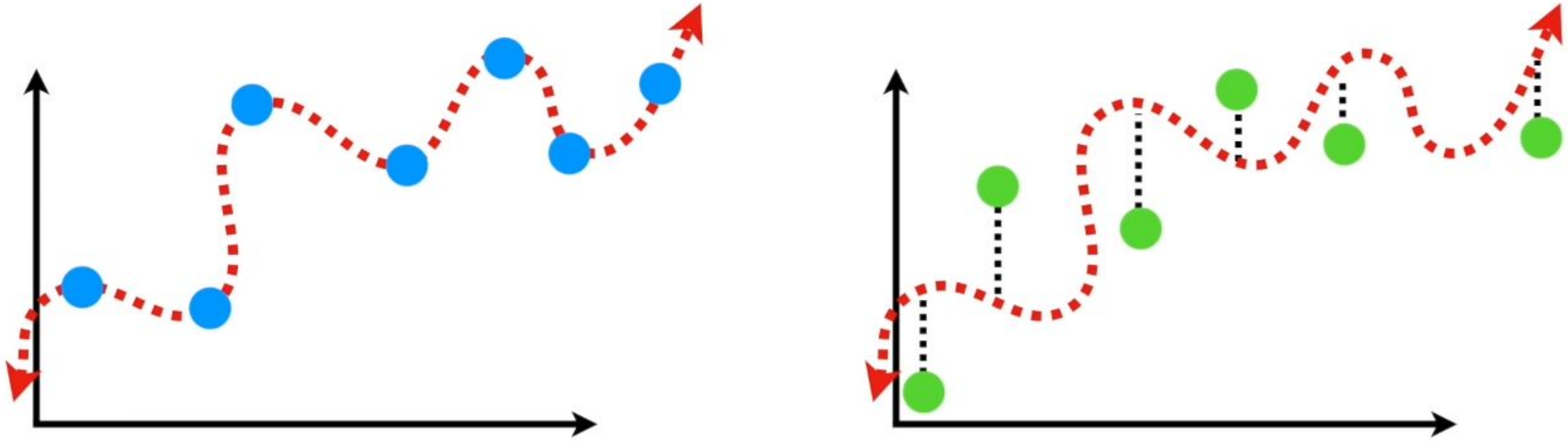
Performance on Testing Data



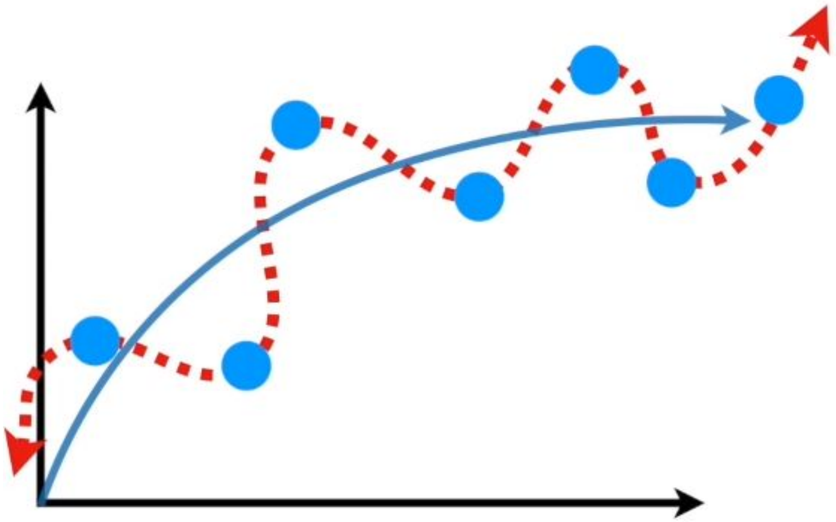
Performance on Testing Data

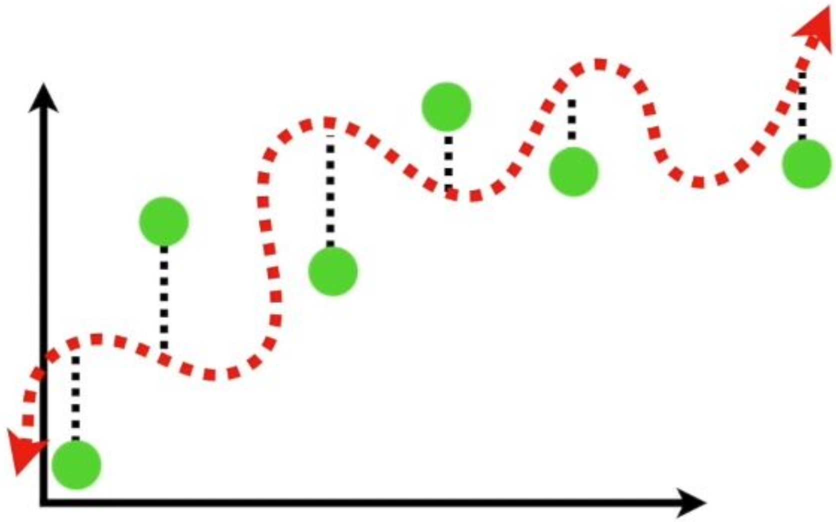


Variance

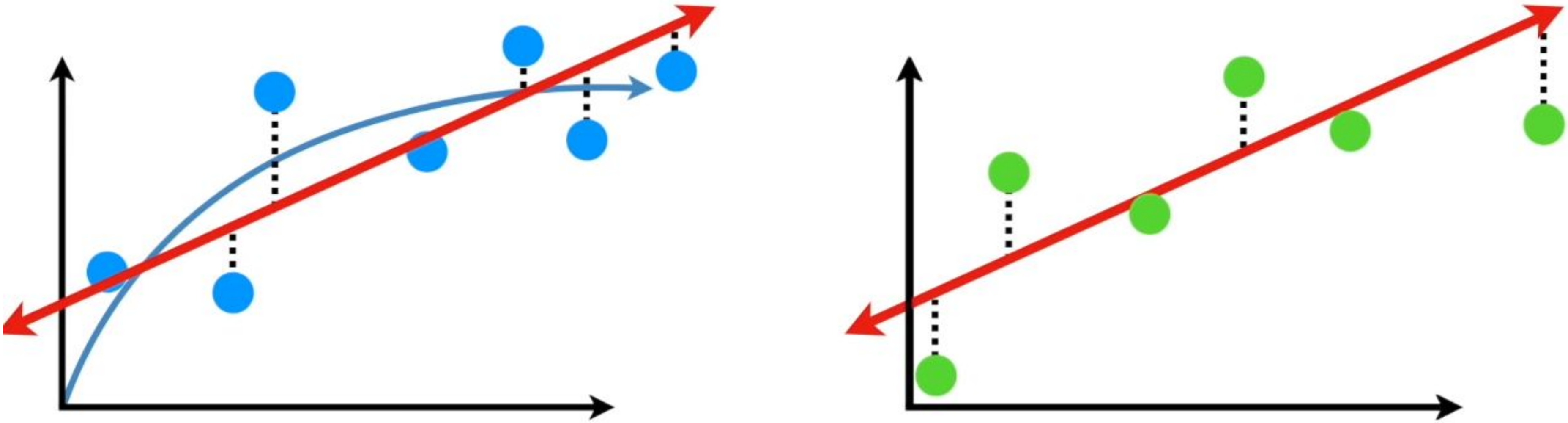


Low Bias and High Variance: Overfit

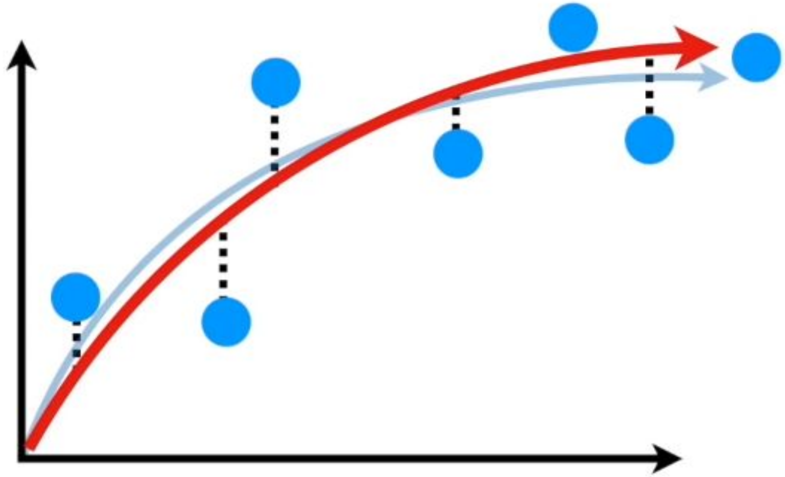


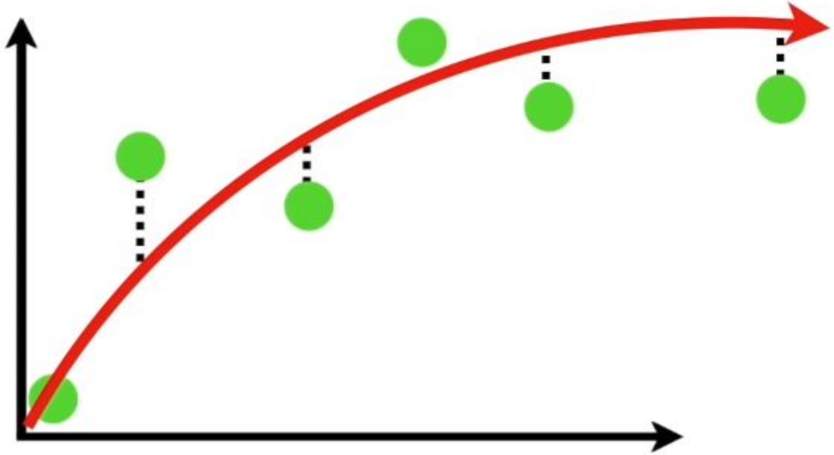


High Bias and Low Variance: Underfit

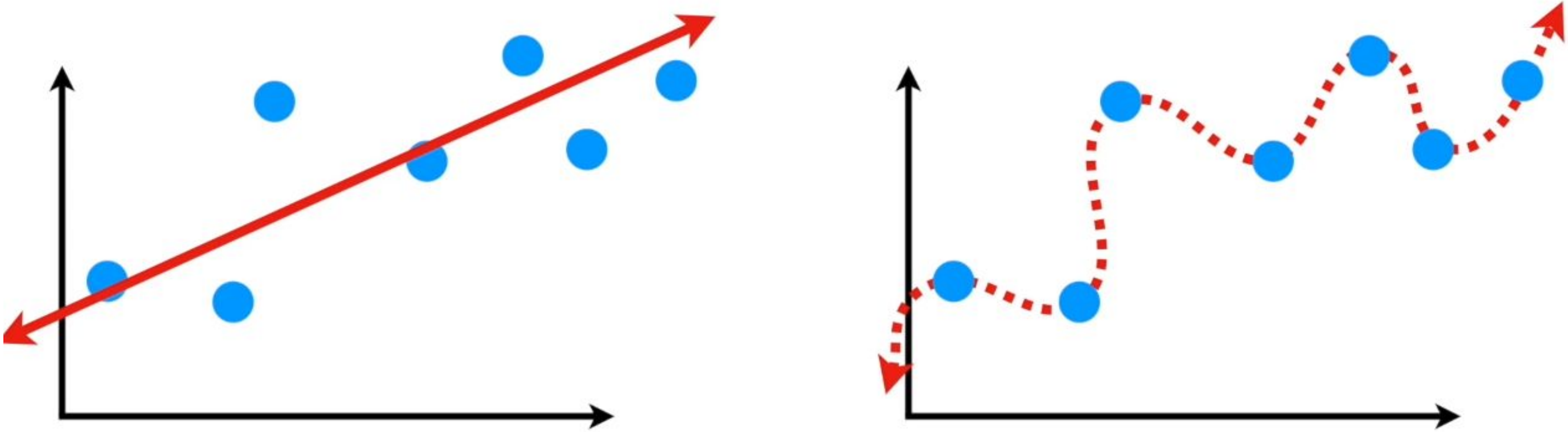


Ideal: Low Bias and Low Variance

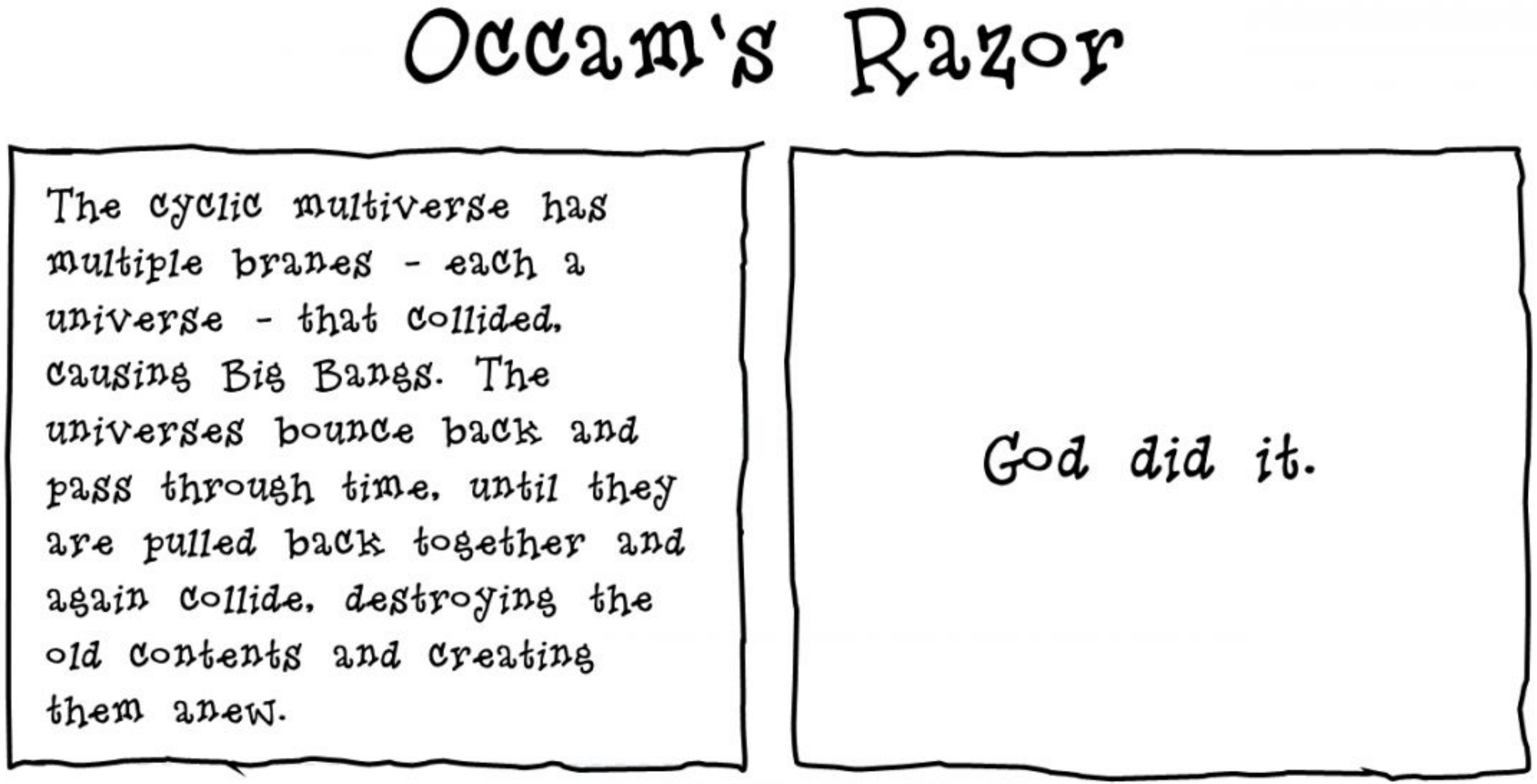




Optimal Solution in between



Occam’s Razor



Revisiting MSE

- Known (Observed) function is $y = f(x) + \varepsilon$, where
  - Observed value = y; actual value = f(x)
  - ε is normally distributed with zero mean and standard deviation σ

- Given a set of training examples, $\{(x_i, y_i)\}$,
  - we fit an hypothesis $h(x) = w^T x + b$ to the data to minimize the squared error; $MSE = \Sigma_i [y_i - h(x_i)]^2$

- Given a new data point x^* (with observed value $y^* = f(x^*) + \varepsilon$), we would like to understand the expected prediction error
  $E[(h(x^*) - y^*)^2]$

Bias-Variance-Noise Decomposition: Lemma

- Let Z be a random variable with probability distribution P(Z)
- Let $\underline{Z} = E_P[\,Z\,]$ be the average value of Z
- Lemma: $E[(Z - \underline{Z})^2] = E[Z^2] - \underline{Z}^2$

- $E[(Z - \underline{Z})^2] = E[Z^2 - 2\,Z\underline{Z} + \underline{Z}^2]$
  $= E[Z^2] - 2E[Z]\underline{Z} + \underline{Z}^2$
  $= E[Z^2] - 2\underline{Z}^2 + \underline{Z}^2$
  $= E[Z^2] - \underline{Z}^2$

- Corollary: $E[Z^2] = E[\,(Z - \underline{Z})^2\,] + \underline{Z}^2$

Bias-Variance-Noise Decomposition: Derivation

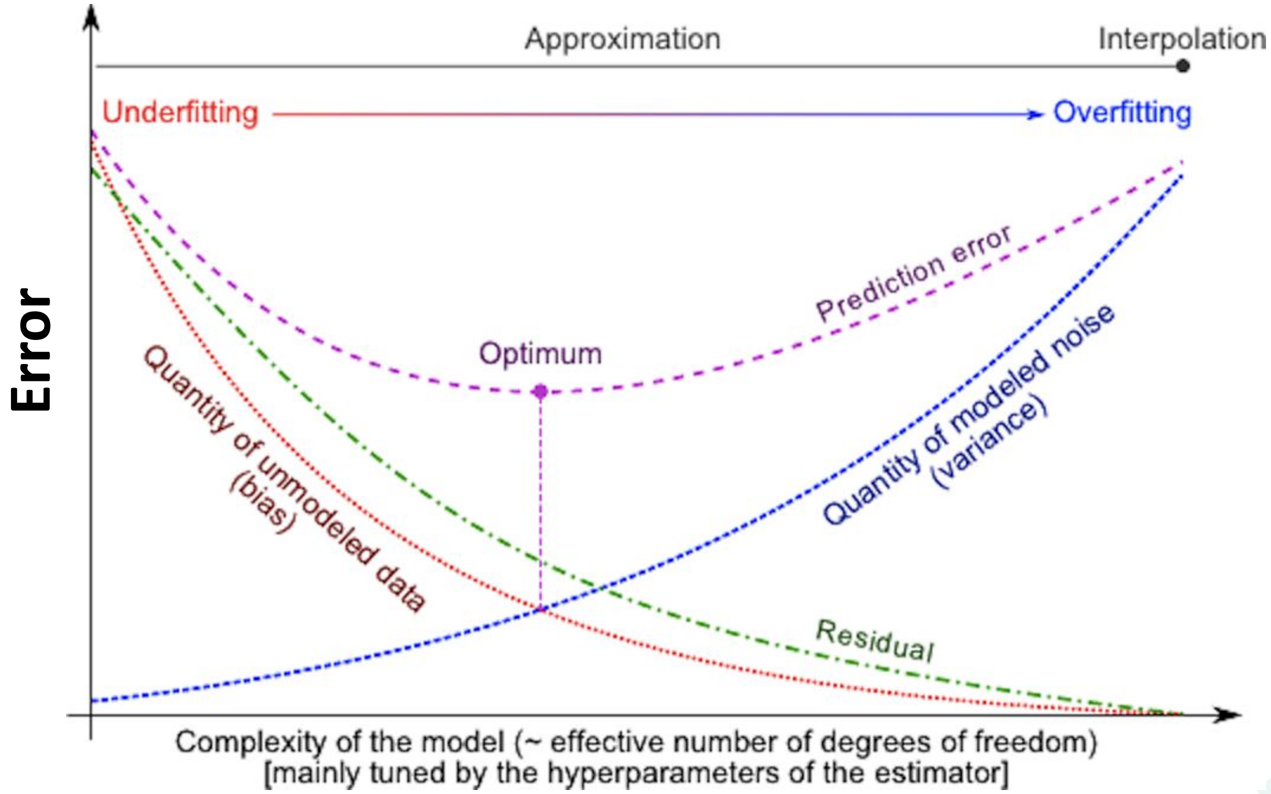
- $E[(h(x^*) - y^*)^2] = E[h(x^*)^2 - 2h(x^*)\, y^* + y^{*2}]$
  $= E[\, h(x^*)^2] - 2E[h(x^*)]E[y^*] + E[y^{*2}]$
  $= E[(h(x^*) - \underline{h(x^*)})^2] + \underline{h(x^*)}^2$ (lemma)
  $- 2\, \underline{h(x^*)} f(x^*)$ $(E(y^*) = E[f(x^*) + \varepsilon] = f(x^*))$
  $+ E[(y^* - f(x^*))^2] + f(x^*)^2$ (lemma)
  $= E[\, (h(x^*) - \underline{h(x^*)})^2] +$ [variance]
  $(\underline{h(x^*)} - f(x^*))^2 +$ [bias²]
  $E[\, (y^* - f(x^*))^2]$ [noise]

Bias-Variance-Noise Decomposition

- $E[\ (h(x^*) - y^*)^2\ ] = E[\ (h(x^*) - \underline{h(x^*)})^2] + (\underline{h(x^*)} - f(x^*))^2 + E[\ (y^* - f(x^*))^2\ ]$, where
  $\underline{h(x^*)} = E_p[\ h(x^*)\ ]$ be the average value of $h(x^*)$
  $= Var(h(x^*)) + Bias(h(x^*))^2 + E[\ \varepsilon^2\ ]$
  $= Var(h(x^*)) + Bias(h(x^*))^2 + \sigma^2$
- Expected prediction error = Variance + Bias^2 + Noise^2

- Variance = Describes how much the model varies from one training set to another training set.
- Bias^2 = Describes the <u>average-error</u> of the model.
- Noise^2 = Describes how much *actual value* varies from *known value*

Bias vs. Variance Analysis



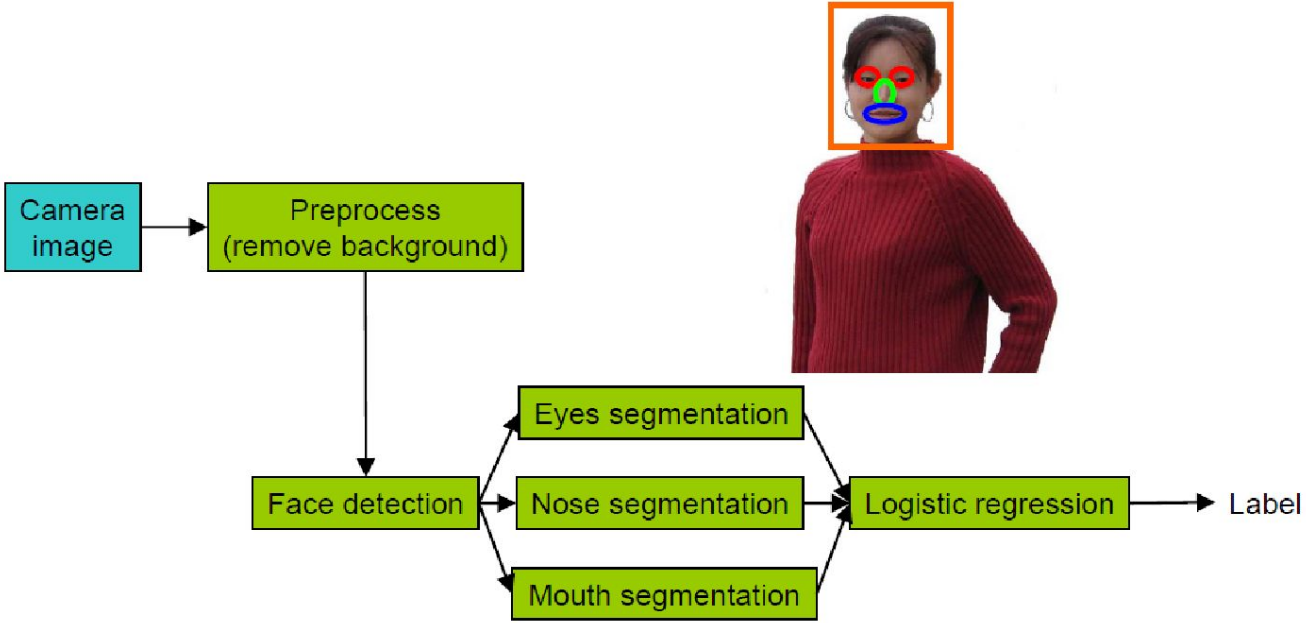
Diagnostics for ML Algorithms

- Try getting more training examples.
- Try a smaller set of features.
- Try a larger set of features.
- Try changing the features.
- Run gradient descent for more iterations.
- Try Newton's method instead of gradient descent.
- Use a different value for reg. parameter λ.
- Try using a different model (e.g., SVM).

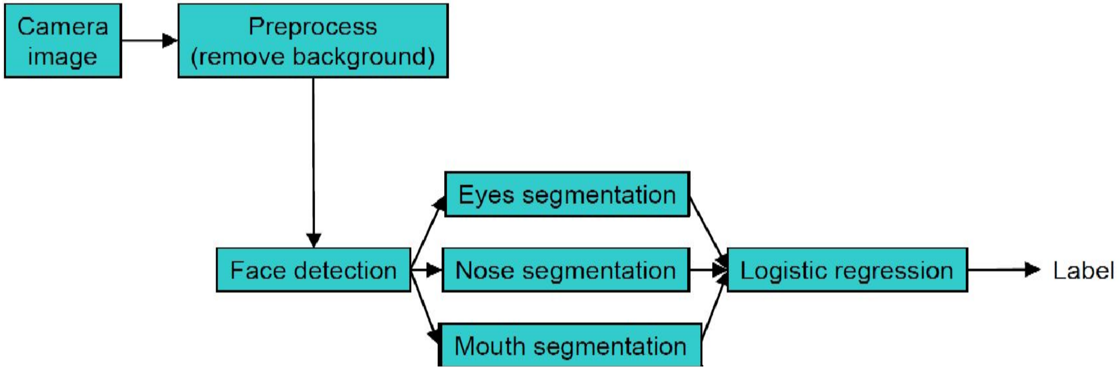
- Fixes high variance.
- Fixes high variance.
- Fixes high bias.
- Fixes high bias.
- Fixes optimization algorithm.
- Fixes optimization algorithm.
- Fixes optimization objective.
- Fixes optimization objective

Debugging ML Systems

- Many applications combine many different learning components into a “pipeline”, e.g., Face recognition from images [toy example].



Error Analysis



How much error is attributable to each of the components?

Plug in ground-truth for each component, and see how accuracy changes.

Conclusion: Most room for improvement in face detection and eyes segmentation.

Component	Accuracy
Overall system	85%
Preprocess (remove background)	85.1%
Face detection	91%
Eyes segmentation	95%
Nose segmentation	96%
Mouth segmentation	97%
Logistic regression	100%