Validation and Polynomial Regression

Learning Goals I

At the end of this lecture you should be able to:

Validation

- Understand what validation is and when to do it
- Split your data in good train and test sets
- Understand k-fold cross validation and when to use it
- Validate a model using cross validation

Learning Goals II

At the end of this lecture you should be able to:

Polynomial Regression

- Fit a **Polynomial** Regression model

Cost Functions

- Understand what is a **Cost Function**
- Understand **Mean Square Error, Bias** and **Variance**
- Understand the Bias / Variance **tradeoff**
- Understand **Underfitting** and **Overfitting**
- Know how to **fix** Underfitting and Overfitting models

Linear Regression Recap

Recap

What are the steps for Linear Regression?

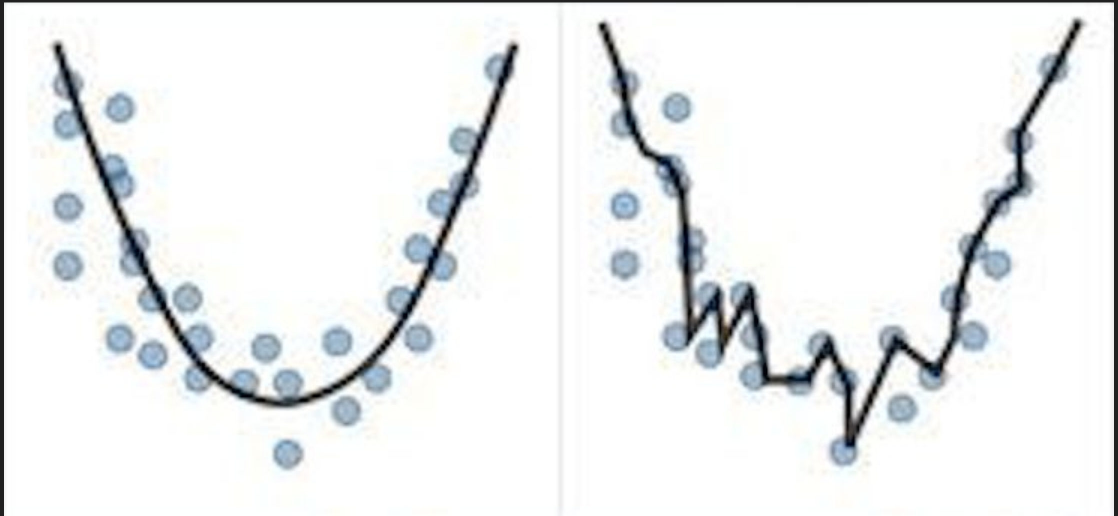
How will you know you've done it well?

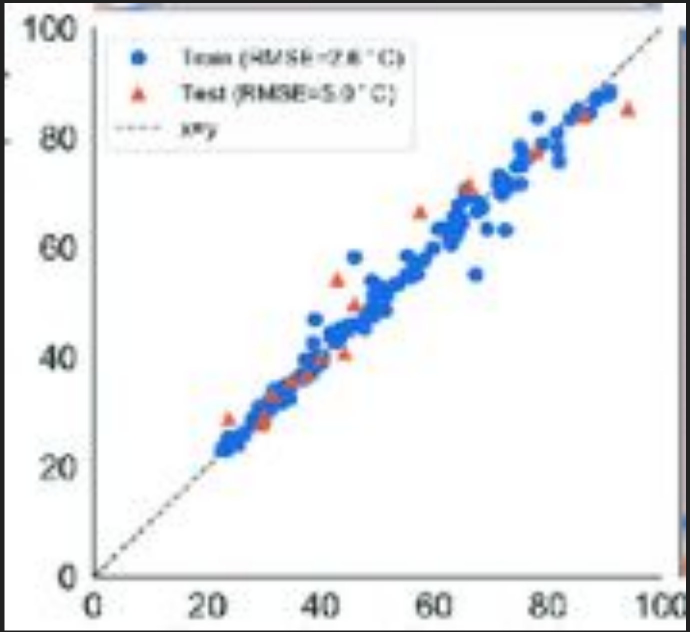
Validation

Validation

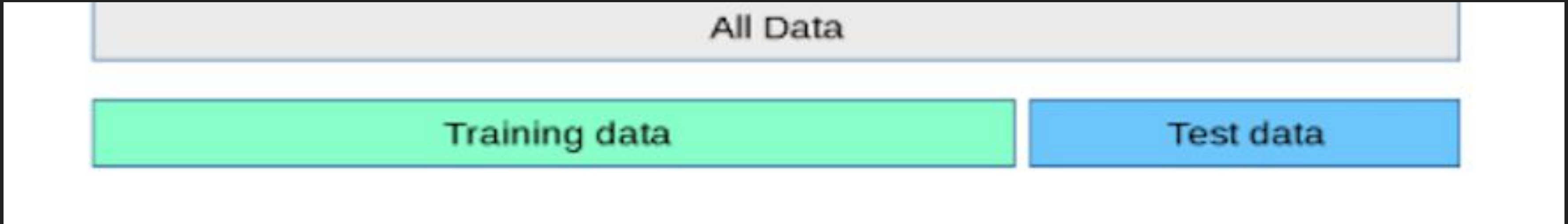
Process of checking the performance of your model on unseen data

Helps us to figure out if the model will **Generalise**





Train test split

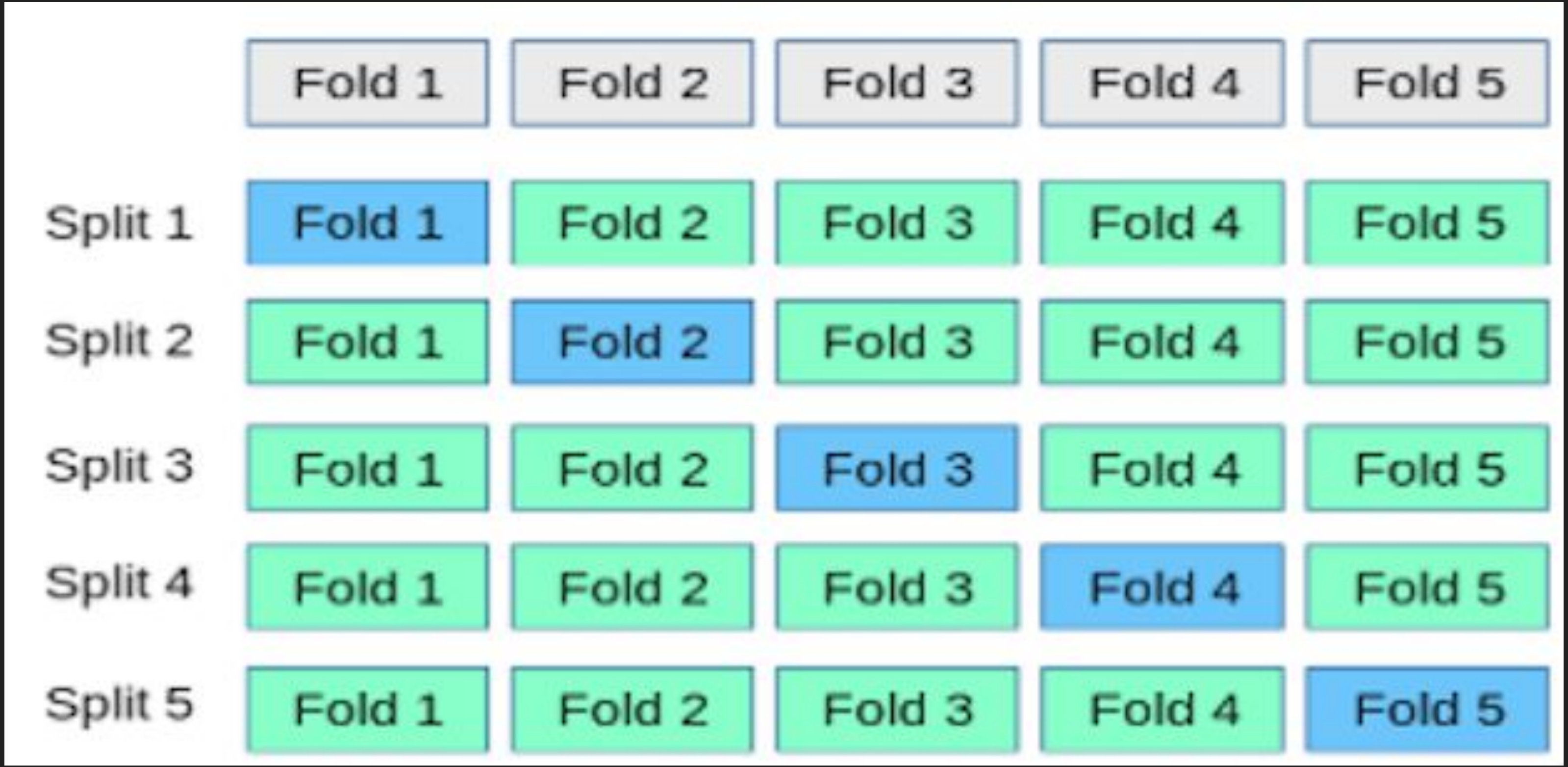


Train test split (6 steps)

1. Get your data in a numpy or pandas format
2. Separate your **predictors** from your **target** variable (X,y)
3. Import the **train_test_split** function (from sklearn.model_selection)
4. Pass X, y and your desired % of data to be allocated to the test set to the function
5. Train your model on the train dataset
6. Score your model on the test dataset

DEMO

Your turn to practice train_test_split
Are you ready?!


Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 1
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 2
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 3
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 4
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 5
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5

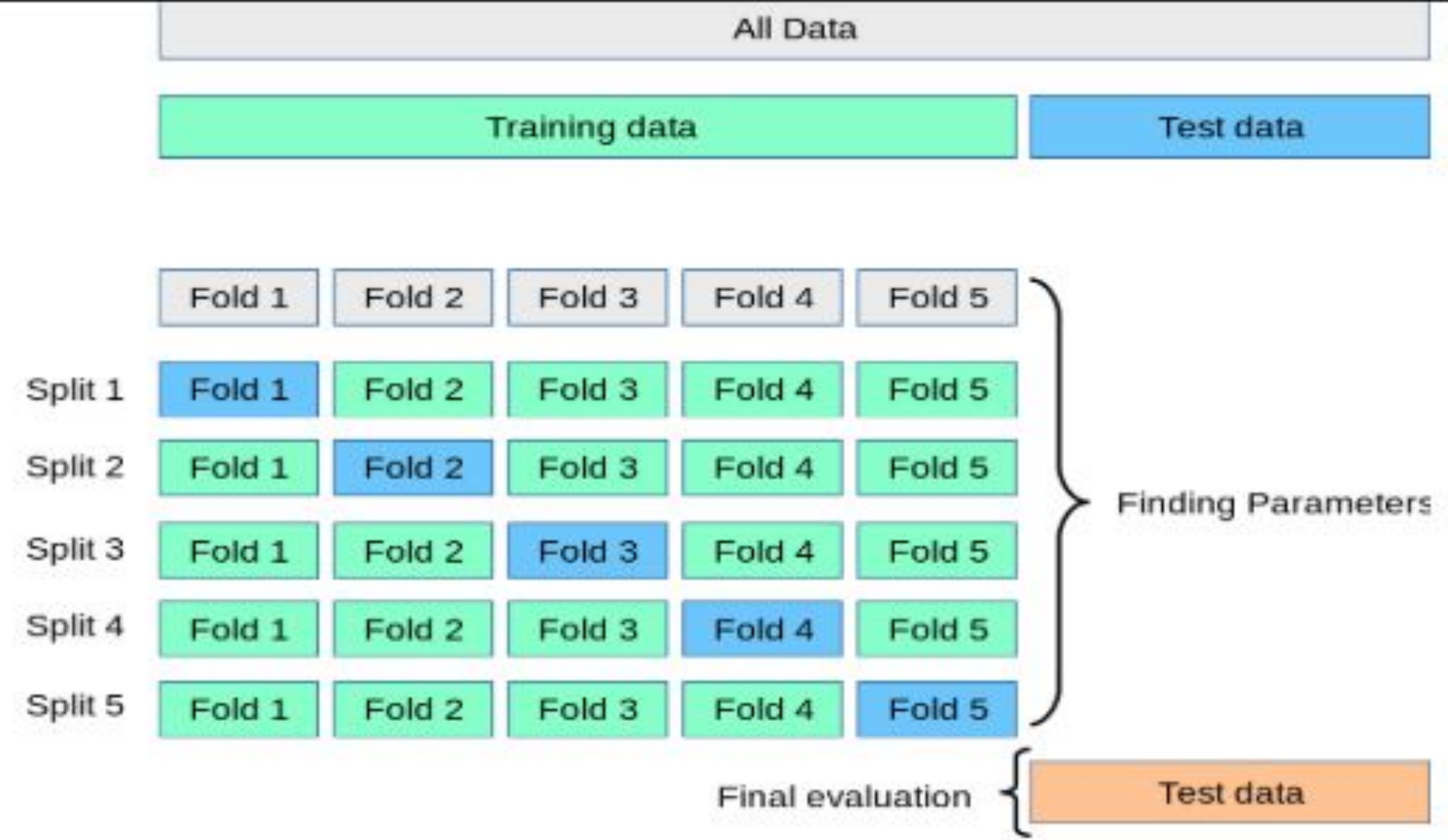
cross_val_score (4 steps)

1. Get your data in a numpy or pandas 2d format
2. Separate your **predictors** from your **target** variable (X,y)
3. Import the **cross_val_score** function (from sklearn.model_selection)
4. Pass learner, X, y and your desired **k** folds

DEMO

Your turn to practice Cross Validation

Are you ready?!


All Data
Training data
Test data
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 1
Split 2
Split 3
Split 4
Split 5
Finding Parameters
Final evaluation
Test data

Polynomial Regression

Polynoms!!

Classifying Polynomials by Degree		
Name	**Degree**	**Example**
Constant	0	-9
Linear	1	$x - 4$
Quadratic	2	$x^2 + 3x - 1$
Cubic	3	$x^3 + 2x^2 + x + 1$
Quartic	4	$2x^4 + x^3 + 3x^2 + 4x - 1$
Quintic	5	$7x^5 + x^4 - x^3 + 3x^2 + 2x - 1$

SQUARE OF SUM

$$(a + b)^2 = a^2 + 2ab + b^2$$

CUBE OF SUM

$$(a + b)^3 = a^3 + 3a^2b + 3ab^2 + b^3$$

Polynomic transformation of inputs (6 steps)

1. Get your data in a numpy or pandas 2d format
2. Separate your **predictors** from your **target** variable (X,y)
3. Import PolynomialFeatures (from sklearn.preprocessing)
4. Create an instance passing the degree of the polynom and exclude the bias
5. Fit it by passing your X and create a new object with your transformed data!
6. Run sklearn regression as usual :-)

DEMO

Your turn to fit a Polynomial model

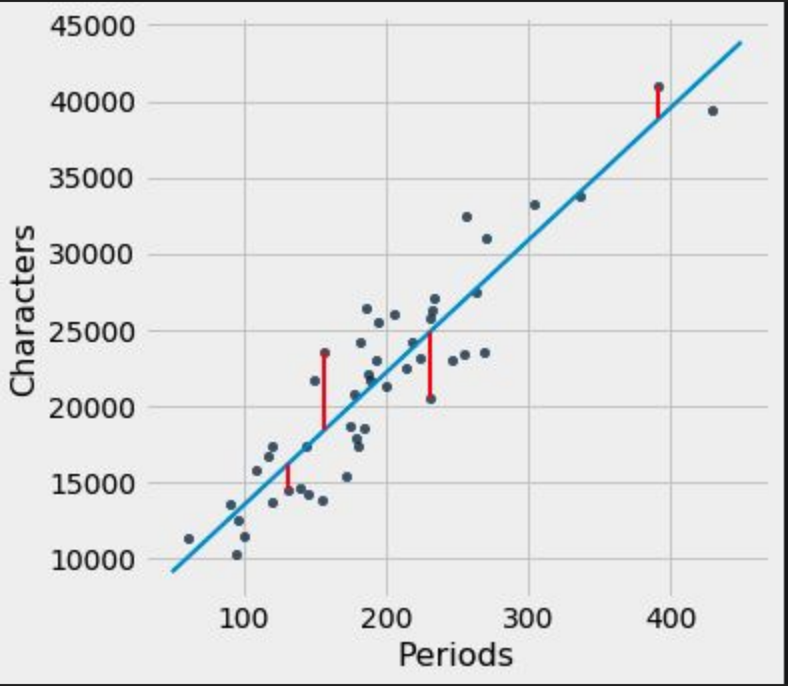
Are you ready?!

Mean Square Error

Mean Squared Error (MSE)

$$\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

$*\ n$ is the number of data points
$*\ Y_i$ represents observed values
$*\ \hat{Y}_i$ represents predicted values



MSE (2 options)

sklearn

1. Import mean_squared error (from sklearn metrics)
2. Pass your y and X

Code it from scratch

1. preds = model.predict(X)
2. error = (y - preds)
3. MSE = np.average(error**2)

DEMO

Your turn to calculate MSE

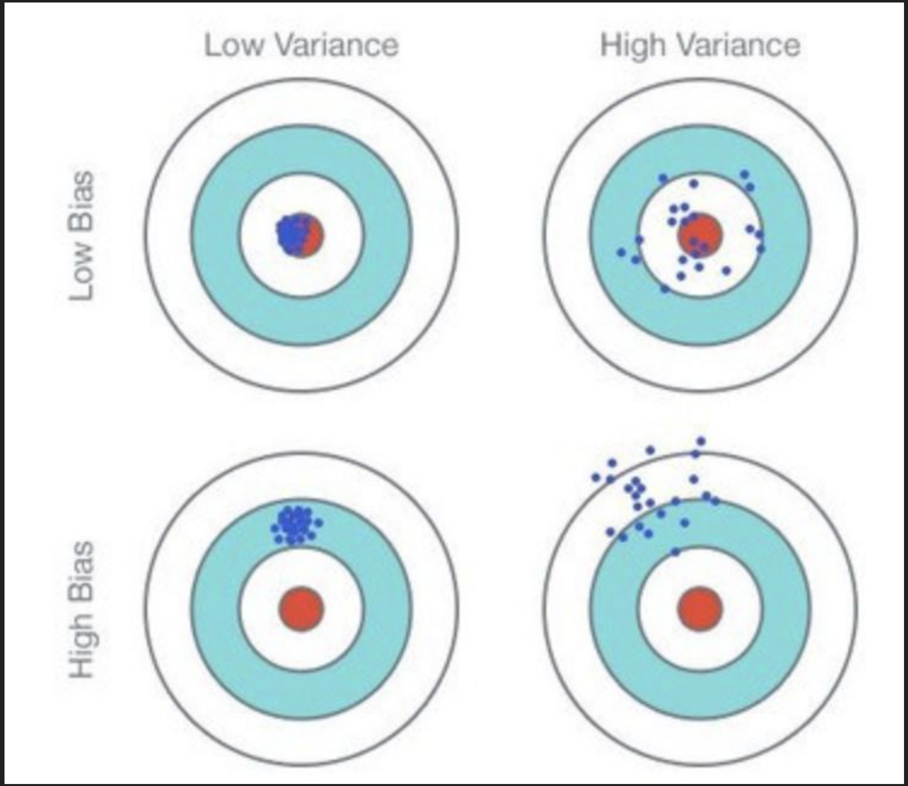
Are you ready?!

Bias-Variance Trade-off

Bias vs Variance

Bias: mean of predictions - mean of actual target values

Variance (of the error): The mean of the squares of the distances from each error to their mean



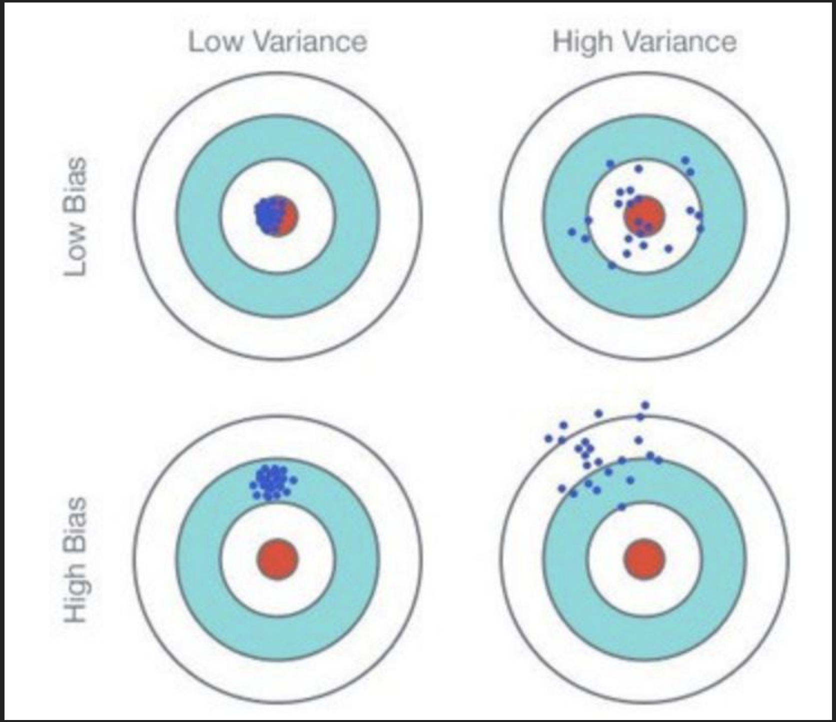
Proof

Variance of random variable formula:

- Var(X) = E(X**2) - [E(X)]**2

Making **Error** our X:

- Var(Error) = E(Error**2) - [E(Error)]**2
- Var(Error) = MSE - Bias**2
- MSE = Var(Error) + Bias**2



Explain to your partner
The MSE decomposition

Let’s code them!!

Bias

1. exp_y = np.average(y)
2. exp_preds = np.average(preds)
3. bias = exp_preds - exp_y

Variance

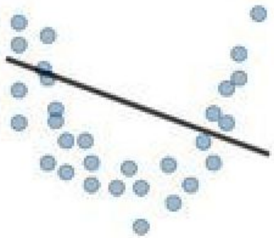
1. error = (y - preds)
2. exp_error = np.average(error)
3. var = np.average((error - exp_error)**2)

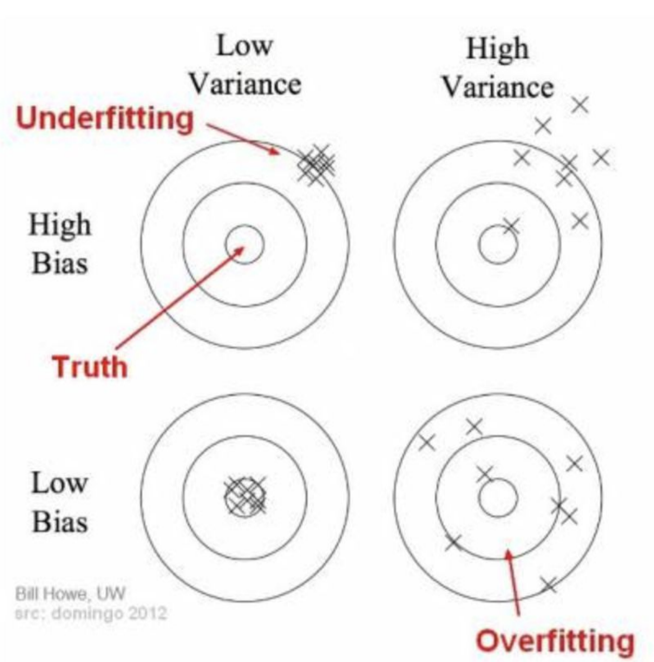
DEMO

Your turn to calculate MSE

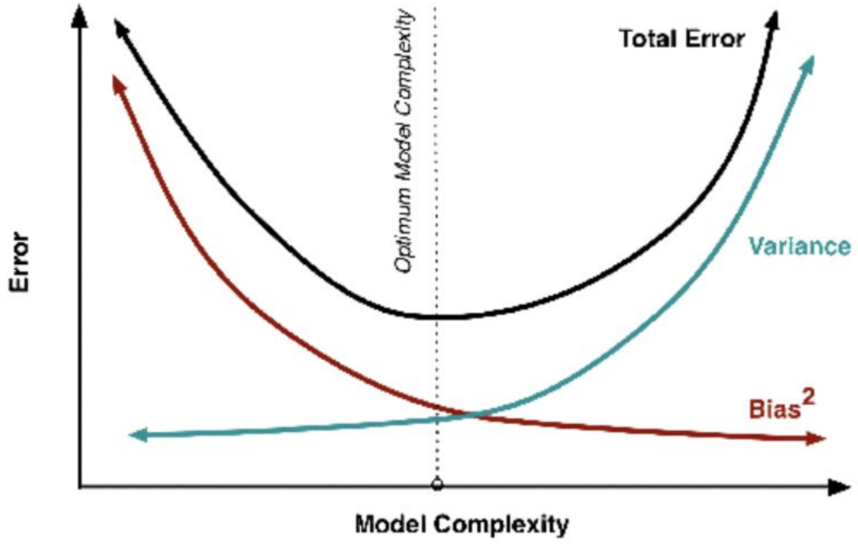
Are you ready?!

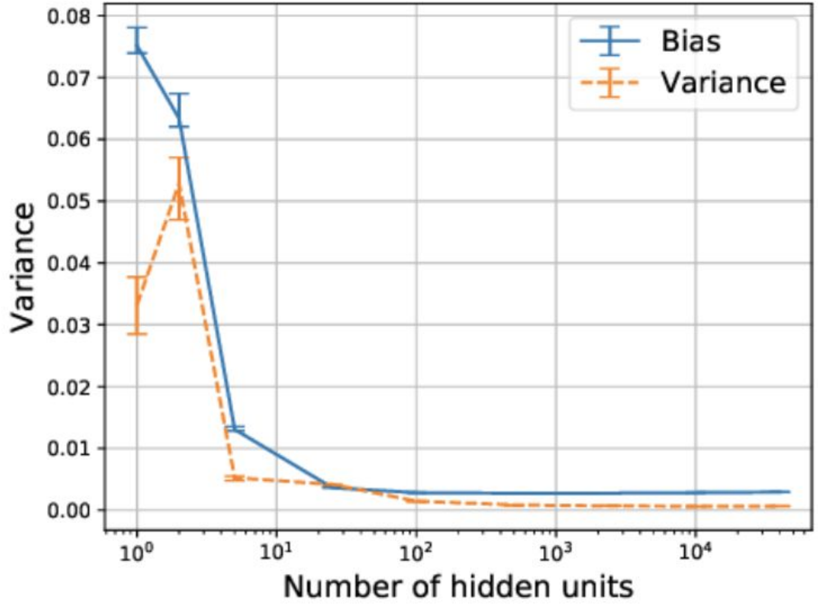
Bias / Variance
Underfitting / Overfitting

	Underfitting	Just right	Overfitting
Symptoms	• High training error • Training error close to test error • High bias	• Training error slightly lower than test error	• Very low training error • Training error much lower than test error • High variance
Regression illustration			



Bias / Variance Trade-off





More Practice

[dsc-bias-variance-trade-off-lab]

Reflection

What have you learned?

What outcomes do you still feel you need to work on?

What steps are you going to take to make that happen?