Machine Learning

| Learning Curves

Prof. Dr. Mostafa Elhosseini

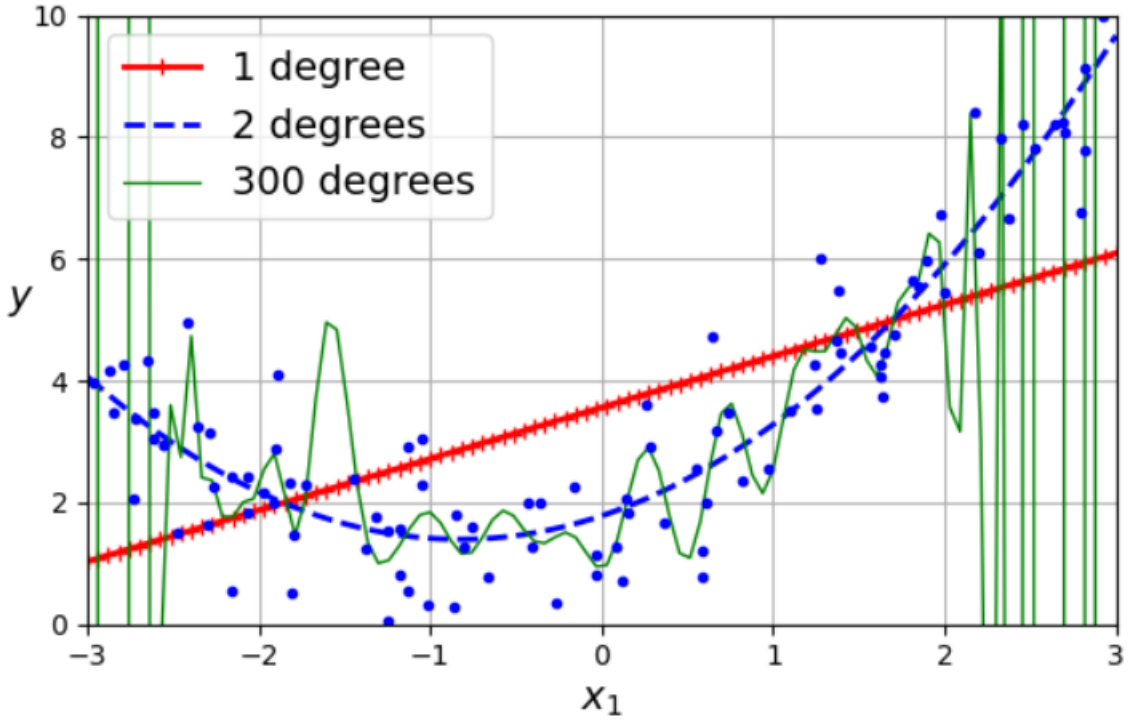
https://youtube.com/@ProfElhosseiniTechVerse

Agenda

- How complex your model should be?
- overfitting - underfitting
- Cross-validation
- Learning curves
  - overfitting - underfitting
- Bias
- Variance
- Bias / Variance tradeoff

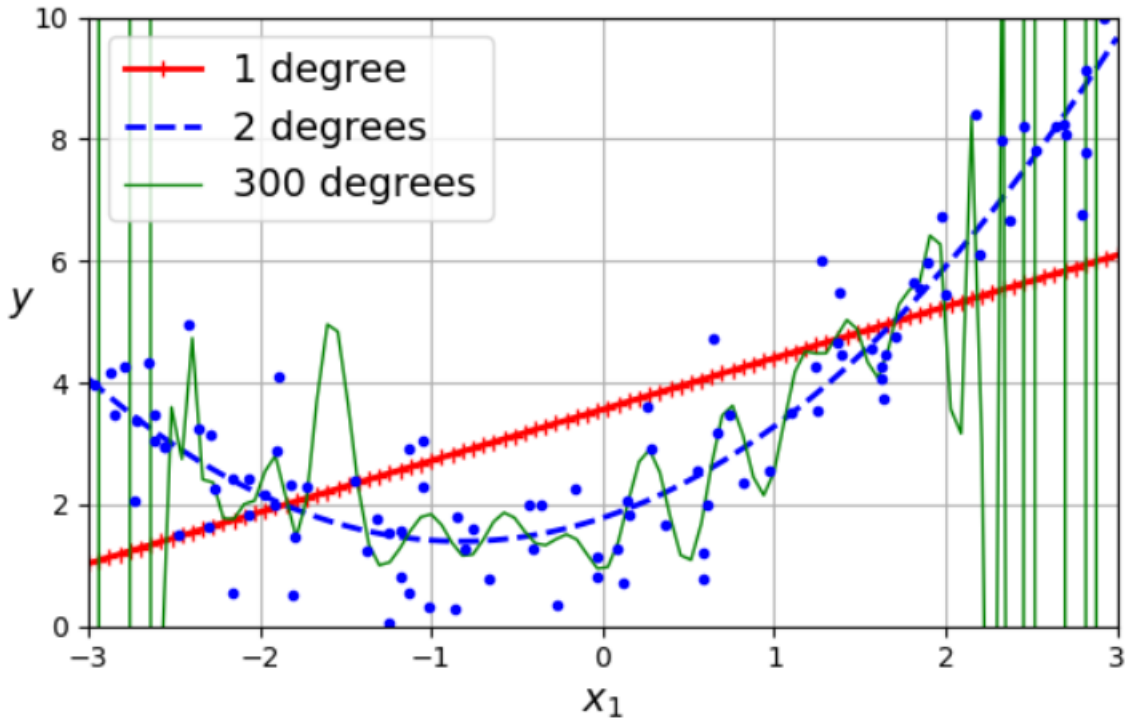
Intro

- This **high-degree** polynomial regression model is severely **overfitting** the training data, while the **linear** model is **underfitting** it



Intro

- **Quadratic** model **generalize best** because the data was generated using a quadratic model.
- But in general, **you won't know** what function generated the data



How complex your model should be?

- How can you say that your model is **overfitting** or **underfitting** the data?
  - Cross-validation
  - Learning curves

Cross-validation

- The dataset is **divided** into multiple smaller sets or "**folds**."
- **Each fold** acts as the **test set once**, and as the **training set k-1 times**.
- In each iteration of the cross-validation, **k-1 folds** are combined to form the **training** dataset.
- The **remaining single fold**, which is different in each iteration, serves as the **testing** dataset
- This process helps in ensuring that **every data point** gets to be in a test set **exactly once** and in a **training set k-1 times**.

Cross-validation

- **Evaluate** your model on the same **k-1 folds** used for **training**.
- Calculate metrics such as accuracy, precision, recall, and F1-score based on the model's performance on this training set.
- **Record** these metrics to analyze later. This will **show how well** the model is learning and adapting to the data it was **trained on**.

- **Test** the model on the **validation fold** (the one fold not used in training).
- This step is critical for assessing the **model's generalization ability** but is separate from calculating training metrics.

Cross-validation

- **Overfitting** occurs when a model **performs exceptionally** well on the **training** data but **poorly** on new, **unseen** data.
- **Underfitting** occurs when a model is **too simple** to capture the underlying pattern of the data
- **Low** performance across **both** the training and validation folds
- The model is **not complex enough** to capture the underlying patterns in the data

Example

- Suppose we have a **binary classification** model tasked with predicting whether an **email is spam or not**.
- using a **5-fold** cross-validation scenario. We'll consider simple performance metrics like **accuracy** for simplicity

Example

- **Underfitting**: Both training and testing accuracies are low and close to each other, indicating the model is too simple.

Fold Number	Training Accuracy	Testing Accuracy
Fold 1	60%	59%
Fold 2	62%	61%
Fold 3	59%	58%
Fold 4	61%	60%
Fold 5	60%	59%

Metric	Average Accuracy
Training	60.4%
Testing	59.4%

Example

- **Overfitting**: High training accuracies versus much lower testing accuracies suggest the model is too complex and is memorizing the training data rather than learning from it.

Fold Number	Training Accuracy	Testing Accuracy
Fold 1	95%	75%
Fold 2	94%	76%
Fold 3	96%	74%
Fold 4	95%	73%
Fold 5	94%	75%

Metric	Average Accuracy
Training	94.8%
Testing	74.6%

Learning Curves

- Plots of the model's training error and validation error as a function of the training iteration.

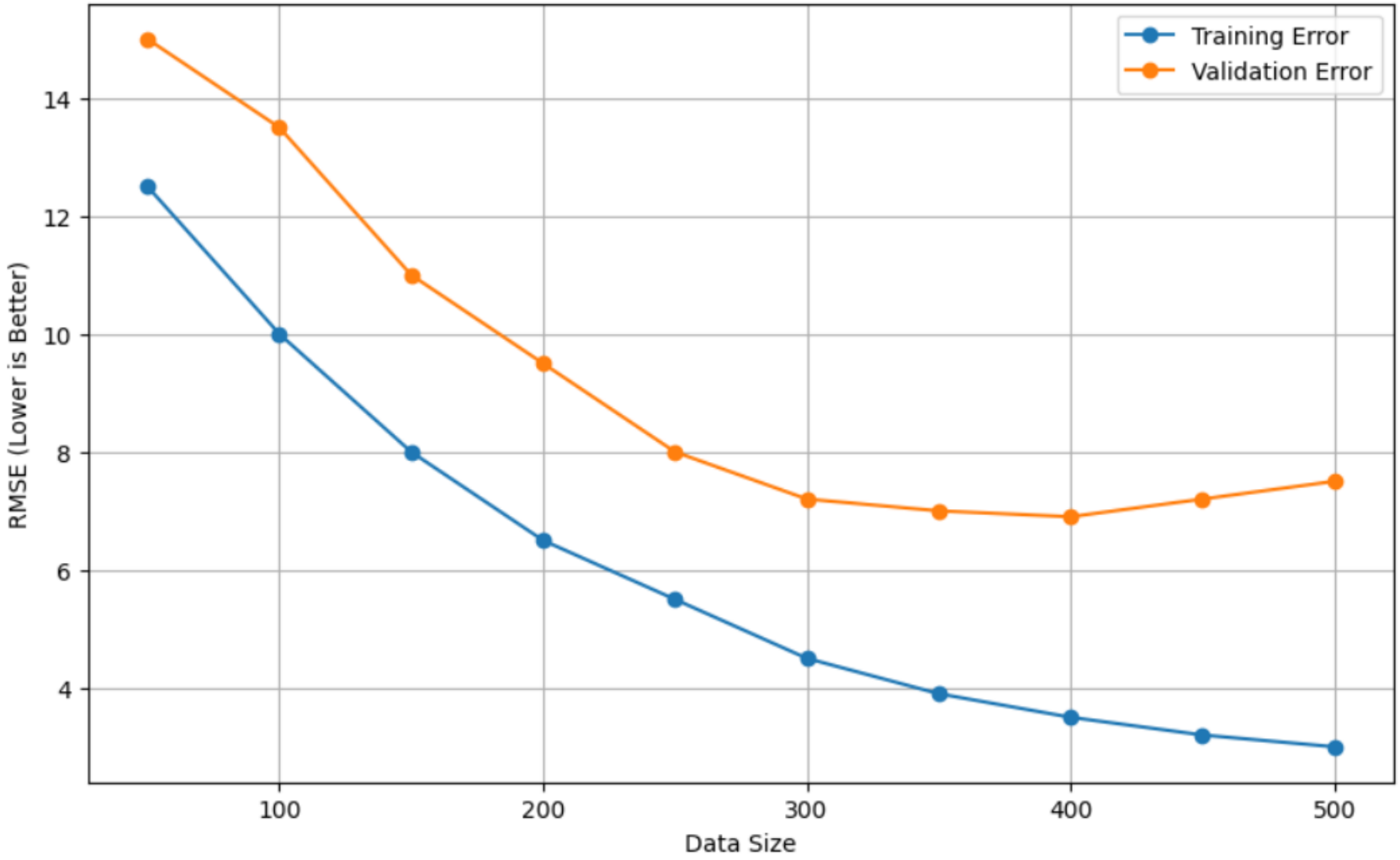
- A learning curve is a graph that shows how the **performance** of a machine learning model improves **as** it learns from an **increasing amount** of training data.

- It helps to understand **how much learning benefits** from **more data** and whether the model is learning effectively.

Learning Curves

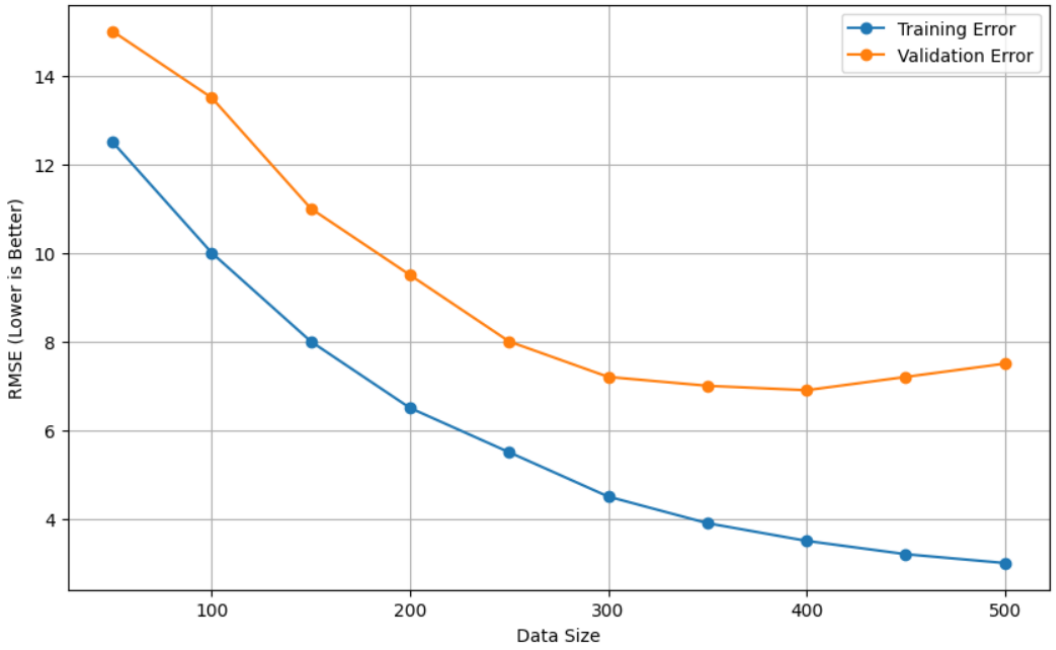
Data Size	Training Error (RMSE)	Validation Error (RMSE)
50	12.5	15.0
100	10.0	13.5
150	8.0	11.0
200	6.5	9.5
250	5.5	8.0
300	4.5	7.2
350	3.9	7.0
400	3.5	6.9
450	3.2	7.2
500	3.0	7.5

Learning Curves



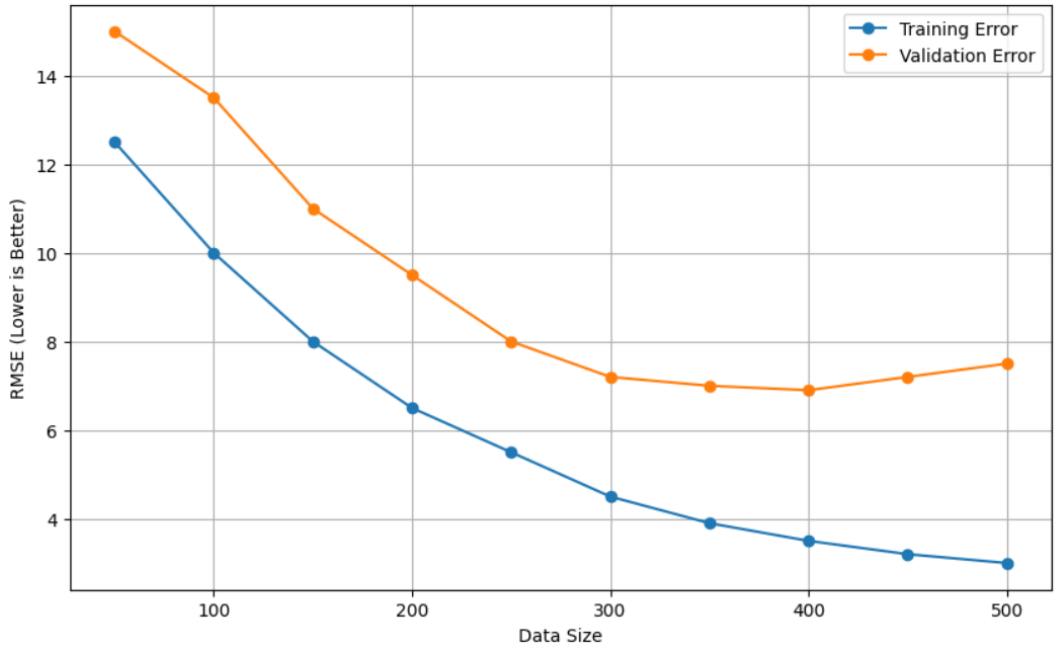
Learning Curves

- **Training Error**: Decreases as the data size increases, indicating the model is learning and fitting the data better with more examples.
- **Validation Error**: Initially decreases, reflecting improved generalization as the model is trained with more data. However, after a certain point, the validation error starts to increase slightly, suggesting the beginning of overfitting as the model becomes too tailored to the training data.



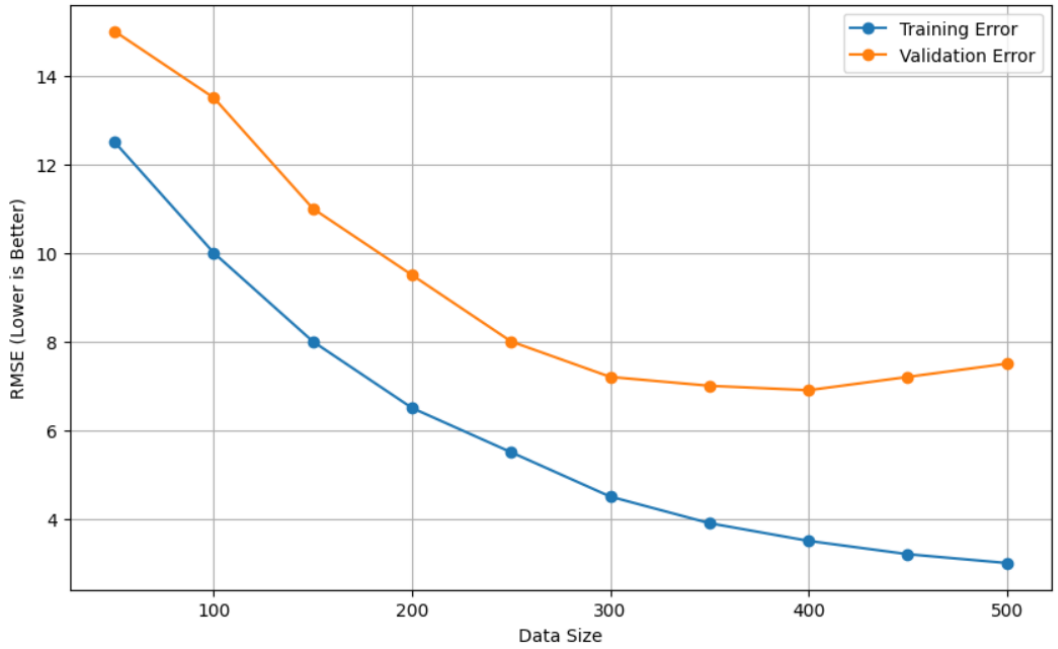
Learning Curves

- **Underfitting**: At the start, both training and validation errors are high, indicating underfitting where the model is too simple to capture the underlying pattern in both the training and validation datasets.



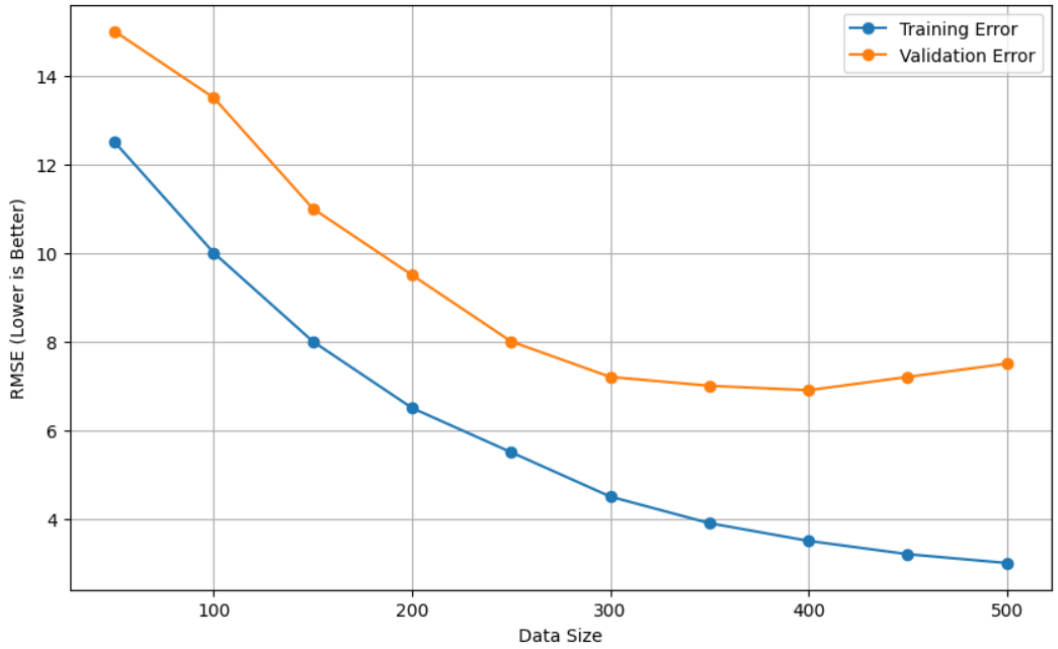
Learning Curves

- **Good Fit**: As more data is added, the training and validation errors converge to a low value, suggesting that the model is generalizing well and capturing the underlying patterns effectively.



Learning Curves

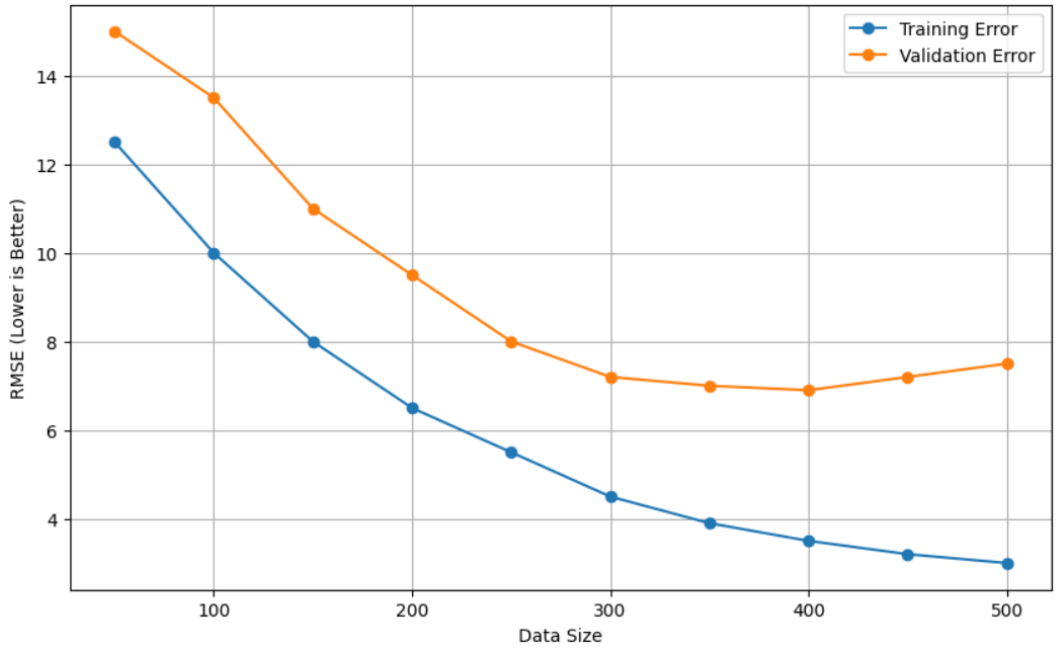
- **Overfitting**: Towards the end, as the training error continues to drop but the validation error begins to rise, it suggests the model is starting to overfit. It's getting too complex and fitting the noise in the training data rather than generalizing from the true signal.



Learning Curves

Note:

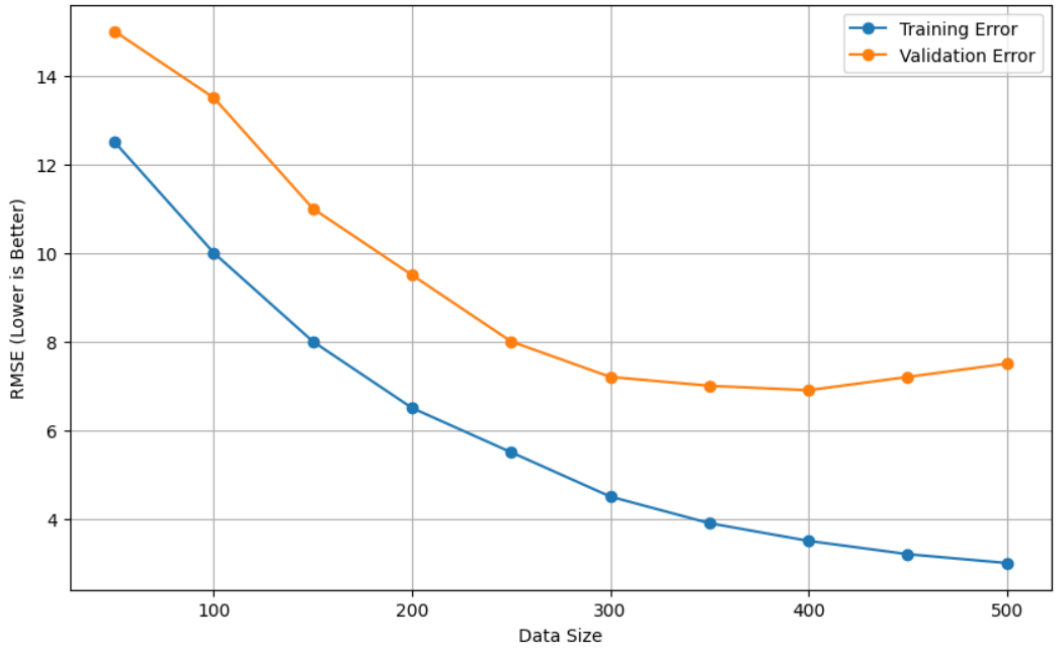
- **Without Cross-Validation**: Each point on the learning curve is derived from a single partition of data into training and validation sets.
- **With Cross-Validation**: Each point on the learning curve is the average of errors across multiple runs, where each run uses a different validation subset (k-fold cross-validation).



Learning Curves

Note:

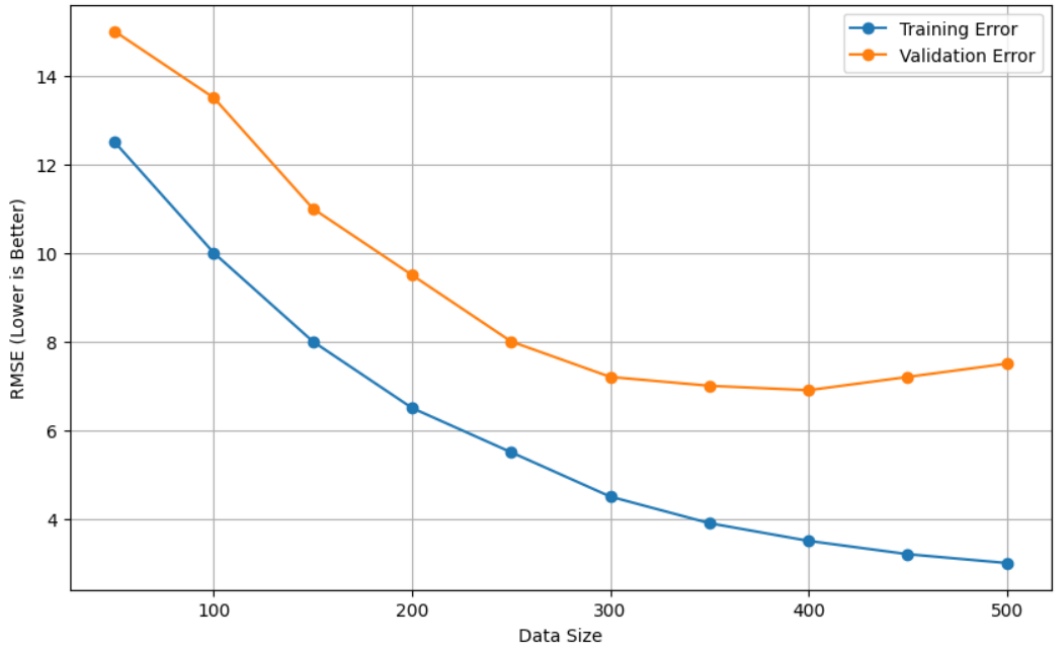
- **Data Size** refers to the number of training examples used to train the model, not the number of epochs.
- For each point on the learning curve, the model is trained on a **different subset of the total available data, gradually increasing in size**. This helps in understanding how the model's performance improves as more data is made available for training.
- Increasing the data size helps determine how much the model benefits from more training data.



Learning Curves

Note:

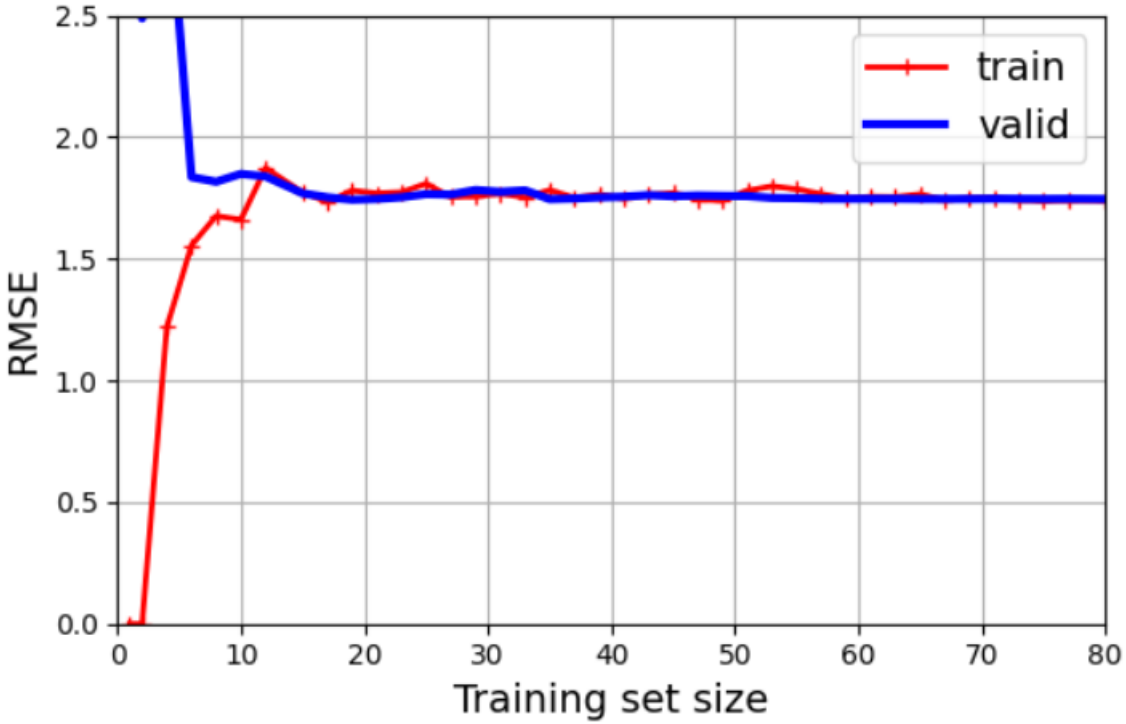
- An **epoch** refers to one complete pass through the entire training dataset
- Increasing the number of epochs usually aims to see if further iterations of training on the same data can improve the model's performance, possibly at the risk of overfitting if done excessively.



Learning curves - Linear regression model

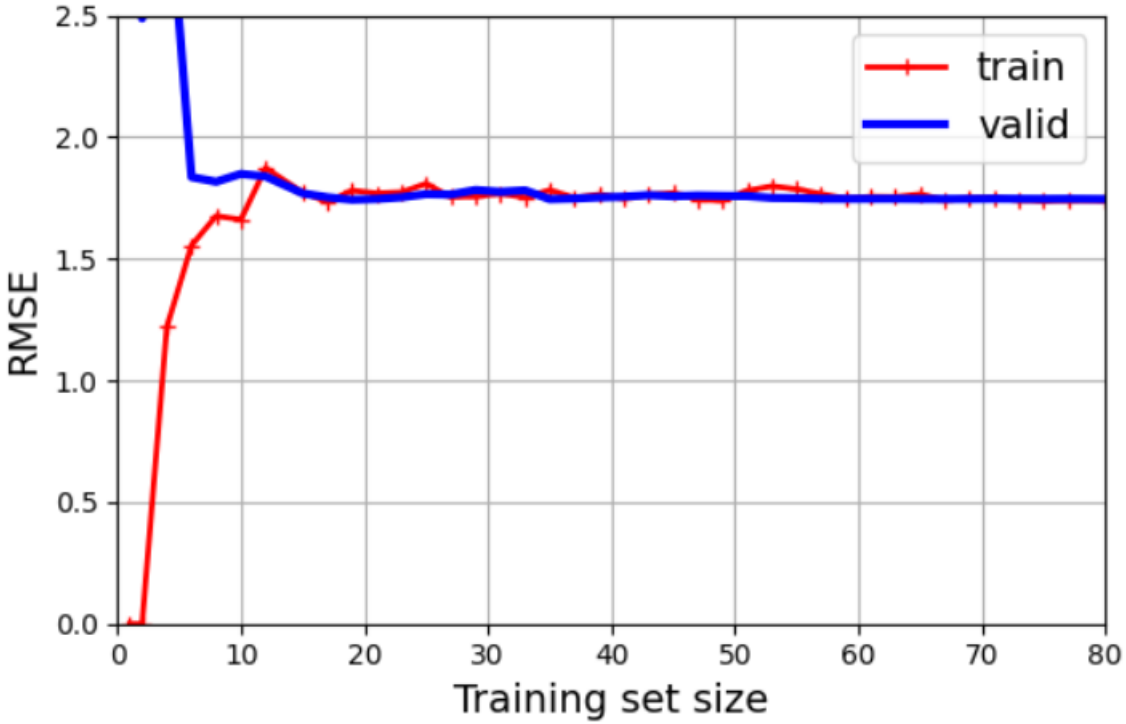
Training Error

- when there are just **one or two instances** in the training set, the model can fit them **perfectly**, which is why the **curve starts at zero**.



Learning curves - Linear regression model

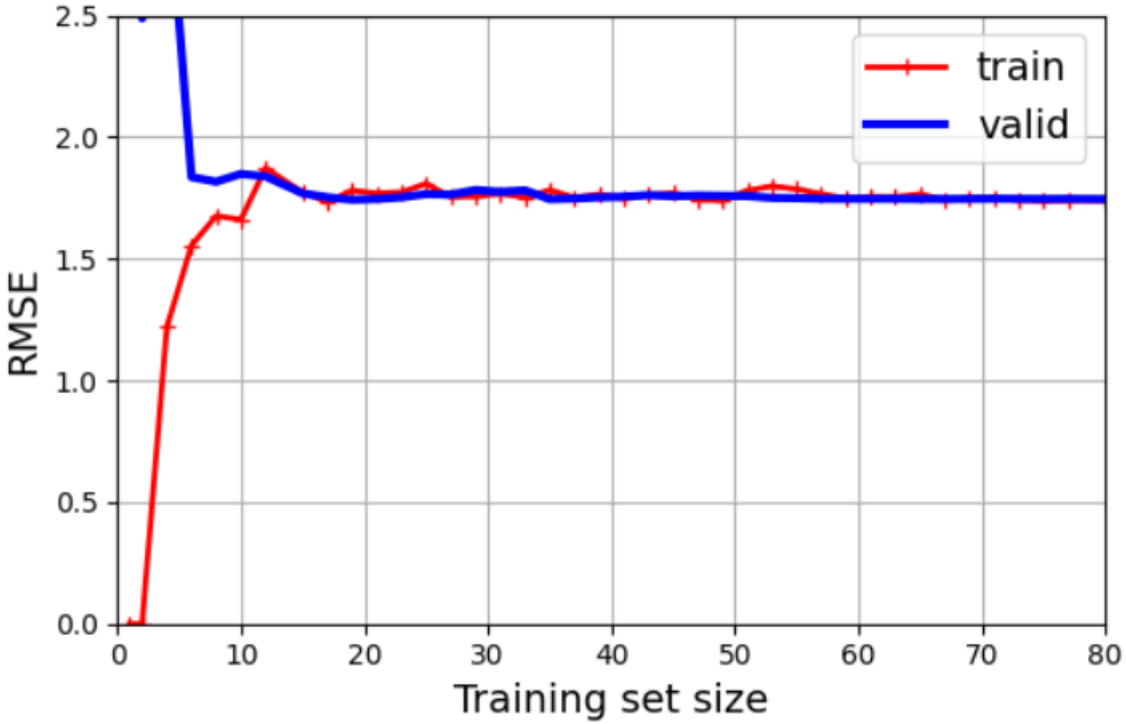
- But as **new instances are added** to the training set, it becomes **impossible for the model to fit** the training data perfectly, both because the **data is noisy**, and it is **not linear** at all.
- So the **error** on the training data **goes up** until it reaches a plateau, at which point **adding new instances** to the training set **doesn't make** the average error much better or worse



Learning curves - Linear regression model

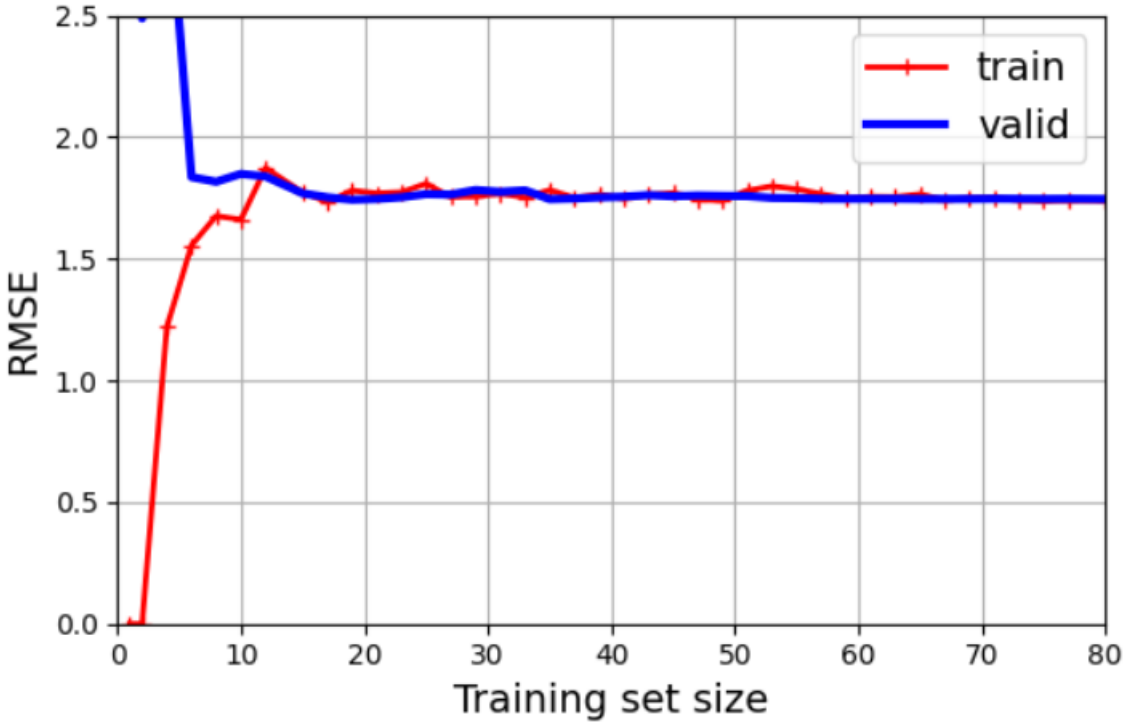
Validation Error

- When the model is trained on very few training instances, it is incapable of generalizing properly, which is why the validation error is initially quite big.



Learning curves - Linear regression model

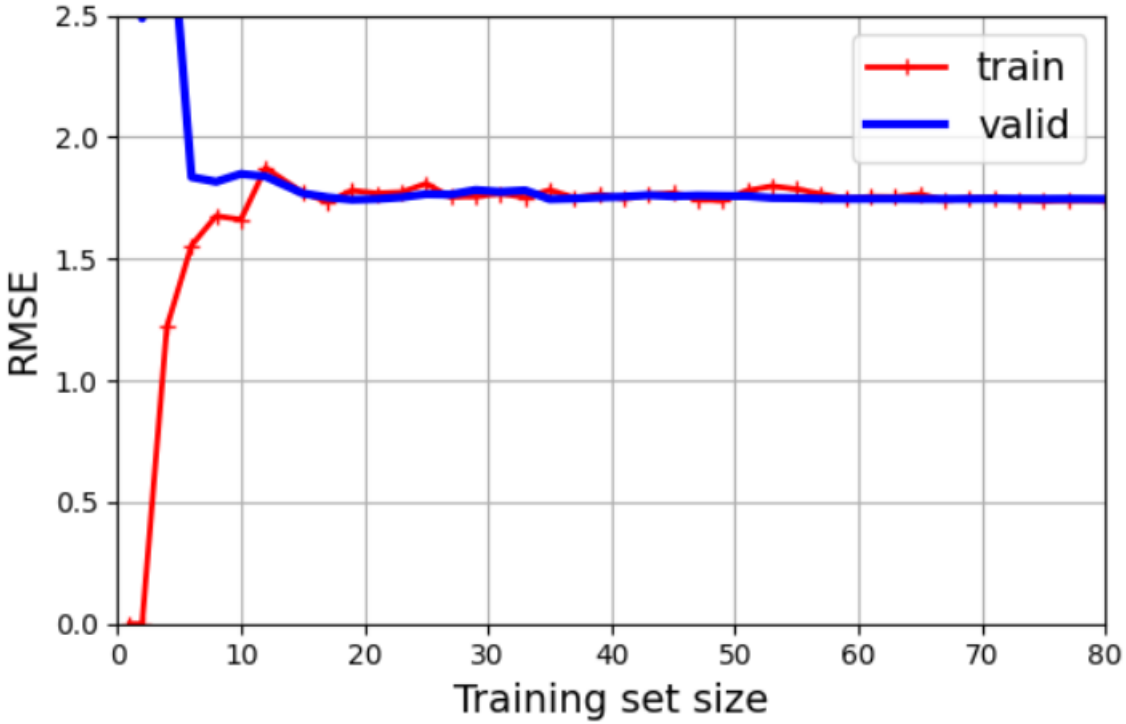
- Then as the model is shown more training examples, it learns and thus the validation error slowly goes down
- However, once again a straight line cannot do a good job modeling the data, so the error ends up at a plateau, very close to the other curve.



- These learning curves are typical of a model that's **underfitting**. Both curves have **reached a plateau**; they are **close and fairly high**

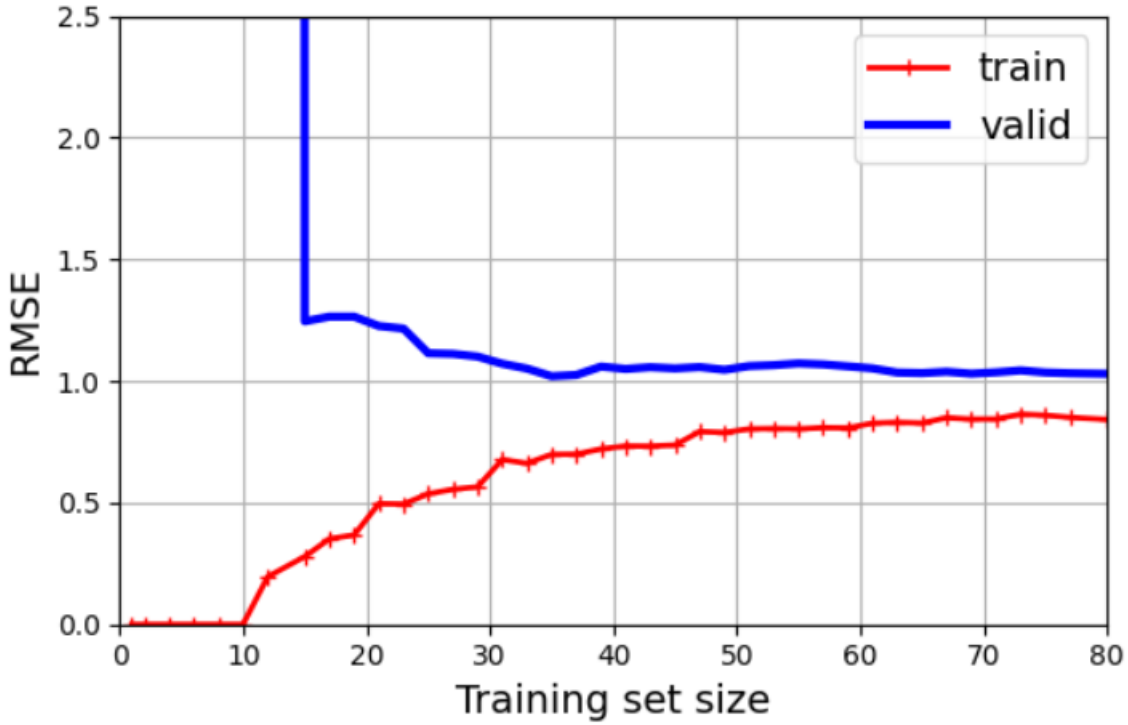
Learning curves - Linear regression model

- These learning curves are typical of a model that's underfitting. Both curves have reached a plateau; they are close and fairly high
- If your model is underfitting the training data, adding more training examples will not help. You need to use a better model or come up with better features.



Learning curves – 10-th degree Polynomial

- The error on the training data is much lower than before.
- There is a gap between the curves.
- This means that the model performs significantly better on the training data than on the validation data, which is the hallmark of an **overfitting model**
- One way to improve an overfitting model is to feed it more training data until the validation error reaches the training error.



Identifying Underfitting from Learning Curves

- High error on both training and validation sets
- Both curves are relatively flat and close together
- Little improvement as training data increases
- Indicates the model is too simple to capture the underlying patterns

Detecting Overfitting - Learning Curves

- Low error on training set, high error on validation set
- Large gap between training and validation curves
- Training error continues to decrease while validation error plateaus or increases
- Suggests the model is memorizing training data rather than generalizing

Ideal learning curve

- Both training and validation errors decrease and converge
- Small gap between training and validation curves
- Indicates good generalization

Bias

- Bias is the error introduced by **approximating a real-world problem**, which may be complex, with a **simplified model**.

- **High bias** can cause an algorithm to **miss the relevant relations** between features and target outputs (**underfitting**)

Variance

- Variance is the error that occurs due to the model's sensitivity to small fluctuations in the training dataset.
- High variance can cause a model to model the random noise (high sensitivity to **data variations** in the training data), rather than the intended outputs (overfitting)
- A model with many degrees of freedom (such as a high-degree polynomial model) is likely to have high variance, and thus to overfit the training data.
- A model that has very low errors on the training data but high errors on new, unseen data typically has a high variance

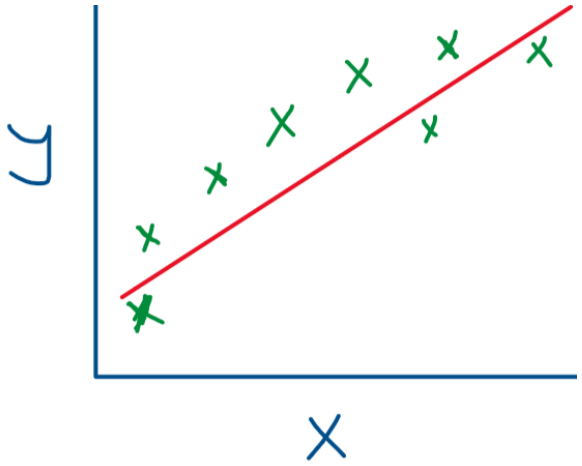
Bias / Variance tradeoff

- Model's generalization error can be expressed as the sum of three very different errors:
- **Bias**
- **Variance**
- **Irreducible error**: This part is due to the noisiness of the data itself. The only way to reduce this part of the error is to clean up the data (e.g., fix the data sources, such as broken sensors, or detect and remove outliers)

Bias / Variance tradeoff

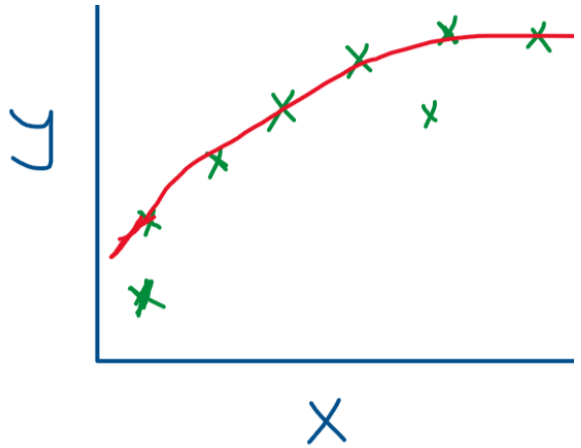
- Increasing a model's complexity will typically
  - increase its variance and
  - reduce its bias.
- Conversely, reducing a model's complexity
  - increases its bias and
  - reduces its variance.
- This is why it is called a tradeoff

Bias / Variance tradeoff



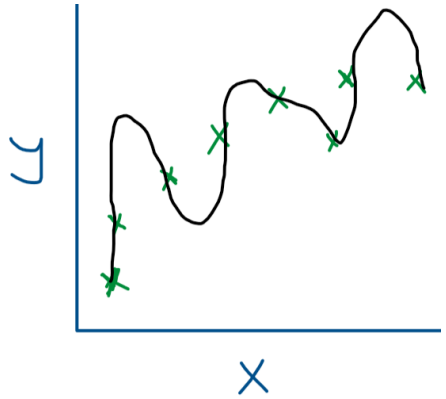
- Underfitting
  - High bias
  - Bias: we have a strong preconception that there should be a linear fit

$$y = \theta_0 + \theta_1 x$$



- Quadratic function
  - Works well
  - Just right

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



- Overfitting
  - Perform perfect on training data
  - High variance
  - Not able to generalize on unseen data
  - If we have too many features

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$$

Regularization

- If you have **lots of features and little data** – **overfitting** can be a problem
- simple way to **regularize** a polynomial model is to reduce the number of polynomial degrees
- The core idea behind regularization is to prevent overfitting by **adding a penalty** for complexity to the model's loss function
- Improve generalization to new, unseen data
- Next Lecture