Introdução à Aprendizagem Automática (IAA)

SUSANA BRÁS

SUSANA.BRAS@UA.PT

IAA – L7

Cross Validation
- Model tuning
- Data Splitting
- Modeling process
- Model selection

Bias vs Variance
- Underfit, overfit, good fit
- Identification of high bias
- Identification of high variance
- Actions to mitigate high bias
- Actions to mitigate high variance

Learning Curves
- Definition
- Interpretation of learning curves
- Identification of model performance
- Identification of model problems

Cross Validation

Model Tunning

Model parameters may be estimated directly from data.

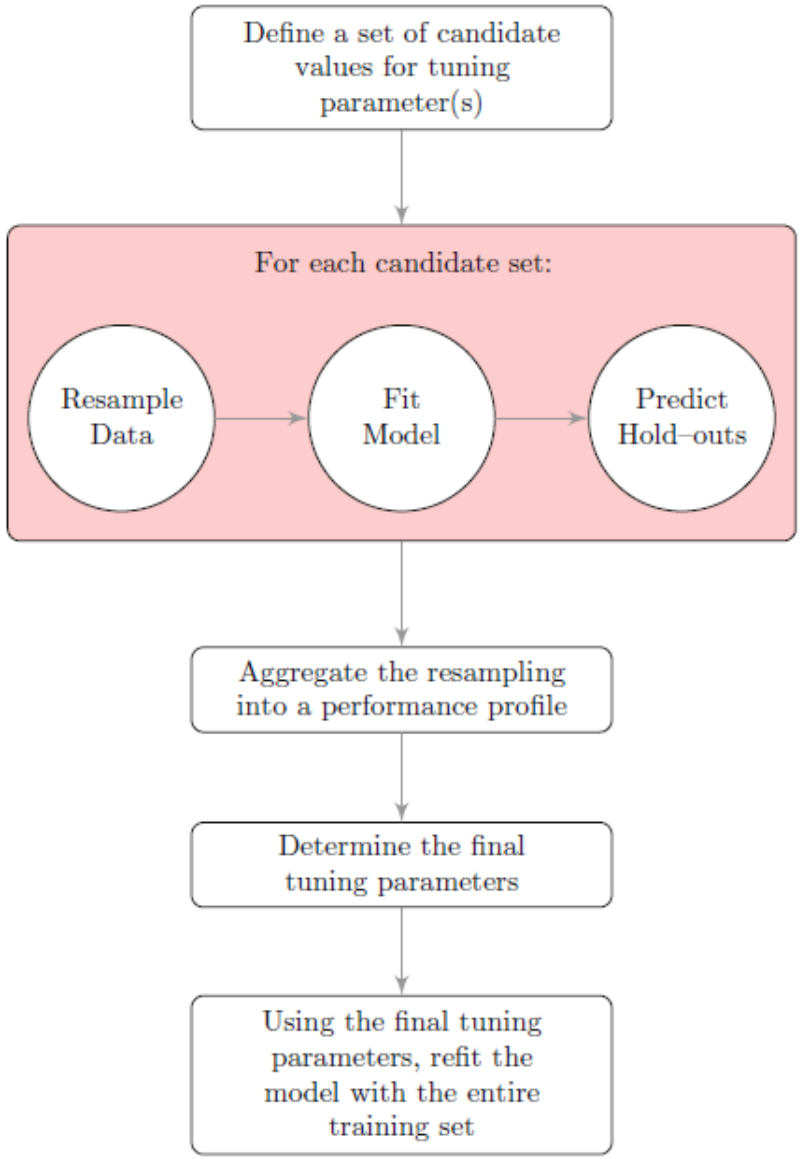
Nevertheless, there are parameters that data cannot estimate.

There are different approaches to searching for the best parameters. A general approach that can be applied to almost any model is to define a set of candidate values, generate reliable estimates of model utility across the candidate values, and then choose the optimal settings.

A more difficult problem is obtaining trustworthy estimates of model performance for these candidate models.

The apparent error rate can produce extremely optimistic performance estimates.

A better approach is to test the model on samples that were not used for training.



Data Splitting

The correct division between train and test is essential to obtain a correct approach to the effectiveness of the model.

Holdout

- **Random splitting**
- **Stratified random selection**

Leave one out

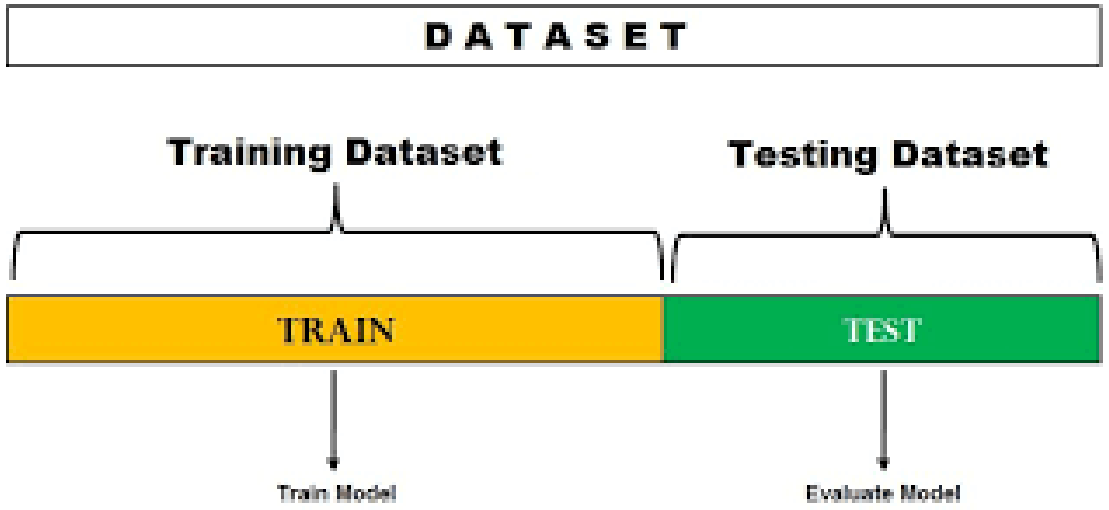
K-fold

Bootstrap

Data Splitting

The correct division between train and test is essential to obtain a correct approach to the effectiveness of the model.

Holdout: Define x% for training and (100-x)% for test.



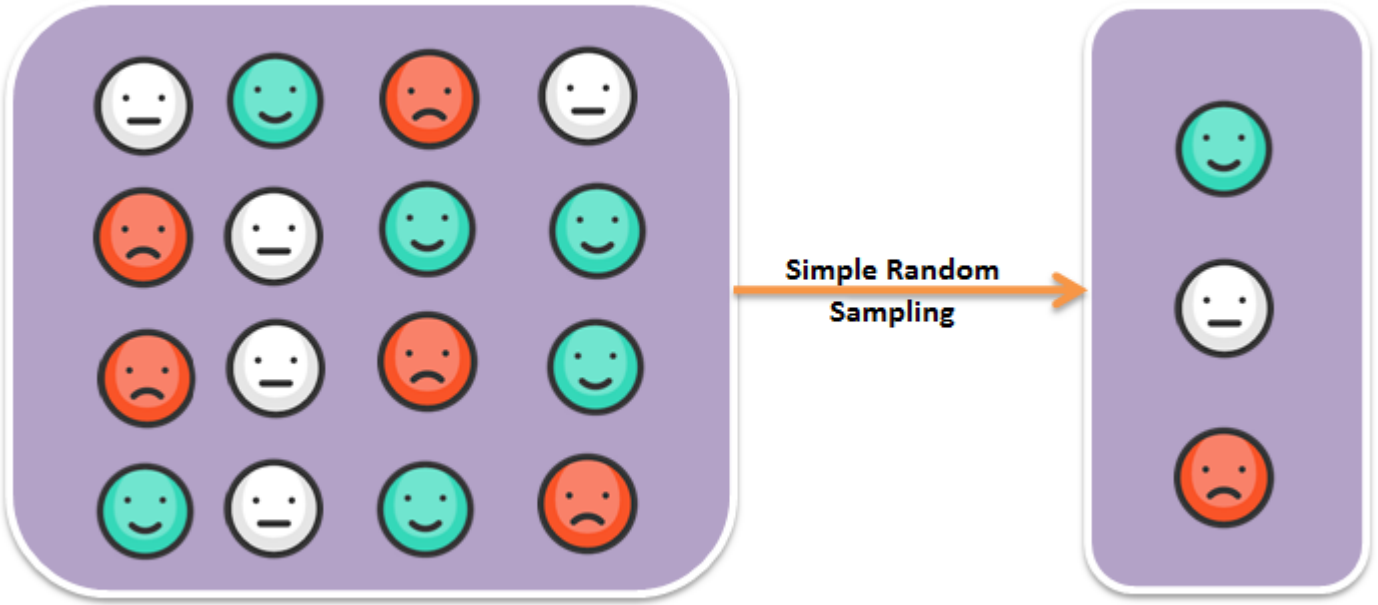
Data Splitting

The correct division between train and test is essential to obtain a correct approach to the effectiveness of the model.

Holdout: Define x% for training and (100-x)% for test.

Random splitting: randomly select samples to the train set.

- If the dataset is homogeneous, it will produce an homogeneous and representative train and test sets of all classes.



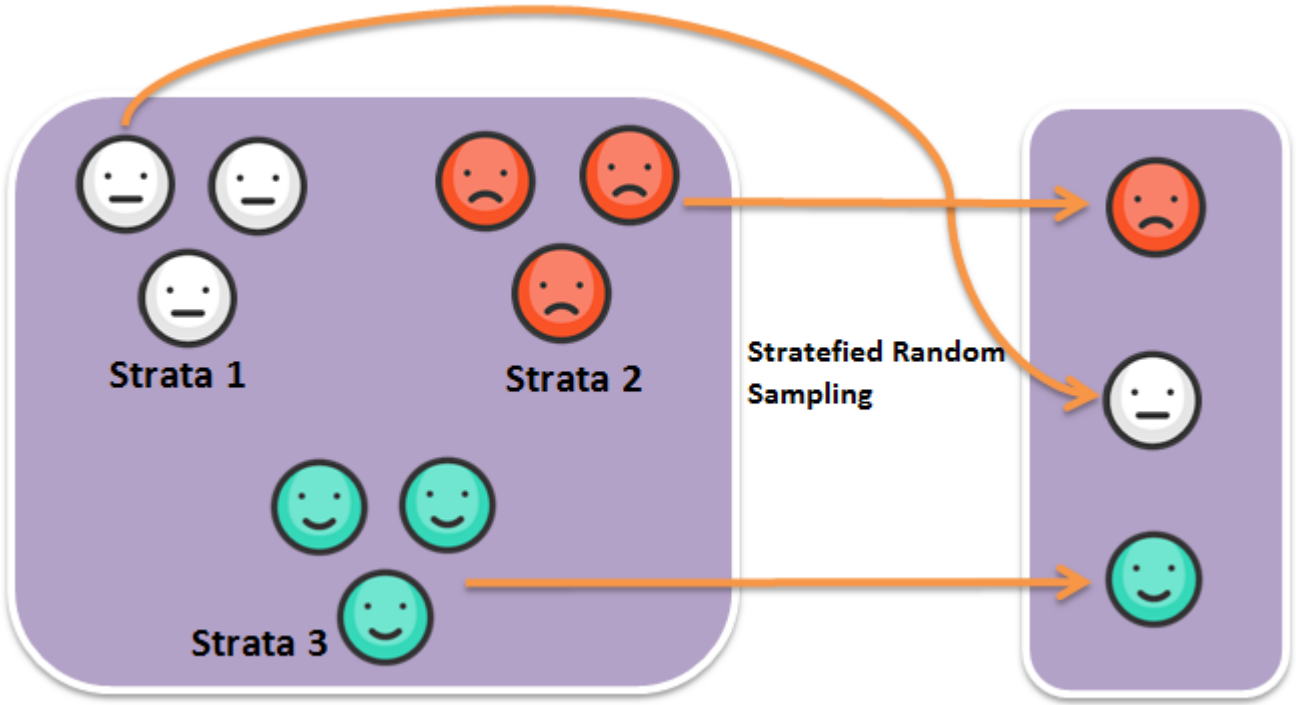
Data Splitting

The correct division between train and test is essential to obtain a correct approach to the effectiveness of the model.

Holdout: Define x% for training and (100-x)% for test.

Stratified random selection: random selection of samples in a similar proportion of the size of the class on the amount of data. (Random selection within each class).

- There is a higher likelihood that the outcome distributions will match.



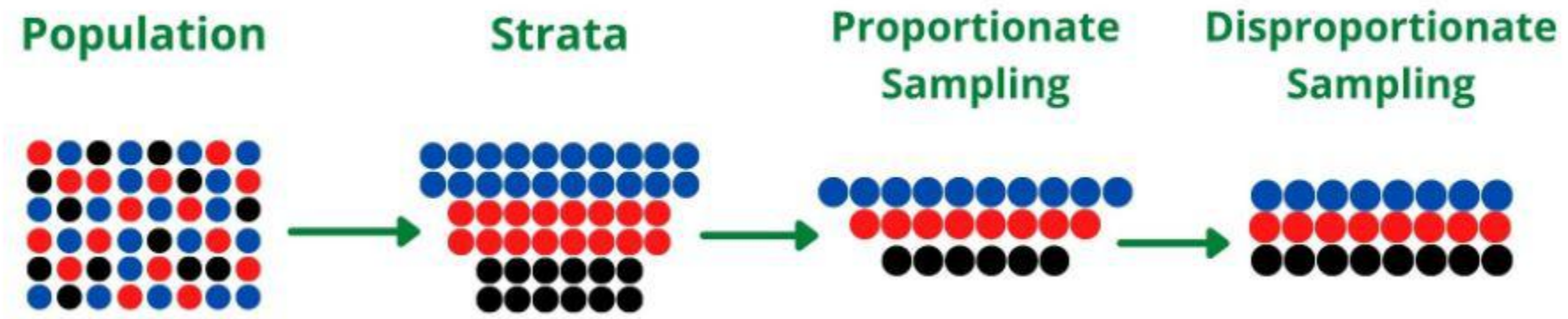
Data Splitting

The correct division between train and test is essential to obtain a correct approach to the effectiveness of the model.

Holdout: Define x% for training and (100-x)% for test.

Stratified random selection: random selection of samples in a similar proportion of the size of the class on the amount of data. (Random selection within each class).

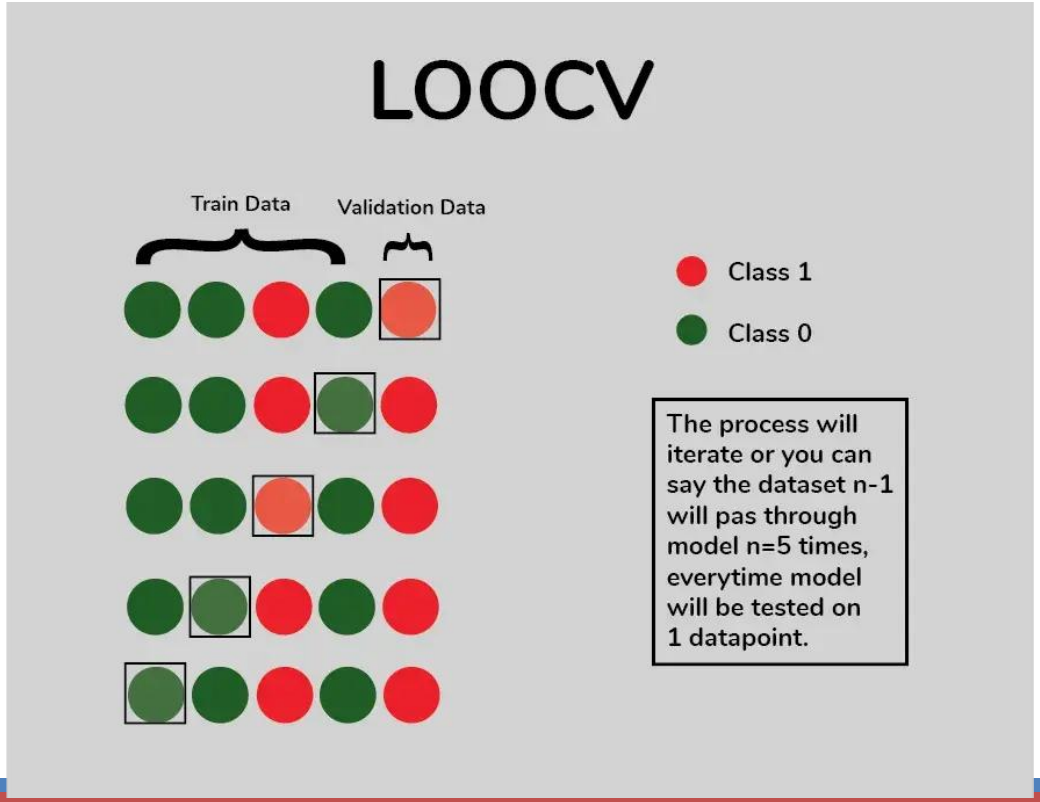
- There is a higher likelihood that the outcome distributions will match.



Data Splitting

The correct division between train and test is essential to obtain a correct approach to the effectiveness of the model.

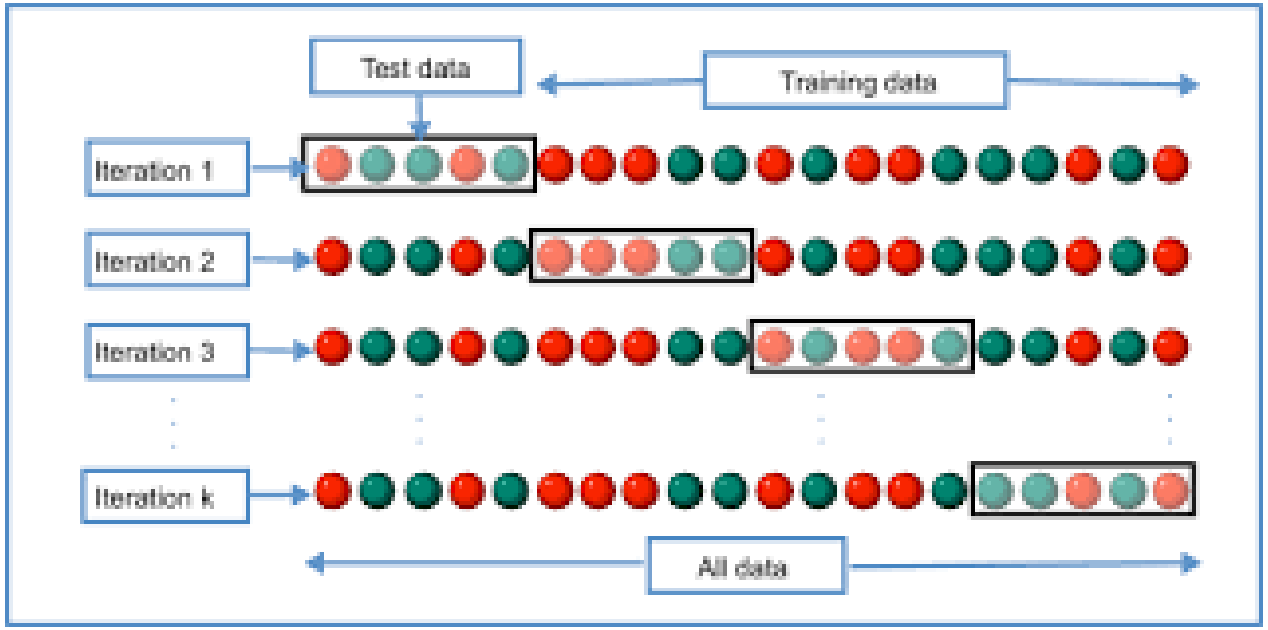
Leave one out: when the dataset is small, use N-1 samples to train the model, and test in the remaining one. Run the method until all samples were evaluated as test.



Data Splitting

The correct division between train and test is essential to obtain a correct approach to the effectiveness of the model.

K-fold: the training subset is defined by k-1 partitions of the data, and the test is the remaining one. It should be repeated until all partitions were evaluated as test.



- The final validation error is the average error of K trainings.
- Choose the best model or the best hyper-parameter the one that minimises the validation error.

Data Splitting

The correct division between train and test is essential to obtain a correct approach to the effectiveness of the model.

Bootstrap: the training subset is defined by sampling with replacement.

- The bootstrap sample is the same size as the original data set.
- Some samples will be represented multiple times in the bootstrap sample while others will not be selected at all ("out-of-bag").
- If the training set size is small, this bias may be problematic, but will decrease as the training set sample size becomes larger.

Data Splitting

The correct division between train and test is essential to obtain a correct approach to the effectiveness of the model.

Holdout: Define x% for training and (100-x)% for test.

Random splitting: randomly select samples to the train set.

- If the dataset is homogeneous, it will produce an homogeneous and representative train and test sets of all classes.

Stratified random selection: random selection of samples in a similar proportion of the size of the class on the amount of data. (Random selection within each class).

- There is a higher likelihood that the outcome distributions will match.

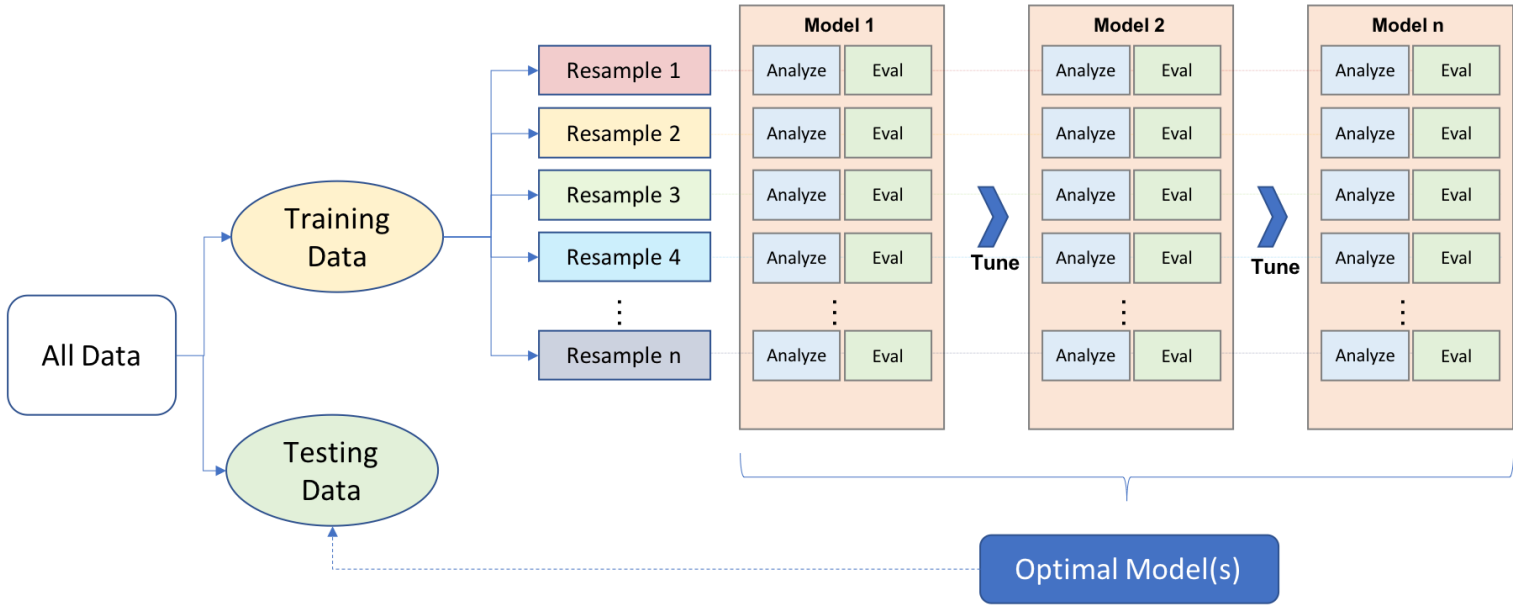
Leave one out: when the dataset is small, use N-1 samples to train the model, and test in the remaining one. Run the method until all samples were evaluated as test.

K-fold: the training subset is defined by k-1 partitions of the data, and the test is the remaining one. It should be repeated until all partitions were evaluated as test.

Bootstrap: the training subset is defined by sampling with replacement.

Another important aspect of a data splitting technique is the uncertainty (i.e., variance or noise). An unbiased method may be estimating the correct value (e.g., the true theoretical performance) but may pay a high price in uncertainty. This means that repeating the data splitting procedure may produce a very different value (but done enough times, it will estimate the true value).

The modeling process



Step 1: Optimize the parameters using the same training set for all models. Compute some perf. metrics with the training data: **Training Error**

Step 2: Test the optimized models from step 1 with the CV set and choose the model with the min CV error: **Cross validation Error**

Step 3: Retrain the best model from step 2 using both the training and cross-validation sets, starting with the parameters obtained in step 2. Evaluate the retrained model using the test set and calculate its performance on the test data. metric: **Test Error**

The modeling process

The optimal model:

- The simplest approach is to pick the settings associated with the numerically best performance estimates.
- In general, it may be a good idea to favor simpler models over more complex ones and choosing the tuning parameters based on the numerically optimal value may lead to overly complicated models.

Considerations:

- Understand what tuning machine learning model is
- Find Your Score Metric
- Obtain Accurate Forecasting Score (data split)
- Diagnose Best Parameter Value Using Validation Curves (cost vs parameter value or cost vs performance metric)
- Use Grid Search To Optimise Hyperparameter Combination

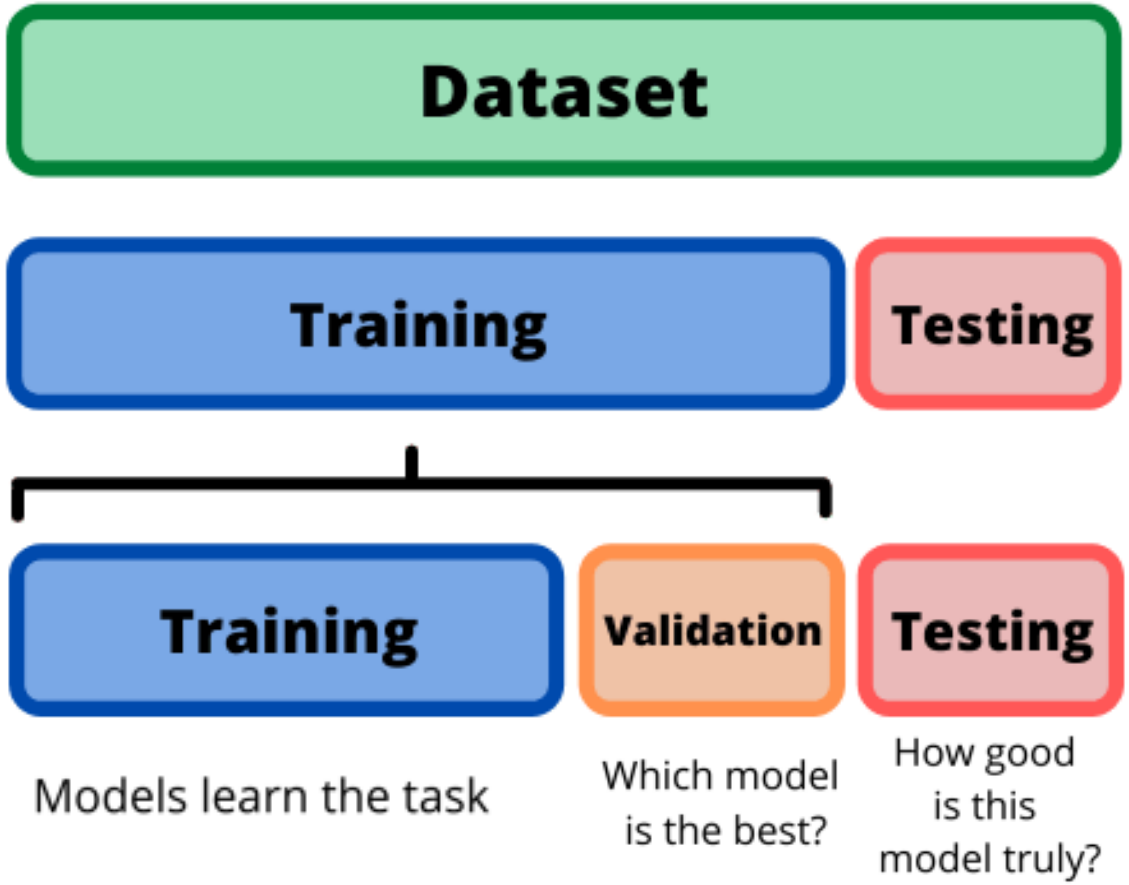
The modeling process

Data Splitting Recommendations

- A test set is a single evaluation of the model and has limited ability to characterize the uncertainty in the results
- Proportionally large test sets divide the data in a way that increases bias in the performance estimates.
- With small sample sizes:
  - The model may need every possible data point to adequately determine model values.
  - The uncertainty of the test set can be considerably large to the point where different test sets may produce very different results.

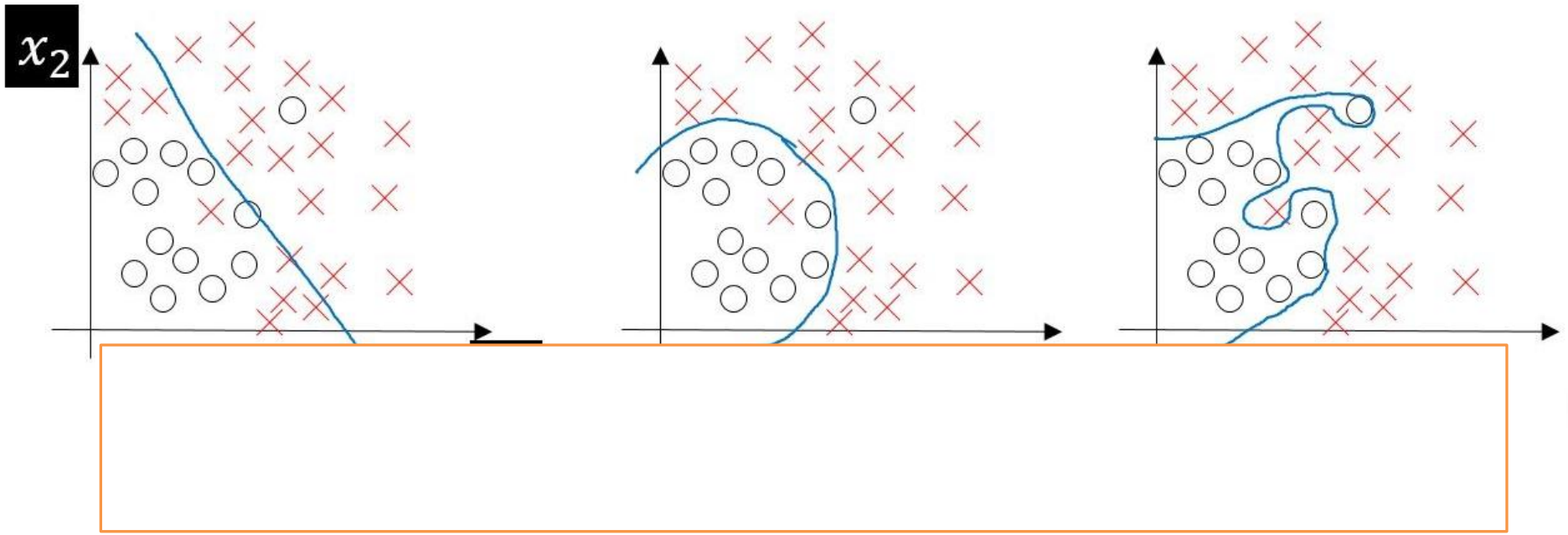
- Data Splitting techniques can produce reasonable predictions of how well the model will perform on future samples.

- No Data Splitting technique is uniformly better than another.


Dataset
Training
Testing
Training
Validation
Testing
Models learn the task
Which model is the best?
How good is this model truly?

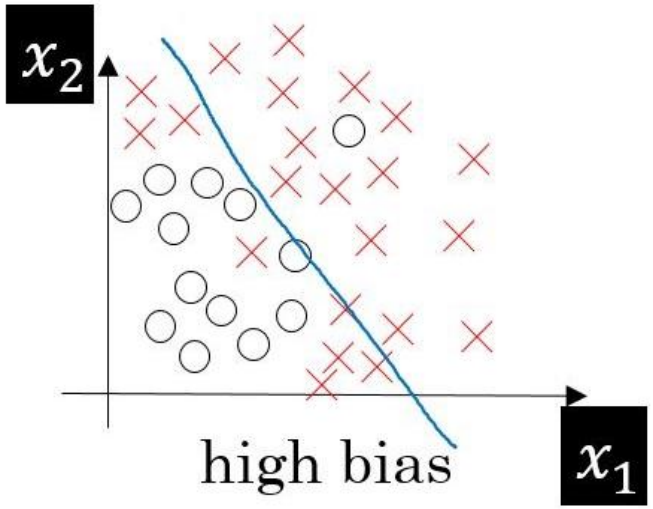
Bias vs Variance

Bias vs. Variance ?

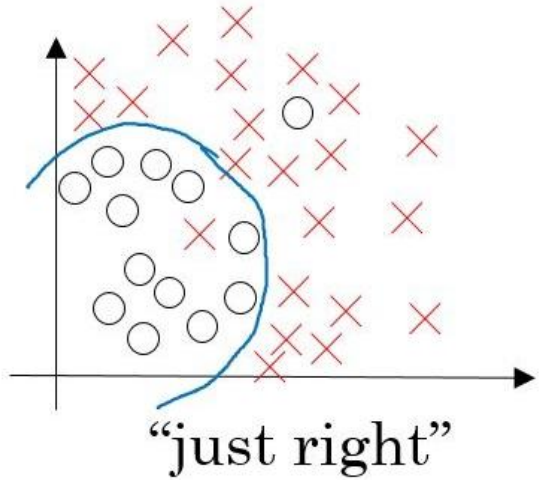


Bias vs. Variance ?

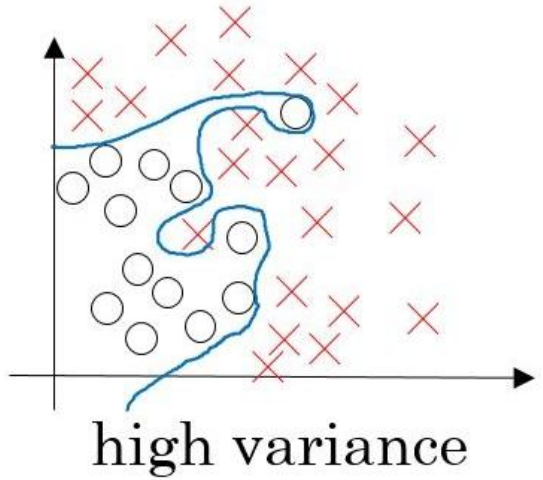
An important concept in ML is the bias-variance tradeoff.
Models with high bias are not complex enough and underfit the training data.
Models with high variance are too complex and overfit the training data.



underfiting data
(very simple model)



(good model)



overfiting data
(very complex model)

Bias vs. Variance ?

How to diagnose if we have a high bias problem or high variance problem ?

High Bias (underfitting) problem:
Training error and Validation error are both **high**

High Variance (overfitting) problem:
Training error is **low** and Validation error is much **higher** than training error.

Bias vs. Variance ?

High Bias (underfitting) :

•Increasing model complexity allows your model to capture more intricate patterns in the data.

•Adding relevant features gives your model more information to learn from.

•Reducing regularization provides your model more freedom to learn complex patterns.

•Implementing boosting algorithms like XGBoost or AdaBoost can systematically improve your model's performance.

High Variance (overfitting) :

•Increasing your training data helps your model learn more generalizable patterns.

•Implementing stronger regularization constrains your model's complexity.

•Reducing model complexity helps prevent memorization of training data.

•Using bagging algorithms like Random Forests can effectively reduce variance through ensemble learning.

The relationship between bias and variance → bias-variance trade-off

Optimize the Bias Variance balance

- ✓ **Start Simple**: Begin with a simple model to establish a baseline.
- ✓ **Monitor Performance**: Use cross-validation to track training and validation metrics.
- ✓ **Adjust Complexity**: Gradually increase complexity while observing changes in bias and variance.
- ✓ **Use Regularization**: Apply techniques like dropout or L2 regularization.
- ✓ **Analyze Learning Curves**: Identify whether adding more data or complexity improves performance.

Choose your optimization strategy based on (suggestion):

- ➢ Use **boosting** techniques when dealing with high **bias**
- ➢ Apply **bagging** methods when fighting high **variance**
- ➢ Consider **ensemble** approaches for **complex** scenarios

Learning Curves

Learning Curves

Until now, we have looked at the problem of evaluating classifier performance.

What about the evaluation of the learning algorithm itself?

How efficient is the given induction technique computationally?

And how good are the classifiers it induces?

Will better results be achieved if we choose some other machine-learning framework?

Now, we will study the costs of learning, considering:

- ✓ The number of examples needed for successful induction
- ✓ The computational time consumed

Learning Curves

The learning curve shows how the classification performance of the classifier depends on the size of the training set. The horizontal axis represents the number of training examples; and the vertical axis represents the classification performance/ error of the classifier.

Computational Costs:

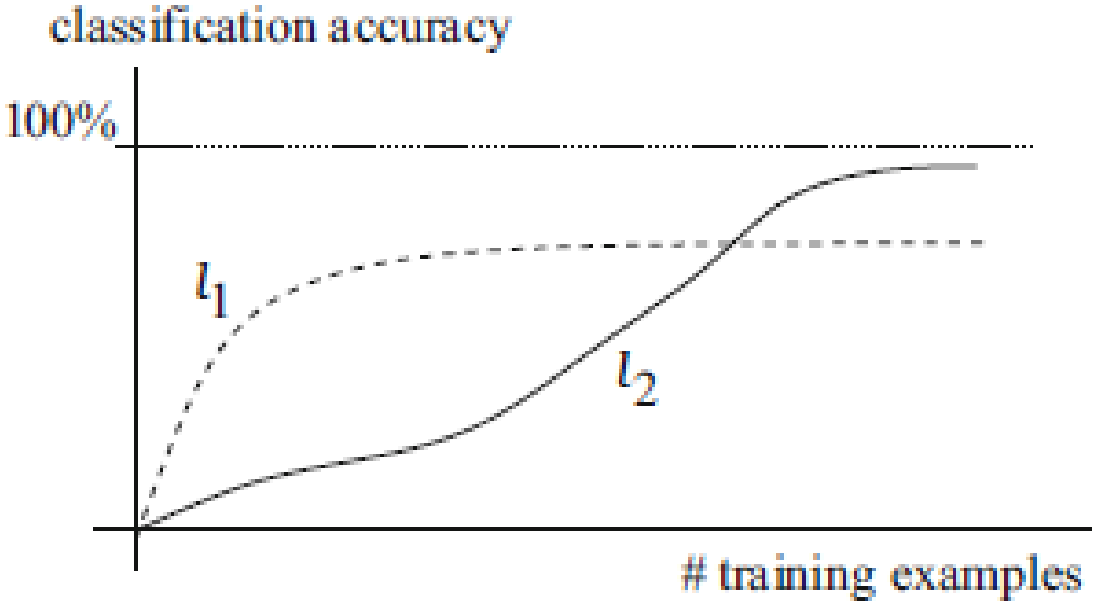
1. The time needed for the induction of the classifier from the available data.
2. The time it takes to classify a set of examples with the classifier thus induced.

They depend on:

- the number of the training examples
- the number of attributes describing them

Training and classification costs vary across different classifiers; this should be a variable in the decision on the selection of the most appropriate machine-learning paradigm.

Learning Curves



Which one do you select as best model?

Learning Curves

A learning curve is a visual and mathematical representation of how proficiency at a task improves over time or with repeated experience.

Learning curves:

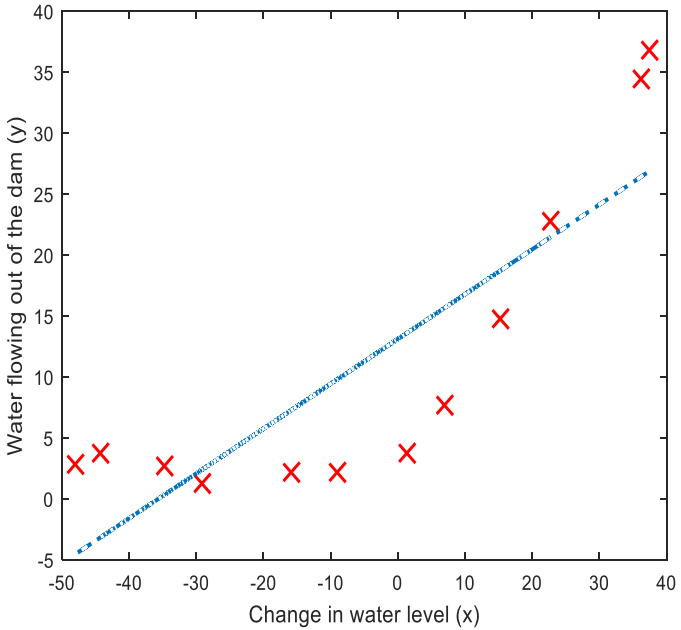
- plots that show changes in learning performance over time in terms of experience.
- plots used to show a model's performance as the training set size increases.
- evaluation on the train and validation datasets can be used to diagnose an underfit, overfit, or well-fit model.
- can be used to diagnose whether the train or validation datasets are not relatively representative of the problem domain.

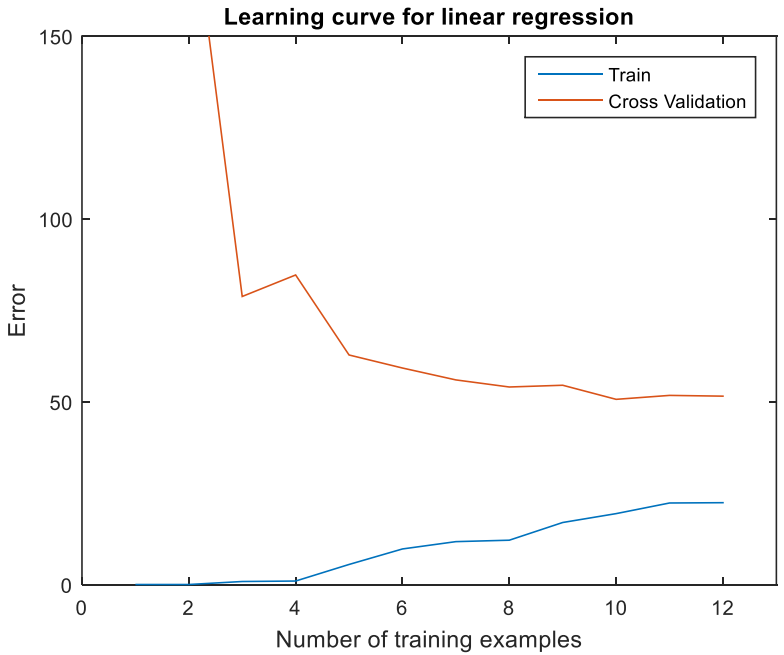
Train Learning Curve: Learning curve calculated from the training dataset that gives an idea of how well the model is learning.

Validation Learning Curve: Learning curve calculated from a validation dataset that gives an idea of how well the model is generalizing.

Learning Curves

$$h_\theta(x) = \theta_0 + \theta_1 x$$



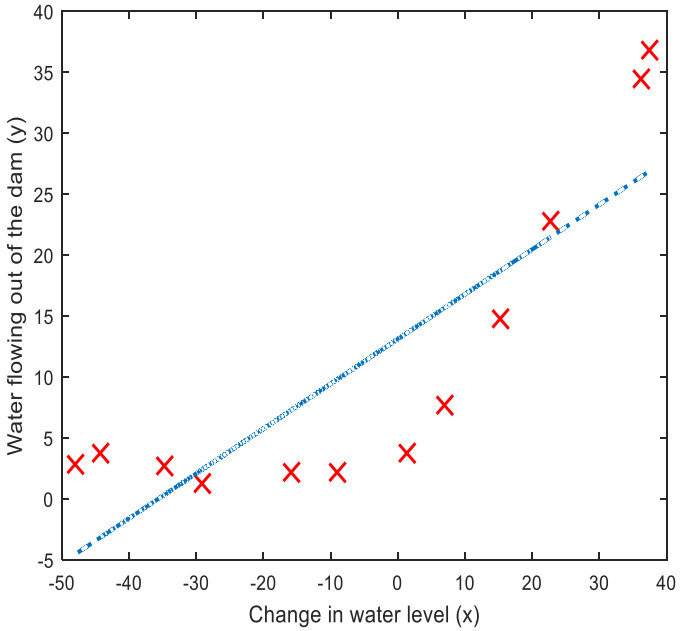


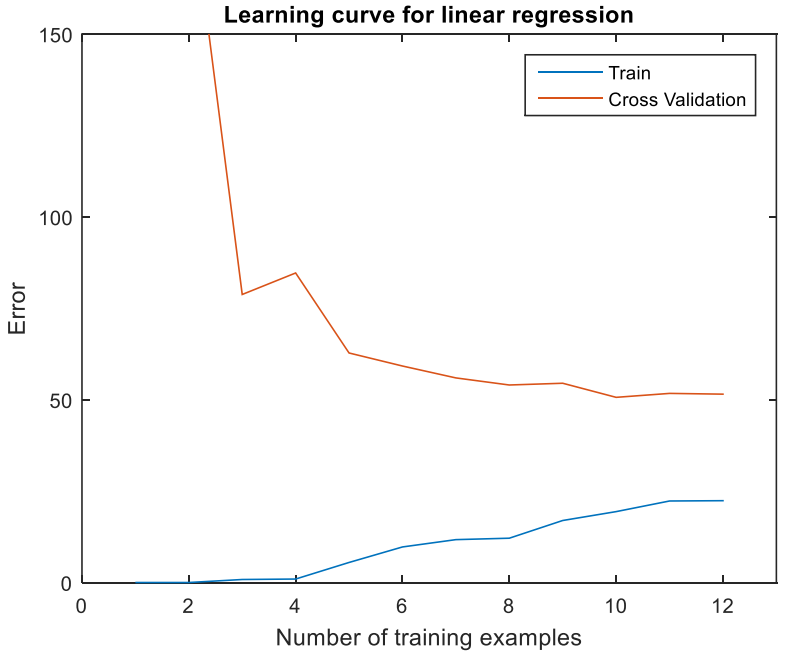
What is the problem: high bias or high variance ?

Learning curves are plots that display the performance of a machine learning model as the number of training samples increases. We use them to evaluate the model's ability to generalize to new, unseen data, and to identify issues, such as overfitting and underfitting.

Learning Curves

$$h_\theta(x) = \theta_0 + \theta_1 x$$

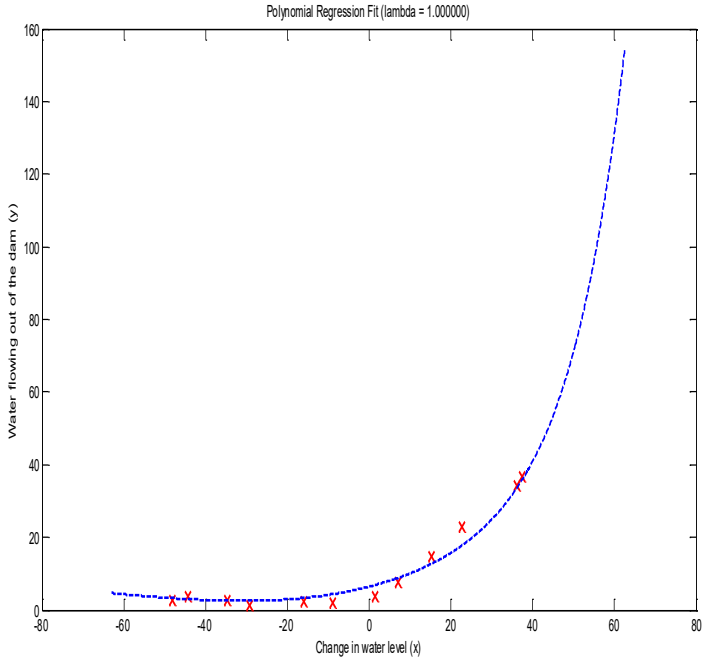


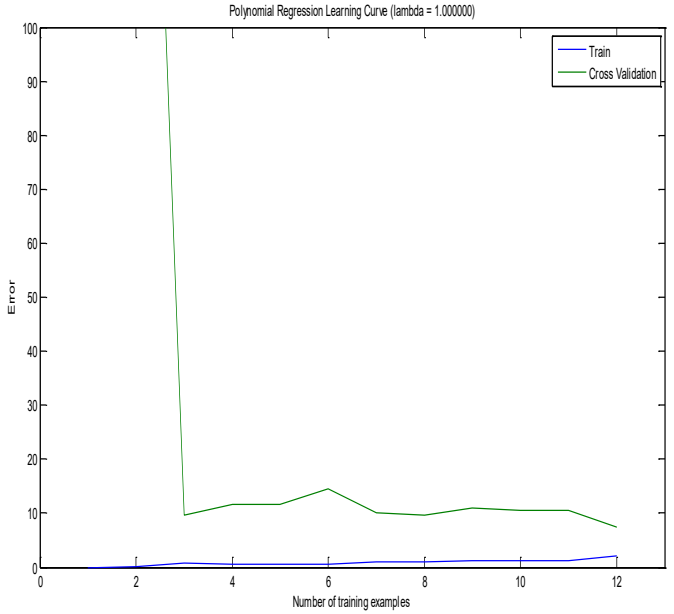


If a learning algorithm is suffering from high bias, getting more training data will not help much.

Underfitting. In this case, the training and validation curves can even converge to a plateau, but the plateau is far from the desired value of the cost function.

Learning Curves





If a learning algorithm is suffering from high variance, getting more training data is likely to help.

Overfitting. The two curves can even converge to a plateau, but there is a huge gap between them, showing that there is a high difference in the value of the cost function chosen.

Regularization and Learning Curves

For a given model, grid search on different values of λ =[0, 0.01, 0.1, 1,….]

Step 1: For each λ, optimize parameters ϑ using the training set

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

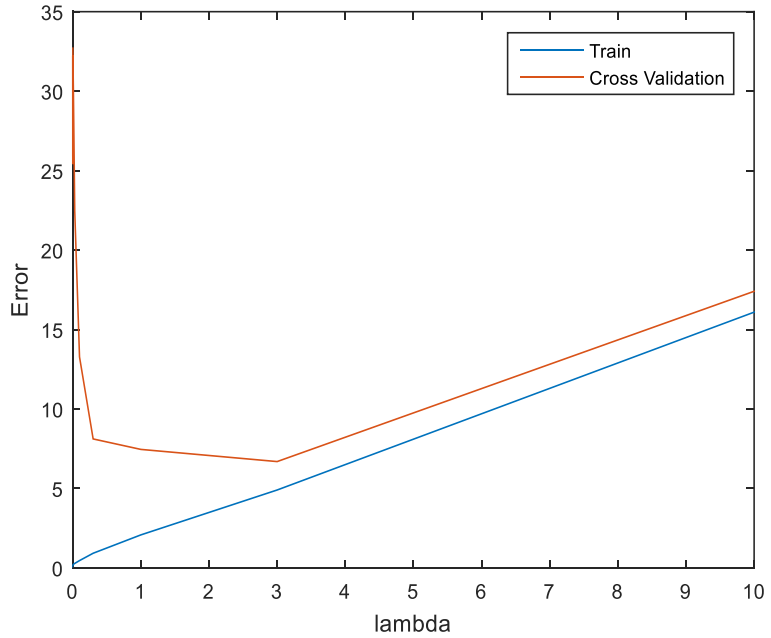
Step 2: Test the optimized models from step 1 with the CV set and choose the model with λ that gets min CV error:

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

Step 3: Retrain the model with the best λ from step 2 with both train and CV sets, starting from the parameters ϑ got at step 2. Test the retrained model with the test set and compute the error:

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

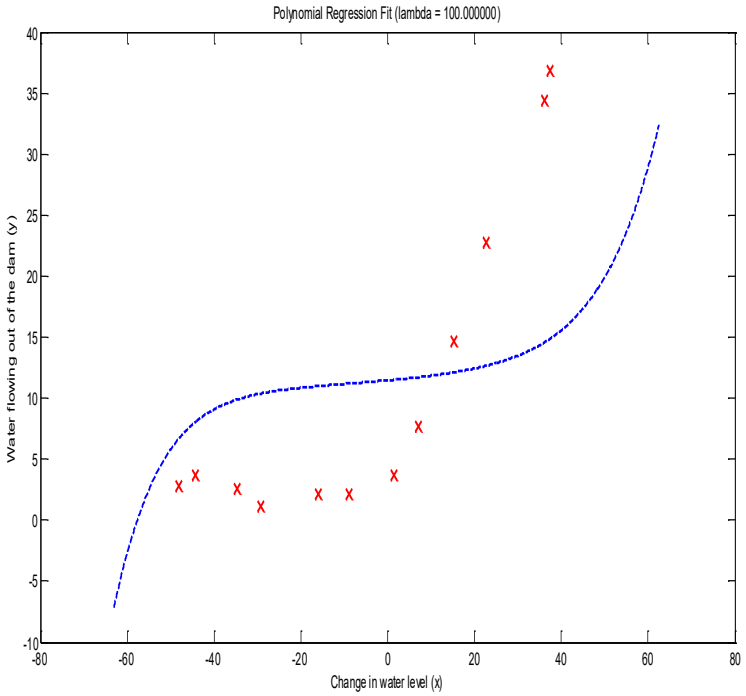
Regularization and Learning Curves



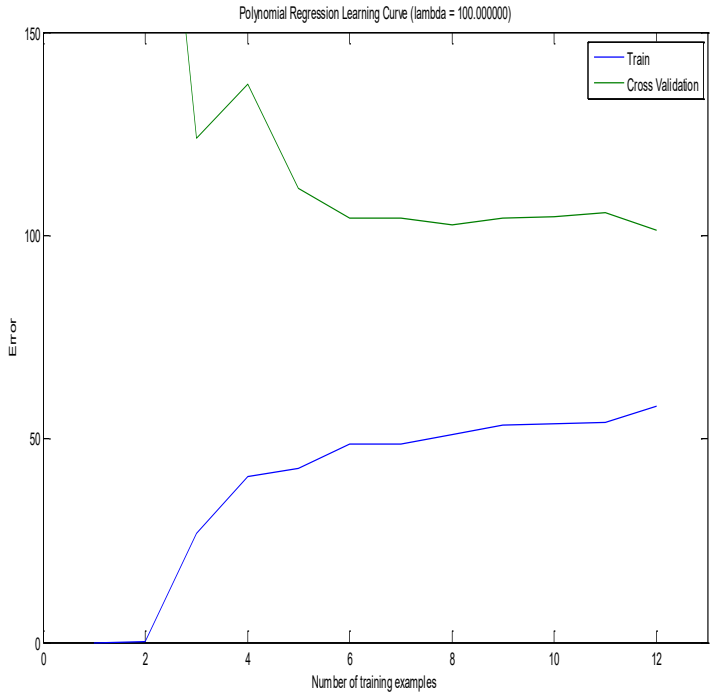
Best λ =3

Why?

Regularization and Learning Curves



Polynomial regression, λ =100



Learning curve, λ =100

Hints to improve ML model

Suppose you have learned a data model (hypothesis).
However, when you test your hypothesis on a new set of data, you find that it makes unacceptably large errors in its prediction (regression or classification).

What should you try next?

- ✓ Get more training examples – fixes high variance
- ✓ Try smaller sets of features – fixes high variance
- ❑ Try getting additional features – fixes high bias
- ❑ Try adding polynomial features - fixes high bias

Hints to improve ML model

Suppose you have learned a data model (hypothesis).
However, when you test your hypothesis on a new set of data, you find that it makes unacceptably large errors in its prediction (regression or classification).
What should you try next?

Underfit -> high bias

A regression problem where the model is experiencing unacceptably large errors on new data.

Probably the model is underfitting (high bias):

- the model isn't complex enough to capture the underlying patterns in the data
- resulting in high bias and poor performance both on the training set and new data
- In the case of regression, this would manifest as large prediction errors

Hints to improve ML model

Suppose you have learned a data model (hypothesis).
However, when you test your hypothesis on a new set of data, you find that it makes unacceptably large errors in its prediction (regression or classification).
What should you try next?

Overfit -> high variance

A regression problem where the model is experiencing unacceptably large errors in prediction when testing on new data.

In an overfitting scenario, a model learns the noise and specific details in the training data so well that it doesn't generalize to new, unseen data. This is often characterized by:

- Low error on the training data (because the model fits it too closely)
- High error on new test data (because the model overfits to the training set's peculiarities and doesn't generalize well).

Hints to improve ML model

Suppose you have learned a data model (hypothesis).
However, when you test your hypothesis on a new set of data, you find that it makes unacceptably large errors in its prediction (regression or classification).
What should you try next?

The description primarily suggests underfitting (high bias), but it could also suggest overfitting (high variance).

To distinguish between the two:

- If your model performs poorly on both training and test data, it is underfitting (high bias).
- If your model performs well on training data but poorly on new data, it is more likely overfitting (high variance).

No free lunch

The No-Free-Lunch Theorem

No universal best algorithm:

- ✓ There is no “holy grail” machine-learning method that works best in all situations.
- ✓ Every learning paradigm has strengths and weaknesses.
- ✓ A method that performs well in one type of problem may fail in another.

Implication for practice:

- ✓ The choice of model cannot be made in the abstract.
- ✓ Engineers must rely on systematic experiments and empirical evaluation.
- ✓ Different classifiers and learning algorithms should be tested for the specific task and dataset.

The No-Free-Lunch Theorem

- ➢ Mathematically proven result in machine learning.
- ➢ No algorithm outperforms all others across all possible problems.
- ➢ Superior performance on some problems implies worse performance on others.

The goal is not to find the best algorithm overall, but to find the best algorithm for the specific problem at hand.