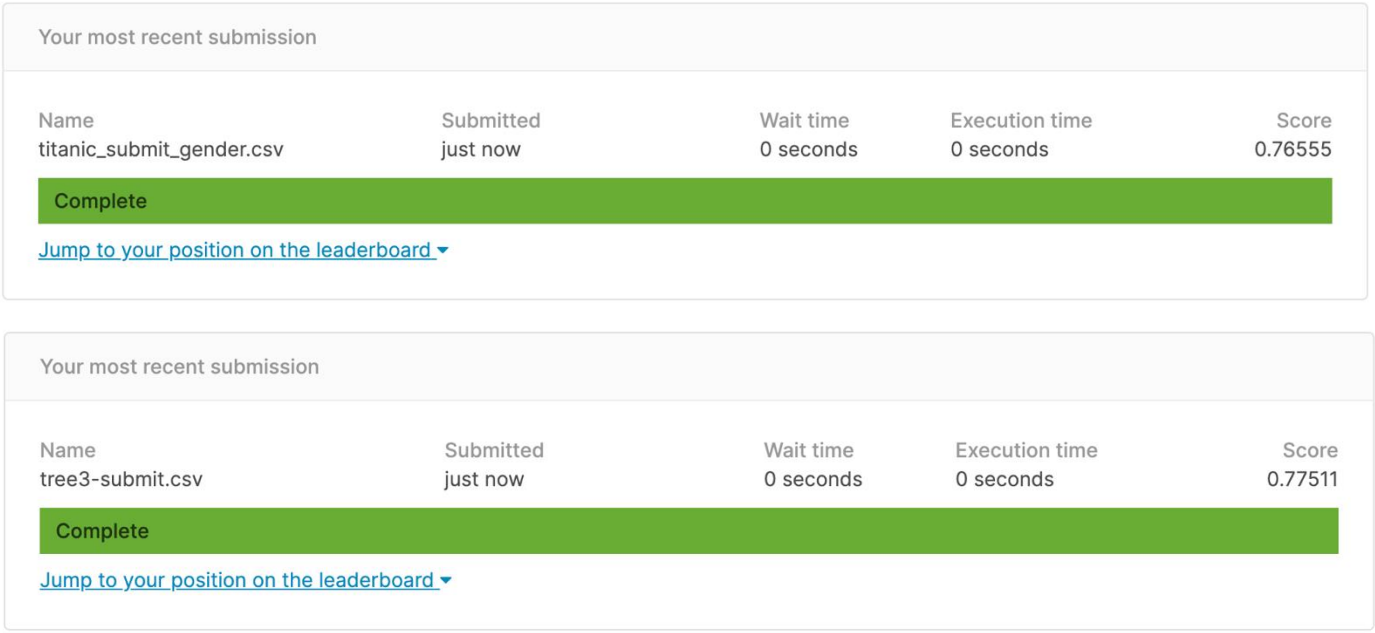
Model Evaluation

Harry Jiannan Wang, Ph.D.

Professor of Management Information Systems

University of Delaware

Performance Measures



How is the Kaggle score calculated?

- For classification problems:
  - Confusion matrix
  - Accuracy, Precision, and Recall

Confusion Matrix

- A basic confusion matrix for a binary classification with two classes:

Actual Label			
True	True Positive (TP)	False Negative/Type II Error (FN)
False	False Positive/Type I Error)	True Negative (TN)
	True	False

Predicted Label

Green: Correct Predictions
Red: Wrong Predictions

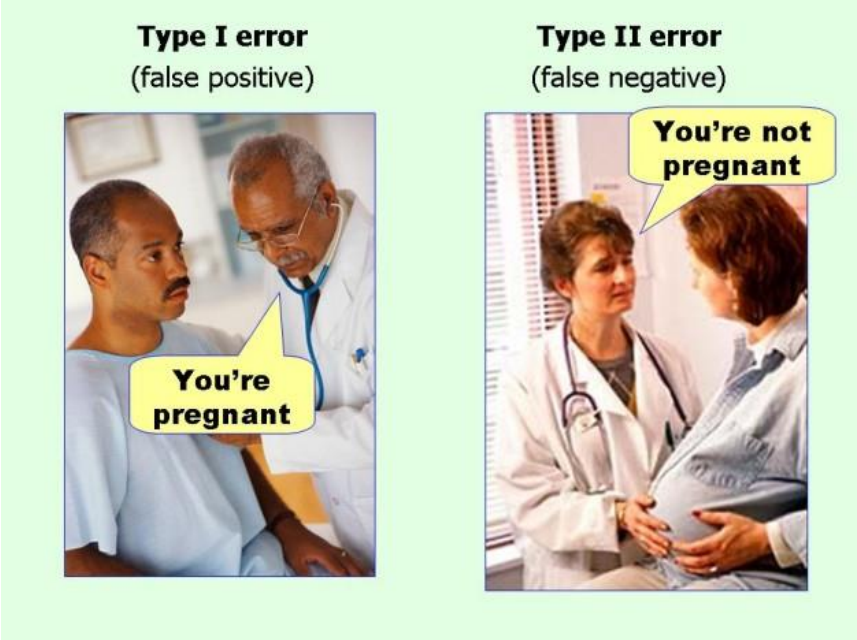


Image Credit: https://bit.ly/2UIxPIe

Accuracy

Actual Label		
True	True Positive (TP)	False Negative/Type II Error (FN)
False	False Positive/Type I Error)	True Negative (TN)
	True	False

Predicted Label

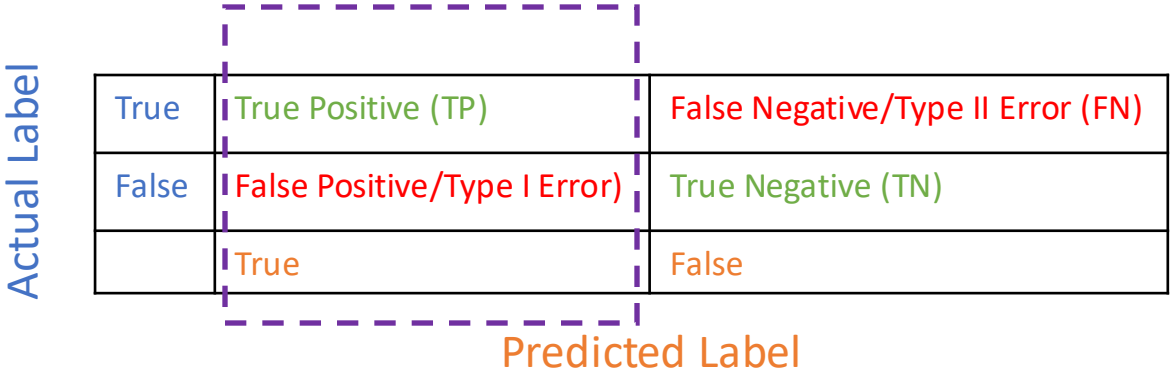
- Accuracy: the ratio of correct predictions to all predictions

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of instances in testing set}}$$

$$\text{Accuracy} = (TP + TN)/(TP + TN + FP + FN)$$

Accuracy is the Kaggle Score for the Titanic Competition

Precision



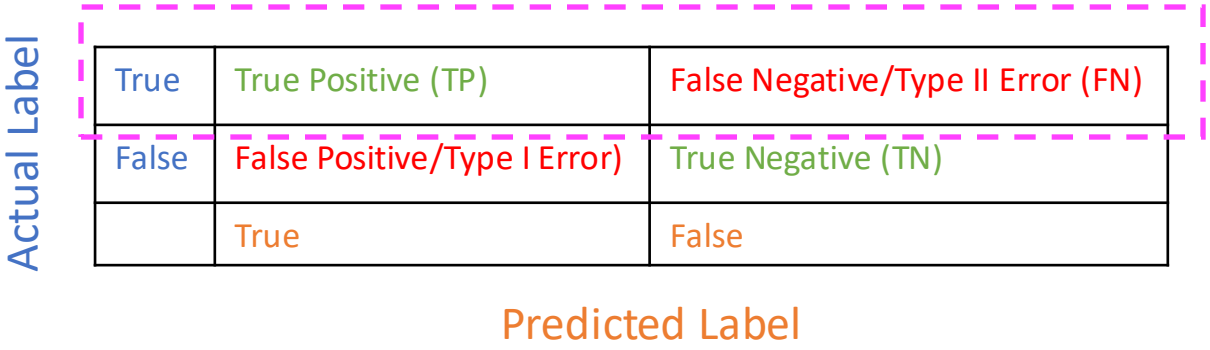
Precision (P) is the ratio of correct positive predictions to all positive predictions:

Precision = TP/TP+FP

*For passengers predicted to be "survived", how many of them are **actually** survived.*

https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html

Recall



Recall (R) is the ratio of correct positive predictions to all true positives.

Recall = TP/TP+FN

For all survived passengers, how many are predicted to be "survived"

https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html

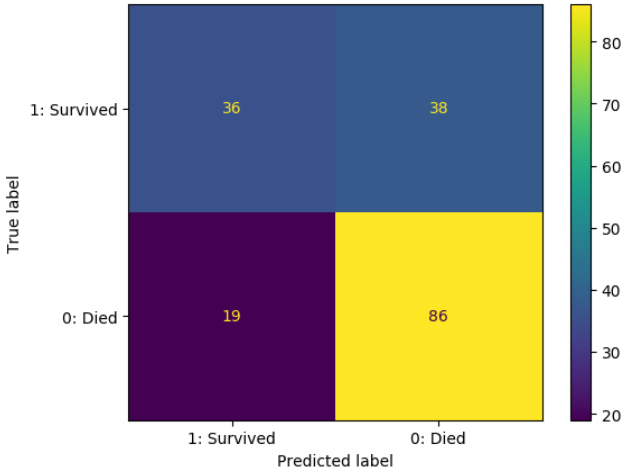
F1 Score

- F1 Score combines the precision and recall of a classifier into a single metric by taking their harmonic mean.

- Harmonic Mean = n / [(1/x1)+(1/x2)+(1/x3)+…+(1/xn)] gives less weight to the large values and large weight to the small values to balance the values
- F1 score is higher for models with balanced precision and recall (e.g., models with high precision but low recall tend to have lower F1 score)

F1 = 2(Precision * Recall)/(Precision + Recall)

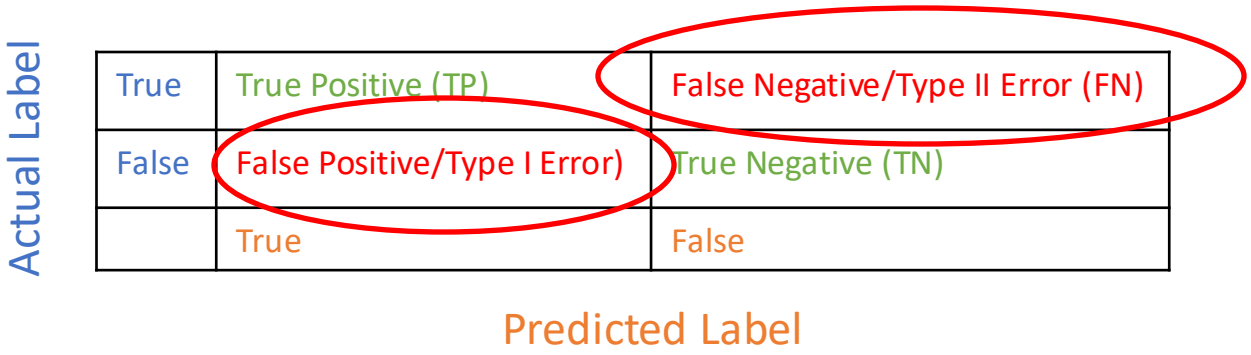
Precision and Recall Exercise



- Positive (Survived), Negative (Died)
  - TP (36): 35 people predicted survived actually survived
  - TN (86): 86 people predicted died actually died
  - FP (19): 19 people predicted survived actually died
  - FN (38): 39 people predicted died actually survived

- Calculate and interpret precision and recall:
  - Precision is 36/36+19= 0.655
  - Recall is 36/74 = 0.486
  - F1 is 2(0.655 * 0.486)/(0.655 + 0.486) = 0.558

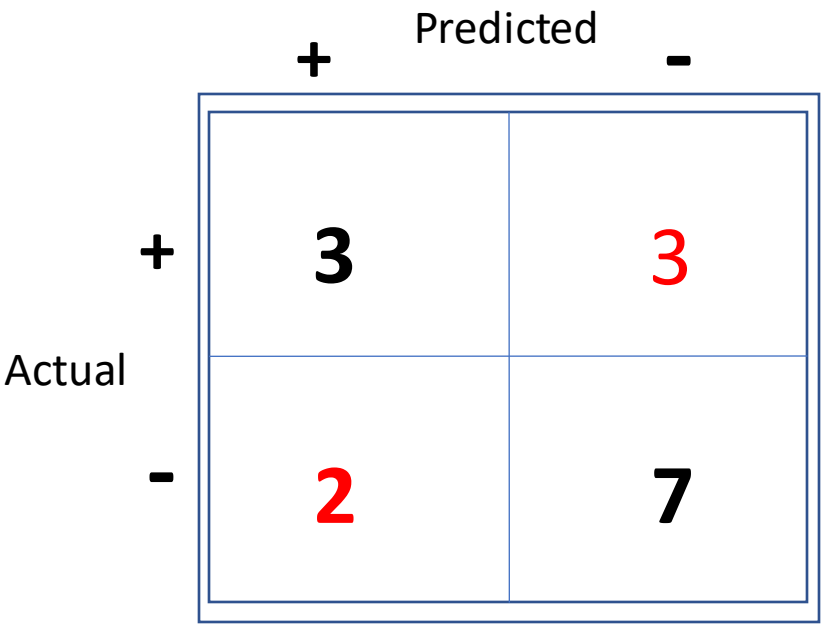
Uneven Costs of Two Types of Prediction Errors



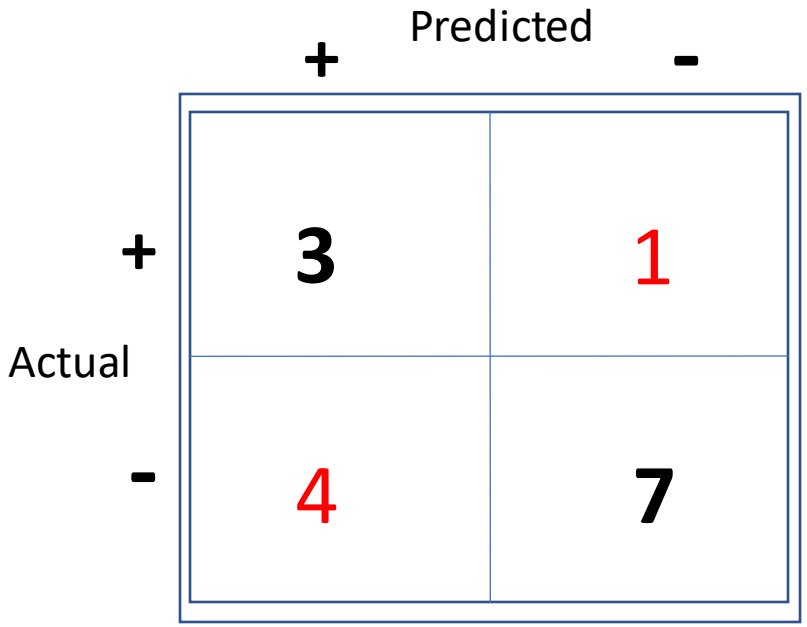
In many applications, different types of errors have different costs

- wrongly diagnose an ill patient as normal costs a lot more than wrongly diagnose a normal people as ill.
- wrongly approve a credit card application costs a lot more than wrongly deny an application
- wrongly filter out a good email costs a lot more than wrongly accept a spam

Evaluate a model using total cost



Error of model 1



Error of model 2

Both models have the same **Accuracy = 10/15 = 0.667**

With equal cost: Cost (FP) = $5 Cost (FN) = $5

Total cost of model 1 = 2*5 +3*5 = **$25**

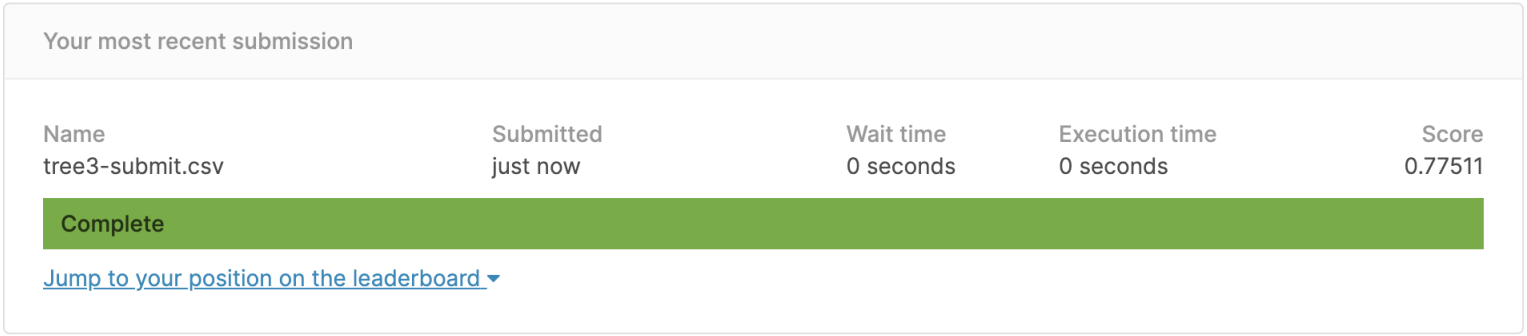
Total cost of model 2 = 4*5 +1*5 = **$25**

With uneven cost: Cost (FP) = $5 Cost (FN) = $20 (such as cancer detection)

Total cost of model 1 = 2*5 +3*20 = **$70**

Total cost of model 2 = 4*5 +1*20 = **$40**

Evaluate without Kaggle?



- In the current process, you don’t know your model’s performance till you upload your prediction to Kaggle to see the score
- How can we know our model’s performance (score) without going to Kaggle?
- In real projects, you don’t have a place as Kaggle to test your model anyway.

Holdout Validation

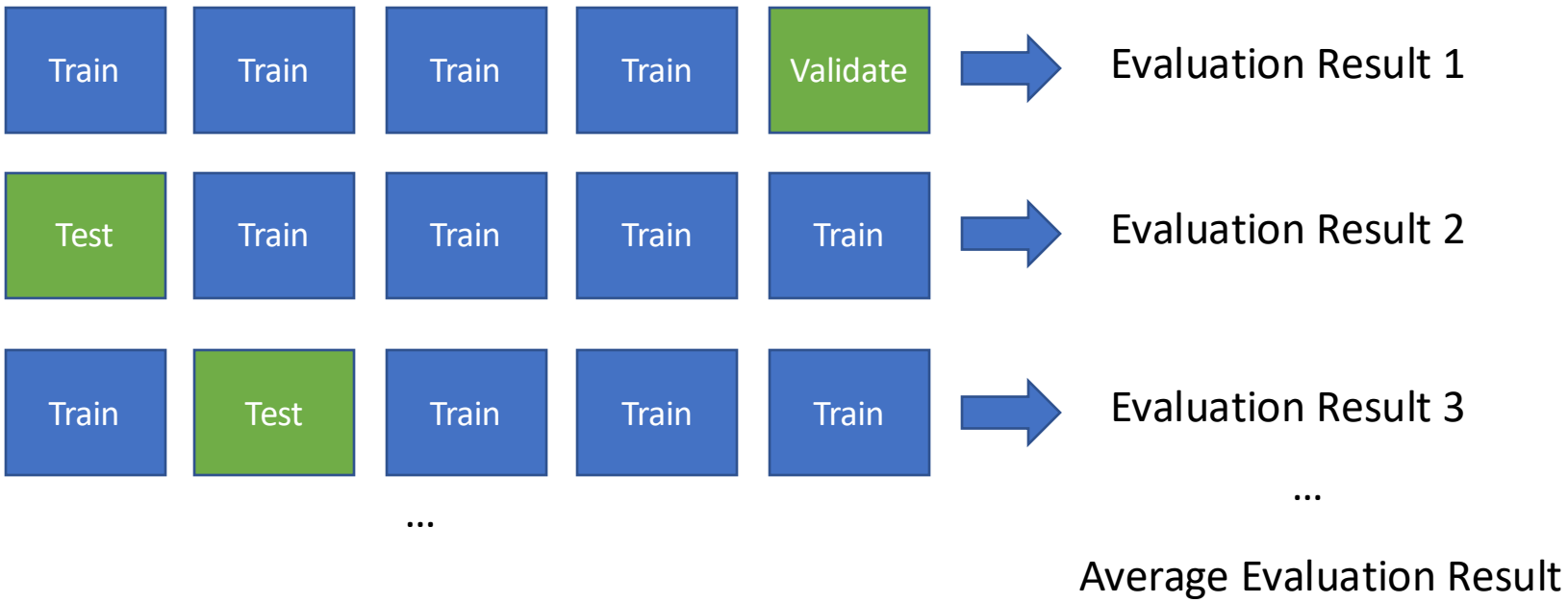
- Split the dataset into Train and Test (like Kaggle)
- Hold a portion of the train dataset to be the validation set
- Use the remaining train dataset to train the model and evaluate the model using the validation set
- Finally use the test set to evaluate the model

- Problems:

1. What if by accident a particularly easy or hard test set is selected?
2. Having a separate test set may reduce the number of samples used for learning the model

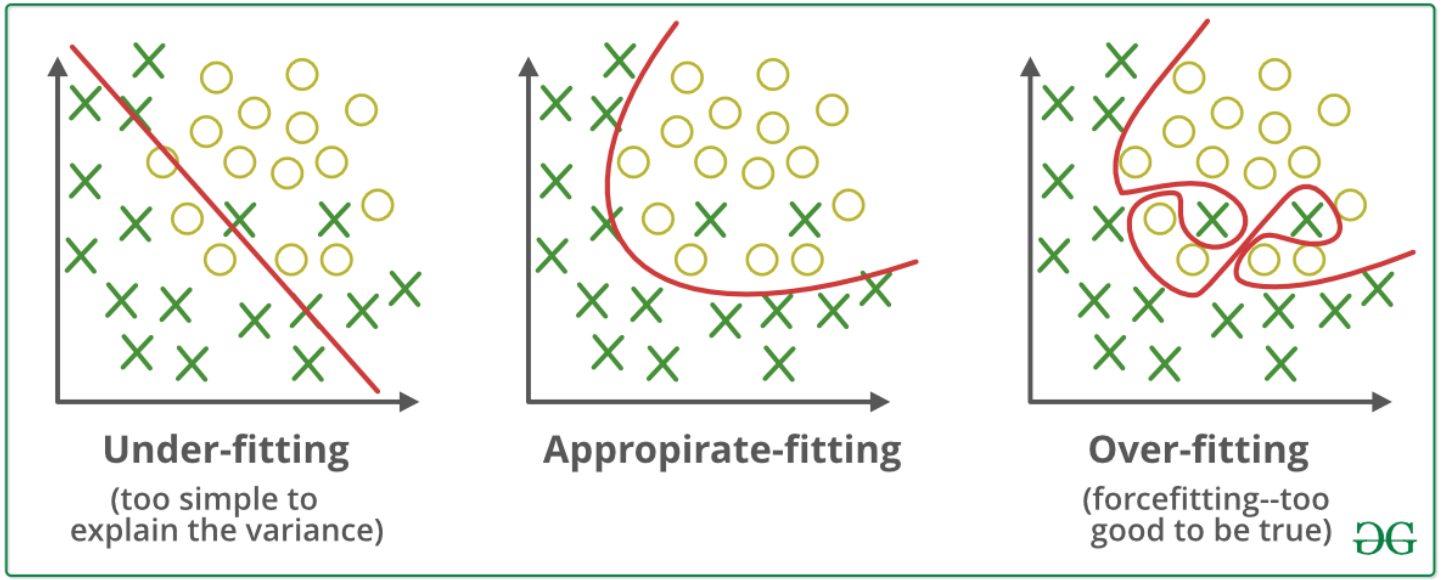
K-Fold Cross Validation

- The train dataset is divided into K folds.
- K-1 folds are used to train and the remaining one fold is used to validate
- Repeat K times and use the **average** evaluation result
- Cross Validation is for model checking (tuning parameters/comparing models) not for model building – once we find the best model, we use **ALL** training data to train the final model and evaluate on the test set.



http://en.wikipedia.org/wiki/Overfitting
https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/

Overfitting and Underfitting



- Underfitting happens when the model cannot capture the underlying patterns of the data and therefore has limited prediction power.
- Overfitting happens when the model corresponds too closely or exactly to a particular set of data and may fail to reliably predict future new unseen data.

https://elitedatascience.com/overfitting-in-machine-learning

Remedies for Overfitting and Underfitting

- Underfitting (detection: low performance for training)
  - Get more data
  - Add more features
  - Use more sophisticated models
- Overfitting (detection: high performance for training, low performance for testing)
  - Cross validation
  - Remove features
  - Regularization (techniques to force the model to be simpler, e.g., pruning a decision tree)
  - Early stopping

Remember the complicated tree we built before?