**Optimization Strategies and Algorithms**

**Section 2**

Optimize hyperparameters in NNWs in the beginning.

Hyperparameters are not directly learned by the learning algorithm, specified outside of the training procedure. Control the capacity of the model → how flexible the model is to fit the data. Prevent over-fitting.

**e.g.** Lasso, Ridge, Elastic net regularization → lambda is the hyperparameter.

Decision Tree → The number of features to evaluate, how deep we want our tree to be. The minimum numbers of the notes.

Random forests: number of trees, Learning rate (GBMs).

Neural Networks: Number of layers, number of neurons per layer, activation function, the dropout rate.

Nearest neighbours: the number of neighbors.

SVM: the kernel functions.

Methods: Fit several GBMs with different hyperparameters, measure performance → rmse

For hyperparameter: can’t define e math formula for them. Try different ones and evaluate.

→critical step: choose how many to try. Also want to lower computation costs.

Hyperparameter response surface

Find the hyperparameters that minimize (or maximize) a performance metric.

Low effective dimension.

**Section 3**

Evaluation Matrics

**Classification Matrics**

Accuracy: percentage of correct prediction.

Confusion Matric: TN, FP, FN, TP

Recall or sensitivity: TP\_rate = TP / (TP + FN)

(correctly predicted positive percentage)

Positive predictive value (precision) PP = TP / (TP+FP)

F1 score - weight ed harmonic mean of precision and recall

F1 = (2\* precision \* recall) /(precision +recall)

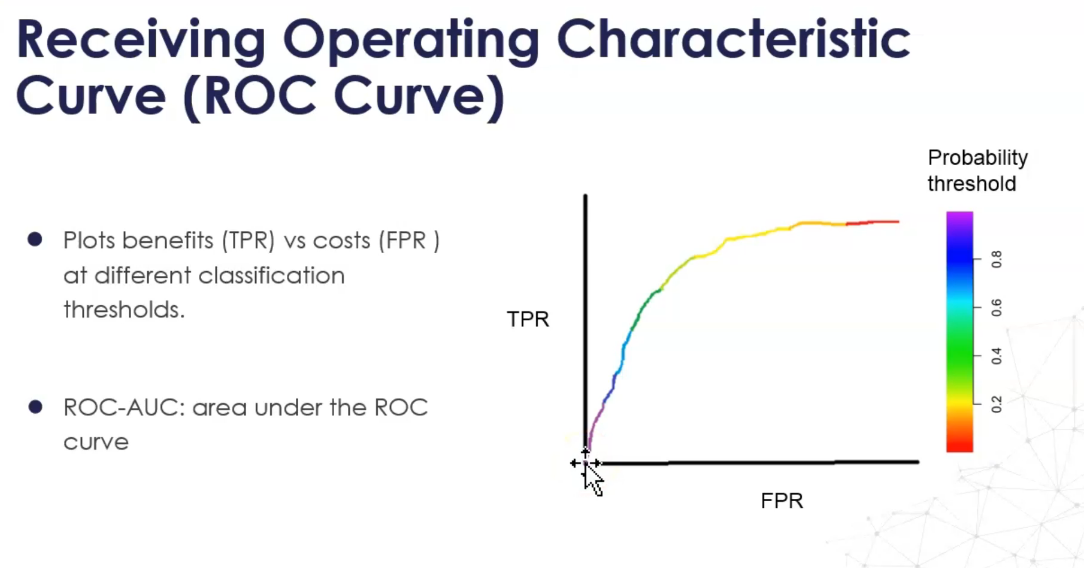
(THE LOWER THE BETTER)

False Positive Rate, FPR = FP / (FP+TN)

False Negative Rate, FNR = FN / (FN + TP)

Receiving operating characteristic curve (ROC)

x → FPR, y→ TPR



The higher ROC-AUC the better.

Lose Function

L(y,p) = -(ylog(p) + (1-y)log(1-p))

**Regression Metrics**

MSE 1/N ( SUM(Y\_i - y\_i^)^2)

RMSE = Sqrt(MSE)

MAE: mean absolute error

R^2 metric: sum of squares / variance of the sample (how much of the variance is explained by the model) → maximize

**Section 4**

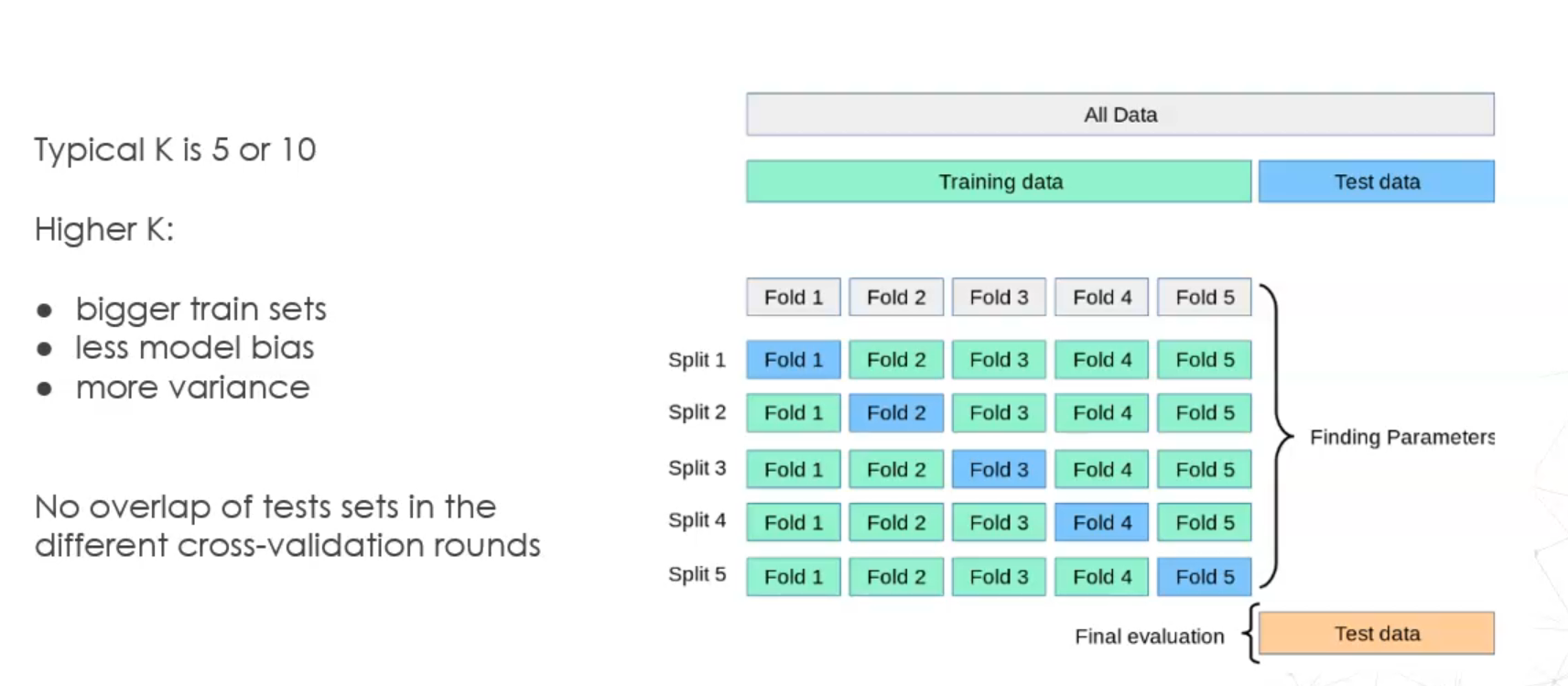
Cross-Validation

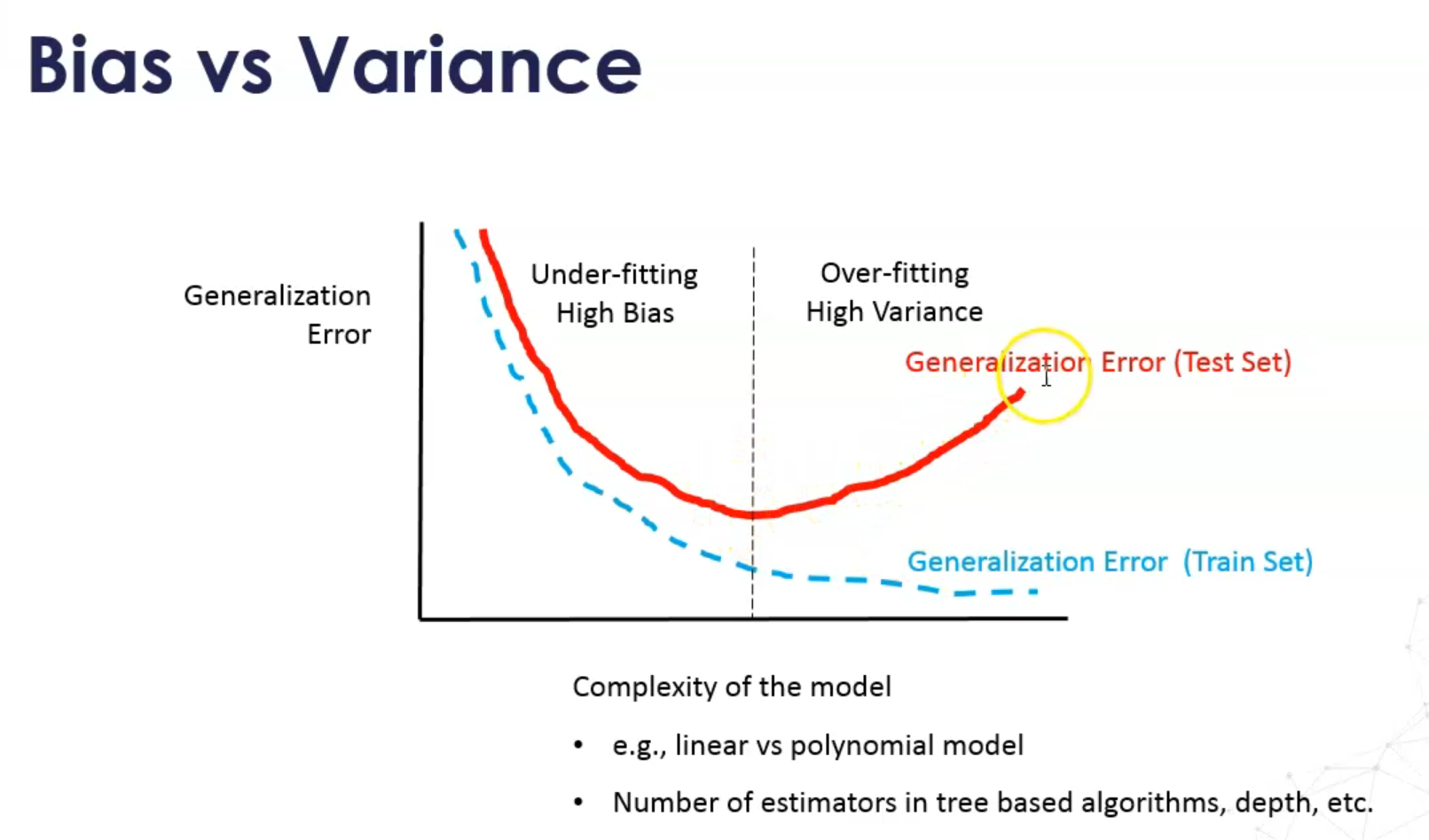
Generalization(effecttiveness scross various input data sets) vs Overfitting(performs well on the training data set, but poorly on the test set)

train-test split won’t help with the hyper-parameters search.

C-V: Make full use of the data set compared to just leaving a data set and divide the rest into multiple training sets.

Cross-validation Schemes





Leave one out cross-validation(good for regression): one data in the data set is the test date, others are the training set. → huge amount of models → can be used if your data set is small.

Leave p out → very computational expensive

**Stratified k-fold c-v**: only for classification

each fold has a similar proportion of observations of each class

useful with very imbalanced datasets.

Special c-v schemes

We assume the data sets are independent

Grouped data

1. Medical data were collected from patients, with multiple samples taken from each patient
2. Voice recognition, is pronounced by various speakers.

We would like to know if a model trained on a particular set of groups generalizeds well to the unseen groups.

To measure this, we need to ensure that all the samples in the validation fold come from groups that are not represented at all in the paired training fold.

(train the model in some subject and test it in another subject)

**Leave one group CV / Leave p out CV**

**Time series: orders.**