Machine learning

Feature Selection

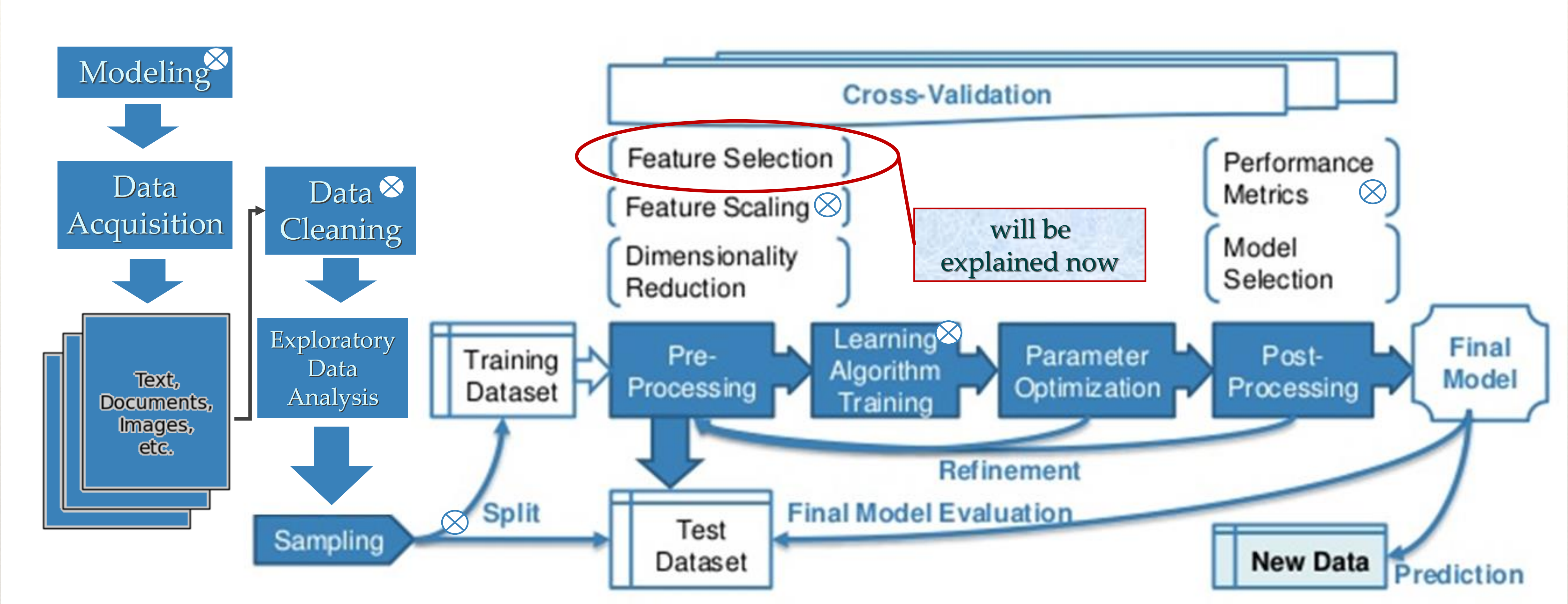
Lecture V

פיתוח:
ד"ר יהונתן שלר
משה פרידמן

What will we talk about

- A typical classification flow summary
- Feature selection

A typical classification flow
- diving in



Data Cleaning
→ Duplicates
→ Missing Data
  →Remove
  →Repair

Train-Test split
+ Validation-set

Data Exploration

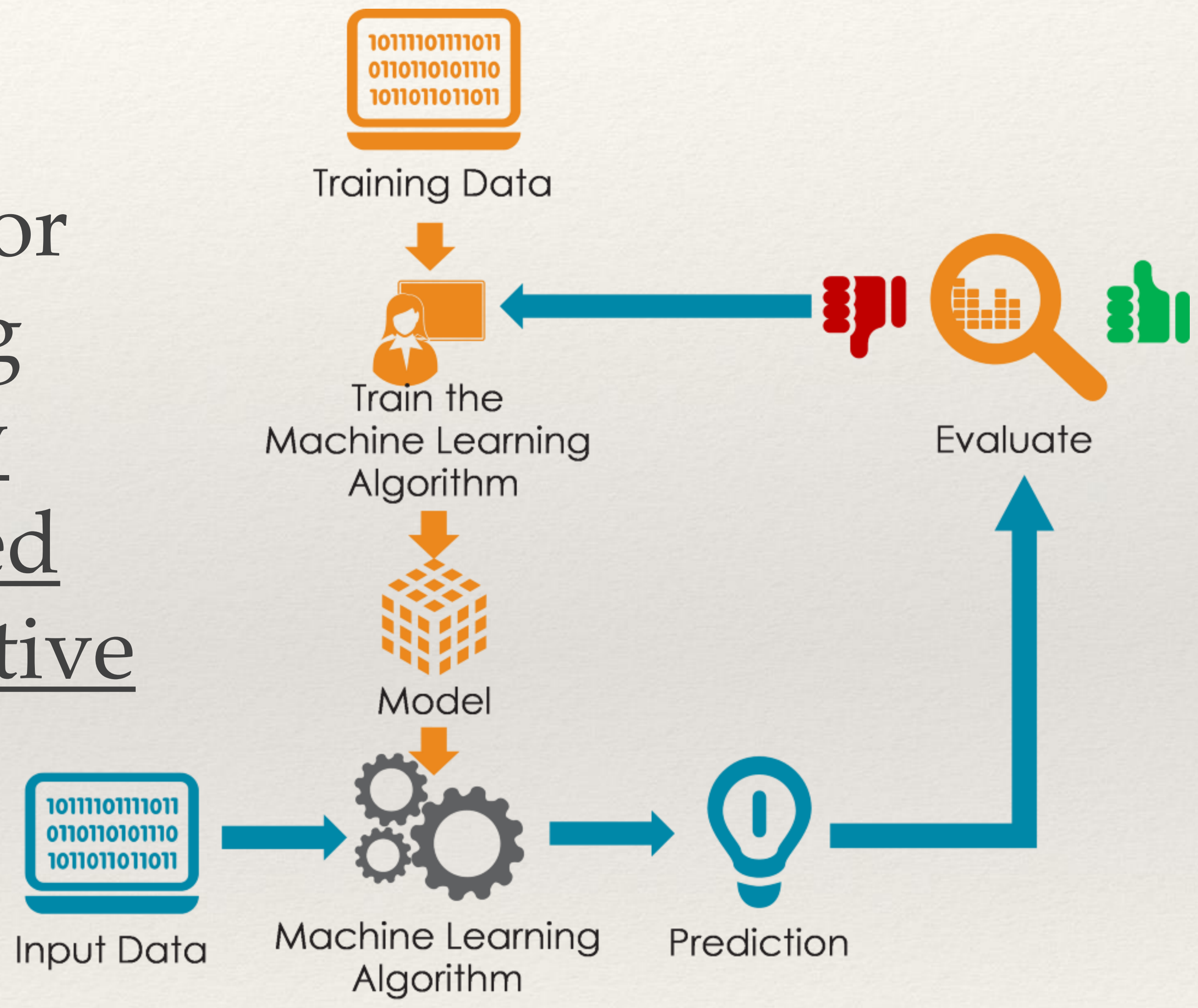
Scaling
→ Minmax norm.
→ t-dist. standardization

Learning Algos.
→ KNN
→Decision Trees
→Naïve Bayes

Evaluation
→Confusion matrix
→ Accuracy ,Error (rate)
→ Precision, Recall
→ F1 (soon)

Machine learning training

- A machine learning algorithm (e.g., classification, regression or clustering) uses a training dataset to determine <u>how can the features be applied to unseen data for predictive purposes</u>.



What is feature selection

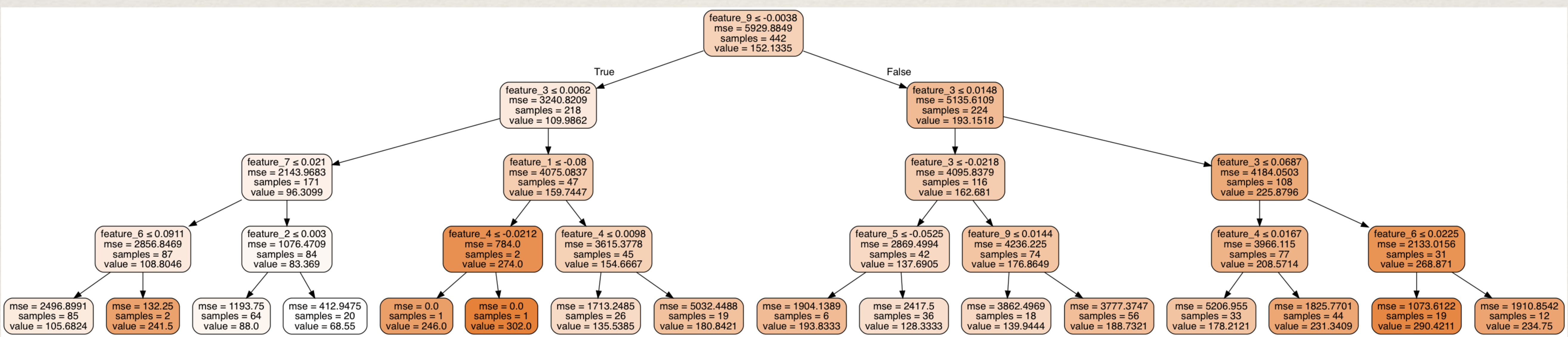
- feature selection is the process of selecting a subset of relevant features for use in model construction” or in other words, the selection of the most important features

All features

Selected Features

Feature selection and Dimensionality Reduction - Motivation

Prevents Overfitting: A high-dimensional dataset having too many features can sometimes lead to overfitting (model captures both real and random effects).



For example – a decision tree could over-fit a trainset with too many features

Feature selection and Dimensionality Reduction - Motivation

Prevents Overfitting: A high-dimensional dataset having too many features can sometimes lead to overfitting (model captures both real and random effects).

Simplicity: An over-complex model having too many features can be hard to interpret.

Feature selection & Dimensionality Reduction - Motivation

Prevents Overfitting: A high-dimensional dataset having too many features can sometimes lead to overfitting (model captures both real and random effects).

Simplicity: An over-complex model having too many features can be hard to interpret especially when features are correlated with each other.

Computational Efficiency: A model trained on a lower-dimensional dataset is computationally efficient (execution of algorithm requires less computational time).

Feature selection – techniques
1. not complex enough for learning

1. a. A trivial case – constant value (variance = 0)

→ We mentioned this example

1. b. Remove features with low variance

- Features with very low variance (under a threshold) are not complex enough for learning.

Feature selection – techniques
2. highly correlated features

2. Remove highly correlated features

- High correlated features could cause distortion of distance functions (KNN).
- Features that are highly correlated or co-linear can cause overfitting (NB)
- When a pair of variables are highly correlated, we can remove one without much loss of information.
    - Which one should we keep?
        - The one with a higher correlation to the target (see ahead)

Feature selection – techniques
2. highly correlated features

2. Remove highly correlated features

- Features that are highly correlated or co-linear can cause overfitting.
  - zero imply weak or no correlation
  - coefficients are used to measure the strength of the relationship between two variables.
  - A trivial case – duplicate features

Feature selection – techniques
2.a. highly correlated features - Pearson correlation

2. Remove highly correlated features

- Features that are highly correlated or co-linear can cause overfitting.
  - Pearson correlation is the one most commonly used
  - Linear correlation
  - Values always range:-1 (strong negative relationship) and +1 (strong positive relationship). 0 – no correlation.

Feature selection – techniques
2.a. highly correlated features - Pearson correlation

Covariance in the population:

$$Cov(X,Y) = \frac{\sum_{i=1}^{n}(x_i - \bar{X})\cdot(y_i - \bar{Y})}{n}$$

Where:

X_i – the values of the X-variable

Y_j – the values of the Y-variable

$\bar{X}$ – the mean (average) of the X-variable

$\bar{Y}$ – the mean (average) of the Y-variableהקלד משוואה כאן.

n – the number of data points

Covariance in the sample:

$$Cov(X,Y) = \frac{\sum_{i=1}^{n}(x_i - \bar{X})\cdot(y_i - \bar{Y})}{n-1}$$

Positive covariance: Indicates that two variables tend to move in the same direction.

Negative covariance: Reveals that two variables tend to move in inverse directions.

Covariance Measures relationship, not strength

Feature selection – techniques
2. a. highly correlated features - Pearson correlation (2)

2. Remove highly correlated features

- Features that are highly correlated or co-linear can cause overfitting.

Pearson correlation - between x,y

$$\rho_{X,Y} = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y}$$

$$= \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}}$$

Feature selection – techniques
2. highly correlated features – NMI

2. Remove highly correlated features

- Mutual information between two features (f1, f2)

Normalized Mutual information

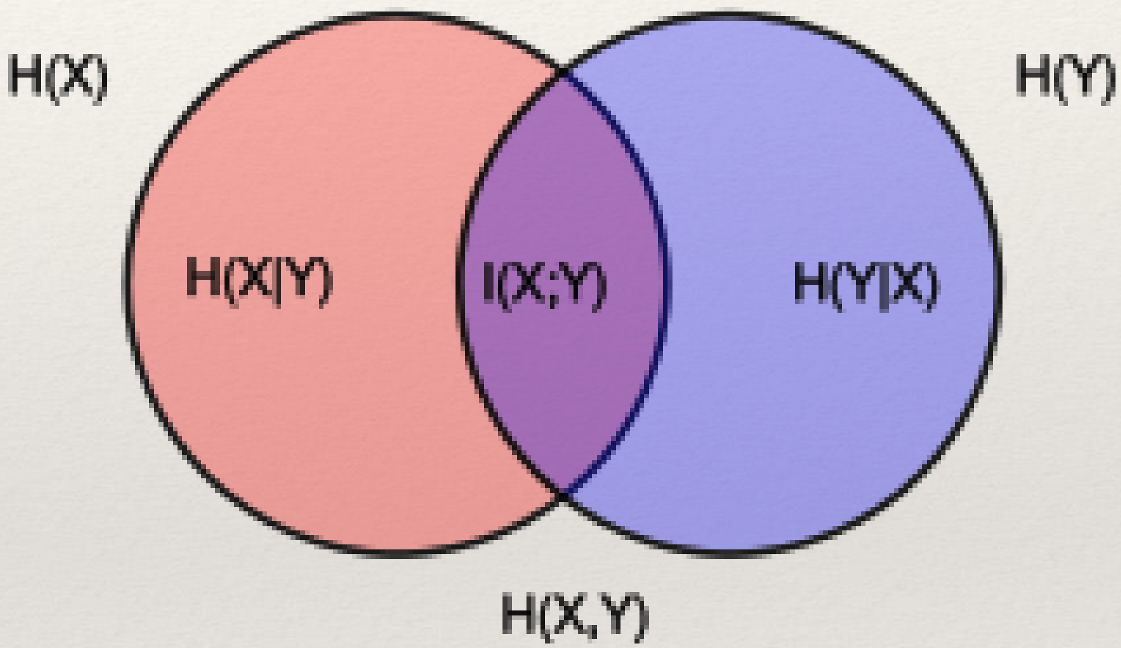
$$\text{NMI} = \frac{IG(f1; f2)}{|H(f1) + H(f2)|/2}$$

- NMI value close to 1
  → high similarity
- NMI value close to 0
  → high dissimilarity

Feature selection – techniques
3. features with high correlation to target

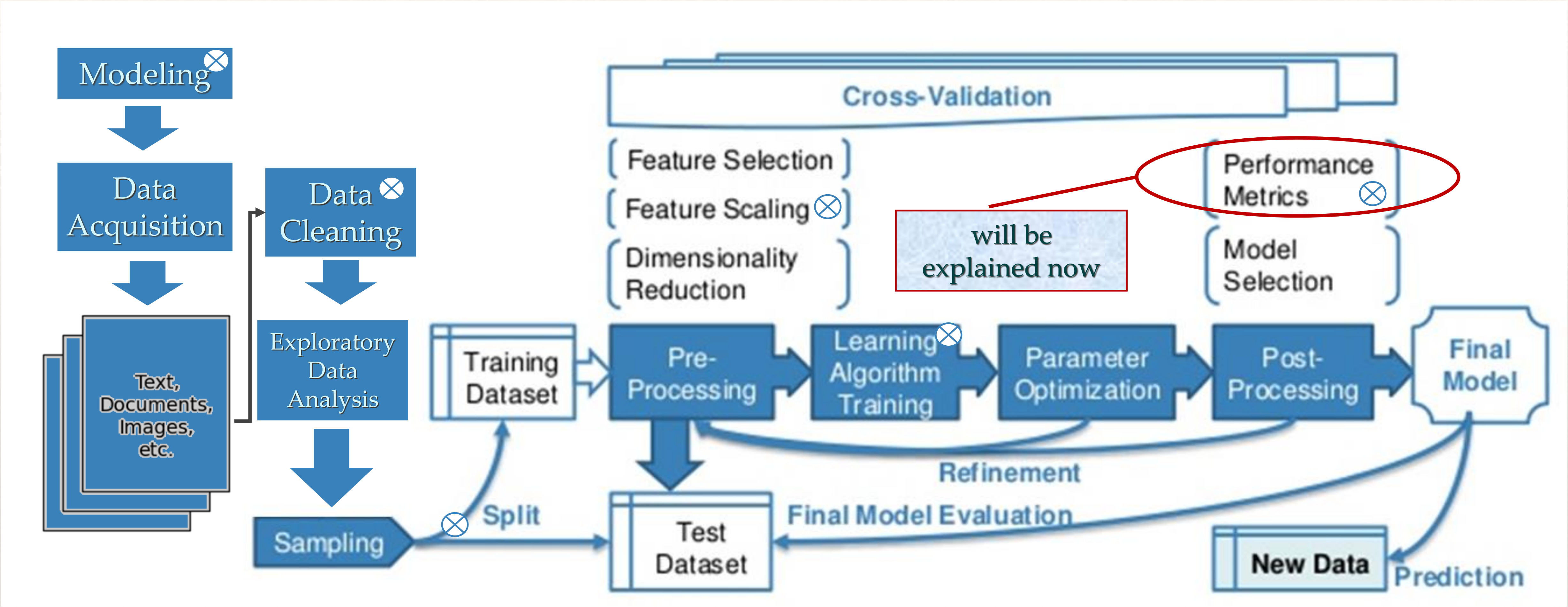
3. Select features with high correlation to target

- Features that are high correlation to the class.
- How to use?
  - Choose top k features
  - Choose features passing threshold
- E.g., Mutual information based



$$\text{NMI} = \frac{IG(X|Y)}{|H(X) + H(Y)|/2}$$

A typical classification flow - diving in



Data Exploration

Data Cleaning
→ Duplicates
→ Missing Data
→Remove
→Repair

Train-Test split
+ Validation-set

Scaling
→ Minmax norm.
→ t-dist. standardization

Learning Algos.
→ KNN
→Decision Trees
→Naïve Bayes

Evaluation
→Confusion matrix
→ Accuracy ,Error (rate)
→ Precision, Recall
→ F1 (soon)

Classification Measures – Confusion Matrix - Reminder

	Class 1 Predicted	Class 2 Predicted
Class 1 Actual	TP	FN
Class 2 Actual	FP	TN

- Class 1: Positive
- Class 2: Negative
- TP: True Positive
- FN: False Negative
- FP: False Positive
- TN: True Negative

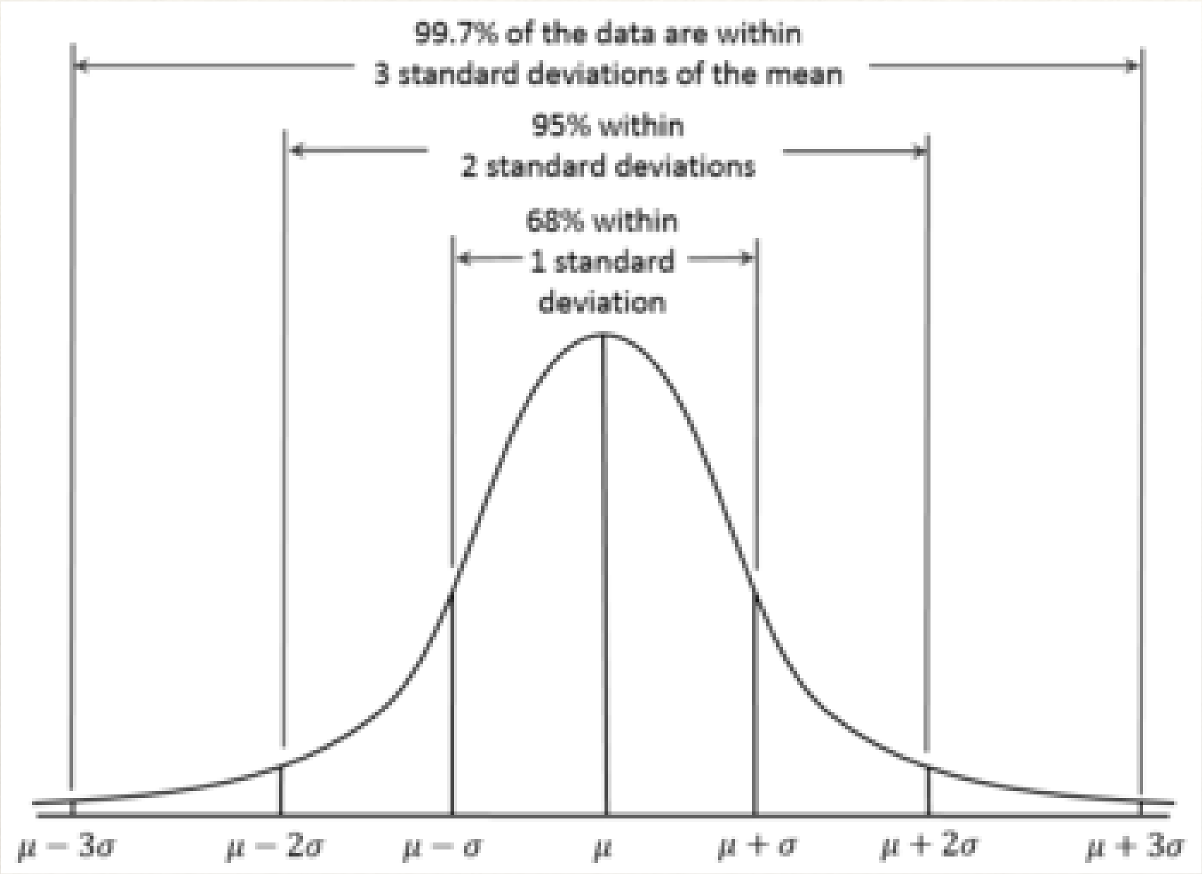
- Classification Rate / Accuracy: $$\frac{TP + TN}{TP + TN + FP + FN}$$
- Precision $$\frac{TP}{TP+FP}$$
- Recall $$\frac{TP}{TP+FN}$$

- High recall, low precision: Most of the positive examples are correctly recognized (low FN) but there are a lot of false positives.
- Low recall, high precision: We miss a lot of positive examples (high FN) but those we predict as positive are indeed positive (low FP).

Classification Measures – F1 Score

- It is useful to have one number to measure the performance of the classifier
- $F_\alpha = (1+\alpha^2)\frac{Precision*Recall}{\alpha^2*Precision+recall}$
- When α=1 → $F_1 = 2\frac{Precision*Recall}{Precision+recall}$

התפלגות נורמלית



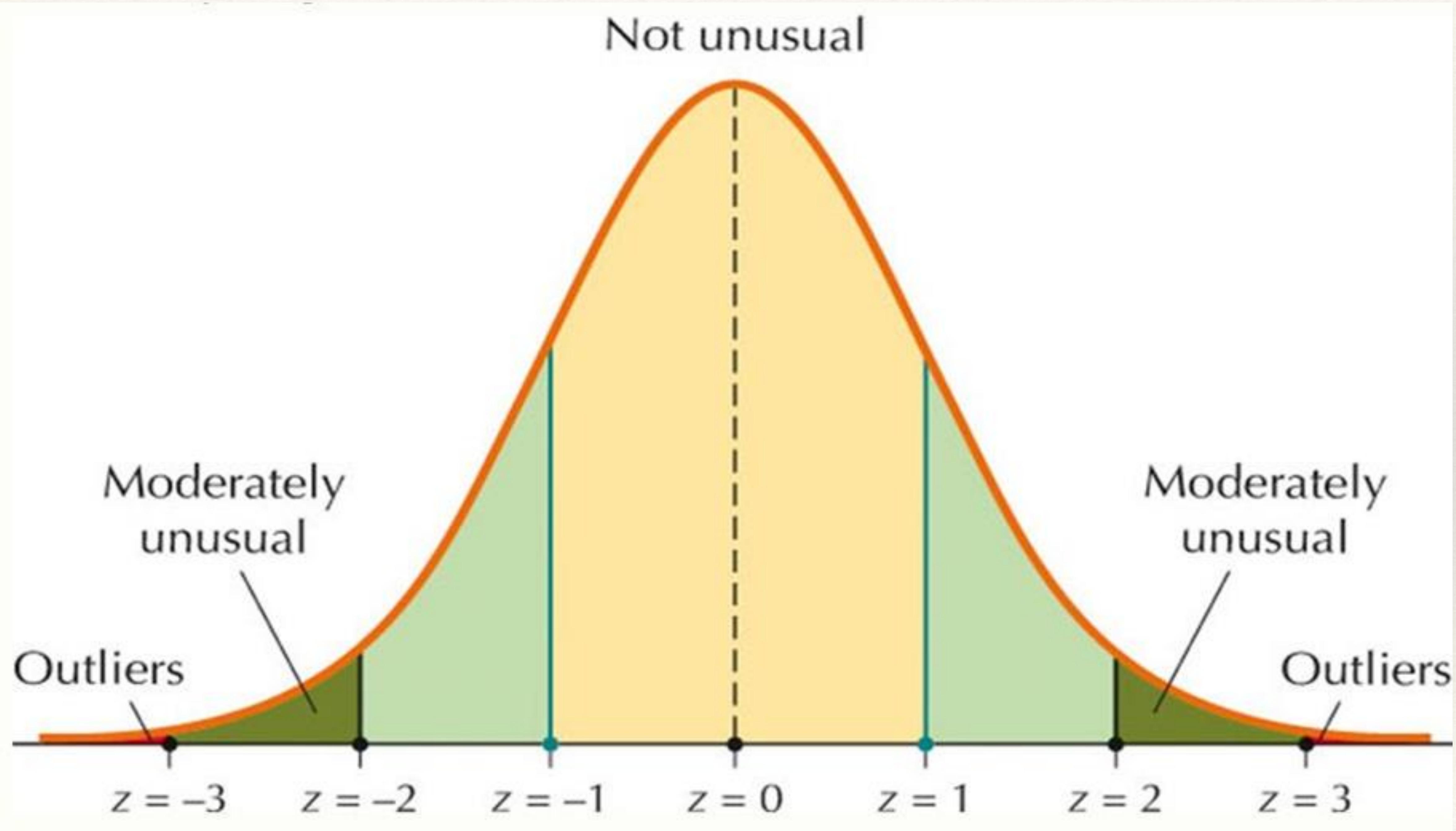
התפלגות נורמלית: נקראת גם גאוסיאן (Gaussian) או עקומת פעמון.

- פונקציית צפיפות סמטרית.

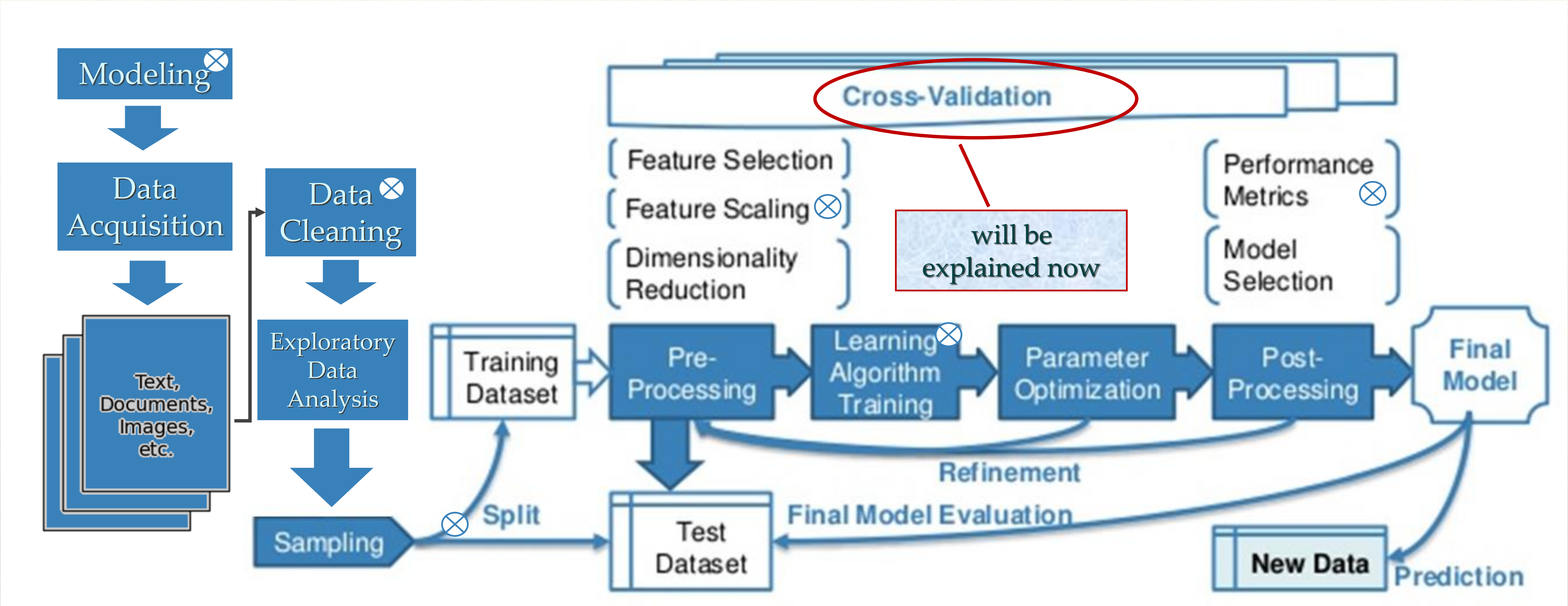
התפלגות z: תת קבוצה של התפלגות נורמלית בו התוחלת/הממוצע=0 וסטיית התקן=1.

- כל התפלגות נורמלית ניתן להפוך להתפלגות z

Outlier detection



A typical classification flow - diving in



Data Exploration

Data Cleaning

→ Duplicates

→ Missing Data

→Remove

→Repair

Train-Test split

+ Validation-set

Scaling

→ Minmax norm.

→ t-dist. standardization

Learning Algos.

→ KNN

→Decision Trees

→Naïve Bayes

Evaluation

→Confusion matrix

→ Accuracy ,Error (rate)

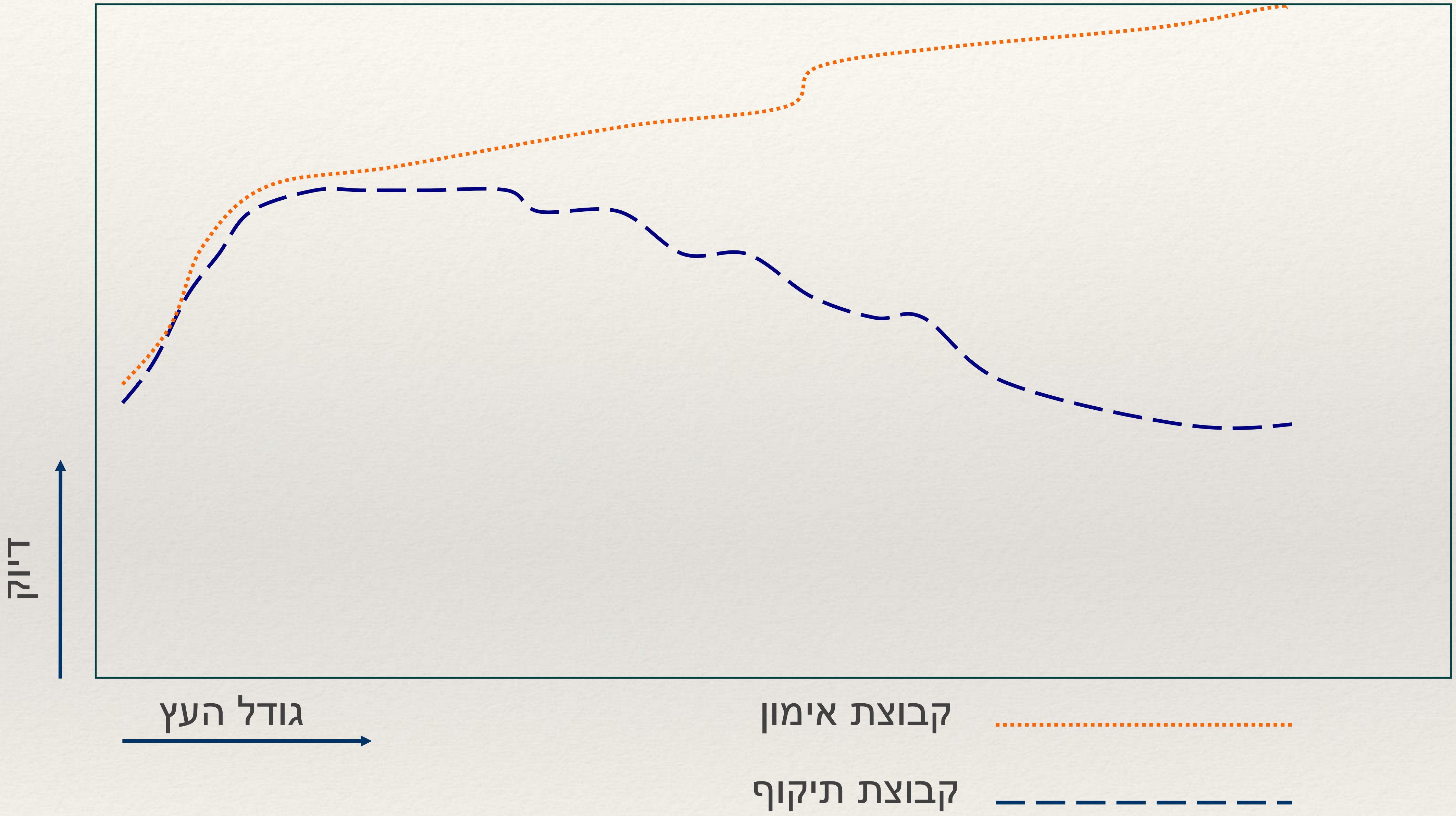
→ Precision, Recall

→ F1 (soon)

נושאים

- Overfitting
- Model selection
- Validation

Overfitting

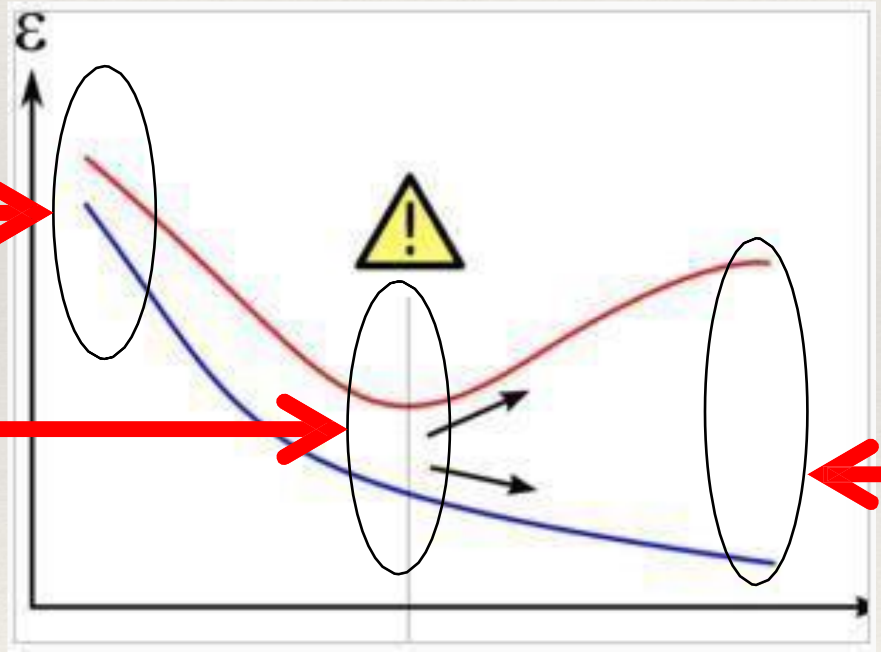


Overfitting

- *Given a hypothesis space H, h∈H overfits the training data if there exists some alternative hypothesis h' ∈ H such that h has smaller error than h' over the training examples, but h' has smaller error than h over the entire distribution of instances.*

Underfitting

Just right

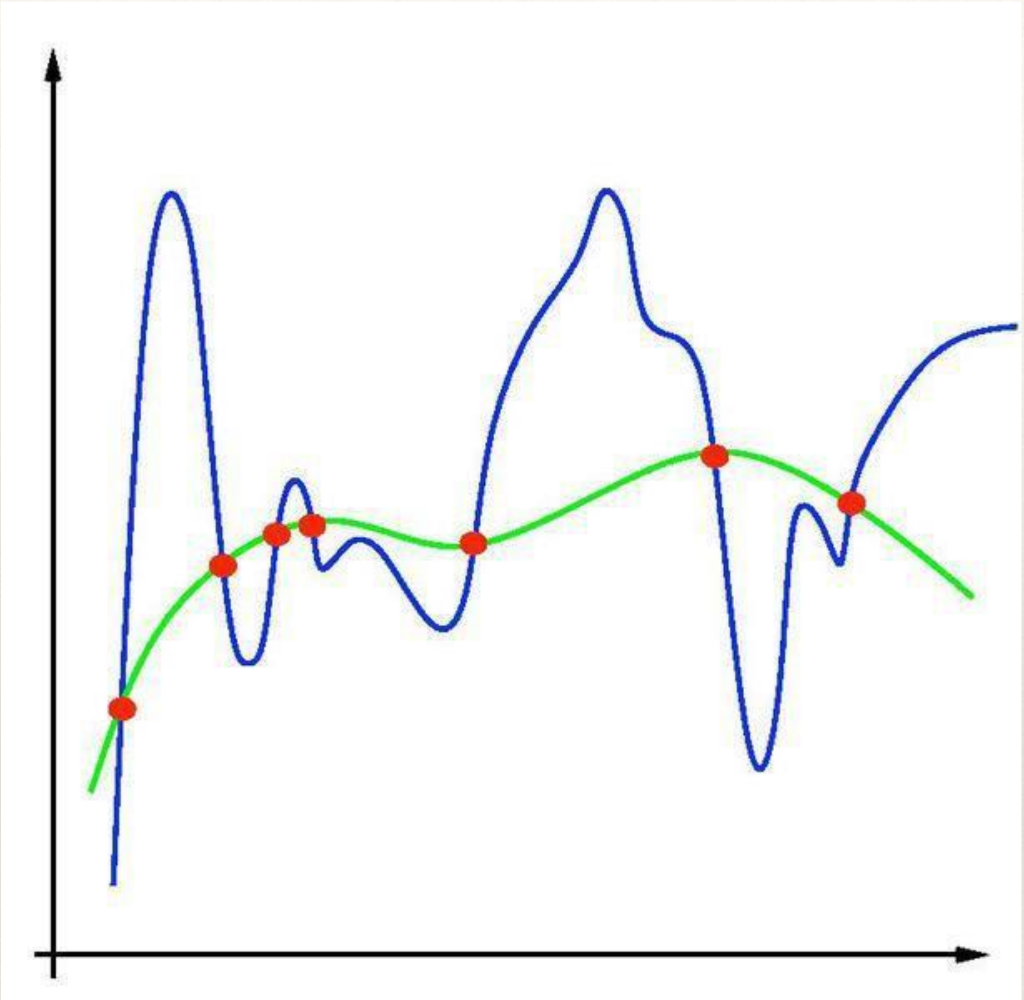


- Red: error on Test set (unseen examples)
- Blue: error on Training set

Overfitting

- Overfitting: Small error on training set, but large error on unseen examples.
- Underfitting: Larger error on training and test sets.

Overfitting

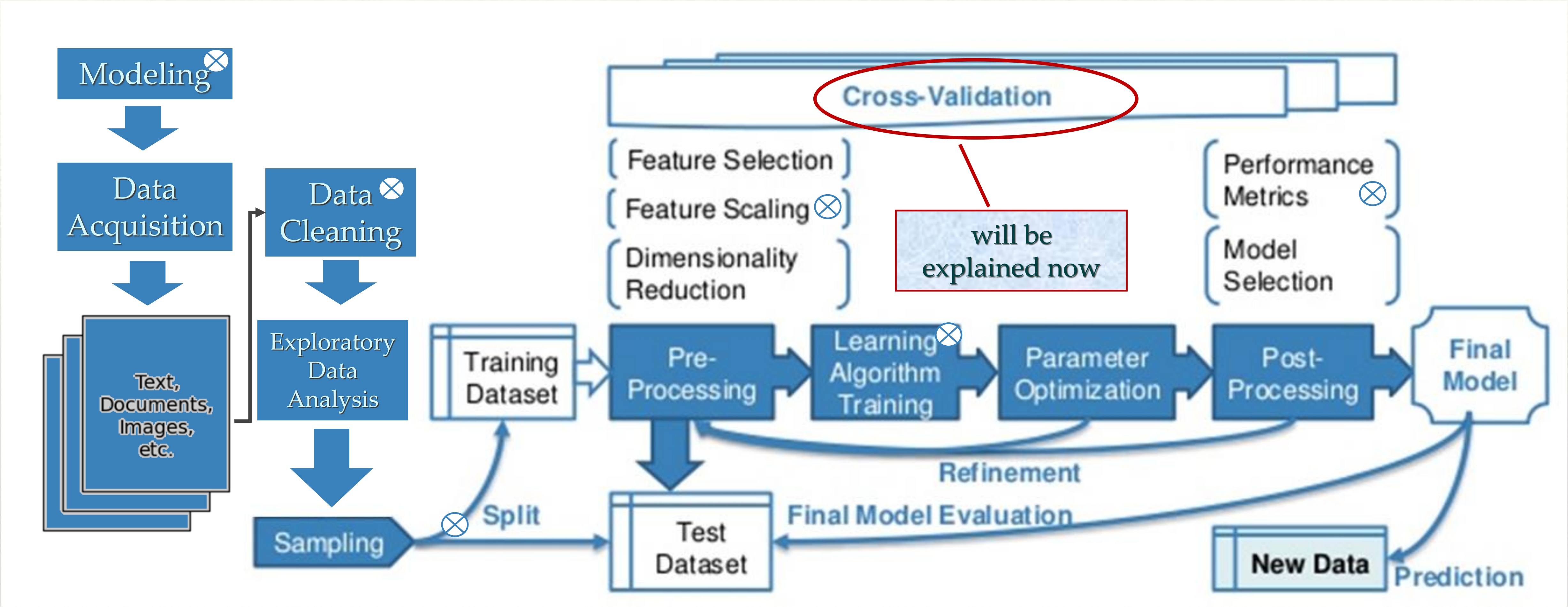


- Green: True target function
- Red: Training points
- Blue: What we have learned (overfitting)

(by Tomaso Poggio, [http://www.mit.edu/~9.520/spring12/slides/class02/class02.pdf)](http://www.mit.edu/~9.520/spring12/slides/class02/class02.pdf)

- The algorithm has learned perfectly the training examples, even the noise present in the examples and cannot generalise on unseen examples.

A typical classification flow



Data Cleaning

→ Duplicates

→ Missing Data

→Remove

→Repair

Train-Test split

+ Validation-set

Scaling

→ Minmax norm.

→ t-dist. standardization

Learning Algos.

→ KNN

→Decision Trees

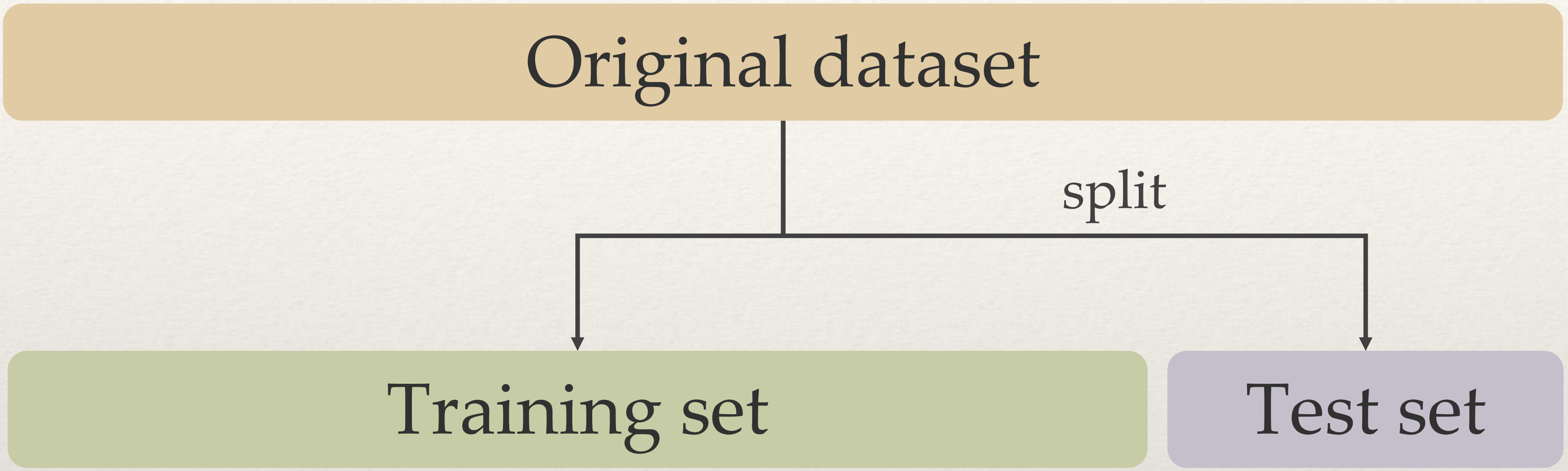
Evaluation

→ Error (rate)

→ Accuracy

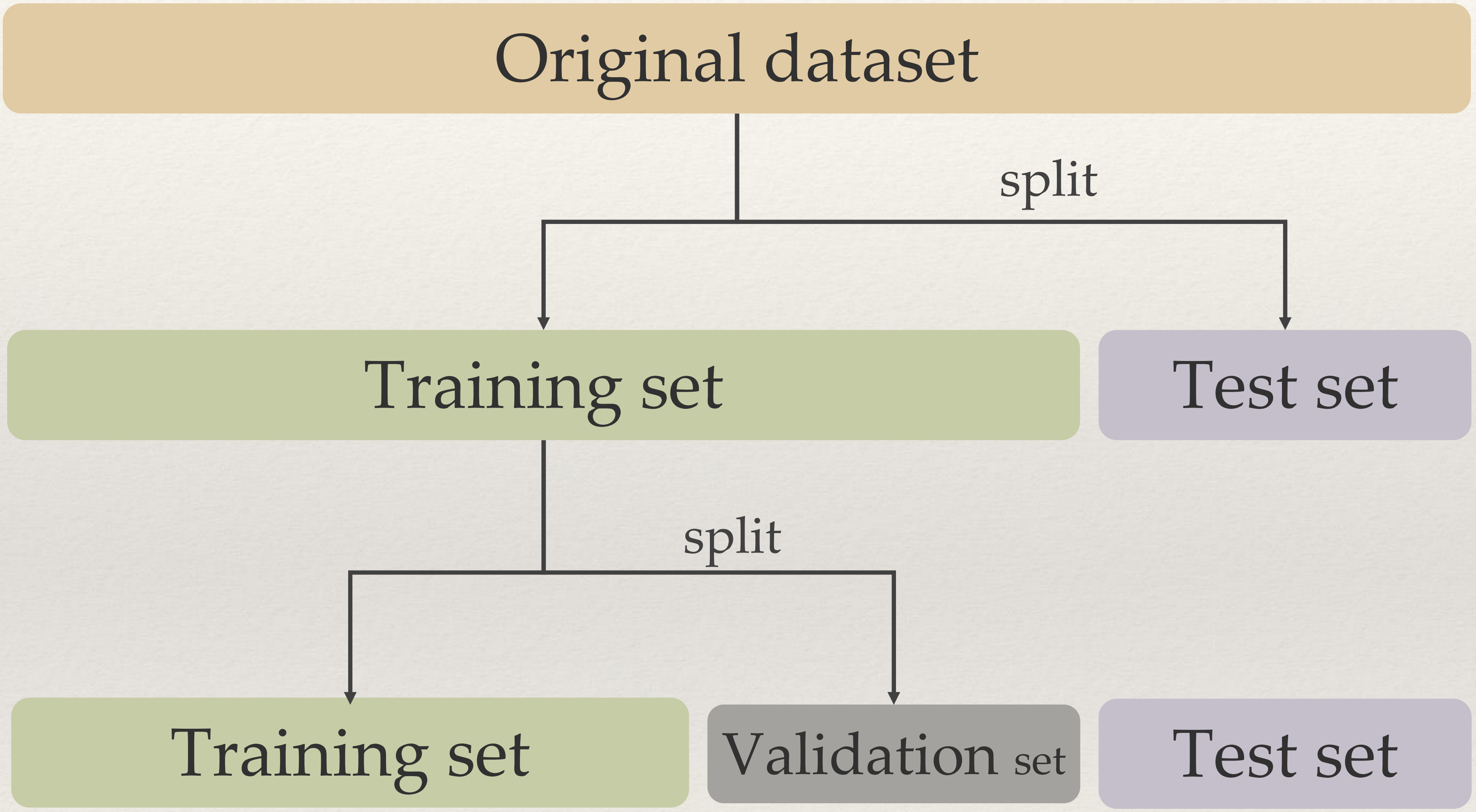
→Confusion matrix

dataset – train-set and test-set



Validation set

dataset – train-test-validation



שימושים ל validation set

- סיוע במניעת overfitting
- Model selection
- בחירת hyperparameters מיטביים
- תהליכים משלימים לתהליך האימון
  - Post pruning של עצי החלטה
- שיערוך המודל, בהיעדר test מקובל cross validation (בהמשך ...)

ועוד ...

What is Model Selection?

Given a set of models M={M1,M2,...,MR}, choose the model that is expected to do the best on the test data. The set M may consist of:

- Instances of same model with different complexities or hyperparams. E.g.,
- K-Nearest Neighbors: Different choices of K
- Decision Trees: Different choices of the number of levels/leaves
- Architecture of a deep neural network (# of layers, nodes in each layer, activation function, etc)
- Naïve Bayes – smoothing methods

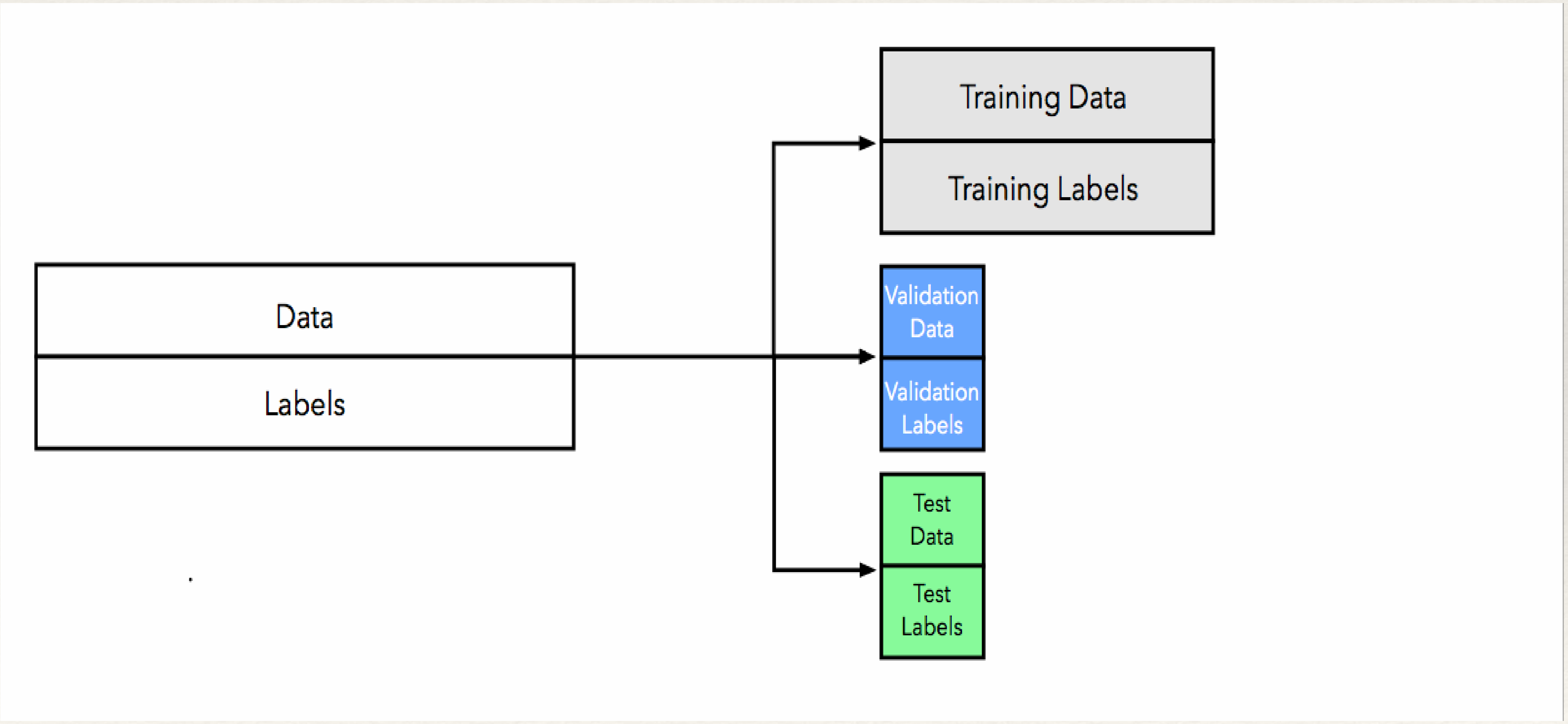
- Different types of learning models (e.g.KNN, DT, etc.)

Hyperparameters שלמדנו

- kNN - שיערוך k, שיטת מרחק, p בשיטת Minikowski distance, משקול מרחקי הנקודות
- עצי החלטה – עומק מקסימלי, מינימום דוגמאות בעלה, כמות המאפיינים לבדיקה כשמחפשים split מסוים
- Naive Bayes - פרמטרי החלקה, שיטות החלקה

Holdout Method

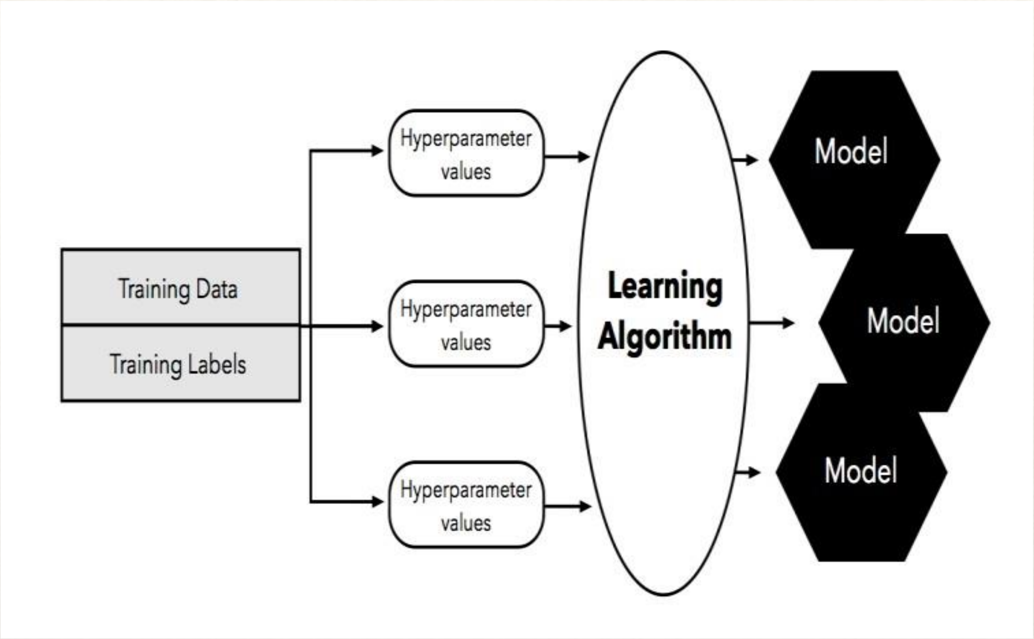
- Split your dataset into 3 disjoint sets: Training, Validation, Test
- If a lot of data are available then you can try 50:25:25 otherwise 60:20:20.

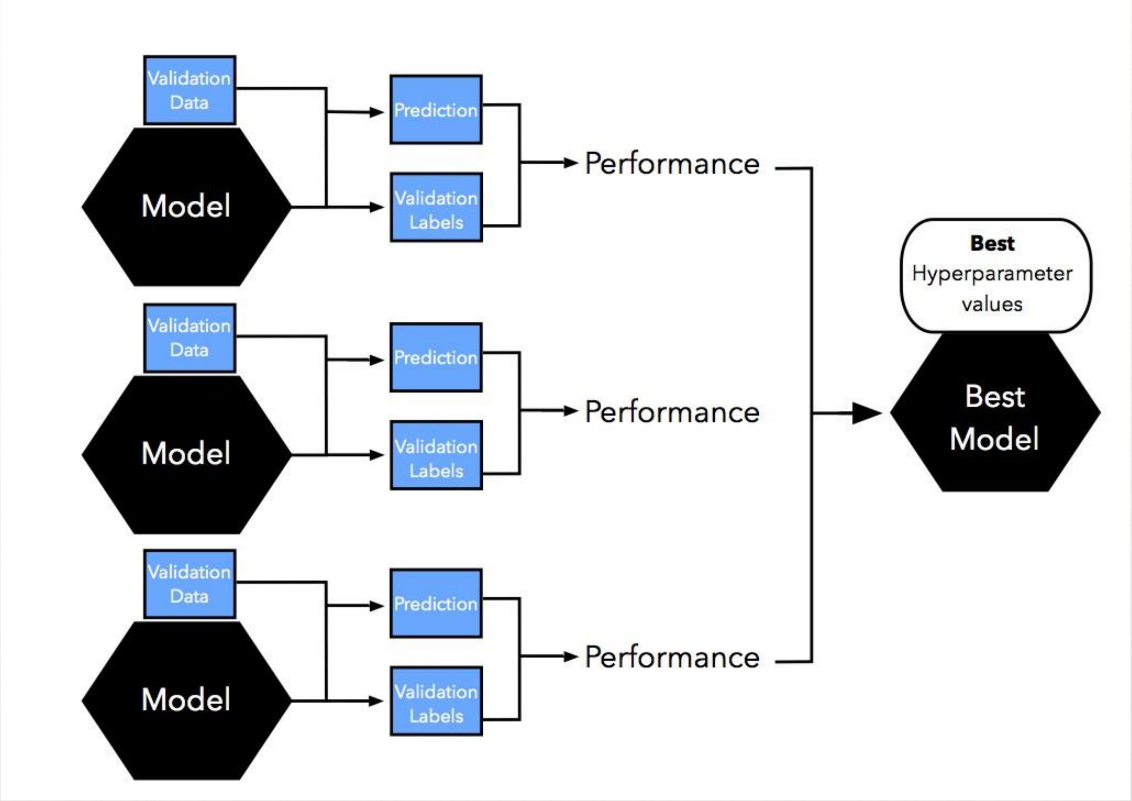


Holdout Method – Hyperparameter tuning

- Identify which parameters need to be optimized
  - e.g., number of hidden neurons, number of hidden layers etc
- Select a performance measure to evaluate the performance on the validation set
  - Accuracy, Precision, Recall etc
  - Appropriate measure depends on the application, if the test set is imbalanced etc

Holdout Method – Hyperparameter tuning

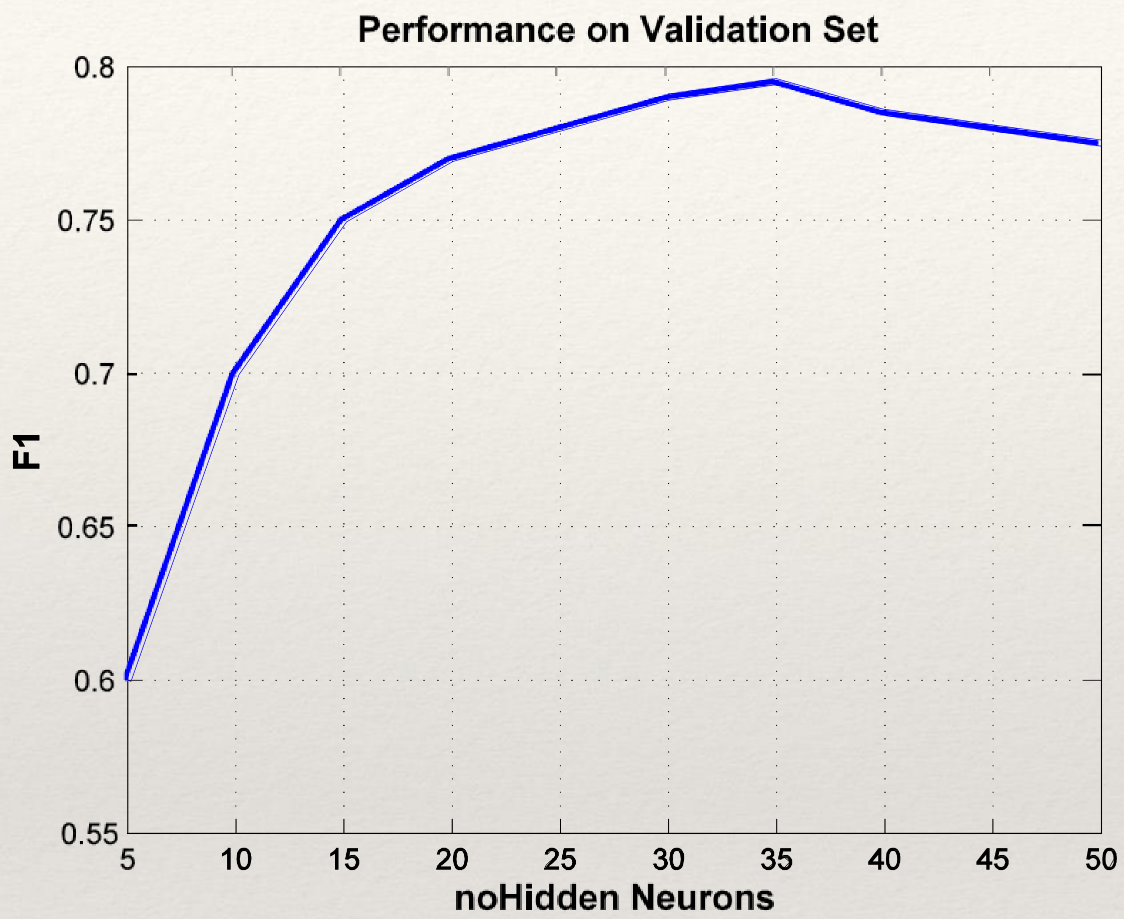




From: https://sebastianraschka.com/blog/2016/model-evaluation-selection-part3.html

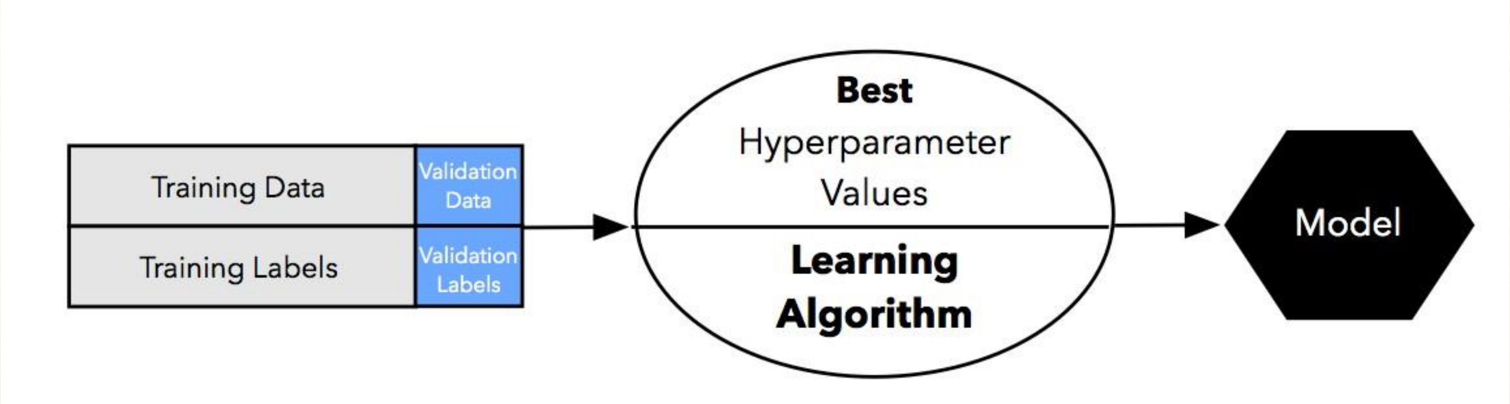
- Train your algorithm on the training set multiple times, each time using different values for the parameters you wish to optimise.
- For each trained classifier evaluate the performance on the validation set (using the performance measure you have selected).

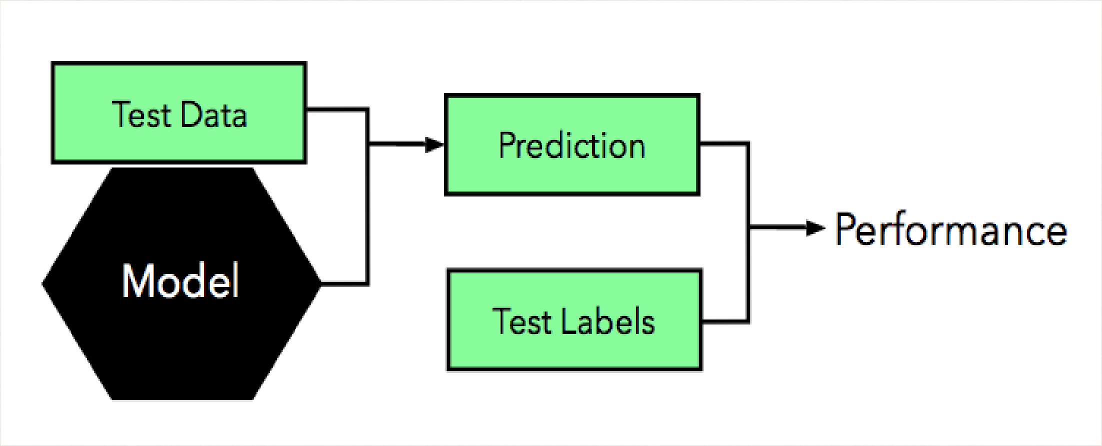
Holdout Method – Hyperparameter tuning



- Keep the classifier that leads to the maximum performance on the validation set (in this example the one trained with 35 hidden neurons).
- This is called parameter optimization/tuning, since you select the set of parameters that have produced the best classifier.

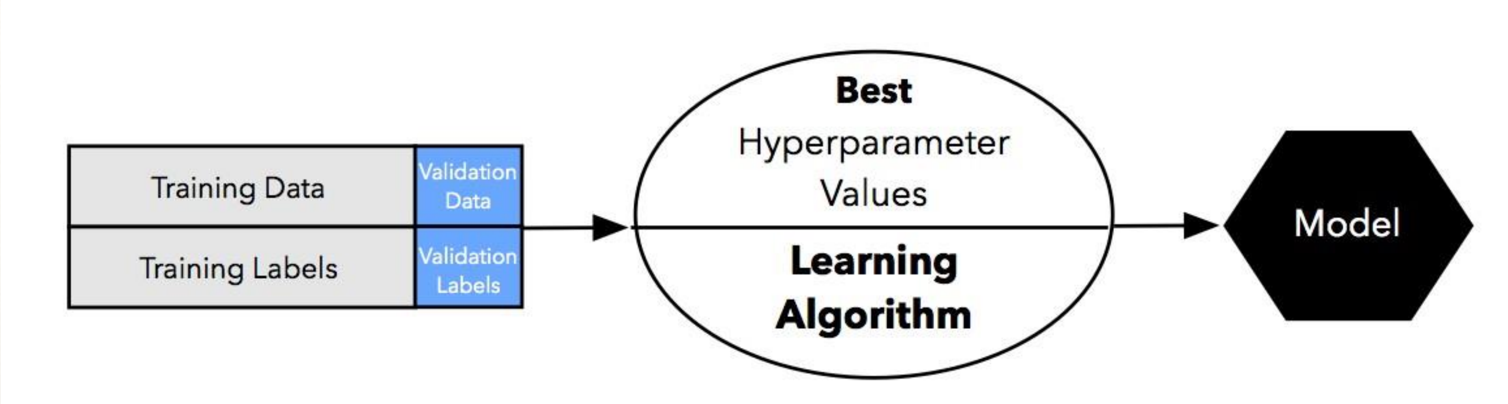
Holdout Method – Hyperparameter tuning – using parameters in the model

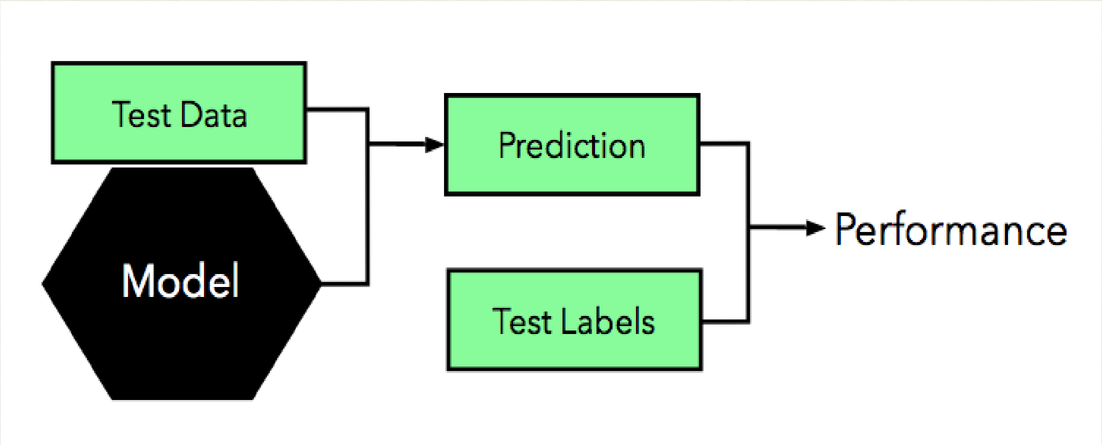




- You can either merge the training and validation sets and train a new classifier using the optimal set of parameters OR you can simply use the best classifier (trained only on the training set).
- Test the performance on the test set.

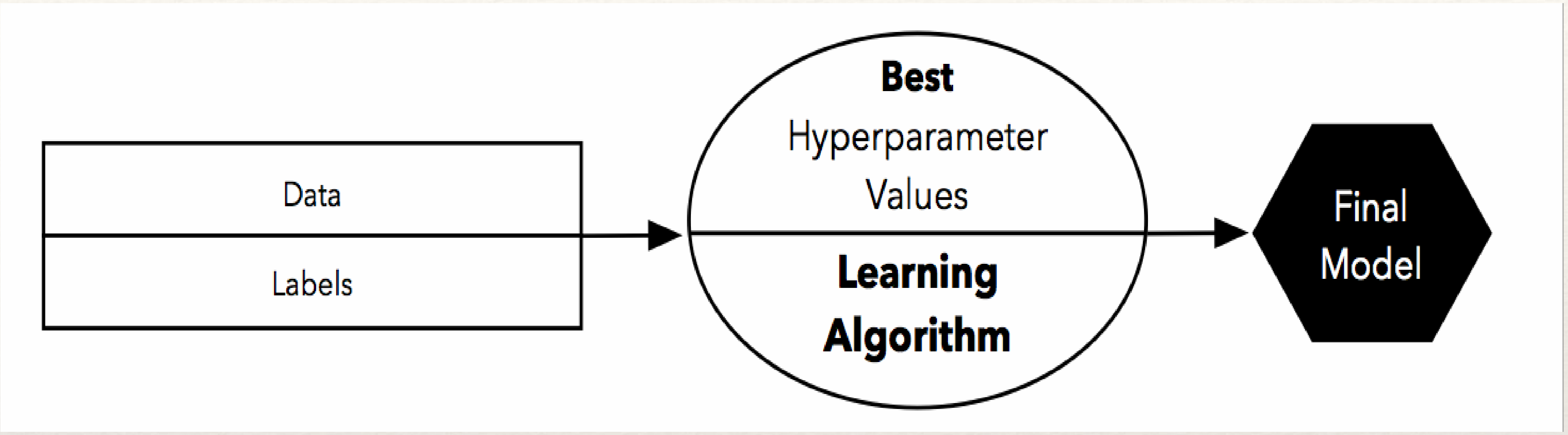
Holdout Method





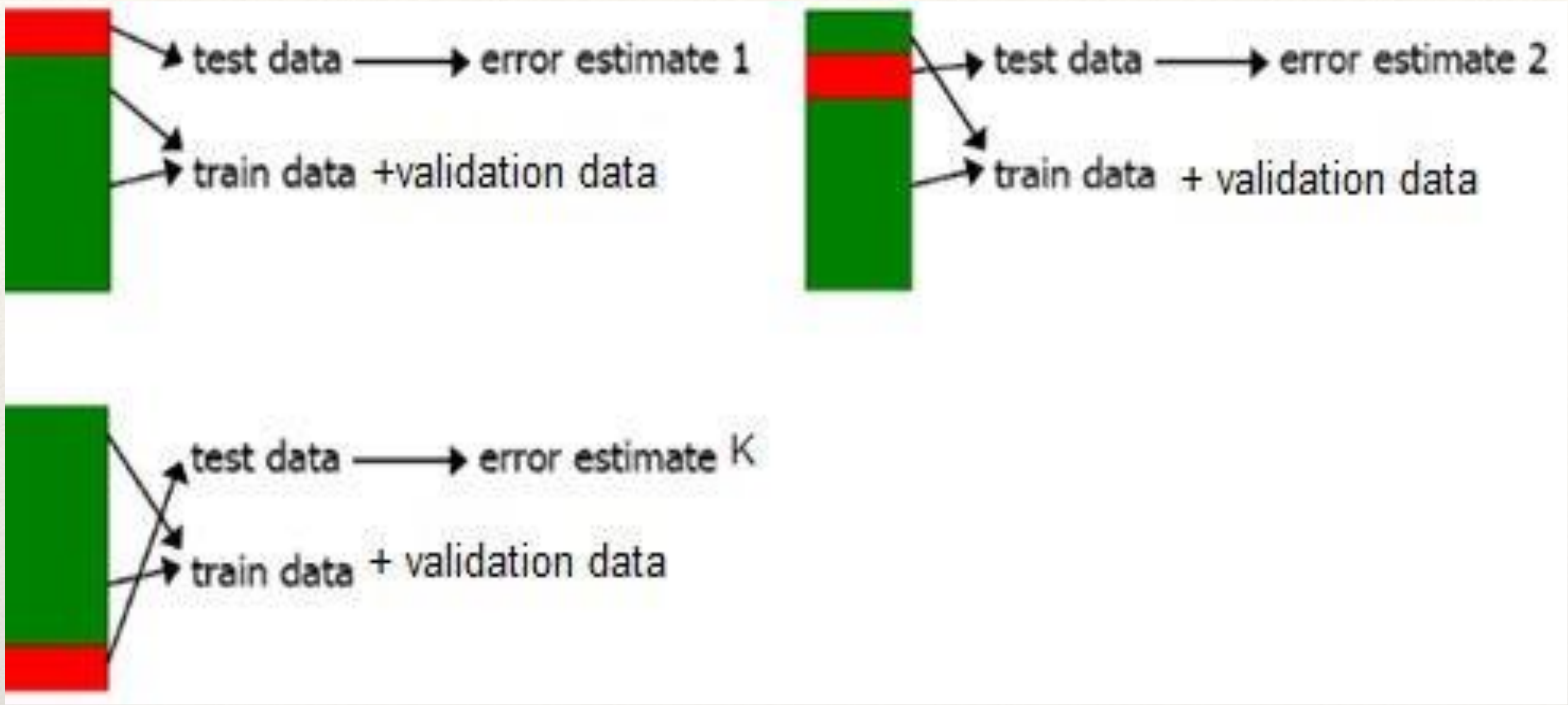
- The test set should **NOT** be used for training or validation. It is used **ONLY** in the end for estimating the performance on unknown examples, i.e. how well your trained classifier generalizes.
- You should assume that you do not know the labels of the test set and only after you have trained your classifier they are given to you.

Holdout Method



- We need a model which we will use for classifying new examples.
- Either use the one trained on the training set or on training + validation sets OR train a new model on the entire dataset using the optimal set of parameters.

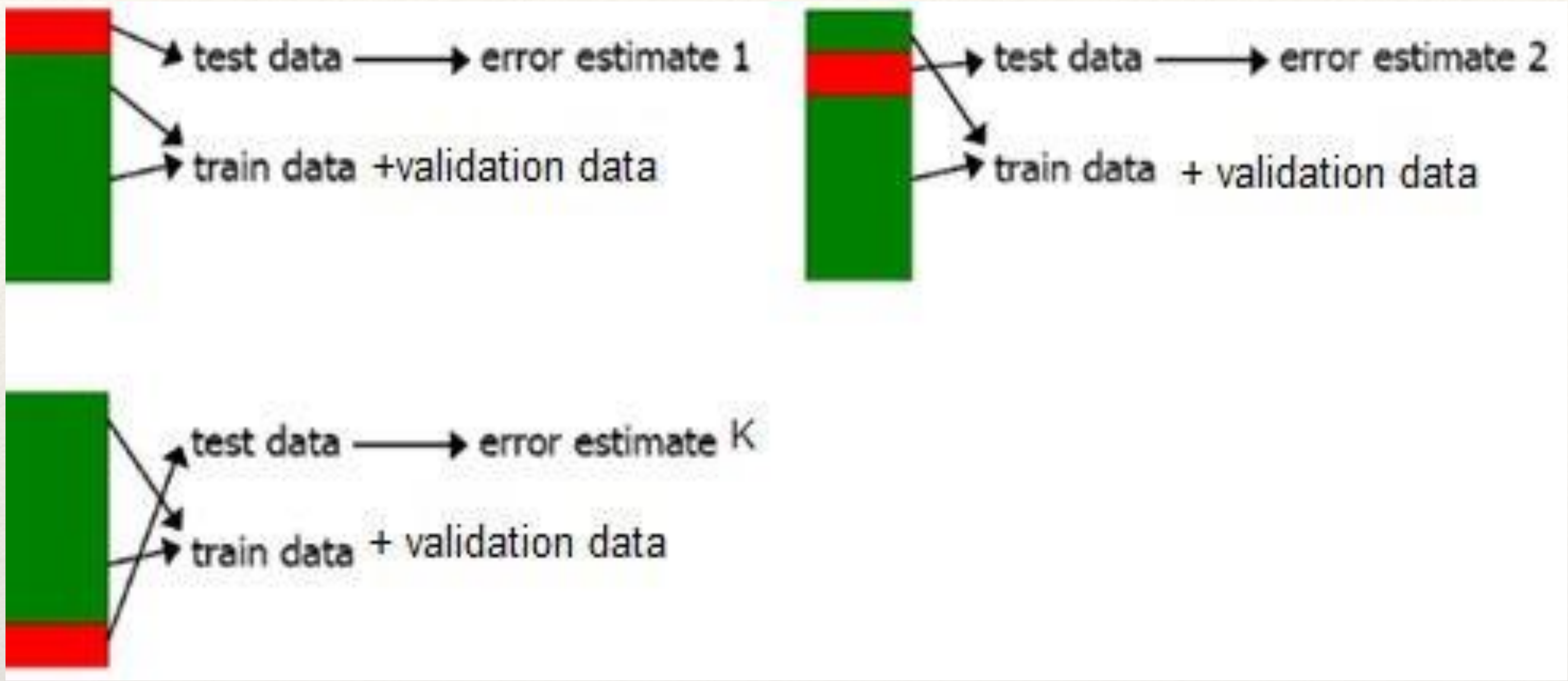
Cross Validation



- When we have a lot of examples then the division into training/validation/test datasets is sufficient.

- When we have a small sample size then a good alternative is cross validation.

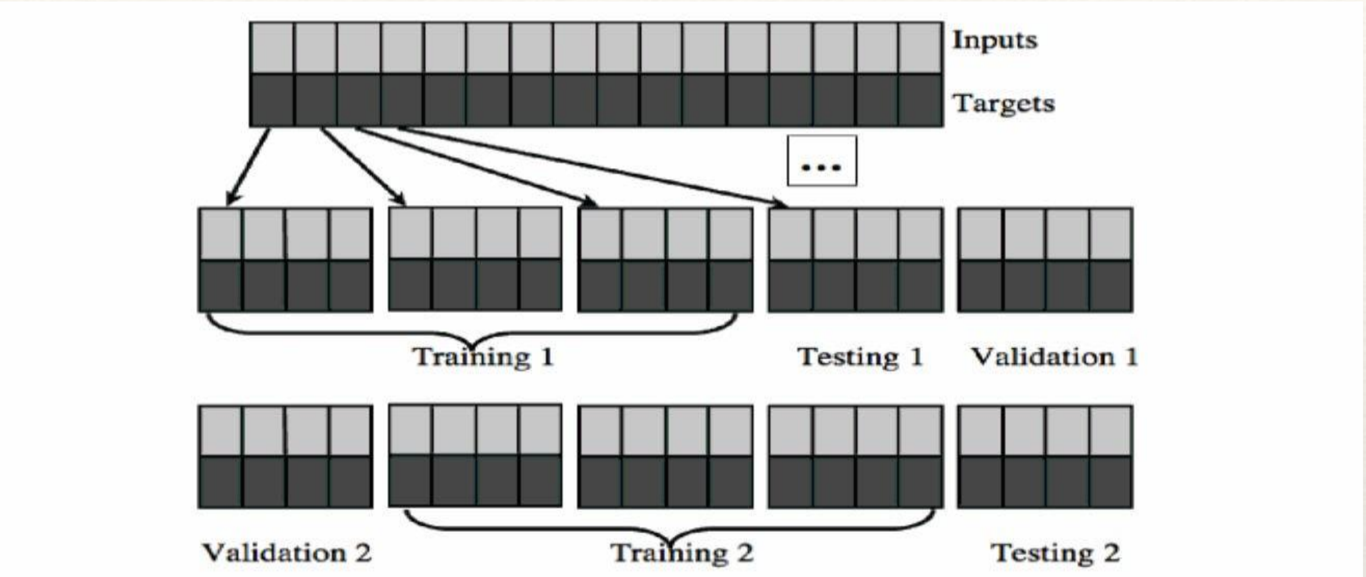
Cross Validation – Test Set Performance Estimation



- Divide dataset into *k* (usually 10) folds using *k*-1 for training + validation and one for testing
- Test data between different folds should never overlap!
- Training + Validation and test data in the same iteration should never overlap!
- In each iteration the error on the left-out test set is estimated
- Error estimate: average of the *k* errors

Cross Validation – Test Set Performance Estimation

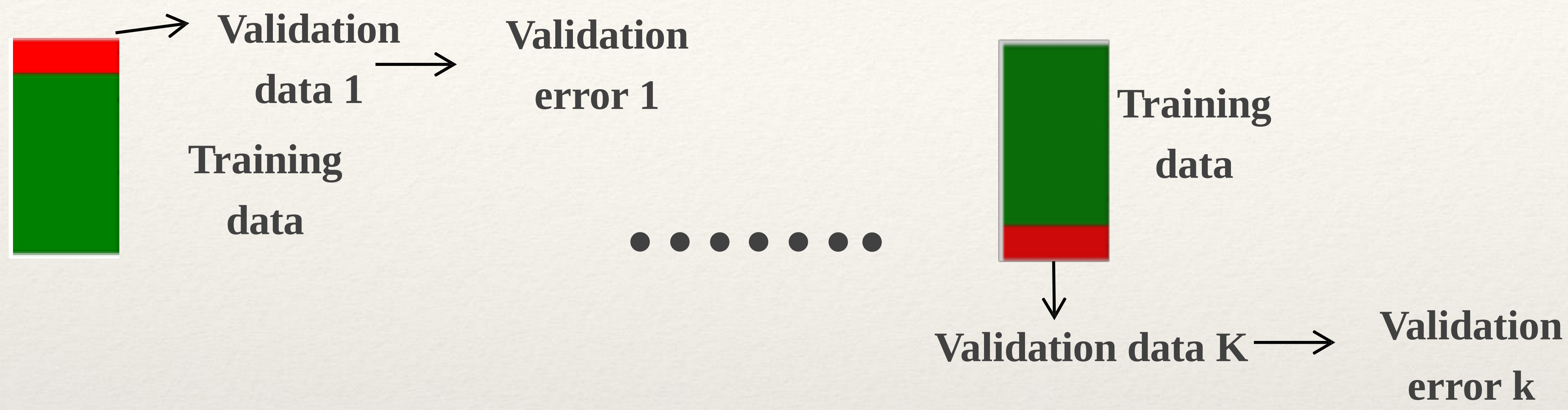
- The *k-1* folds should be divided into training and validation folds, e.g. *k-2* folds for training and 1 for validation.



S. Marsland, Machine learning: An algorithmic perspective

- Train on the training set, optimize parameters on the validation set and test on the test set.
- We can only estimate the test set performance. In other word we evaluate how our implementation (and the way we optimize the parameters) generalizes on unknown test sets.
- We know nothing about the optimal set of parameters. We find a different set of optimal parameters in each fold.

Cross Validation – Parameter Estimation



- We can use cross validation to estimate the optimal set of parameters
- *k-1*folds for training, 1-fold left out for validation (using the entire dataset)
- For each parameter set run the *k* fold cross-validation
- Select the parameters that result in the best average performance over all *k* left out folds

Cross validation – hyperparameter tuning with Grid Search

- **Grid Search** is a method for adjusting the hyperparameters.
- With Grid Search, we try all possible combinations of the parameters of interest and find the best ones.
- Practically – we try best practice values and choose the best combination

Parameter Optimization – Performance Estimation - Summary

- **CASE 1:** A lot of data are available (Holdout Method)
  1) Tune parameters on validation set
  2) Estimate generalization performance using the test set
  3) Train on entire dataset using optimal set of parameters

- **CASE 2:** Data are limited (Cross validation)
  1) Run cross validation to estimate the test set performance
     - Training, validation, test folds
     - Optimize parameters in each iteration
  2) Run cross validation to estimate optimal parameters
     - Training, Validation folds only (also called Grid Search)
  3) Train on entire dataset using optimal set of parameters