CEE 1323/2323
PRACTICAL DATA SCIENCE AND MACHINE LEARNING

Prediction, Errors, Cross Validation, Data Preprocessing, Model Validation

Amir Alavi
University of Pittsburgh

Before we get started…

- Last Time:
  - Principles of data science and machine learning
- Today:
  - Prediction, Errors, Cross Validation, Data Preprocessing, Model Validation

Machine Learning

Machine learning: Design of computer programs (agents) capable of learning from past experience or adapting to changes in the environment

Learner (a computer program) processes data *D* representing past experiences and tries to build a model that either:
– Generates appropriate response to future data, or
– Describes in some meaningful way the data seen

Machine Learning

Types of learning problems

• Supervised learning
–Takes data that consists of pairs (**X, Y**)
–Learns mapping ***f:*** **X**(input) → **Y** (output, response)

• Unsupervised learning
–Takes data that consist of vectors X
• Learns relations **X** among vector components
• Groups/clusters data into the groups

•Reinforcement learning
–Learns mapping ***f:*** **X**(input) → **Y** (desired output)
– **X** is an input, **Y** is a response chosen by the user/system

Supervised learning

Data: D = {d_1, …, d_n} a set of *n* examples where d_i = < x_i, y_i >

x_i is input vector, and y_i is desired output

Objective: Learn mapping ***f:*** **X**(input) → **Y** (output, response)

$$\mathbf{Y} = f(\mathbf{X})$$

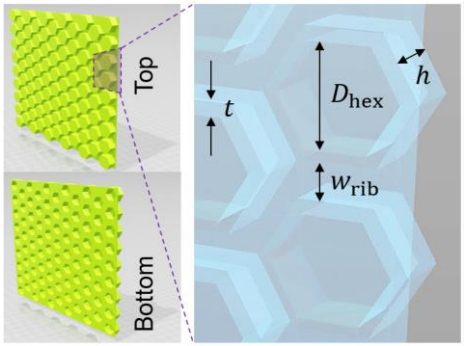
Two types of problems:

- **Regression:** X discrete or continuous, Y is continuous
- **Classification:** X discrete or continuous, Y is discrete

Regression

Hexagonal, single-layer microplates

Inputs						Output
Diameter Dhex [um]	**Rib width Wrib [um]**	**Thickness t [nm]**	**Height h [um]**	**Length L [mm]**	**Width W [mm]**	**Bending Stiffness [N/m]**
180	10	150	20	4	12	901.1
45	5	45	15	4	1	294.655



Classification

Input Parameters				Output
Z Score 1 M	**Z Score 1 M**	**Z Score 2 M**	**Z Score 2 M**	**Class**
-40.37605655	33.64966629	-5.88841007	5.88307946	1
1.28377939	1.28210237	0.18722530	-0.18753898	2

Damage State 1: Intact (a= 0 mm) ===> Class 1

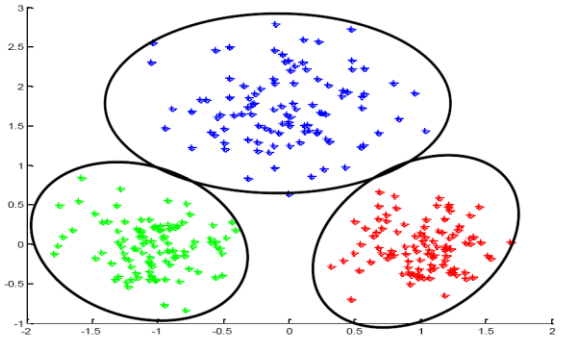
Damage State 2: a = 10 mm ===> Class 2

Unsupervised learning

Data: D = {d_1, …, d_n} a set of *n* examples where $d_i = x_i$
x_i is a vector of values

No target value (output) Y!

Objective: learn relations between samples, components of samples



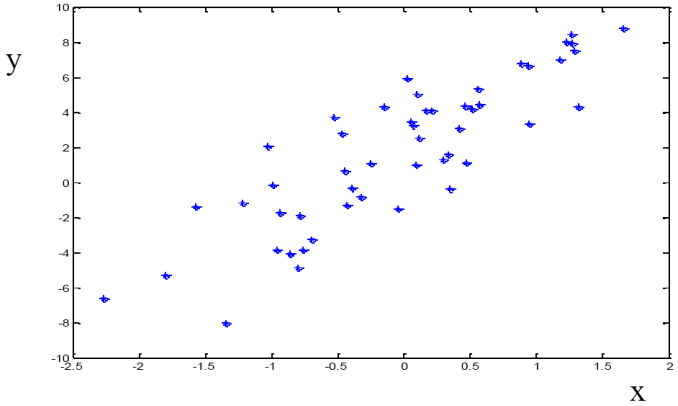
Types of problems:

• Clustering
– Group together “similar” examples

• Dimension reduction
– Reducing the dimension of the data by condensing variables together

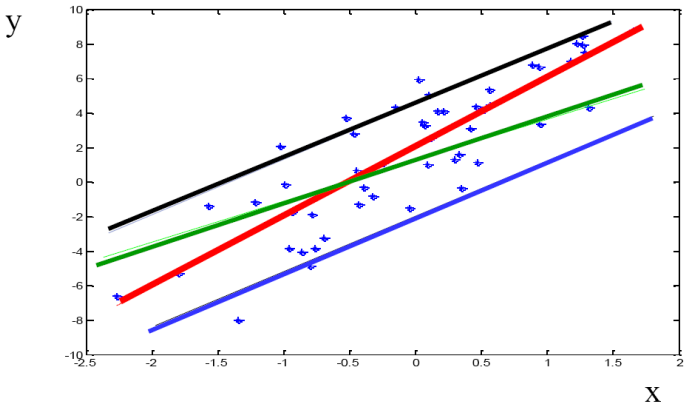
Learning

We get the data D. Now what should we do?!



Problem: many possible functions exist for representing the mapping between **X** and **Y.** Which one to choose?

We want the **best set** of model parameters to reduce the misfit between the model **M** and observed data ***D***



Basic Cycle of a Learning System

1. **Data**

2. Model development and selection
– Select a model or a set of models (with parameters)

3. Choose the objective function
– Reduce error
– Increase accuracy

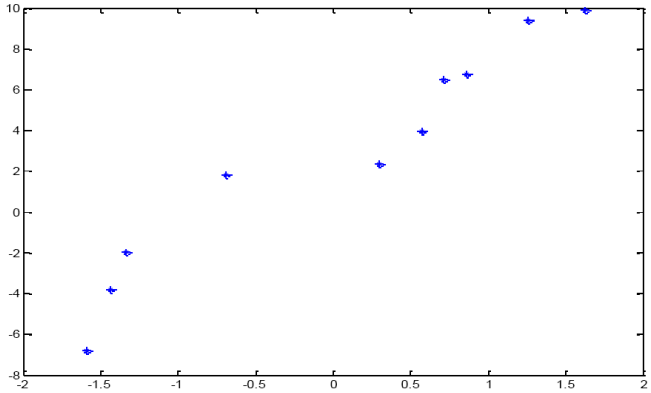
4. Learning
– Find the set of parameters optimizing the error function
–The model and parameters with the smallest error

5. Evaluation (testing/validation)
– Evaluate on unseen data

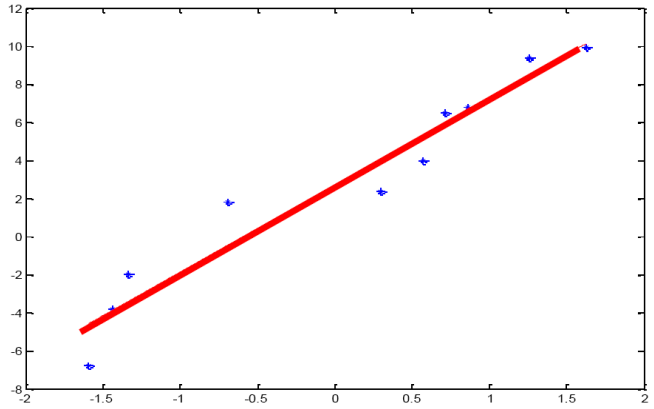
6. Application
– Apply the learned model to new data

Overfitting

Assume we have a set of 10 points and we consider polynomial functions as our possible models

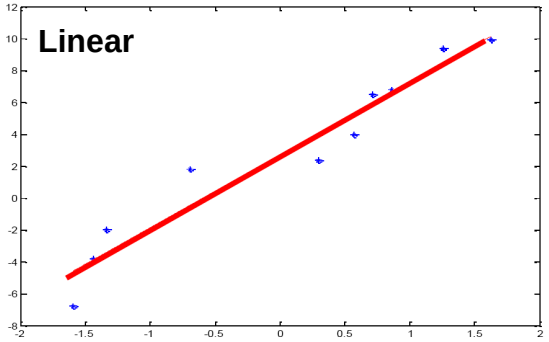


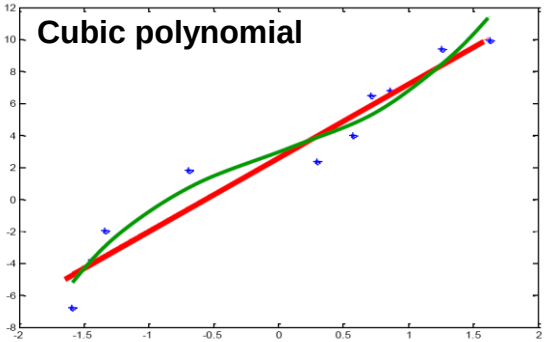
- Fitting a linear function with the square error



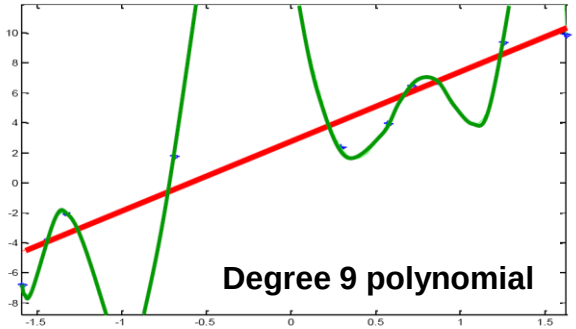
Overfitting

Which model would give us a smaller error for the least squares fit?





Is it always good to minimize the **training** error?



Overfitting

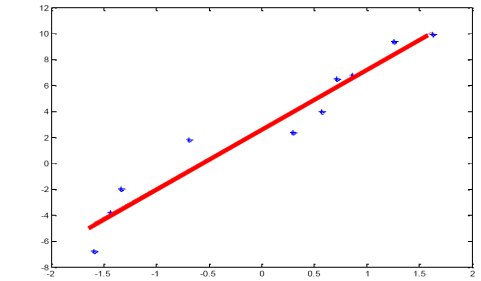
• **More important:** How do we perform on the **unseen data**?

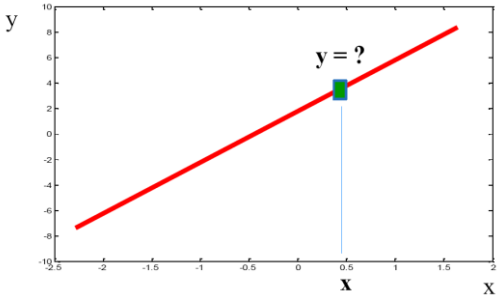
An **overfitted model** ➔ Learns the detail and noise in the training/calibration data to the extent that it negatively impacts the performance of the model on new data.

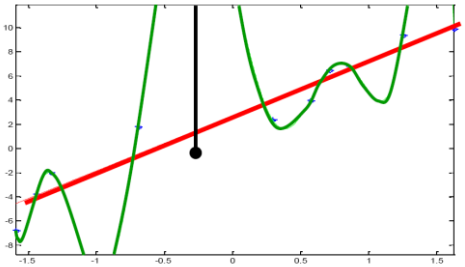
An **overfitted model** ➔ Contains more parameters than can be justified by the data.

Overfitting: Situation when the training error is low and the generalization/testing error is high.

Generalization: How well does a learned model generalize from the data it was trained on to a new test set.







Underfitting: Model is too “simple” to represent all the relevant class characteristics. High training error and high test error!

How to evaluate the learner's performance?

Causes of the overfitting:
- Model with a large number of parameters
- Small data size (as compared to the complexity of the model)

- **Generalization error** is the **true error** for the population of examples we would like to optimize

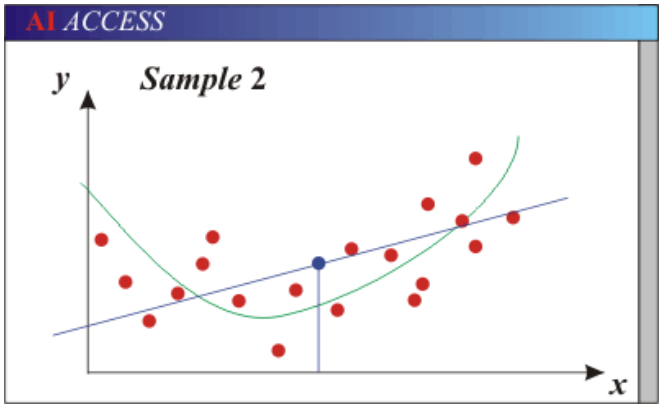
- **Optimizing the training error can lead to the overfit, i.e.** training error may not reflect properly the generalization error

Practical solution to avoid overfitting: Use a separate data set with *m* data samples to test the model

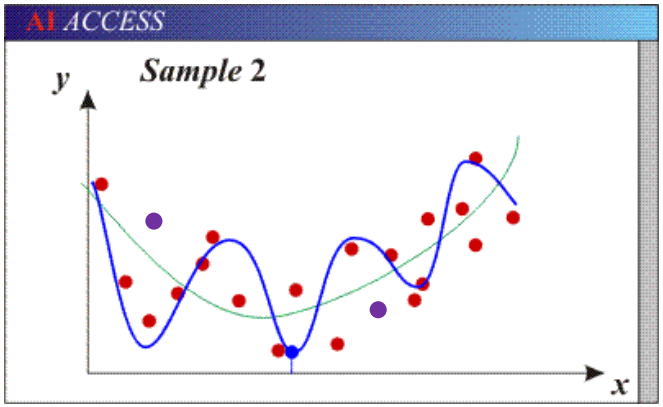
CEE 1323/2323: Practical DS & ML

Red dots = training data Green curve = true underlying model Blue curve = our predicted model/fit

Purple dots = possible test points



- Models with too few parameters are inaccurate because of a large bias (not enough flexibility).



- Models with too many parameters are inaccurate because of a large variance (too much sensitivity to the sample).

Key: Complexity and flexibility, as well as suitable training data

Adapted from D. Hoiem

CEE 1323/2323: Practical DS & ML

Generalization

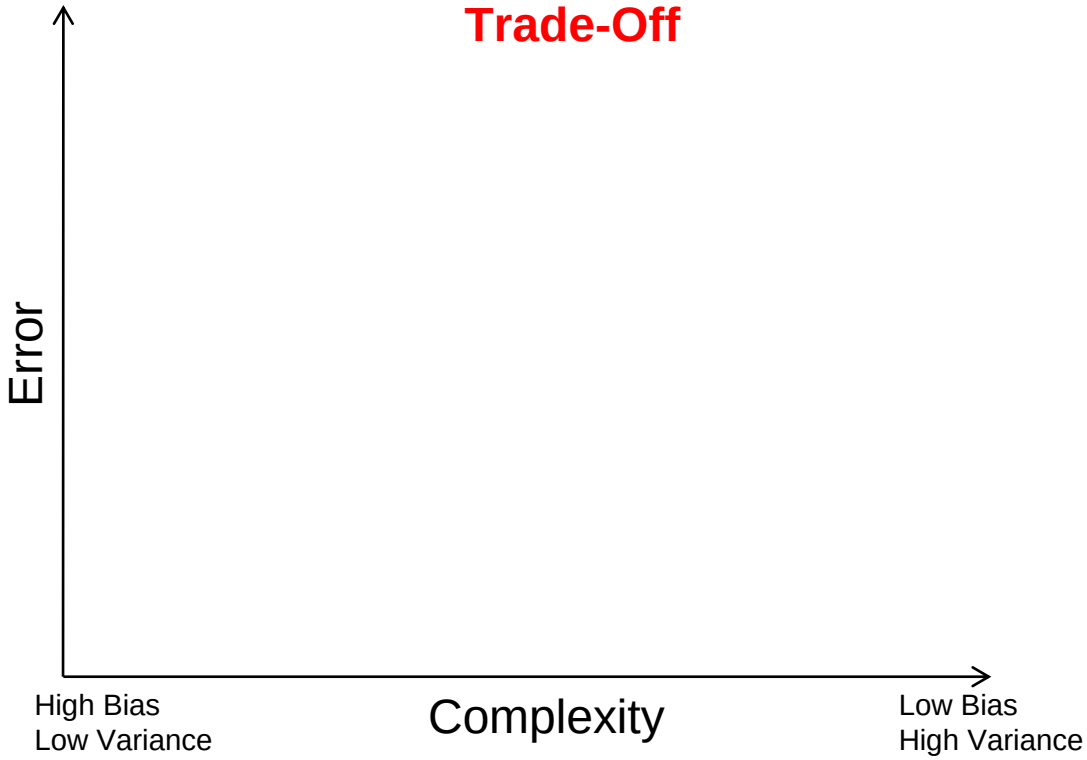
Components of expected loss (error):

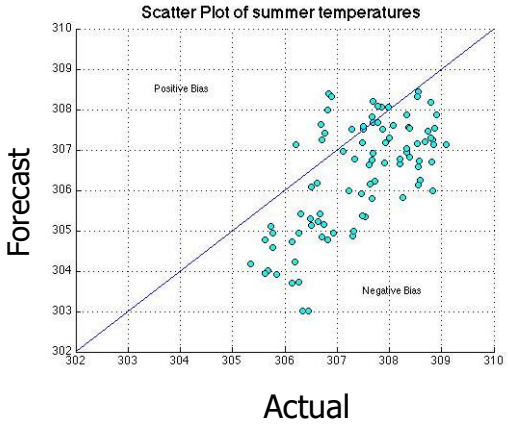
- **Noise in our observations**: unavoidable
- **Bias:** how much the average model's predictions over all training sets differs from the true target values
  - Error due to inaccurate assumptions/simplifications made by the model
- **Variance:** how much the estimate of the target function will alter if different training data were used

Training vs Test error

Underfitting **Overfitting**

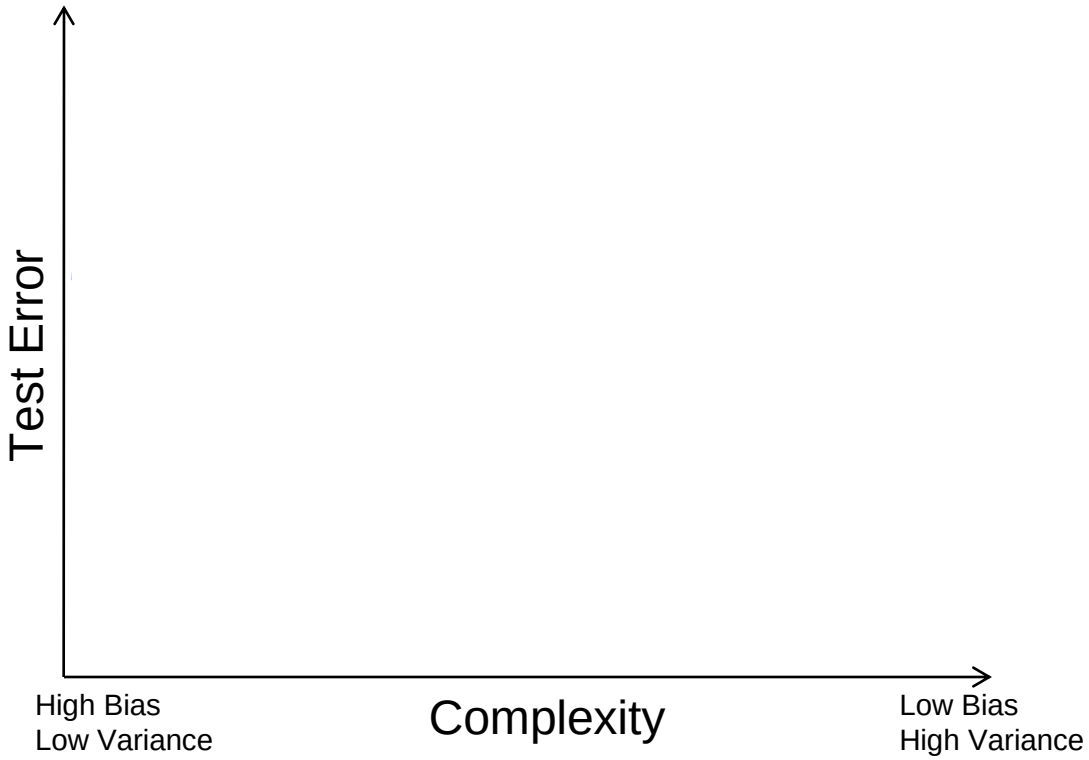
The Bias-Variance Trade-Off





Slide credit: D. Hoiem

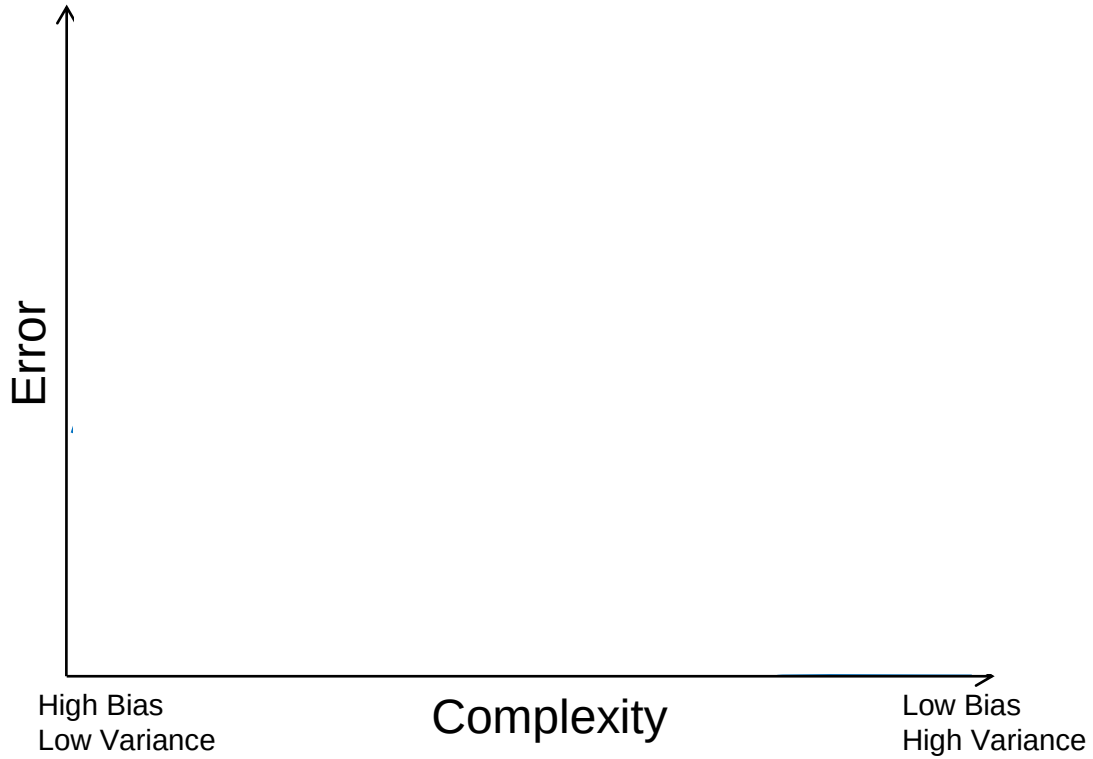
The Effect of Training Set Size



Slide credit: D. Hoiem

Choosing the trade-off between bias and variance

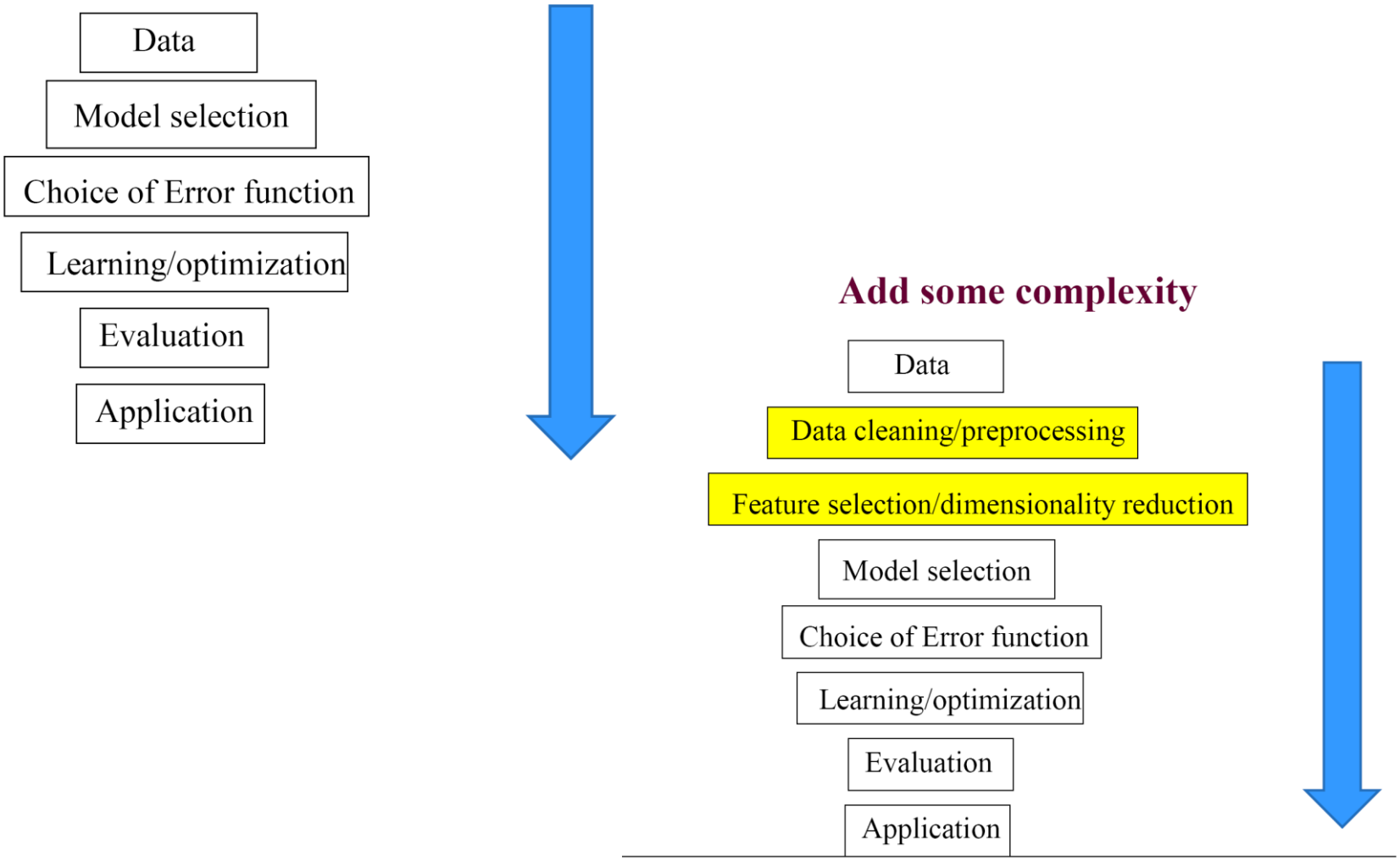
- Need validation set (separate from the test set)



- **Get more training data**
- **Choose a simpler classifier**

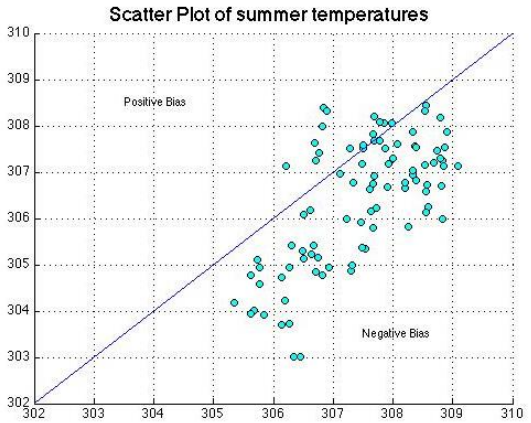
Slide credit: D. Hoiem

Steps Taken When Designing an ML System



Steps Taken When Designing an ML System

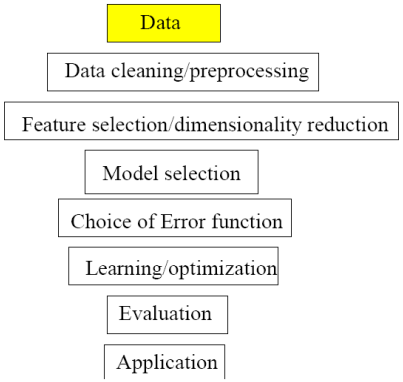
Data source and data biases



Data biases refer to systematic errors or distortions in a dataset that can result in inaccurate or unfair conclusions

- Understand the data source
- Understand the data your models will be applied to
- Watch out for data biases:
  - Make sure the data we make conclusions on are the same as data we used in the analysis
  - It is very easy to derive “unexpected” results when data used for analysis and learning are biased

- **Results (conclusions) derived for a biased dataset do general !!!**



Steps Taken When Designing an ML System

Data cleaning and preprocessing

Data you receive may not be perfect:

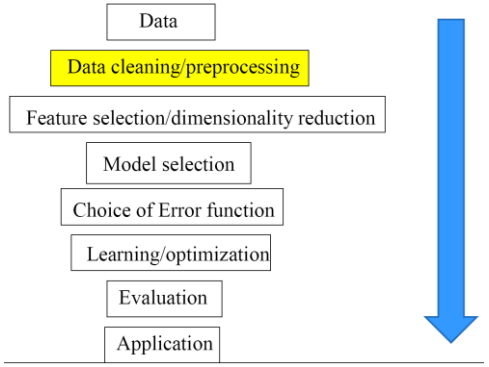
- Cleaning
- Preprocessing (conversions)

Cleaning:

– Get rid of errors, noise,
– Removal of redundancies

Preprocessing:

– Renaming
– Rescaling (normalization)
– Discretization
– Abstraction
– Aggregation
– …



Steps Taken When Designing an ML System

Data cleaning and preprocessing

Renaming (relabeling**)** categorical values to numbers
- dangerous in conjunction with some learning methods
- numbers will impose an order that is not warranted

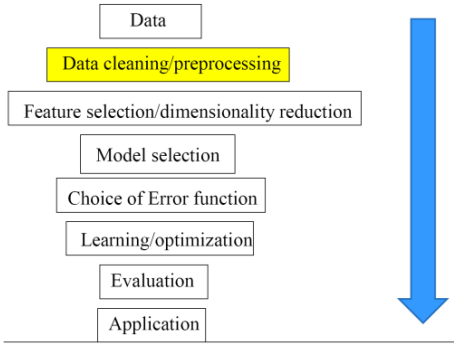
High → 2
Normal → 1 √
Low → 0

Red → 2
Blue → 1 ×
Green → 0

Problem: How to safely represent the different categories as numbers when no order exists?

Solution: Use indicator vector (or one-hot) representation.
- **Example: Red, Blue, Green colors**

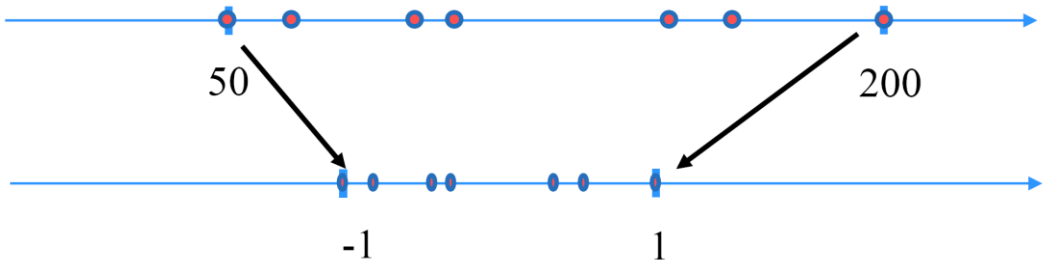
3 categories → use a vector of binary (0,1) values of size 3
Encoding: **Red:**(1,0,0); **Blue:**(0,1,0); and **Green:**(0,0,1)

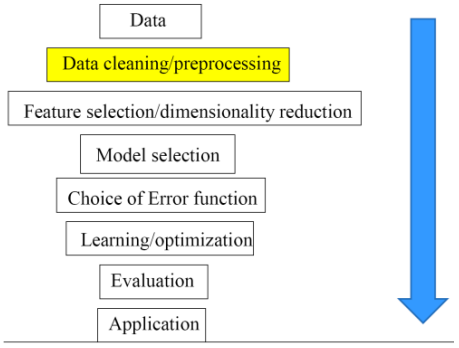


Steps Taken When Designing an ML System

Rescaling (normalization): continuous values are transformed to some range, typically [-1, 1] or [0,1].

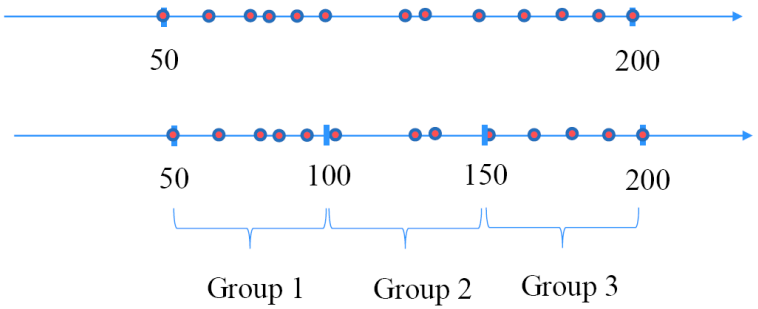
- Why normalization?

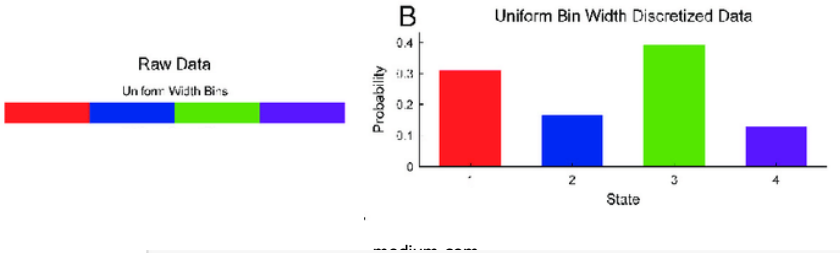




Steps Taken When Designing an ML System

• **Discretization (binning):** continuous values to a finite set of discrete values



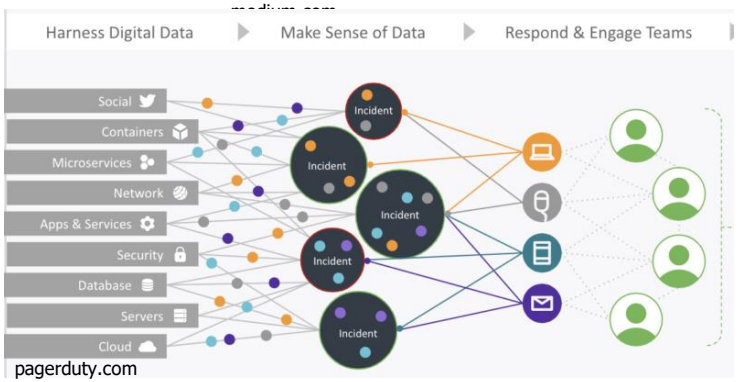


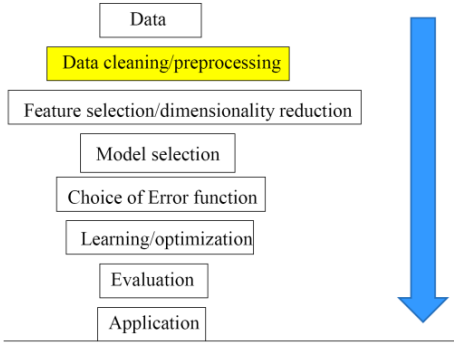
• **Abstraction:** merge together categorical values

• **Aggregation:** summary or aggregation operations, such minimum value, maximum value, average over a set of values etc.

• **New variables:**

– example: Water, Cement ➔ Water/Cement





Steps Taken When Designing an ML System

Feature selection/dimensionality reduction

The dimensionality of each example in the data can be huge

$$\mathbf{x} = (x_1, x_2, \ldots x_d) \qquad d \quad \text{is large}$$

$$\mathbf{x}' = (x_1', x_2', \ldots x_k') \qquad k < d$$

Problems:

– Too many parameters to learn (we many not have enough samples to get good estimates the model parameters) of the model

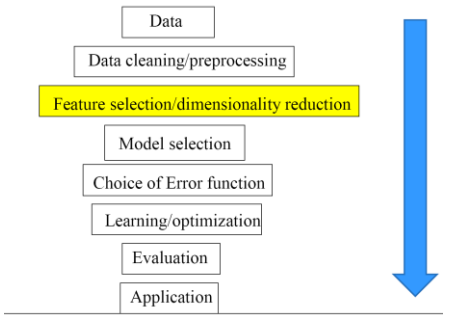
– Some entries are dependent

Objective: reduce the dimensionality of the data while preserving its most important properties

Two types of methods:

A. Pick a subset of inputs (feature selection)

B. Transform the space to a lower dimensional space



Steps Taken When Designing an ML System

Model selection

What is the right model to learn?

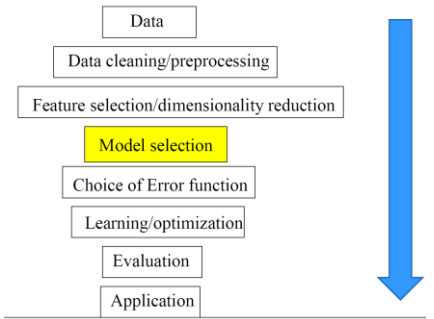
– A prior knowledge helps a lot, but still a lot of guessing
– Initial data analysis and visualization

We can make a good guess about the form of the distribution, shape of the function by looking at data

– Independences and correlations

Overfitting problem

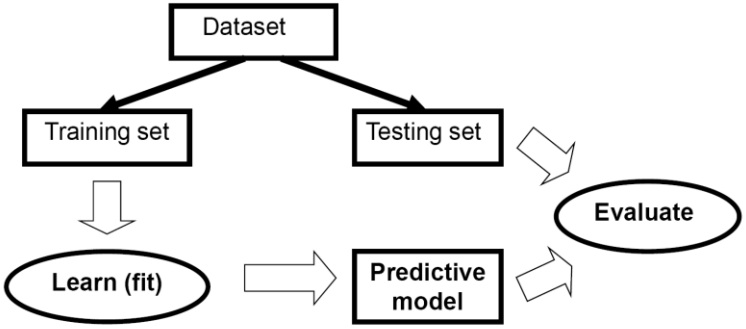
– Take into account the **bias and variance** of error estimates
– Simpler (more biased) model–parameters can be estimated more reliably (smaller variance of estimates)
– Complex model with many parameters –parameter estimates are less reliable (large variance of the estimate)



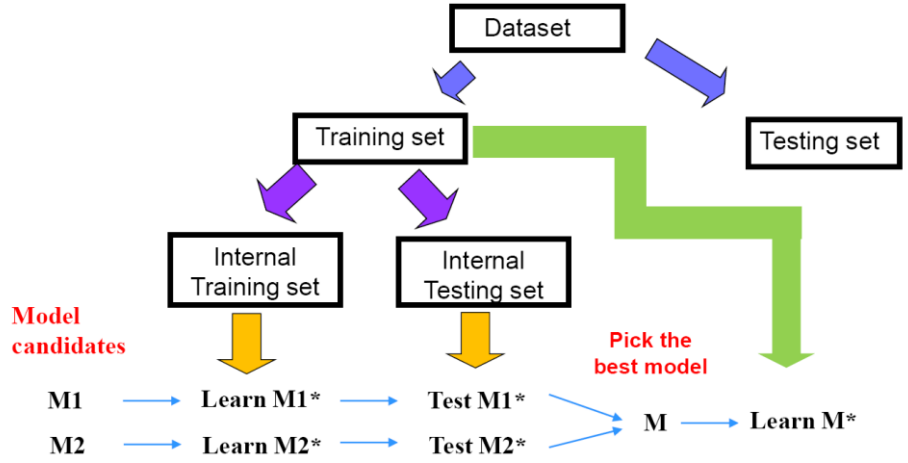
CEE 1323/2323: Practical DS & ML

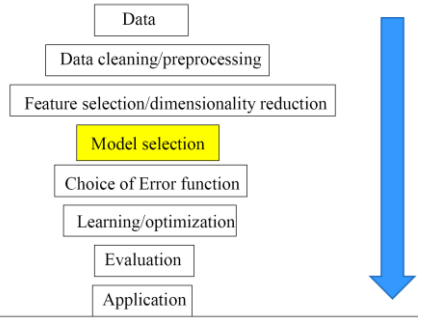
How to assess the generalization performance of ML models?!

- **Simple holdout method**
  - Divide the data into the disjoint training and test data



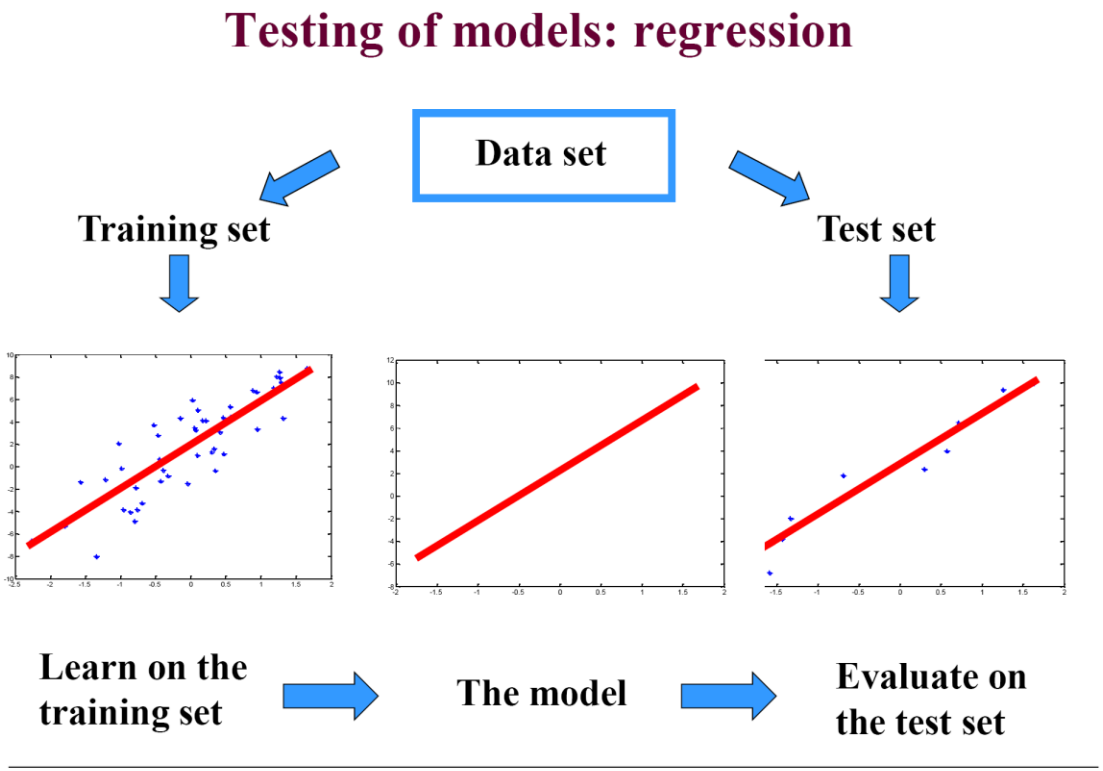
  - Typically 2/3 training and 1/3 testing

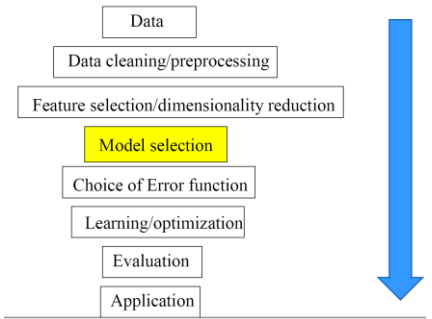




CEE 1323/2323: Practical DS & ML

How to assess the generalization performance of ML models?!

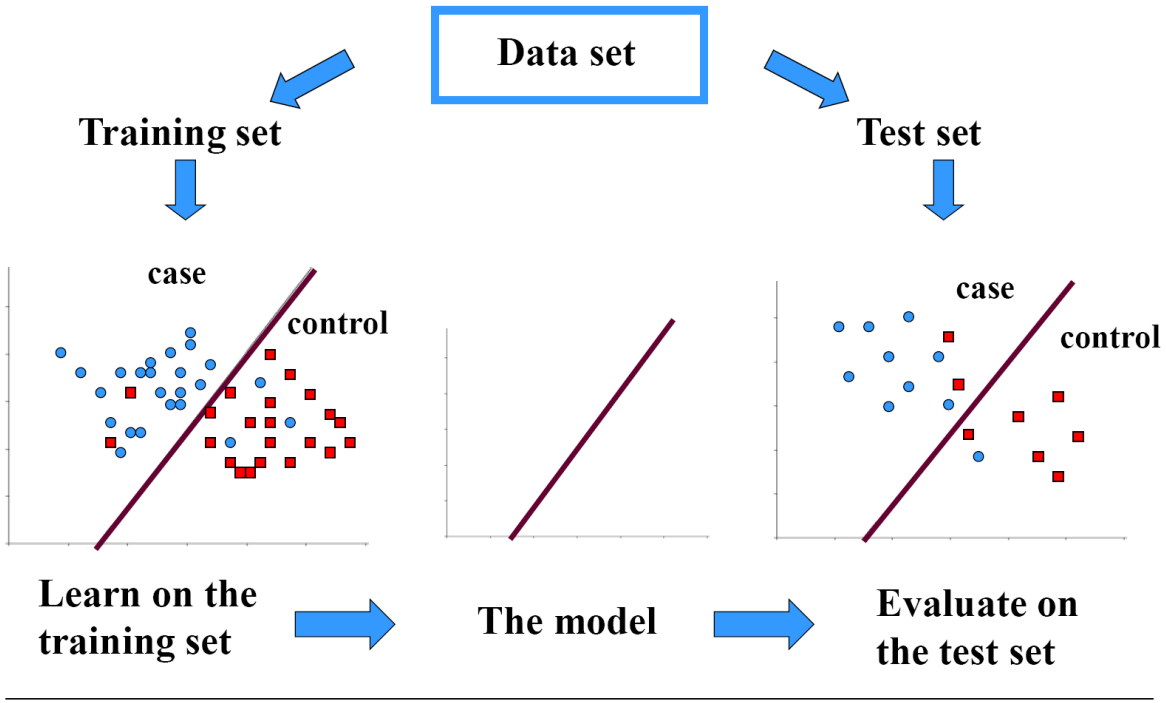


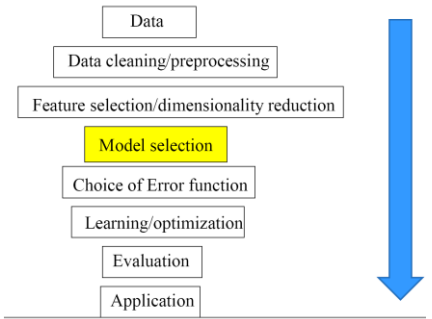


CEE 1323/2323: Practical DS & ML

How to assess the generalization performance of ML models?!

Testing of models: classification





CEE 1323/2323: Practical DS & ML

Choice of Error function & Learning/optimization

Objective function/Error function:

• **Learning = optimization problem**. Various criteria:

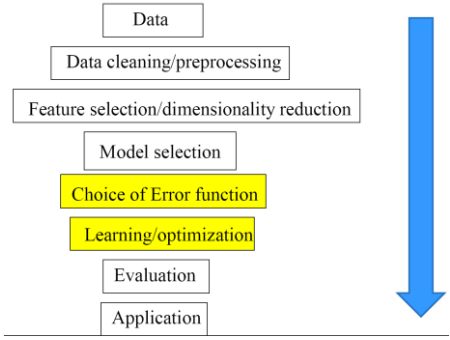
–Mean square error $$Error(\mathbf{w}) = \frac{1}{N} \sum_{i=1,..N} (y_i - f(x_i, \mathbf{w}))^2$$

–Maximum likelihood (ML) criterion $$Error(\Theta) = -\log P(D \mid \Theta)$$

–Maximum posterior probability (MAP) $$P(\Theta \mid D) = \frac{P(D \mid \Theta) P(\Theta)}{P(D)}$$

Optimization problems can be hard to solve. Right choice of a model and an error function makes a difference.

• Parameter optimizations
• Gradient descent, Conjugate gradient
• Newton-Rhapson
• Levenberg-Marquard
…



CEE 1323/2323: Practical DS & ML

Evaluation

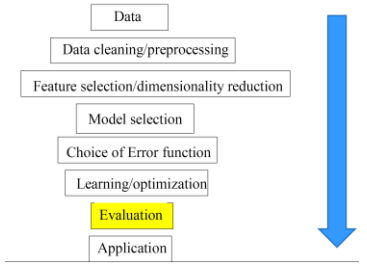
Performance Fitness and Error Metrics (PFEMs) for ML

With the rise of commercial, open-source and user-catered ML tools, a key question often arises whenever ML is applied to explore a phenomenon or a scenario!

What constitutes a good ML model?

A good ML model is one that optimally performs and best describes the phenomenon on hand.

✓**Regression** ✓**Classification**



PFEMs for ML Regression Models

List of commonly used PFEMs for ML regression models

No.	Metric	Definition	Formula	Remarks
1	Error (E)	The amount by which an observation differs from its actual value.	$E = A - P$	• Intuitive • Easy to apply
2	Mean error (ME)	The average of all errors in a set.	$ME = \frac{\sum_{i=1}^{n} E_i}{n}$	• May not be helpful in cases where positive and negative predictions cancel each other out.
3	Mean Normalized Bias (MNB)	Associated with observation-based minimum threshold.	$MNB = \frac{\sum_{i=1}^{n} E_i/A_i}{n}$	• Biased towards overestimations.
4	Mean Percentage Error (MPE)	Computed average of percentage errors.	$MPE = \frac{\sum_{i=1}^{n} E_i/A_i}{n/100}$	• Undefined whenever a single actual value is zero.
5	Mean Absolute Error (MAE)*	Measures the difference between two continuous variables.	$MAE = \frac{\sum_{i=1}^{n} \lvert E_i \rvert}{n}$	• Uses a similar scale to input data. • Can be used to compare series of different scales.

A: actual measurements, *P*: predictions, *n*: number of data points

PFEMs for ML Regression Models

A: actual measurements, *P*: predictions, *n*: number of data points

6	Mean Absolute Percentage Error (MAPE)*	Measures the extent of error in percentage terms.	$MAPE = \frac{100}{n}\sum_{i=1}^{n}\lvert E_i\rvert/\lvert A_i\rvert$	• Commonly-used as a loss function • Cannot be used if there are actual zero values. • Percentage error cannot exceed 1.0 for small predictions.
7	Relative Absolute Error (RAE)	Expressed as a ratio comparing the mean error to errors produced by a trivial model.	$RAE = \sum_{i=1}^{n}\lvert E_i\rvert/\lvert A_i - A_{mean}\rvert$	• E_i ranges from zero (being ideal) to infinity.
8	Mean Absolute Relative Error (MARE)	Measures the average ratio of absolute error to random error.	$MARE = \frac{1}{n}\sum_{i=1}^{n}\lvert E_i\rvert/\lvert A_i\rvert$	• Sensitive to outliers. • Division by zero may occur (if actuals contain zeros).

9	Mean Relative Absolute Error (MRAE)	Ratio of accumulation of errors to cumulative error of random error.	$MRAE = \frac{\sum_{i=1}^{n} \lvert E_i \rvert / \lvert A_i - A_{mean} \rvert}{n}$	• For a perfect fit, the numerator equals to zero.
10	Geometric Mean Absolute Error (GMAE)*	Defined as the n-th root of the product of error values.	$GMAE = \sqrt[n]{\prod_{i=1}^{n} \lvert E_i \rvert}$	• GMAE is more appropriate for averaging relative quantities as opposed to arithmetic mean. • This metric can be dominated by large outliers and minor errors (i.e. close to zero).
11	Fractional Absolute Error (FAE)	Evaluates the absolute fractional error.	$FAE = \frac{1}{n} \sum_{i=1}^{n} \frac{2 \times \lvert E_i \rvert}{\lvert A_i \rvert + \lvert P_i \rvert}$	-
12	Mean Squared Error (MSE)	Measures the average of the squares of the errors.	$MSE = \frac{\sum_{i=1}^{n} {E_i}^2}{n}$	• Scale dependent. • Values closer to zero present adequate state • Heavily weights outliers. • Highly dependent on fraction of data used (low reliability).
13	Root Mean Squared Error (RMSE)	Root square of average squared error.	$RMSE = \sqrt{\frac{\sum_{i=1}^{n} {E_i}^2}{n}}$	• Scale dependent. • A lower value for RMSE is favorable. • Sensitive to outliers. • Highly dependent on fraction of data used (low reliability).

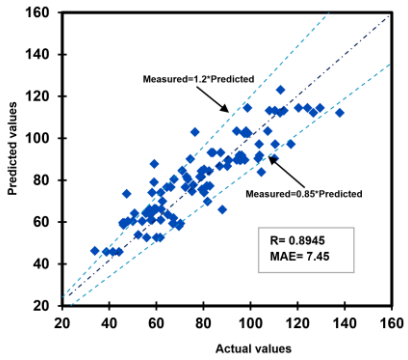
14	Sum of Squared Error (SSE)	Sums the squared differences between each observation and its mean.	$SSE = \sum_{i=1}^{n} {E_i}^2$	• A small SSE indicates a tight fit.
15	Relative Squared Error (RSE)	Normalizes total squared error by dividing by the total squared error.	$RSE = \sum_{i=1}^{n} {E_i}^2 / (A_i - A_{mean})^2$	• A perfect fit is achieved when the numerator equals to zero.
16	Root Relative Squared Error (RRSE)	Evaluates the root relative squared error between two vectors.	$RRSE = \sqrt{\sum_{i=1}^{n} {E_i}^2 / (A_i - A_{mean})^2}$	• Ranges between zero and 1, with zero being ideal.
17	Geometric Root Mean Squared Error (GRMSE)	Evaluates the geometric root squared errors.	$GRMSE = \sqrt[2n]{\prod_{i=1}^{n} {E_i}^2}$	• Scale dependent. • Less sensitive to outliners than RMSE.
18	Mean Square Percentage Error (MSPE)*	Evaluates the mean of square percentage errors.	$MSPE = \frac{\sum_{i=1}^{n} (\lvert E_i \rvert / \lvert A_i \rvert)^2}{n/100}$	• Non-symmetrical.
19	Root Mean Square Percentage Error (RMSPE)*	Evaluates the mean of squared errors in percentages.	$RMSPE = \sqrt{\frac{\sum_{i=1}^{n} (\lvert E_i \rvert / \lvert A_i \rvert)^2}{n/100}}$	• Scale independent. • Can be used to compare predictions from different datasets. • Non-symmetrical.
20	Normalized Root Mean Squared Error (NRMSE)**	Normalizes the root mean squared error.	$NRMSE = \frac{\sqrt{\frac{\sum_{i=1}^{n} {E_i}^2}{n}}}{A_{mean}}$	• Can be used to compare predictions from different datasets.
21	Normalized Mean Squared Error (NMSE)	Estimates the overall deviations between measured values and predictions.	$NMSE = \frac{\frac{\sum_{i=1}^{n} {E_i}^2}{n}}{variance^2}$ $variance = \frac{\sum (x_i - mean)^2}{n-1}$	• Biased towards over-predictions.
22	Coefficient of Determination (R^2)	The square of correlation.	$R^2 = 1 - \sum_{i=1}^{n} (P_i - A_i)^2 / \sum_{i=1}^{n} (A_i - A_{mean})^2$	• R^2 values close to 1.0 indicate strong correlation. • Can be used in predicting material properties.

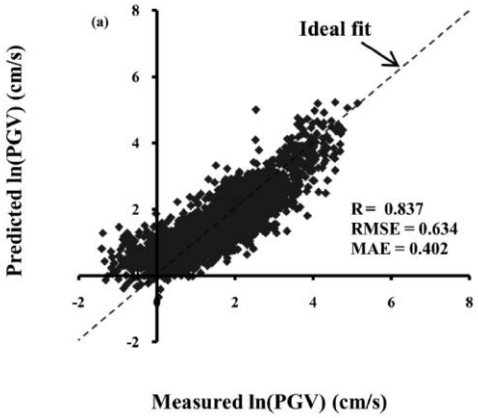
23	Correlation coefficient (R)	Measures the strength of association between variables.	$R = \frac{\sum_{i=1}^{n}(A_i - \overline{A}_i)(P_i - \overline{P}_i)}{\sum_{i=1}^{n}(A_i - \overline{A}_i)^2 \sum_{i=1}^{n}(P_i - \overline{P}_i)}$	• R>0.8 implies strong correlation. • Does not change by equal scaling. • Can be used in predicting material properties.
24	Mean Absolute Scaled Error (MASE)	Mean absolute errors divided by the mean absolute error.	$\frac{\sum_{i=1}^{n}\frac{E_i}{A_i}}{n/100} / (\frac{1}{n} - 1)\sum_{i=1}^{n}\lvert A_i - A_{i-1}\rvert$	• Scale independent. • Stable near zero.
25	Golbraikh and Tropsha's (2003) criterion	-	*At least one slope of regression lines (k or k') between the regressions of actual (A_i) against predicted output (P_i) or P_i against A_i through the origin, i.e. $A_i = k \times P_i$ and $t_i = k' A_i$, respectively.* $k = \frac{\sum_{i=1}^{n}(A_i \times P_i)}{{A_i}^2}$ $k' = \frac{\sum_{i=1}^{n}(A_i \times P_i)}{{A_i}^2}$ $m = \frac{R^2 - {R_o}^2}{R^2}$ $n = \frac{R^2 - {R_{o'}}^2}{R^2}$	• *k* and *k'* need to be close to 1 or at least within the range of 0.85 and 1.15. • *m* and *n* are performance indexes and their absolute value should be lower than 0.1.
26	QSAR model by Roy and Roy (2008)	-	$R_m = R^2 \times (1 - \sqrt{\lvert R^2 - {R_o}^2\rvert})$ *where,* ${R_o}^2 = \frac{\sum_{i=1}^{n}(P_i - A_i^o)^2}{\sum_{i=1}^{n}(P_i - P_{mean})^2}, A_i^o = k \times P_i$ ${R_o}^2 = \frac{\sum_{i=1}^{n}(A_i - P_i^o)^2}{\sum_{i=1}^{n}(A_i - A_{mean})^2}, P_i^o = k \times A_i$	• R_m is an external predictability indicator. $R_m > 0.5$ implies a good fit.
27	Frank and Todeschini (1994)	-	*Recommend maintaining a ratio of 3-5 between the number of observations and input parameters.*	-

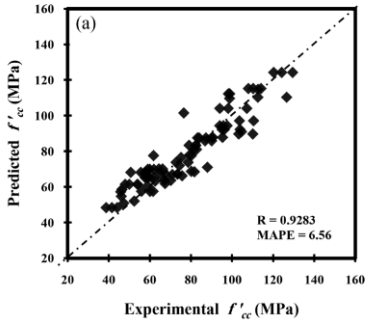
28	Objective function by Gandomi et al. (2013)	A multi-criteria metric.	*Function* $= (\frac{No._{Training} - No._{Validation}}{No._{Training} + No._{Validation}}) RMS + \frac{2No._{Validation}}{No._{Training} + No._{Validation}} RMSE$	• This function needs to be minimized to yield highest fitness.
			where, $No._{Training}$ *and* $No._{Validation}$ *are the number of training and validation data, respectively.*	• Can be used in predicting material properties.
29	Reference index (RI) by Cheng et al. (2014)	A multi-criteria metric that uniformly accounts for RMSE, MAE and MAPE.	$RI = \frac{RMSE + MAE + MAPE}{3}$	• Each fitness metric is normalized to achieve the best performance.

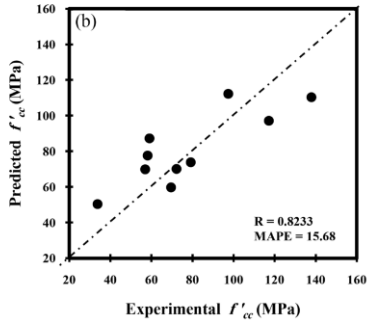
Item	Formula	Condition
1	$R = \frac{(\sum_{i=1}^{n}(O_i - \overline{O_i})(t_i - \overline{t_i}))}{\sqrt{\sum_{1=1}^{n}(O_i - \overline{O_i})^2 \sum_{i=1}^{n}(t_i - \overline{t_i})^2}}$	$0.8 < R$
2	$k = \frac{\sum_{i=1}^{n}(h_i \times t_i)}{{h_i}^2}$	$0.85 < k < 1.15$
3	$k' = \frac{\sum_{i=1}^{n}(h_i \times t_i)}{{t_i}^2}$	$0.85 < k' < 1.15$
4	$m = \frac{R^2 - {R_o}^2}{R^2}$	$\lvert m \rvert < 0.1$
5	$n = \frac{R^2 - {R_{o'}}^2}{R^2}$	$\lvert n \rvert < 0.1$
6	$R_m = R^2 \times \left(1 - \sqrt{\lvert R^2 - {R_o}^2 \rvert}\right)$	$0.5 < R_m$
7	${R_o}^2 = 1 - \frac{\sum_{i=1}^{n}(t_i - {h_i}^o)^2}{\sum_{i=1}^{n}(t_i - \overline{t_i})^2}, {h_i}^o = k \times t_i$	Should be close to 1
8	${R_{o'}}^2 = 1 - \frac{\sum_{i=1}^{n}(h_i - {t_i}^o)^2}{\sum_{i=1}^{n}(h_i - \overline{h_i})^2}, {t_i}^o = k' \times h_i$	Should be close to 1

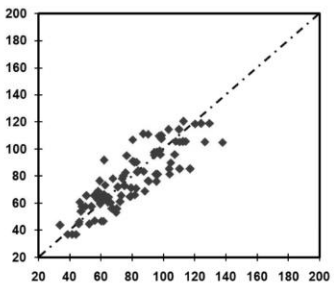
PFEMs for ML Regression Models



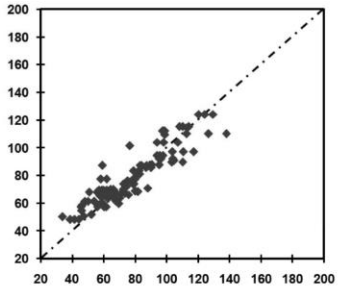




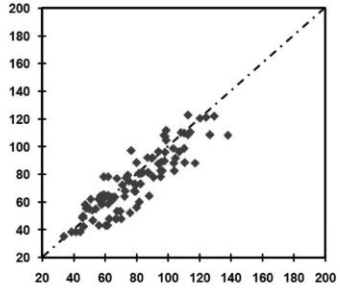




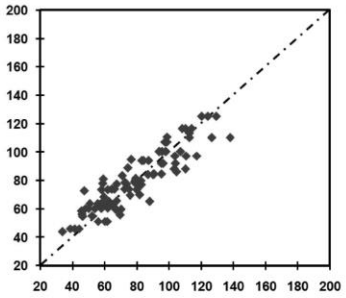
(j) Berthet *et al.* (2006), Eq. (10)
(R=0.659; MAPE=23.16)



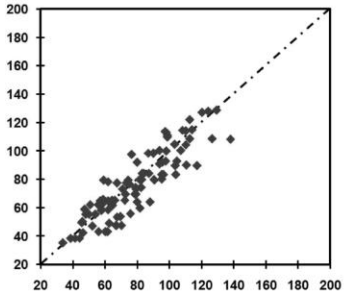
(k) GP/SA, Model I
(R= 0.910; MAPE= 7.27)



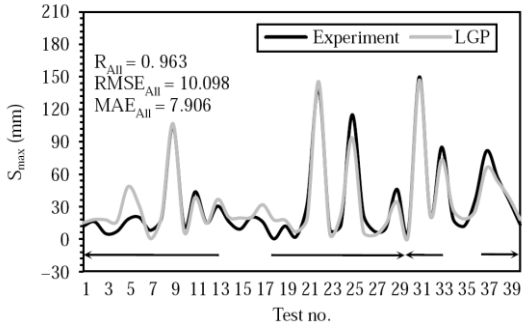
(l) GP/SA, Model II
(R= 0.887; MAPE= 7.45)

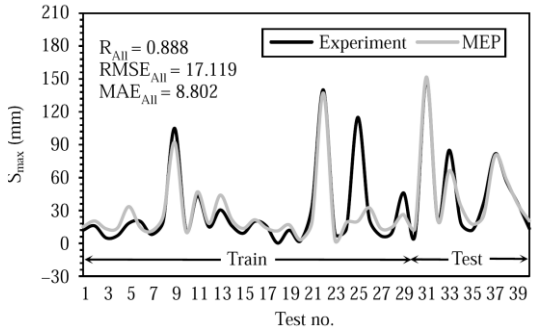


(m) MEP, Model III
(R=0.905; MAPE=7.1)



(n) MEP, Model IV
(R= 0.889; MAPE= 9.21)

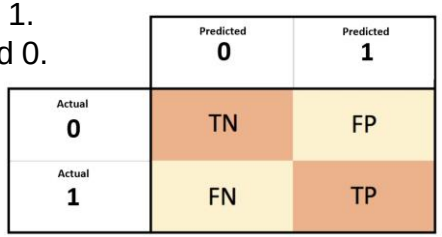




PFEMs for ML Classification Models

No.	Metric	Definition	Formula	Remarks
1	True Positive Rate (TPR) or Sensitivity or Recall	Measures the proportion of actual positives that are correctly identified as positives.	$TPR = \frac{TP}{P} = \frac{TP}{TP + FN} = 1 - FNR$	• Describes the proportion of actual positives that are correctly identified. • Does not account for indeterminate results.
2	True Negative Rate TNR or Specificity or selectivity	Measures the proportion of actual negatives that are correctly identified negatives.	$TNR = \frac{TN}{N} = \frac{TN}{TN + FP} = 1 - FPR$	• Describes the proportion of actual negatives that are correctly identified.
3	Positive Predictive Value (PPV) or Precision	The proportions of positive observations that are true positives.	$PPV = \frac{TP}{TP + FP} = 1 - FDR$	• Has an ideal value of 1 and the worst value of zero.
4	Negative Predictive Value (NPV)	The proportions of negative observations that are true positives.	$NPV = \frac{TN}{TN + FN} = 1 - FOR$	• Has an ideal value of 1 and the worst value of zero.
5	False Positive Rate (FPR)	Measures the proportion of positive cases in that are correctly identified as positives.	$FPR = \frac{FP}{N} = \frac{FP}{FP + TN} = 1 - TNR$	• Describes proportion of negative cases incorrectly identified as positive cases.

TP (true positive) The model predicts that the class is 1 and the class of the given instance is indeed 1.
TN (true negative) The model predicts that the class is 0 and the class of the given instance is indeed 0.
FP (false positive) The model predicts that the class is 1 but the class of the given instance is 0.
FN (false negative) The model predicts that the class is 0 but the class of the given instance is 1.



6	False Discovery Rate (FDR)	Expected proportion of false observations.	$FDR = \frac{FP}{FP + TP} = 1 - PPV$	• Describes proportion of the individuals with a positive test result for which the true condition is negative.
7	False Omission Rate (FOR)	Measures the proportion of false negatives that are incorrectly rejected.	$FDR = \frac{FN}{FN + TPN} = 1 - NPV$	• Describes proportion of the individuals with a negative test result for which the true condition is positive.
8	Positive likelihood ratio (LR+)	Evaluates the change in the odds of having a diagnosis with a positive test.	$LR+= \frac{TPR}{FPR}$	• Measures the ratio of TPR (sensitivity) to the FPR (1 – specificity). • Presents the likelihood ratio for increasing certainty about a positive diagnosis.

9	Negative likelihood ratio (LR-)	Evaluates the change in the odds of having a diagnosis with a negative test.	$LR-=\frac{FNR}{TNR}$	• Describes the ratio of FNR to TNR (specificity).
10	Diagnostic odds ratio (DOR)	Measures the effectiveness of a (diagnostic) test.	$DOR=\frac{LR+}{LR-}=\frac{TP/FP}{FN/TN}$	• Often used in binary classification.
11	Accuracy (ACC)	Evaluates the ratio of number of correct predictions to the total number of samples.	$ACC=\frac{TP+TN}{P+N}$ $=\frac{TP+TN}{TP+TN+FP+FN}$	• Presents performance at a single class threshold only. • Assumes equal cost for errors.
12	F_1 score	Harmonic mean of the precision and recall.	$F_1=\frac{2PPV\times TPR}{PPV+TPR}$ $=\frac{2TP}{2TP+FP+FN}$	• Describes the harmonic mean of precision and sensitivity. • Focuses on one class only. • Biased to the majority class.
13	Matthews Correlation Coefficient (MCC)	Measures the quality of binary classifications analysis.	MCC $=\frac{TP\times TN-FP\times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)}}$	• Measures the quality of binary and multi-class classifications. • Can be used in classes with different sizes. • When MCC equals +1 → perfect prediction, → 0 equivalent to a random prediction and → −1 false prediction. • Considered as a balanced measures as it involves values of all the four quardants of a confusion matrix.
14	Bookmaker Informedness (BM) or Youden's J statistic	Evaluates the discriminative power of the test.	$BM=TPR+TNR-1$	• Describes the probability of an informed decision (vs. a random guess). • Has a range between zero and 1 (being ideal). • Considers both real positives and real negatives.

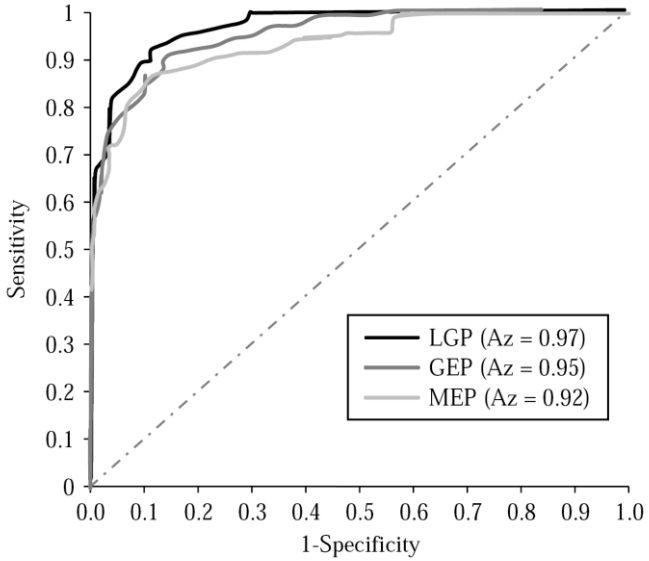
18	Area under the ROC curve (AUC)	Measures the two-dimensional area underneath the entire ROC curve.	$AUC = \sum_{i=1}^{N-1} \frac{1}{2}(FP_{i+1} - FP_i)(TP_{i+1} - TP_i)$ *or* $AUC = \frac{1}{2} w\,(h + h')$, *where, w = width, and h and h' = heights of the sides of a trapezoid histogram*	• Not dependent on a single class threshold. • Associated with increased training times.
19	Precision-Recall curve	Plots the tradeoff between precision and recall for different thresholds.	*Plots precision (in the vertical axis) and the recall (in the horizontal axis) for different thresholds.*	• Applicable in cases of moderate to large class imbalance. • Used in binary classification.
20	Log Loss Error (LLE)	Measures the where the prediction input is a probability value.	$LLE = -\sum_{c=1}^{M} A_i \log P$, *where, M: number of classes, c: class label, y: binary indicator (0 or 1) if c is the correct classification for a given observation.*	• Measures the uncertainty of the probabilities by comparing predictions to the true labels. • Penalizes for being too confident in wrong prediction. • Has probability between zero and 1. • A log loss of zero indicates a perfect model.
21	Hinge Loss Error (HLE)	-	$HLE = max(0, 1 - q \cdot y)$ *where, q= ±1 and y: classifier score*	• Linearly penalize incorrect predictions. • Primarily used in support vector machine.
22	Wilcoxon–Mann–Whitney (WMW) test	-	$WMW = \frac{\sum_{i \in Minor\ class} \sum_{i \in Major\ class} I_{wmw}(P_i}{\lvert Minor\ class \rvert \times \lvert Major\ class \rvert}$ *where, P_i and P_j: outputs when evaluated on an example from the minority and majority classes, respectively*	• Used in scenarios with unbalanced data. • The indicator function I_{wmw} returns 1 if $P_i > P_j$ and $P_i \geq 0$ or 0 if otherwise.
23	Fitness Function *Amse* (FFA)	Measures pattern difference between input and output.	$FFA = \frac{1}{K} \sum_{c=1}^{K} \left(1 - \frac{\sum_{i=1}^{N_c}(1 - sig(P_{ci}) - T_c)}{N_c \times 2}\right)^2$, $sig(x) = \frac{2}{1 + e^{-x}} + 1$	• Used in scenarios with unbalanced data. • Appropriate for genetic programing. • Needs to be scaled to a range of [-1, 1] and hence the

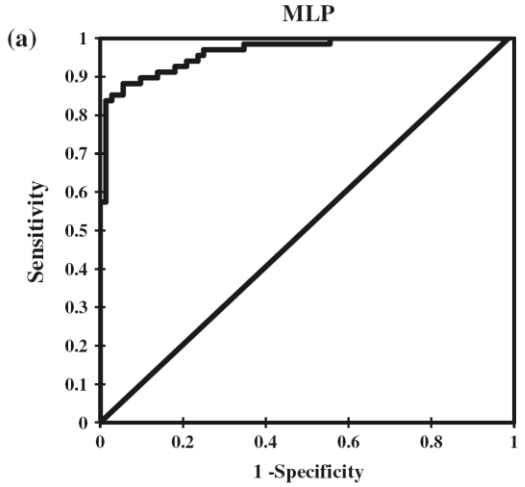
			where, P_{ci}: output of a classifier evaluated on the ith example, N_c: number of examples, K: number of classes, T_c : target values (equals to -0.5 and 0.5 for majority and minority classes, respectively)	need for sigmoid function. • FFA = 1 presents an ideal scenario.
24	Fitness Function *Incr* (FFI)	-	$Incr = \frac{1}{K}\sum_{c=1}^{K}\left(\frac{\sum_{j=1}^{M_c}[I_{zt}(j, D_{cj}, c).\sum_{i=1}^{N_c} Eq($ $\frac{1}{2}N_c(N_c+1)$ $I_{zt}(r,k,c) = \begin{cases} r, & \text{if } k \geq 0 \text{ and } c \in \\ & \text{or if } k < 0 \text{ and } c \\ 0, & \text{othe} \end{cases}$ $Eq(p,q) = \begin{cases} 1, \\ 0, otherwise \end{cases}$	• Used in scenarios with unbalanced data. • Assigns incremental rewards to predictions that fall further away from the class boundary. • Appropriate for genetic programming. • Ranges [0, 1] (zero being worst fitness).
25	Fitness Function Correlation (FFC)	-	$FFC = \frac{1}{K}(r + I_{zt}(1, \mu_{minor}, \mu_{major}),$ $r = \sqrt{\frac{\sum_{c=1}^{K} N_c(\mu_c - \bar{\mu})^2}{\sum_{c=1}^{K}\sum_{i=1}^{N_c}(P_{ci} - \bar{\mu})^2}}$ $\mu_c = \frac{\sum_{i=1}^{N_c} P_{ci}}{N_c}, \bar{\mu} = \frac{\sum_{c=1}^{K} N_c\mu_c}{\sum_{c=1}^{K} N_c}.$ *where, r: correlation ratio, μ_{minor} and μ_{major}: mean for minor and major classes, respectively*	• Used in scenarios with unbalanced data.
26	Fitness Function Distribution (FFD)	Measures the distance between class distributions as a function of class separability.	$FFD = \frac{\lvert\mu_{min} - \mu_{maj}\rvert}{\sigma_{min} + \sigma_{maj}} \times I_{zt}(2, \mu_{min}, \mu_{maj})$ $\mu_c = \frac{\sum_{i=1}^{N_c} P_{ci}}{N_c}, \sigma_c = \sqrt{\frac{1}{N_c}\sum_{i=1}^{N_c}(P_{ci} - \mu_c)^2}.$ *where, μ_c and σ_c: mean and standard deviation of the class distribution, respectively,*	• Used in scenarios with unbalanced data. • Treats predictions as independent distributions. • Measures separability (i.e. distance between class distributions) – high separability (no overlap) and this distance turns large (go to +∞).
27	Canberra Metric (CM)	Measures the distance between pairs of points in a vector space.	$CM = \sum_{i=1}^{n}\frac{\lvert E_i\rvert}{A_i + P_i}$	-

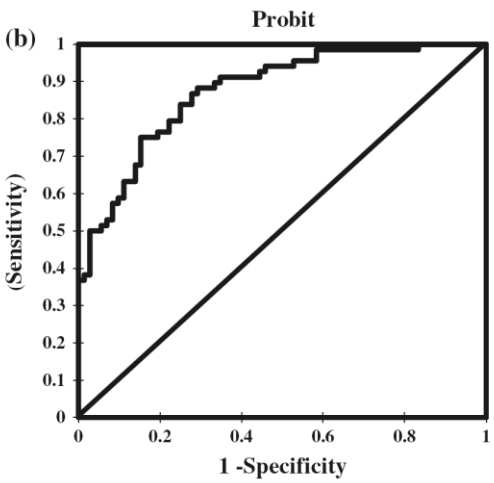
28	Wave Hedges Distance (WHD)	-	$WHD = \sum_{i=1}^{n} \frac{\lvert E_i \rvert}{max\ (A_i, P_i)}$	• Normalizes the difference of each pair of coefficients with its maximum
29	Lift	Measures the performance of a model at predicting or classifying cases.	$LIFT = \frac{\%of\ true\ positives\ above\ the\ th}{\%of\ dataset\ above\ the\ thres}$	• Measures betterness of a classifier than a baseline classifier that randomly predicts positives. • Threshold is set as a static fraction of the positive dataset. • Lift and Accuracy do not always correlate well.
30	Mean Cross Entropy (MXE)	Measures the performance of a model where the output is a probability between zero and one.	$MXE = -\frac{1}{N}\sum True \times ln(Predicted) + (1 - True) \times ln(1 - Predicted)$ *(The assumptions are that Predicted ∈ [0, 1] and True ∈ {0, 1})*	• Minimizing MXE gives the maximum likelihood
31	Probability Calibration (CAL)	-	*1. Order cases 1-100 by their predicted in the same bin.* *2. Evaluate the percentage of true positives.* *3. Calculate the mean prediction for true positives.* *4. …*	• Lengthy procedure.
32	Precision-recall break-even point	Point at which the precision-recall-curve intersects the bisecting line.	*Precision = Recall*	• Defines the point when precision and recall are equal.
33	Average precision (AP)	Combines recall and precision for ranking.	$AP = \sum_{n} (Recall_n - Recall_{n-1}) Percision_n$	• Describes the weighted mean of precision in each threshold with the increase in recall from the previous threshold used.
34	Balanced accuracy	Calculates the average of the correctly identified proportion of individual classes.	*Defined as the average of recall obtained on each class.*	• Used in binary and multiclass classification problems. • Accommodates imbalanced datasets.

35	Brier score (BS)	Measures the accuracy of probabilistic-based predictions.	$BS = \frac{1}{N}\sum_{i=1}^{N}(f_i - A_i)^2$ *in which f_i is the probability that was forecast, A_i the actual outcome of the event at instance i*	• Measures the mean squared difference between the predicted probability and the actual outcome. • Takes on a value between zero and 1 (the lower the score is, the better the predictions).
36	Cohen's kappa (CK)	Measures interrater (agreement) reliability.	$\kappa = (p_o - p_e)/(1 - p_e)$ *where, p_o: empirical probability of agreement on the label assigned to any sample, p_e: expected agreement when both annotators assign labels randomly and this is estimated using a per-annotator empirical prior over the class labels.*	• Measures inter-annotator agreement. • Expresses the level of agreement between two annotators • Ranges between -1 and 1. The maximum value means complete agreement.
37	Hamming loss (HL)	Fraction of the wrongly identified labels.	$HL = \frac{1}{m}\sum_{i=1}^{m} 1_{\widehat{P_{i \neq A_i}}}$	• Describes fraction of labels that are incorrectly predicted. • Optimal value is zero.
38	Fitness (T)	-	$Fitness(T) = Q(T) + \alpha * R(T) + \beta * Cost(T)$ *where, Q(T): accuracy, R(T): sum of $R(T_i)$ in all multi-tests of the T tree, Cost(T): sum of the costs of attributes constituting multi-tests. The default parameters values are: α=1.0 and β=−0.5,* $R(T_i) = \frac{\lvert X_i \rvert}{\lvert X \rvert} * \sum_{j=1}^{\lvert mt_i \rvert - 1} r_{ij}$ *where, X: learning set, X_i: instances in i-th node, and $\lvert mt_i \rvert$: size of a multi-test.* $Cost(T_i) = \frac{\lvert X \rvert}{\lvert X_i \rvert} * C(a_{ij})$ *where: a_{ij}: j-th attribute of the i-th multi-test, $C(a_{ij})$: cost of the a_{ij} attribute.*	• Used for fitting decision trees. • This function needs to be maximized to achieve high performance.

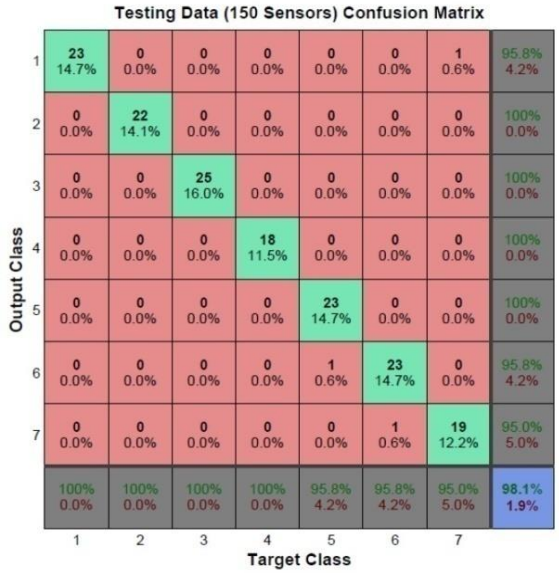
39	F2 score	Measured as the weighted average of precision and recall.	$F_\beta = 1 + \beta 2 \times \frac{precision \times recall}{(\beta 2 \times precision) + recall}$ *where: β = 2.*	• Used in genetic programming and medical fields. • Computes a weighted harmonic mean of Precision and Recall. • Learning about the minority class.
40	Distance score (D score)	-	$D_{sc} = \frac{2 \times C1 \times C2}{C1 + C2}$ *where:* $C1 = \frac{\sum_{i=0}^{N_{maj}} sig(P_{Maji}) \times \lvert T - sig(P_{Maji}) \rvert}{N_{maj}} \times func(1, P_{Maji})$ $sig(x) = \frac{2}{1 + e - x} - 1$ $C2 = \frac{\sum_{i=0}^{N_{min}} sig(P_{Mini}) \times \lvert T - sig(P_{Mini}) \rvert}{N_{min}} \times func(1, P_{Mini})$ $func(1,k) = \begin{cases} 1, & \text{if} k \leq 0 \text{formajorityclassinstan} \\ 1, & \text{if} k > 0 \text{forminorityclassinstan} \\ 0, & \text{otherwise} \end{cases}$ *C1* for majority class and *C2* for minority class.	• Used in genetic programming and medical fields. • Distance score (D score) which learns about both the classes by giving them equal importance and being unbiased. • The range of both C1 and C2 is 0 (worst score) to 1 (best score).





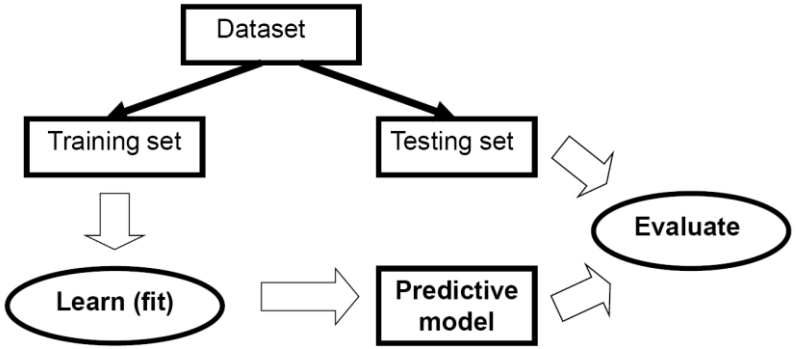


		Predicted Class: Positive	Predicted Class: Negative	
Actual Class	Positive	True Positive (TP)	False Negative (FN) Type II Error	Sensitivity $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) Type I Error	True Negative (TN)	Specificity $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	Negative Predictive Value $\frac{TN}{(TN+FN)}$	Accuracy $\frac{TP+TN}{(TP+TN+FP+FN)}$



CEE 1323/2323: Practical DS & ML

Evaluation

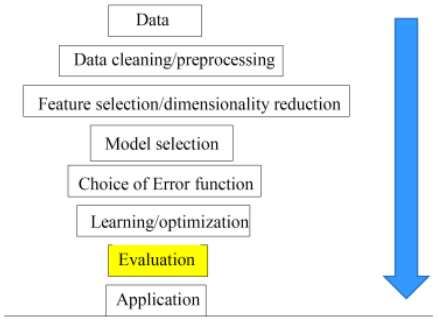


Problem: the mean error results may be influenced by a lucky or an unlucky training and testing split especially for small data sizes

Solution: try multiple train-test splits and average their results

Other more complex methods

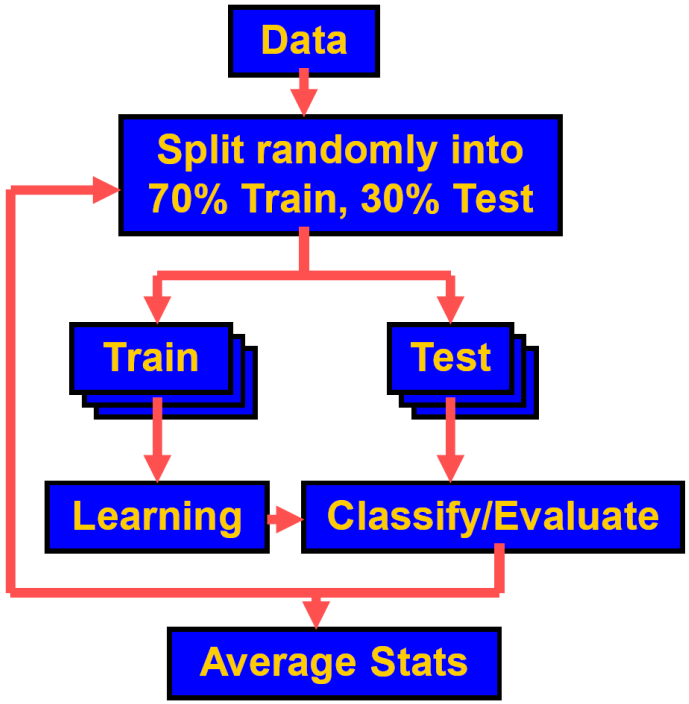
- Use multiple train/test sets
- Based on various random re-sampling schemes:
  1. **Random sub-sampling**
  2. **Cross-validation**

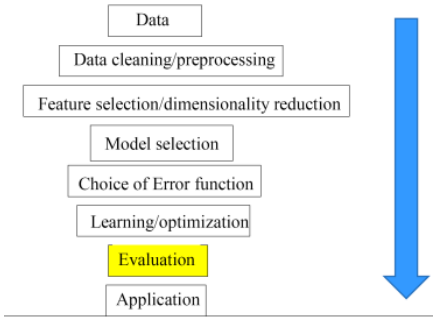


Evaluation

Random sub-sampling

– Repeat a simple holdout method *k* times



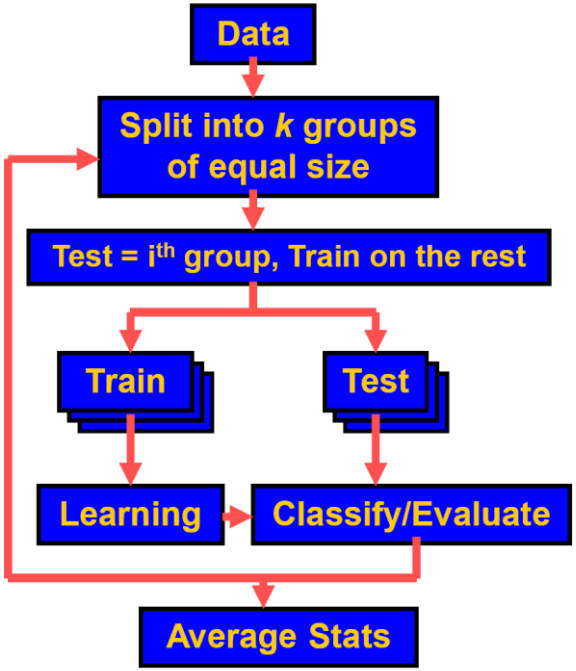


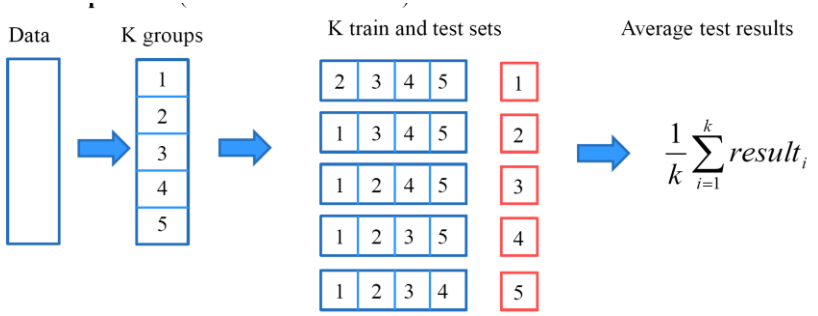
CEE 1323/2323: Practical DS & ML

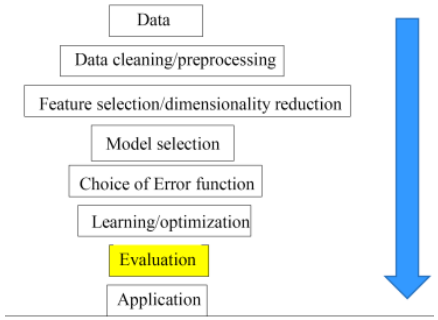
Evaluation

Cross-validation (k-fold)

- Divide data into k disjoint groups, test on i-th group and train on the rest
- Typically 10-fold cross-validation
- Leave one out cross-validation
- Gives k models and k test results



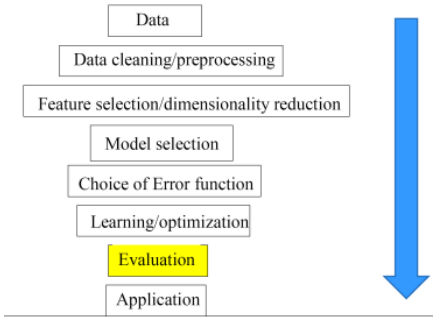




CEE 1323/2323: Practical DS & ML

Evaluation

Parametric and Sensitivity Analyses



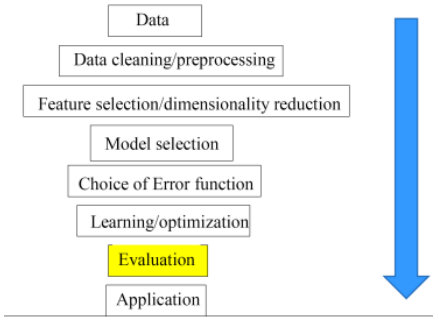
Evaluation

Parametric and Sensitivity Analyses

Parametric analysis ➔ Underlying physical relations governing the behavior of the system

Sensitivity analysis ➔ The most important predictor variables

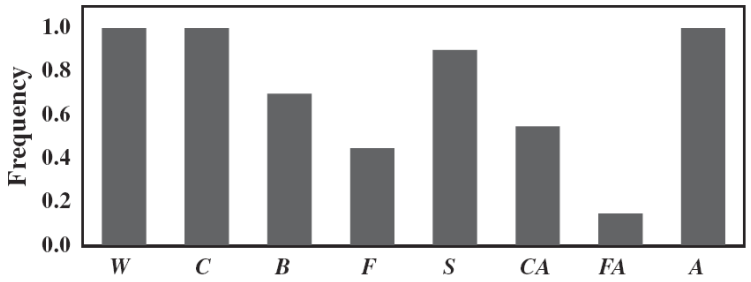
The parametric analysis investigates the response of the predicted output from the ML model to a set of hypothetical input data.

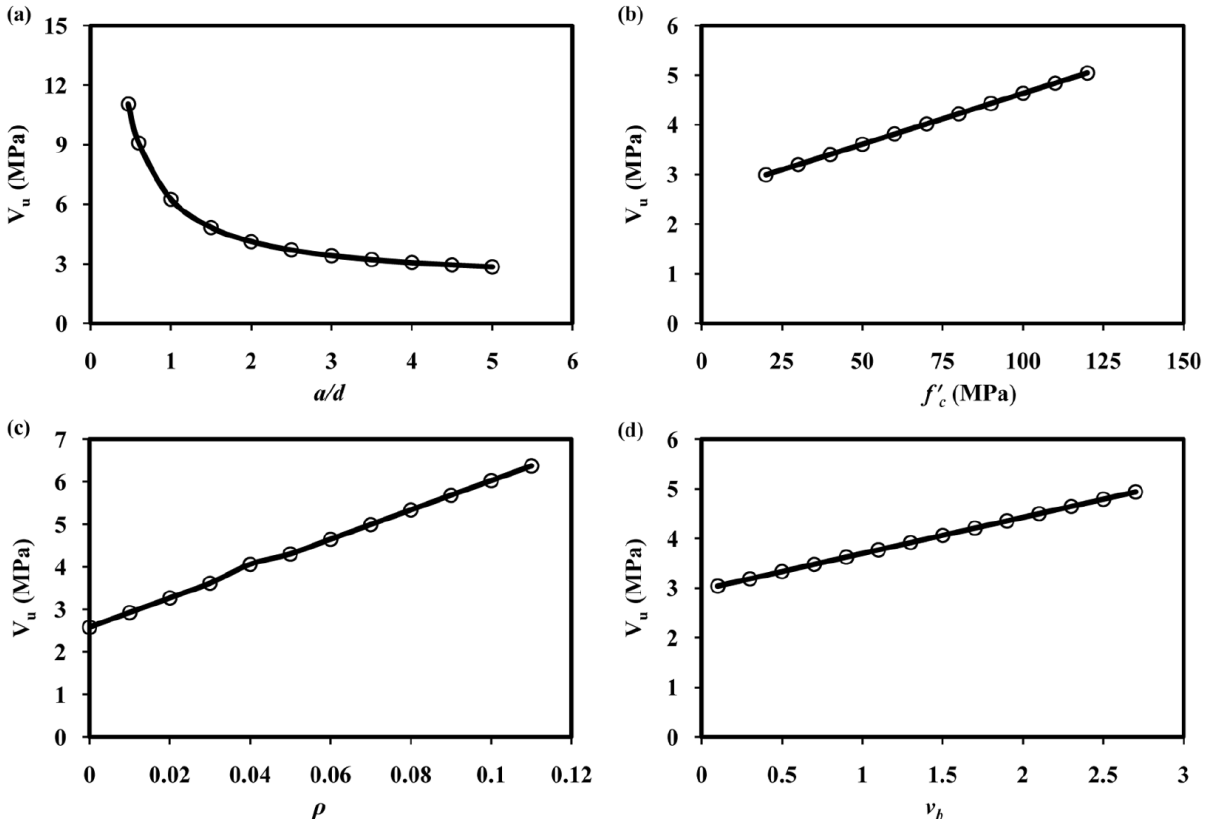


Parametric and Sensitivity Analyses

$$N_i = f_{\max}(x_i) - f_{\min}(x_i)$$

$$S_i = \frac{N_i}{\sum_{j=1}^{n} N_j} \times 100$$





CEE 1323/2323: Practical DS & ML

End: Prediction, Errors, Cross Validation, Data Preprocessing, Model Validation

Next: Regression