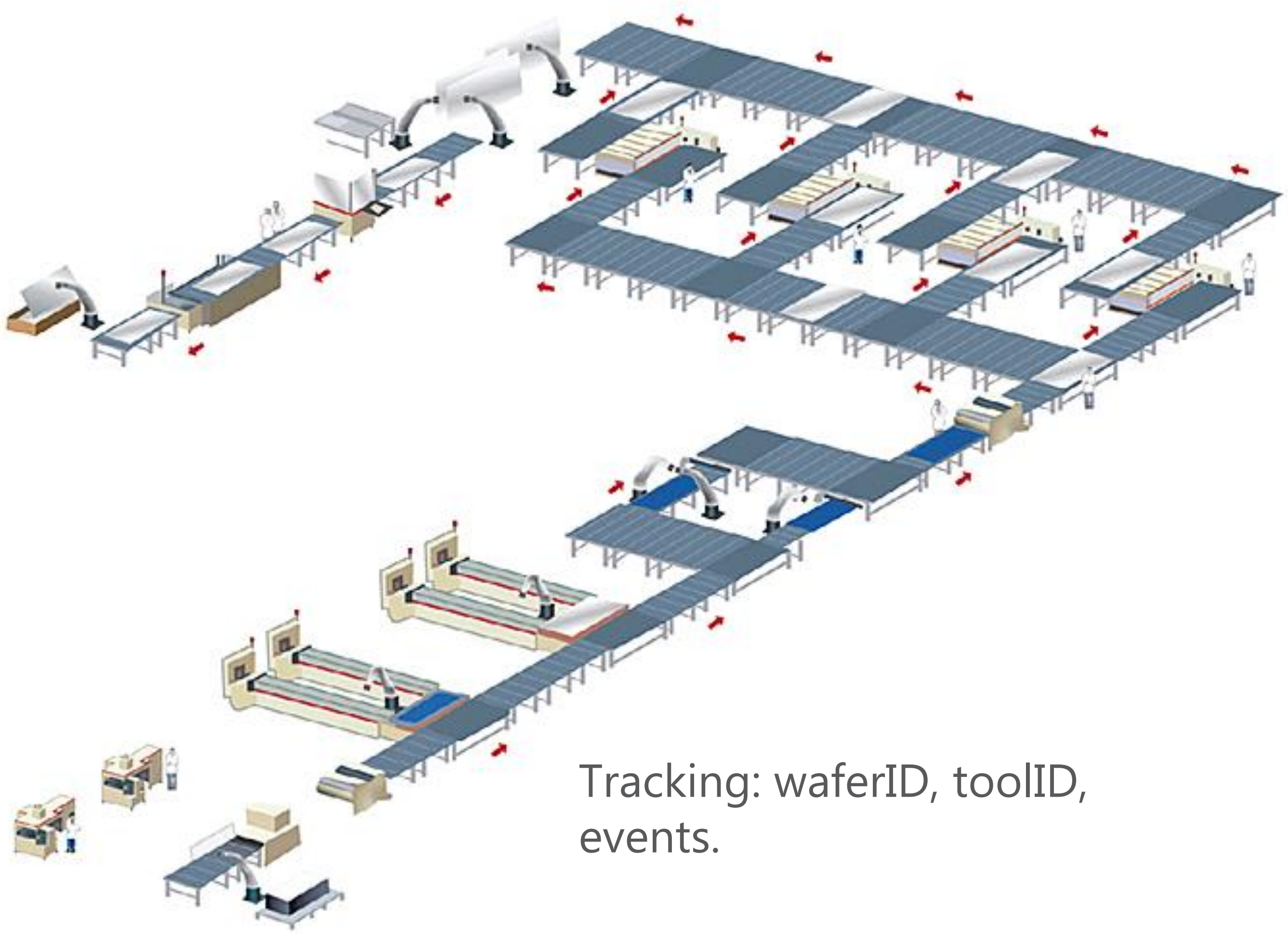
Applications of Data Analytics in Manufacturing

Minh Nguyen, PhD.

Rolls Royce Digital.

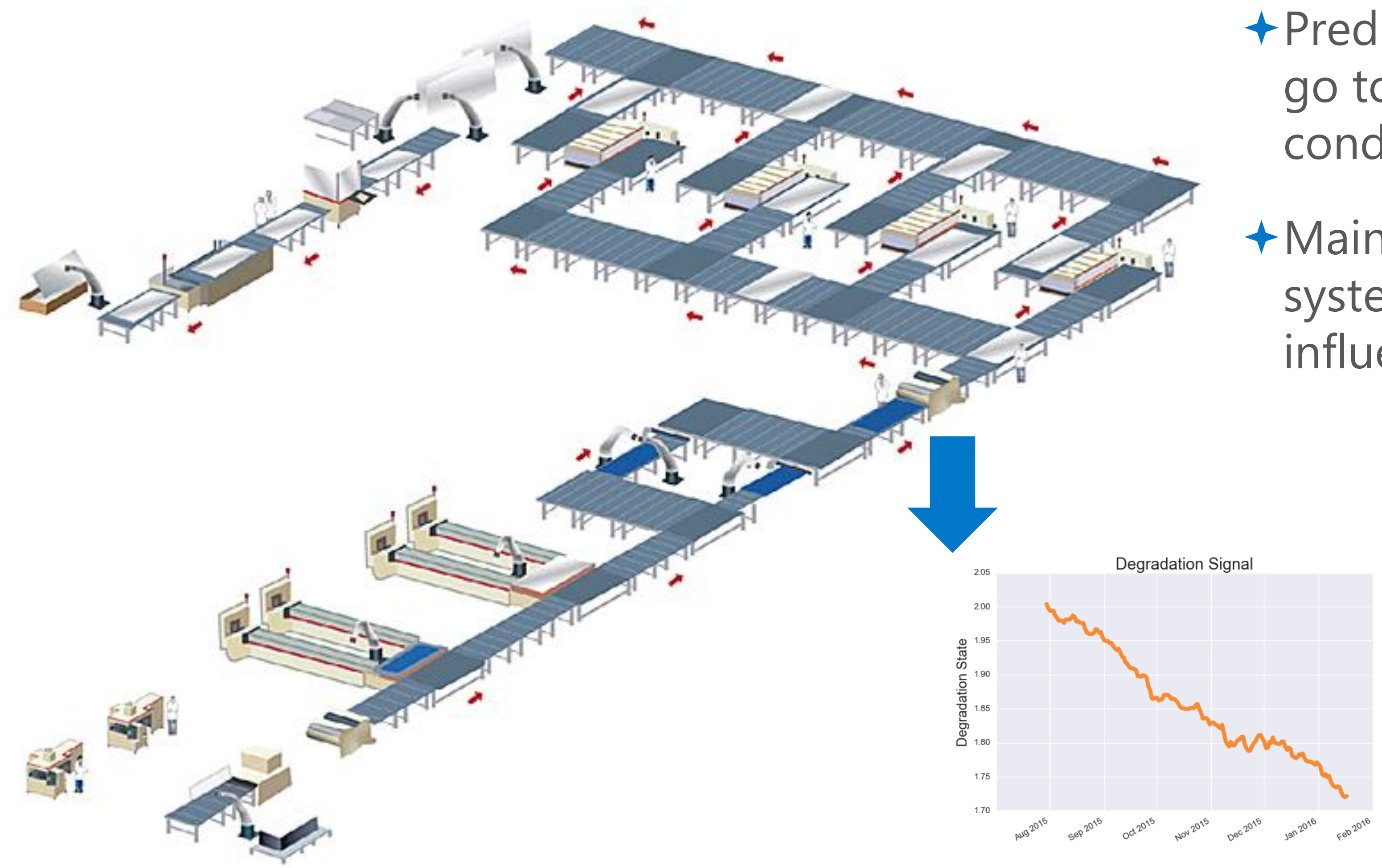
Overview of Production Line



Tracking: waferID, toolID, events.

- Event system: recipe changes, tool maintenance activities (PM), human actions.
- Measurement system: deviation quality gates, yield output.
- Investigation system: find quality deviation root causes

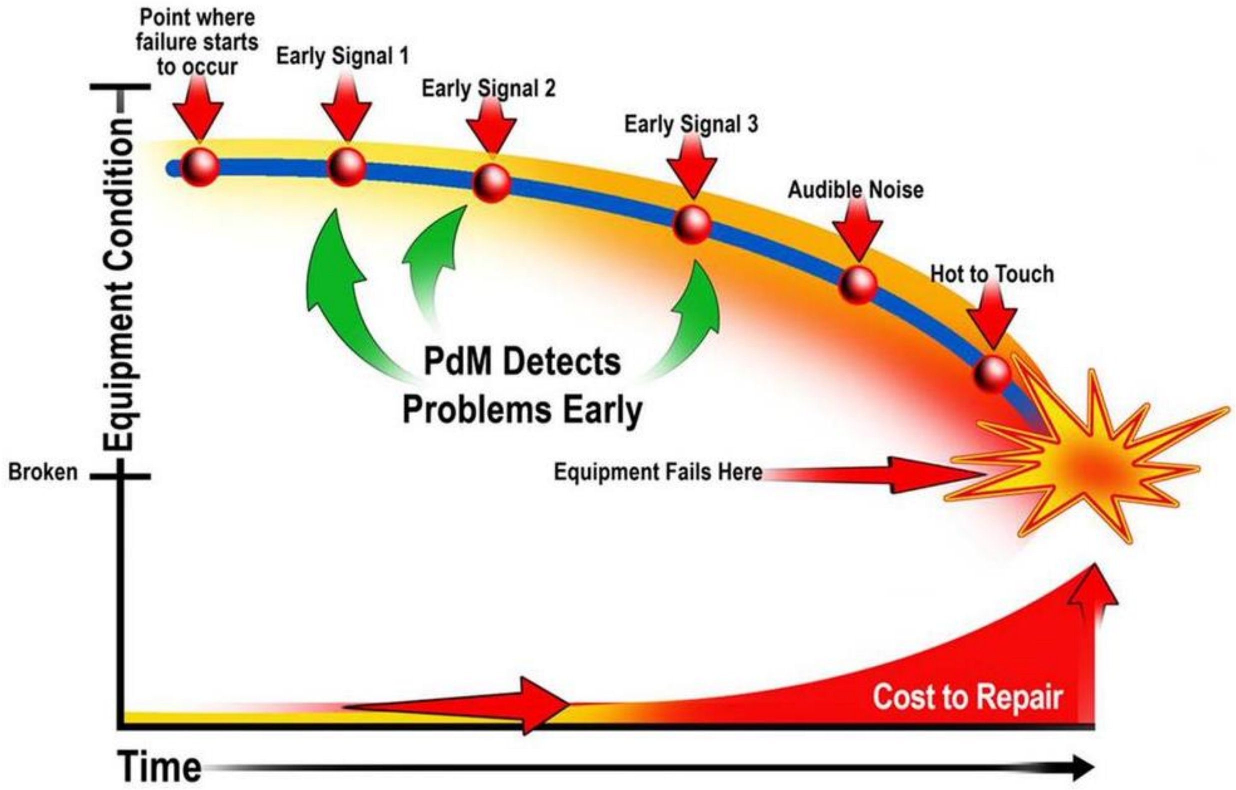
Event System Solution: Predictive Maintenance



- Predict when the tool would go to an unacceptable condition.
- Mainly based on monitoring system, but event data also influences the result.

Predictive Maintenance

- Predict when a tool would go bad.



Sources: www.maintenance.org

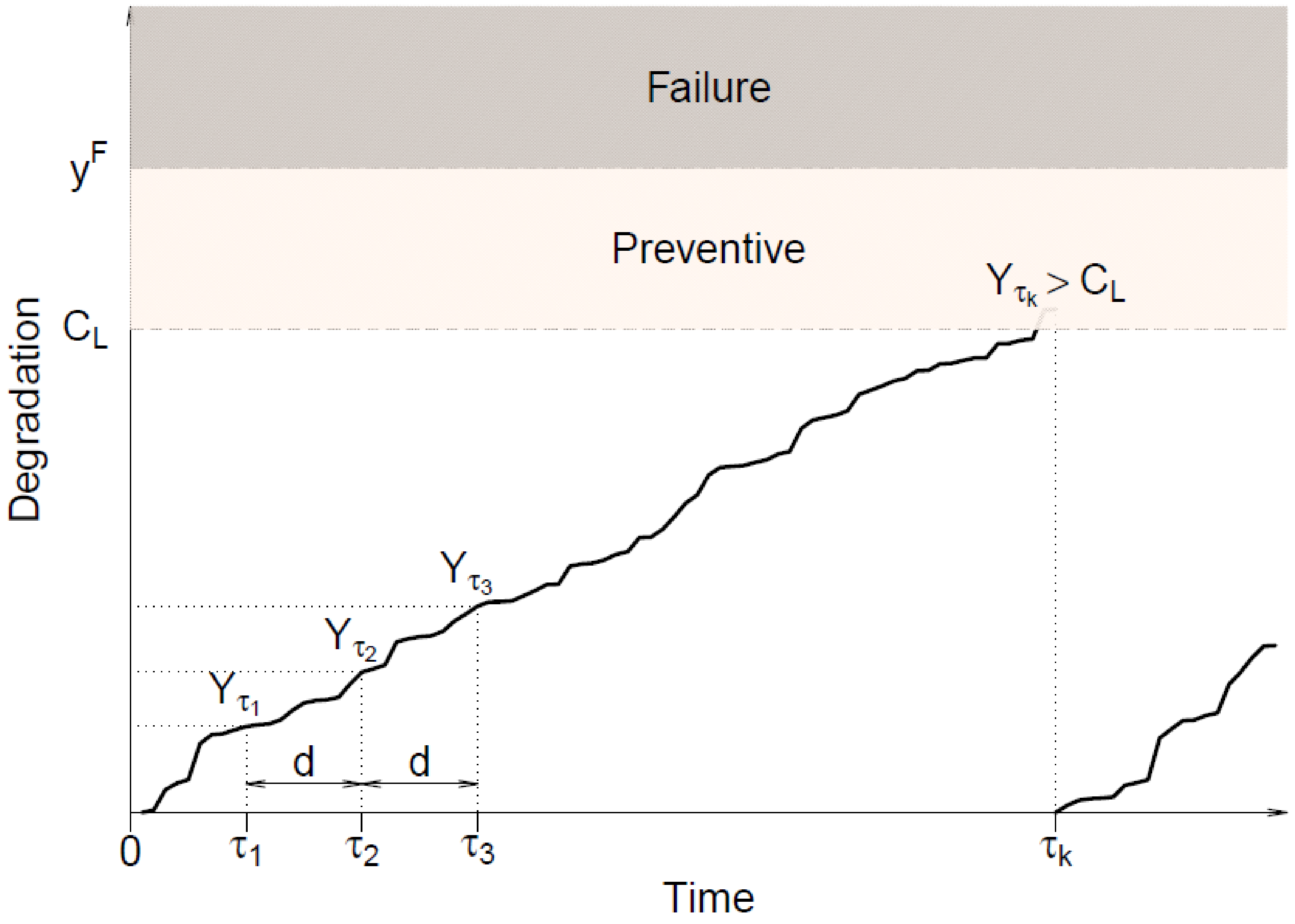
Maintenance Model

Terminologies:

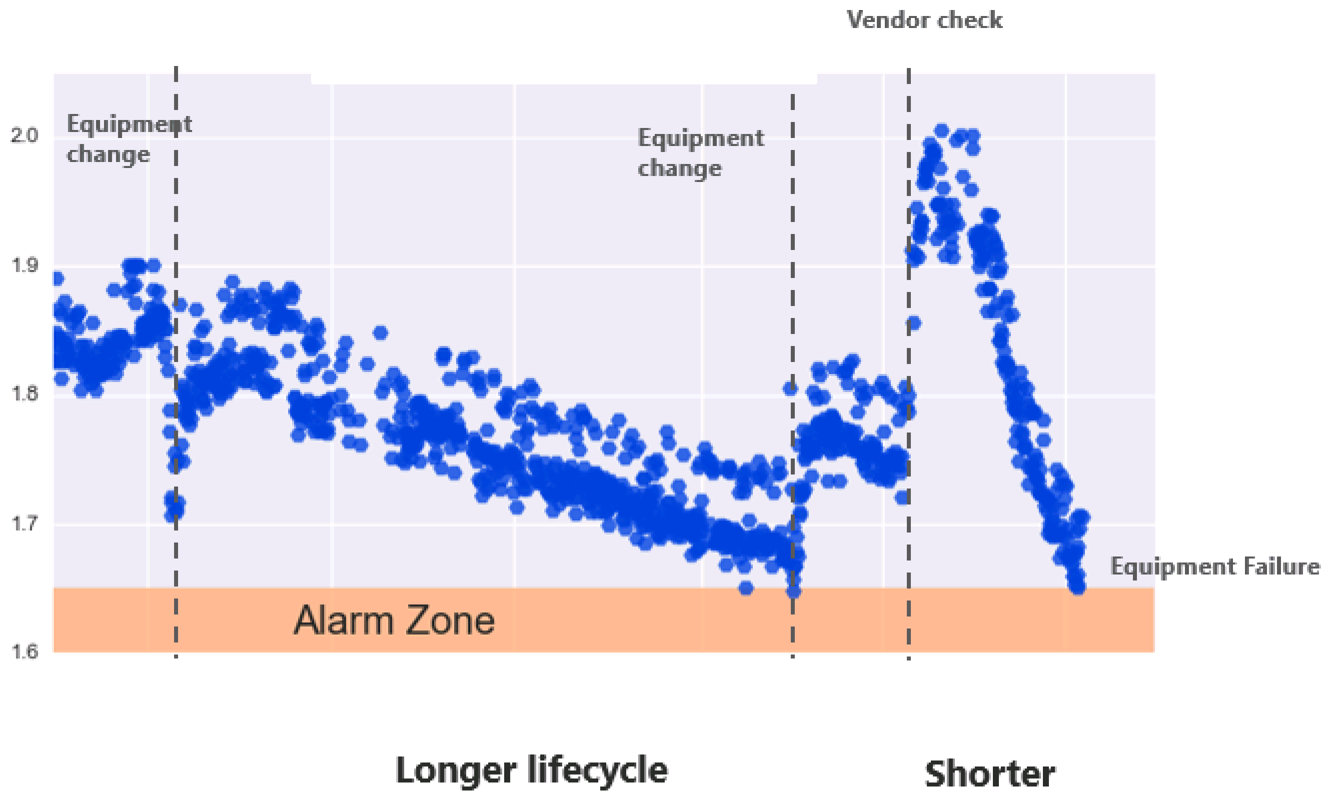
Degradation signal Y:
- Monotonically increasing or decreasing
- Recover after CM/PM

Failure Limit y^F:
- Tool fails when degradation signal hits

Control Limit C_L:
- Perform PM when degradation signal hits



Real Scenario



Problem Formulation

$$\boxed{y = f(X)}$$

Regression

Binary classification

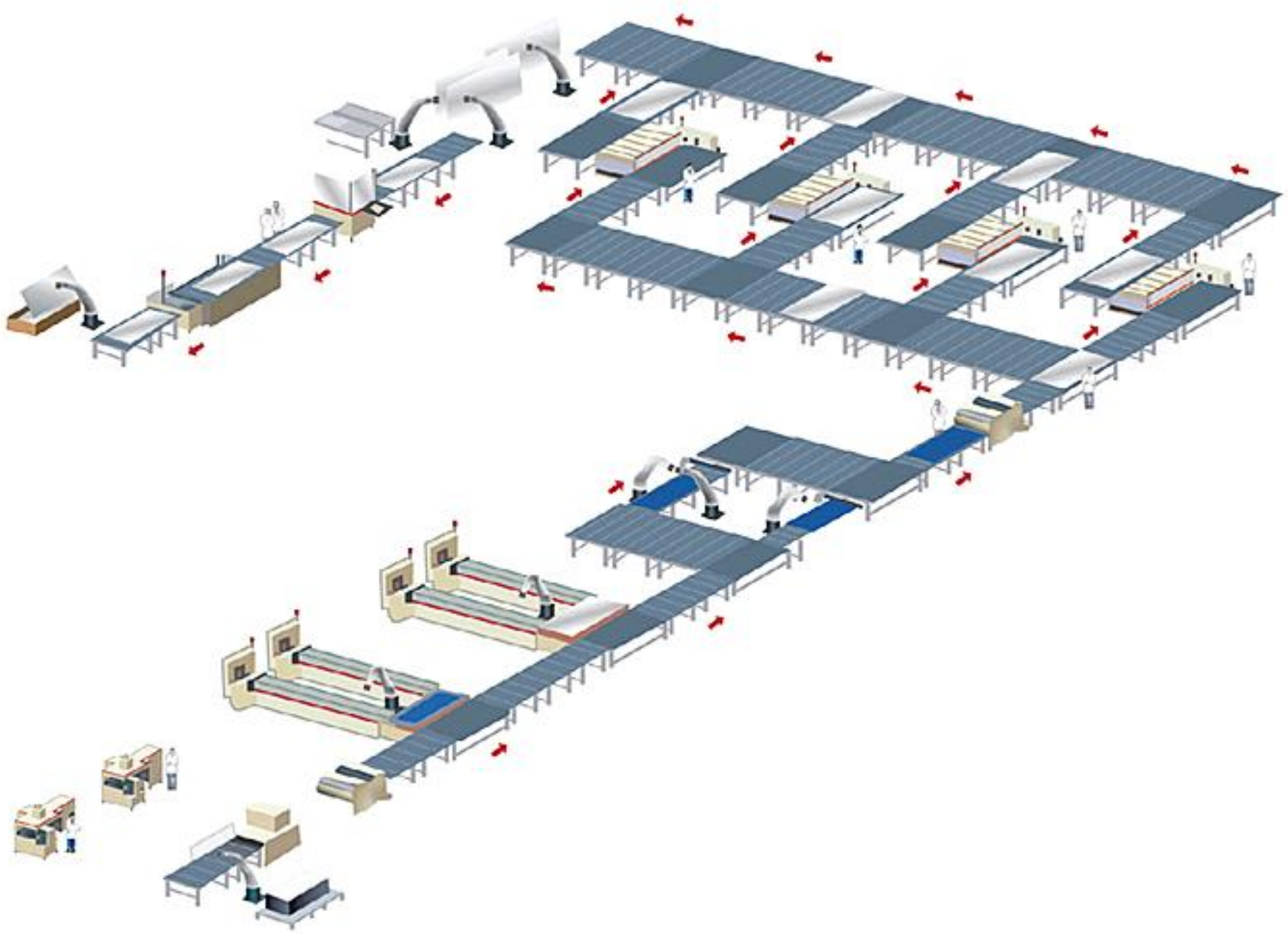
Multi-class classification

id	cycle	...	RUL	label1	label2
1	1		191	0	0
1	2		190	0	0
1	3		189	0	0
1	4		188	0	0
...			...	...	...
1	160		32	0	0
1	161		31	0	0
1	162		30	1	1
1	163		29	1	1
1	164		28	1	1
1	165		27	1	1
1	166		26	1	1
1	167		25	1	1
1	168		24	1	1
1	169		23	1	1
1	170		22	1	1
1	171		21	1	1
1	172		20	1	1
1	173		19	1	1
1	174		18	1	1
1	175		17	1	1
1	176		16	1	1
1	177		15	1	2
1	178		14	1	2
1	179		13	1	2
1	180		12	1	2
1	181		11	1	2
1	182		10	1	2
1	183		9	1	2
1	184		8	1	2
1	185		7	1	2
1	186		6	1	2
1	187		5	1	2
1	188		4	1	2
1	189		3	1	2
1	190		2	1	2
1	191		1	1	2
1	192		0	1	2

w1

w0

Measurement Solution: Virtual Measurement



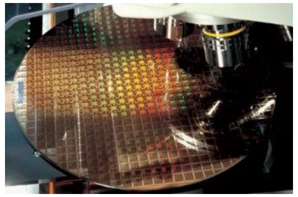
- Measurement is an expensive operation, some destructive. Consequently, its sampling rate is low.
- However, some monitoring systems have 100% sampling rate, either through sensors or scanning images.

Virtual Measurement

- Motivation: Predict low-sampled measurement using 100% sampled images.

Virtual Measurement

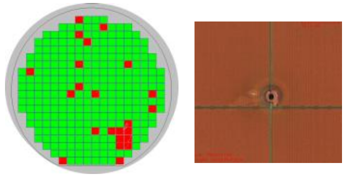
Conventional Macro Inspection



Wafer Process

8% lot-level
1% wafer-level
sampling

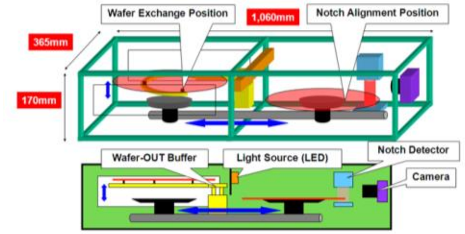
RDA Inspection Tool



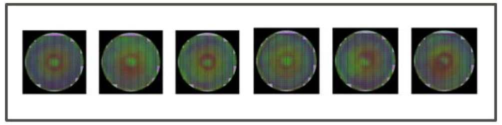
RDA Wafer Map and Defects

Big Data Method

Photo WIS



All wafers will go through multiple photo WIS scans

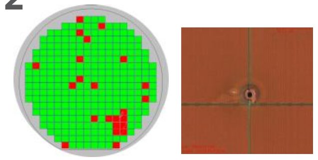


100% sampling!!!

1

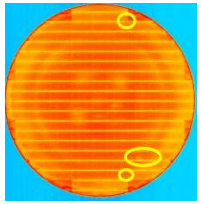
WIS RGB extracted for each pixel

2

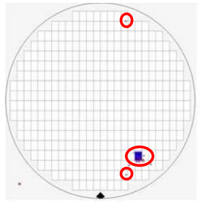


VRDA analysis to flag defect

3



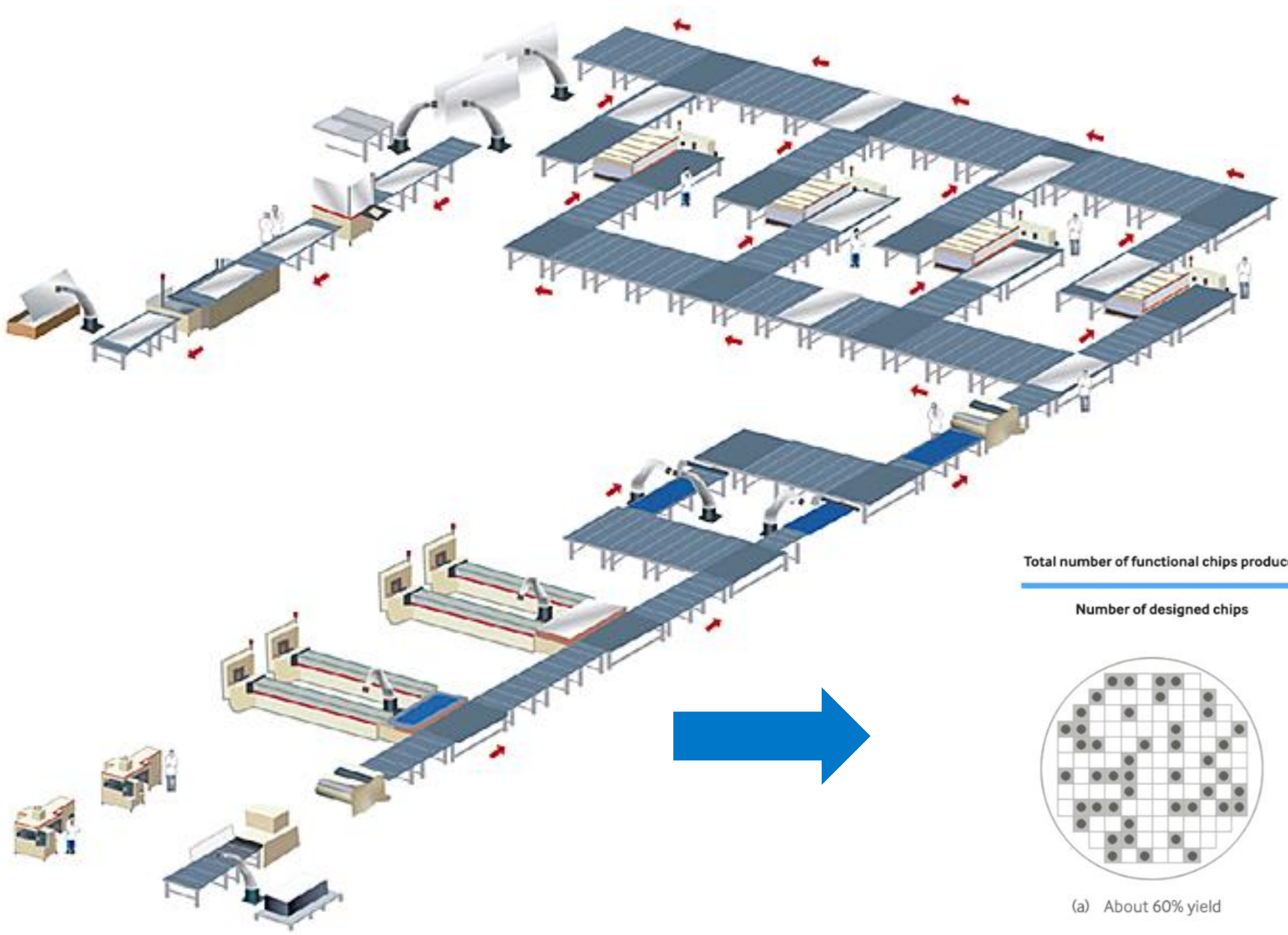
Predicted RDA defects



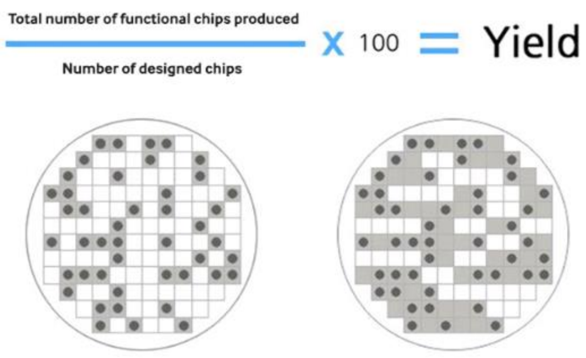
Actual RDA defects

100% wafer sampling →Higher visibility to detect defective wafers

Investigation Solution: Yield Modelling



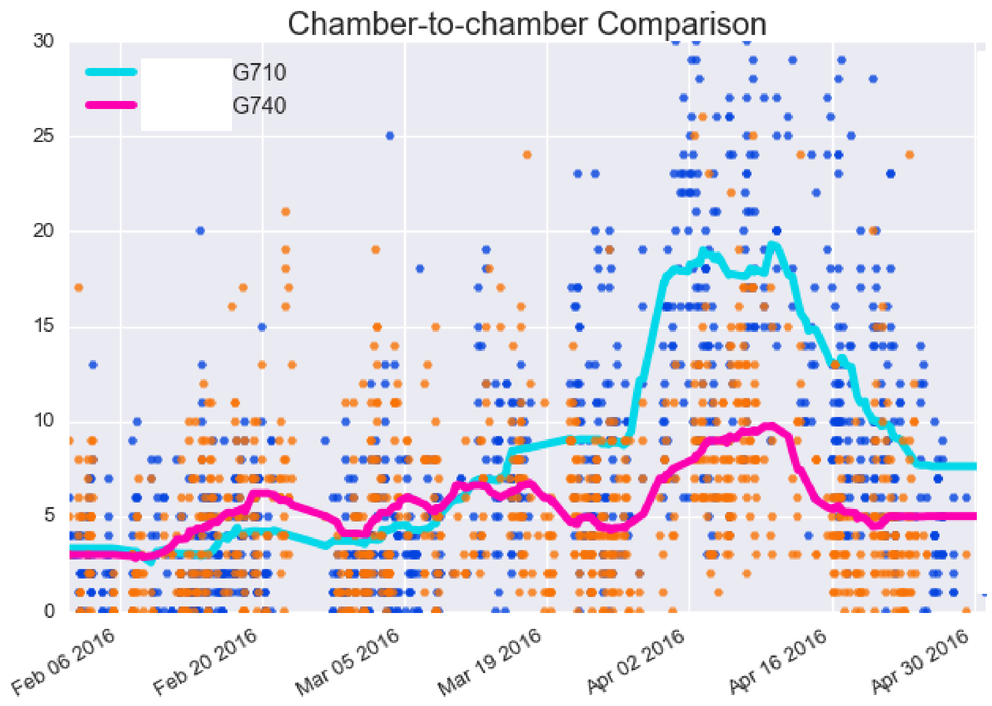
- Ensure all products meet performance criteria
- Have sampling rate 100%
- Be used to investigate what went wrong in the event/monitoring system.

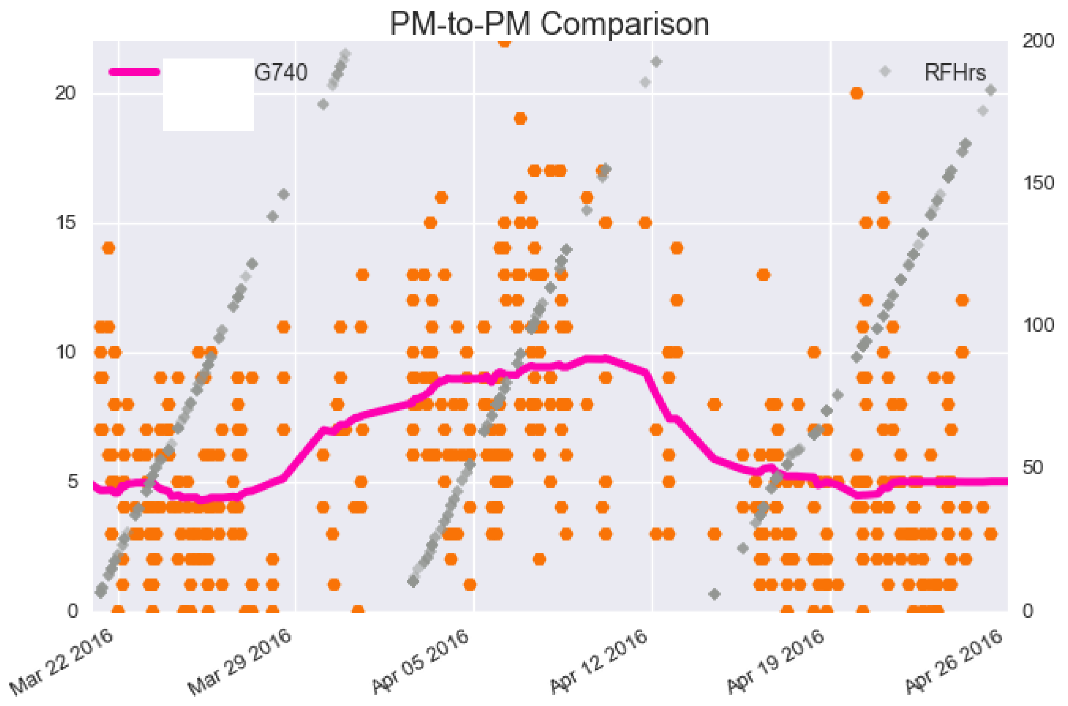


(a) About 60% yield (b) About 30% yield

Yield Modelling

✦Investigate what went wrong with a bad tool

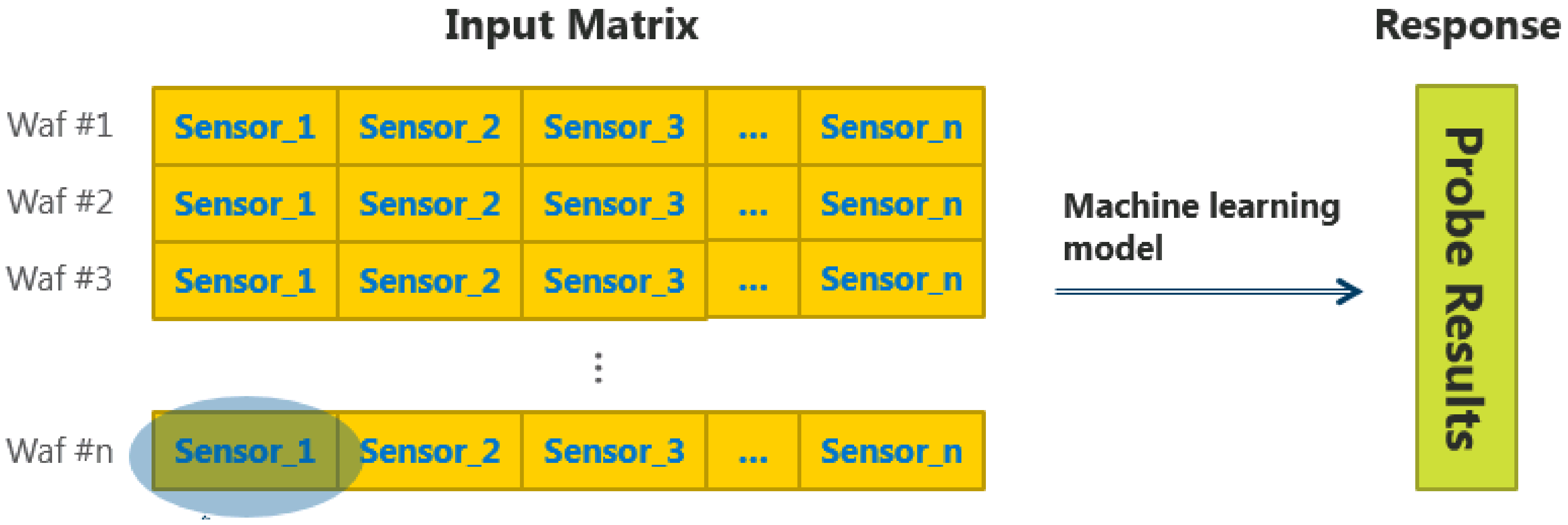




What are the root causes?
How to improve yield?

Answer from tool sensor signals
=> Bad maintenance activity.

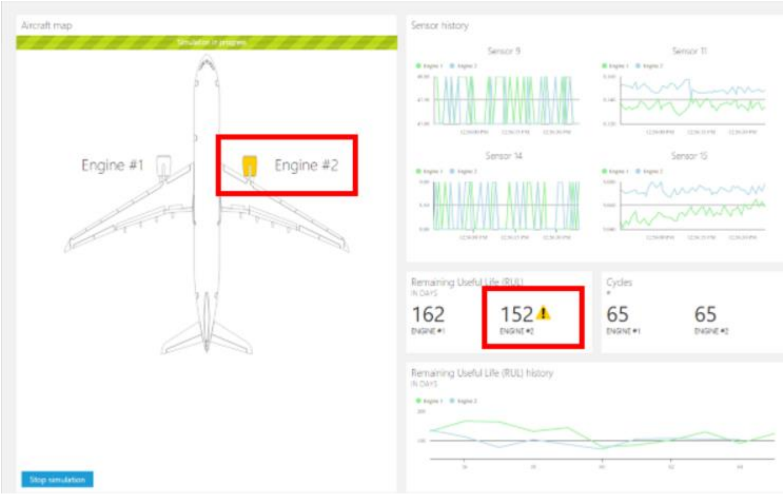
Yield Modelling



Modelling with Lasso and Random Forest

Purposes of Data Modelling

- Inference
  - Explain the relationship among predictors and between predictors and responses.
  - Tell data insights and trigger investigation
- Prediction
  - Estimate responses given predictor values in a set of unobserved samples.



Data Structure

Sample training data

~20k rows,

100 unique engine id

id	cycle	setting1	setting2	setting3	s1	s2	s3	...	s19	s20	s21	RUL	label1	label2
1	1	-0.0007	-0.0004	100	518.67	641.82	1589.7		100	39.06	23.419			
1	2	0.0019	-0.0003	100	518.67	642.15	1591.82		100	39	23.4236			
1	3	-0.0043	0.0003	100	518.67	642.35	1587.99		100	38.95	23.3442			
...	...													
1	191	0	-0.0004	100	518.67	643.34	1602.36		100	38.45	23.1295			
1	192	0.0009	0	100	518.67	643.54	1601.41		100	38.48	22.9649			
2	1	-0.0018	0.0006	100	518.67	641.89	1583.84		100	38.94	23.4585			
2	2	0.0043	-0.0003	100	518.67	641.82	1587.05		100	39.06	23.4085			
2	3	0.0018	0.0003	100	518.67	641.55	1588.32		100	39.11	23.425			
...	...													
2	286	-0.001	-0.0003	100	518.67	643.44	1603.63		100	38.33	23.0169			
2	287	-0.0005	0.0006	100	518.67	643.85	1608.5		100	38.43	23.0848			

?

Sample testing data

~13k rows,

100 unique engine id

id	cycle	setting1	setting2	setting3	s1	s2	s3	...	s19	s20	s21
1	1	0.0023	0.0003	100	518.67	643.02	1585.29		100	38.86	23.3735
1	2	-0.0027	-0.0003	100	518.67	641.71	1588.45		100	39.02	23.3916
1	3	0.0003	0.0001	100	518.67	642.46	1586.94		100	39.08	23.4166
...	...										
1	30	-0.0025	0.0004	100	518.67	642.79	1585.72		100	39.09	23.4069
1	31	-0.0006	0.0004	100	518.67	642.58	1581.22		100	38.81	23.3552
2	1	-0.0009	0.0004	100	518.67	642.66	1589.3		100	39	23.3923
2	2	-0.0011	0.0002	100	518.67	642.51	1588.43		100	38.84	23.2902
2	3	0.0002	0.0003	100	518.67	642.58	1595.6		100	39.02	23.4064
...	...										
2	48	0.0011	-0.0001	100	518.67	642.64	1587.71		100	38.99	23.2918
2	49	0.0018	-0.0001	100	518.67	642.55	1586.59		100	38.81	23.2618
3	1	-0.0001	0.0001	100	518.67	642.03	1589.92		100	38.99	23.296
3	2	0.0039	-0.0003	100	518.67	642.23	1597.31		100	38.84	23.3191
3	3	0.0006	0.0003	100	518.67	642.98	1586.77		100	38.69	23.3774
...	...										
3	125	0.0014	0.0002	100	518.67	643.24	1588.64		100	38.56	23.227
3	126	-0.0016	0.0004	100	518.67	642.88	1589.75		100	38.93	23.274

Data Structure

$$\boxed{y = f(X)}$$

Regression

Binary classification

Multi-class classification

id	cycle	...	RUL	label1	label2
1	1		191	0	0
1	2		190	0	0
1	3		189	0	0
1	4		188	0	0
...			...		...
1	160		32	0	0
1	161		31	0	0
1	162		30	1	1
1	163		29	1	1
1	164		28	1	1
1	165		27	1	1
1	166		26	1	1
1	167		25	1	1
1	168		24	1	1
1	169		23	1	1
1	170		22	1	1
1	171		21	1	1
1	172		20	1	1
1	173		19	1	1
1	174		18	1	1
1	175		17	1	1
1	176		16	1	1
1	177		15	1	2
1	178		14	1	2
1	179		13	1	2
1	180		12	1	2
1	181		11	1	2
1	182		10	1	2
1	183		9	1	2
1	184		8	1	2
1	185		7	1	2
1	186		6	1	2
1	187		5	1	2
1	188		4	1	2
1	189		3	1	2
1	190		2	1	2
1	191		1	1	2
1	192		0	1	2

w1

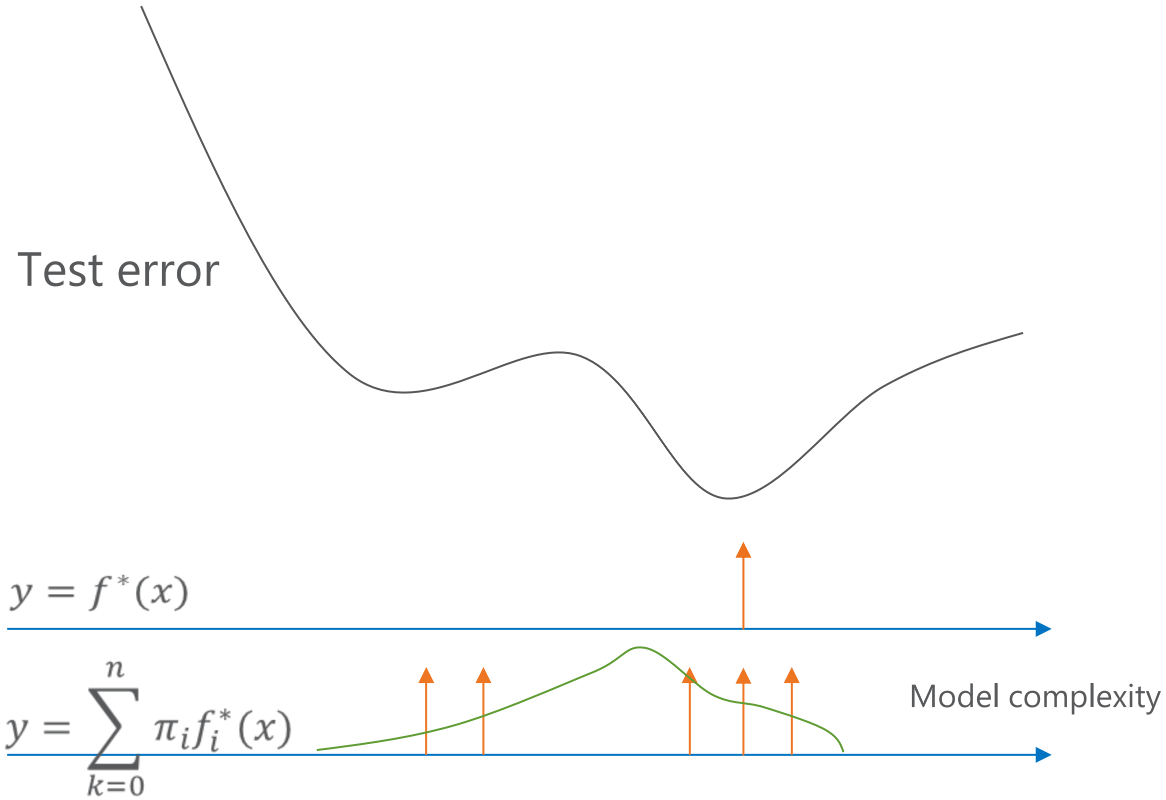
w0

1. LASSO

Basic Model Assumptions

$$\boxed{y = f(X)}$$

- Two approaches to estimate f
  - Underlying model style (*)
  - Bayesian style



Ref: Chapter 5. Deep Learning. Ian Goodfellow, Yoshua Bengio and Aaron Courville

Linear Regression

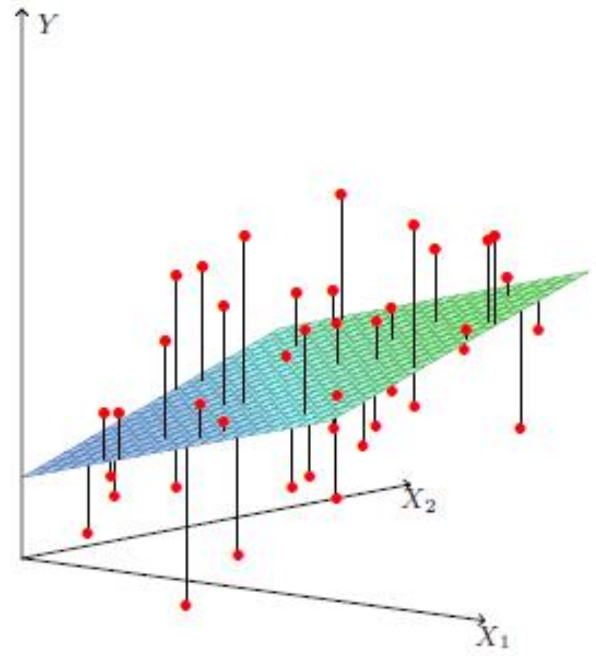
- Assumption: $f(X) = B_0 + \sum_{j=1}^{p} X_j B_j$

To find B

$$J_B = (y - XB)^T (y - XB)$$

$$\frac{\partial J_B}{\partial B} = -2X^T (y - XB)$$

$$\hat{B} = (X^T X)^{-1} X^T y$$



Ridge Regression

$J_B = (y - XB)^T(y - XB) + \lambda B^T B$

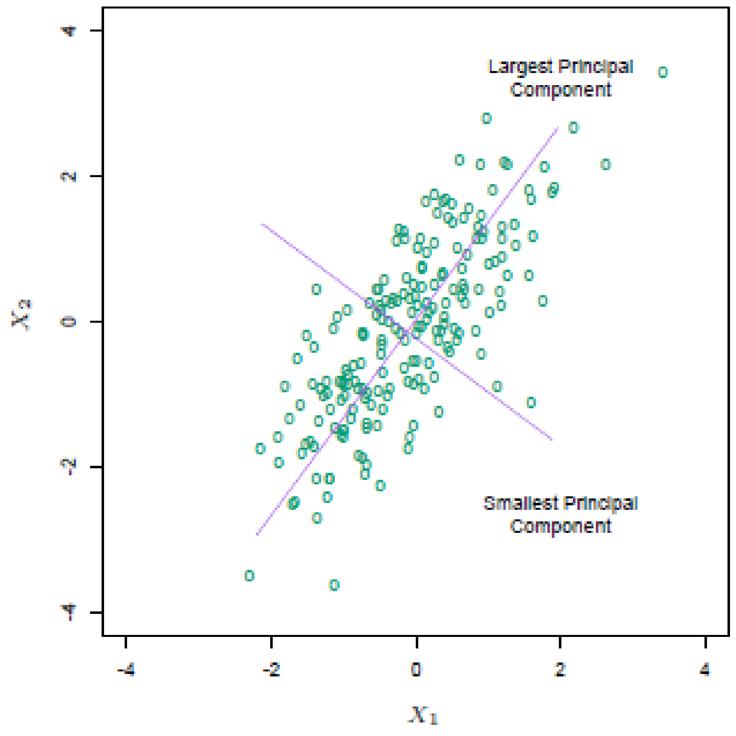
$\hat{B} = (X^T X + \lambda I)^{-1} X^T y$

With $X = UDV^T$

$$\hat{B} = V(D^2 + \lambda I)^{-1} D U^T y$$

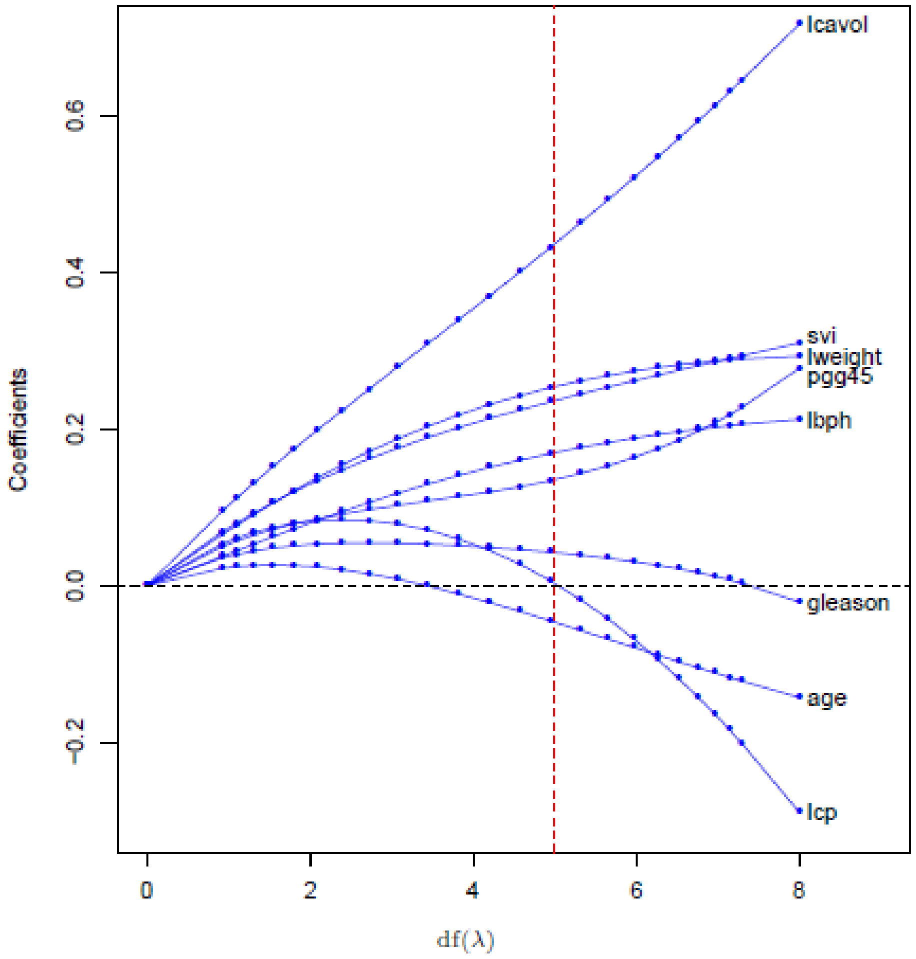
$$\hat{y} = X\hat{B} = \sum_{j=1}^{p} u_j \left(\frac{d_j^2}{d_j^2 + \lambda} \right) u_j^T y$$

u_j columns of U, d_j diagonal elements of D



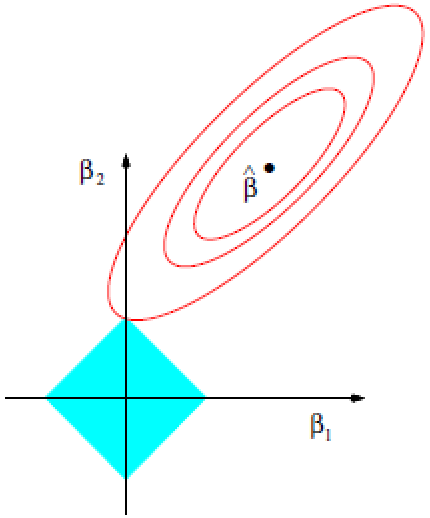
Ref: SVD Tutorial

Ridge Regression

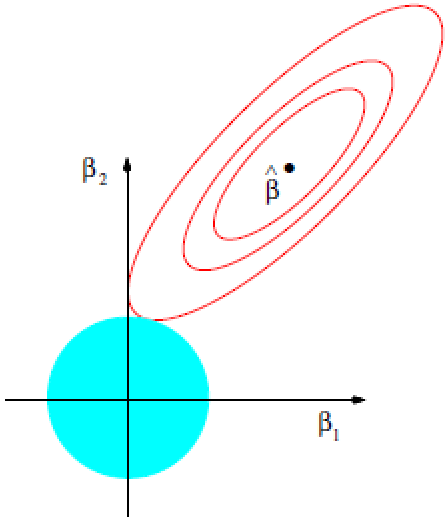


Lasso Regression

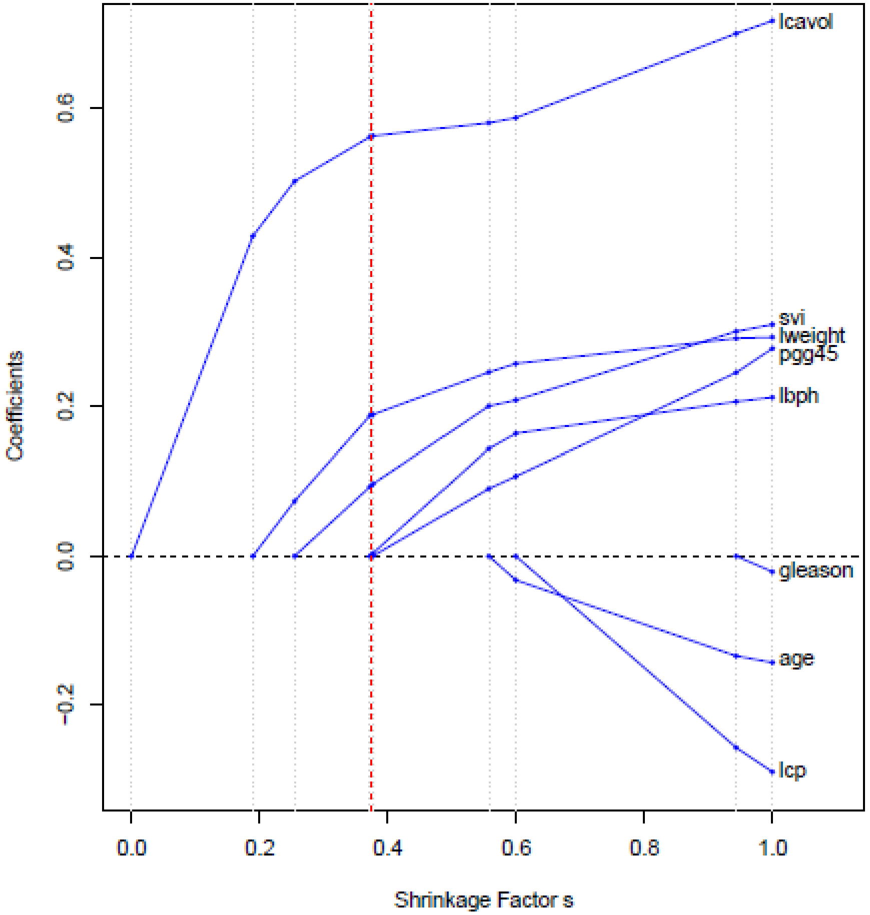
- $J_B = (y - XB)^T(y - XB) + \lambda \sum_{j=1}^{p} |B_j|$



Lasso

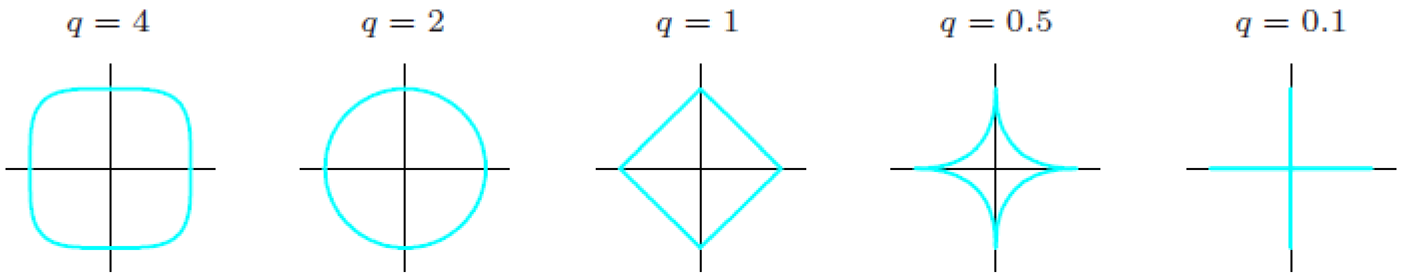


Ridge



G-Formula

$$J_B = (y - XB)^T (y - XB) + \lambda \sum_{j=1}^{p} |B_j|^q$$

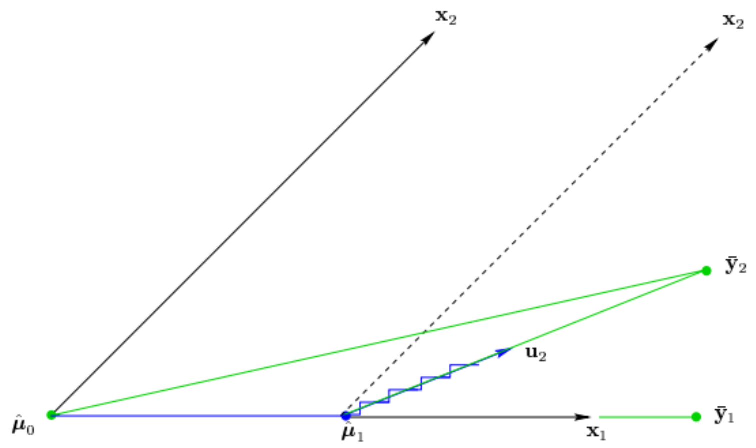


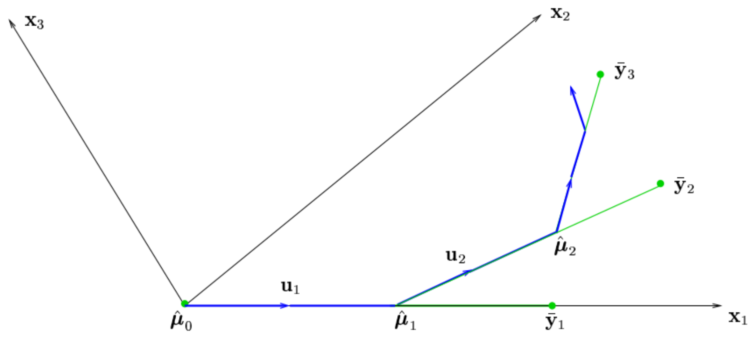
Elastic Net

Lasso Solution – Least Angle Regression (LAR) *

Algorithm 3.2 *Least Angle Regression.*

1. Standardize the predictors to have mean zero and unit norm. Start with the residual $\mathbf{r} = \mathbf{y} - \bar{\mathbf{y}}$, $\beta_1, \beta_2, \ldots, \beta_p = 0$.
2. Find the predictor $\mathbf{x}_j$ most correlated with $\mathbf{r}$.
3. Move β_j from 0 towards its least-squares coefficient $\langle \mathbf{x}_j, \mathbf{r} \rangle$, until some other competitor $\mathbf{x}_k$ has as much correlation with the current residual as does $\mathbf{x}_j$.
4. Move β_j and β_k in the direction defined by their joint least squares coefficient of the current residual on $(\mathbf{x}_j, \mathbf{x}_k)$, until some other competitor $\mathbf{x}_l$ has as much correlation with the current residual.
5. Continue in this way until all p predictors have been entered. After $\min(N-1, p)$ steps, we arrive at the full least-squares solution.



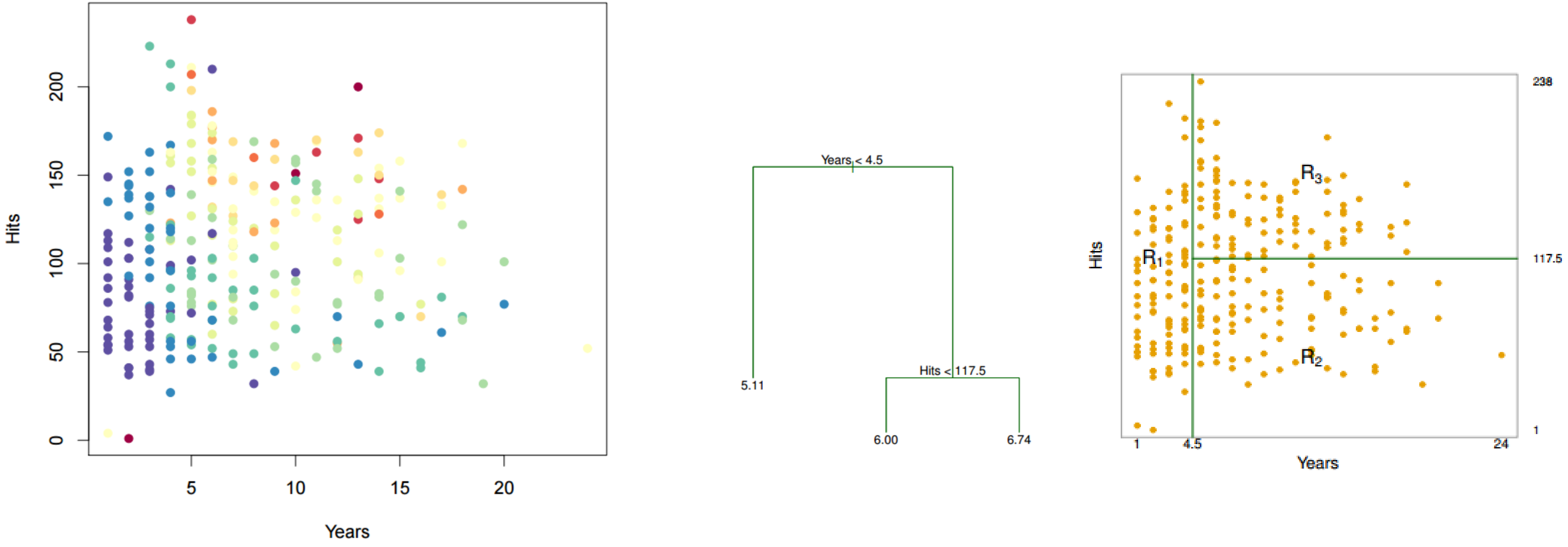


Questions?

2. RANDOM FOREST

Decision Tree - Baseball Player Salary

Salary is color-coded from low (blue, green) to high (yellow,red)



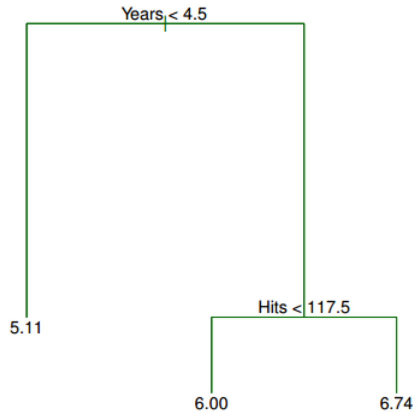
Decision Tree

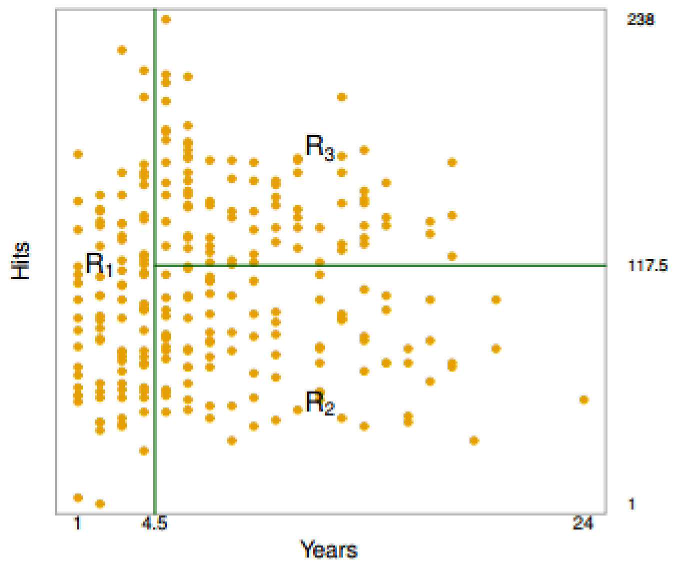
An approach that is known as recursive binary splitting

✦**Top-down**,

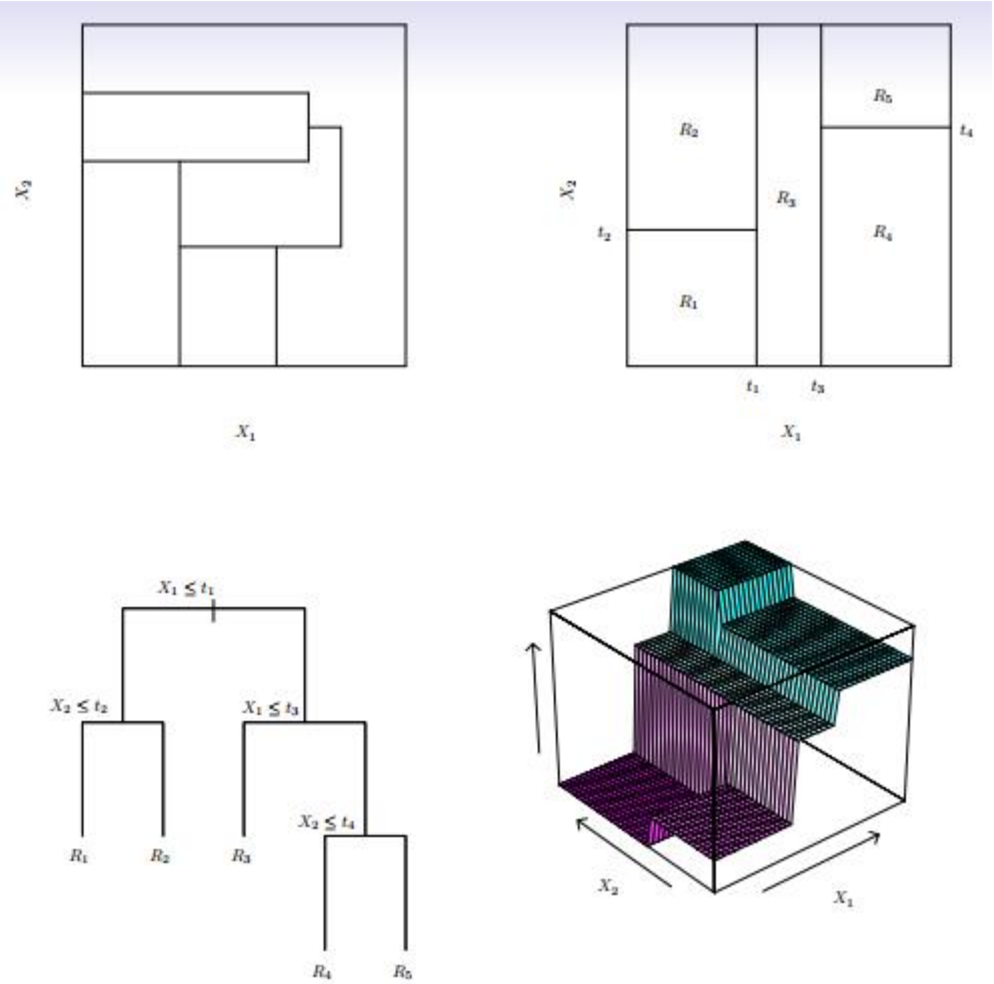
✦**Greedy**

Predict the response for a given test observation using the mean of the training observations in the region

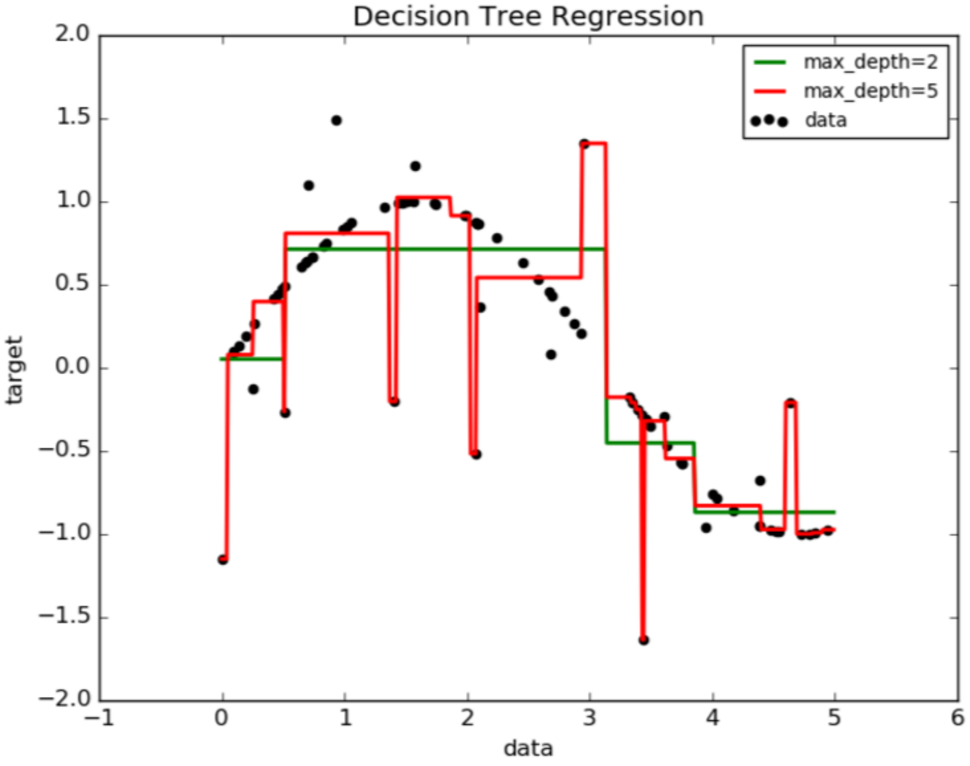




Tree Complexity and Overfitting



Simple Tree



Complex Tree

Tree Pruning

- Motivation: Simple tree is too biased. Complex tree is overfitting.
- Naïve solution: grow the tree only so long as the decrease in the Residual-Sum-of-Square due to each split exceeds some (high) threshold.
  - Result in smaller trees, but is too **short-sighted**: a seemingly worthless split early on in the tree might be followed by a very good split

XOR

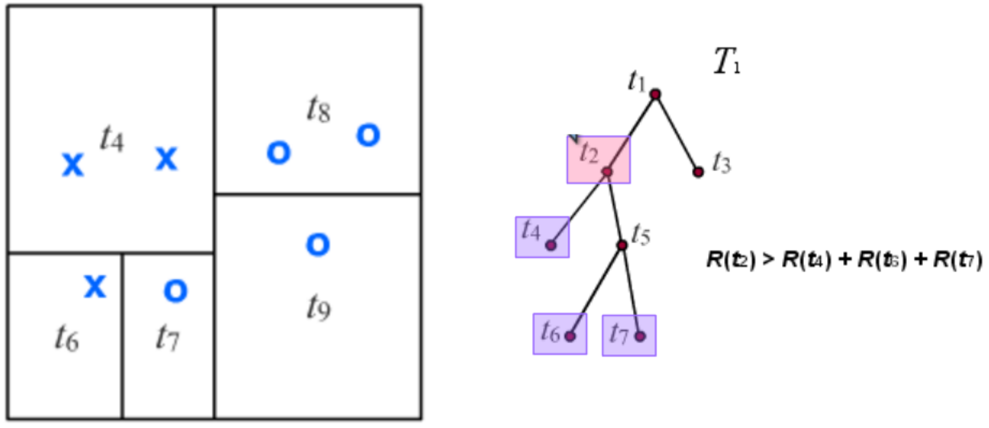
P	Q	P *xor* Q
1	1	0
1	0	1
0	1	1
0	0	0

Error not reduced

But it WILL be reduced

Tree Pruning

- Motivation: Simple tree is too biased. Complex tree is overfitting.
- Better solution: grow a large tree, then merge back nodes to obtain a smaller tree of the right size link



0.25 > 0 + 0 + 0

Cost complexity pruning (or Weakest link pruning) *

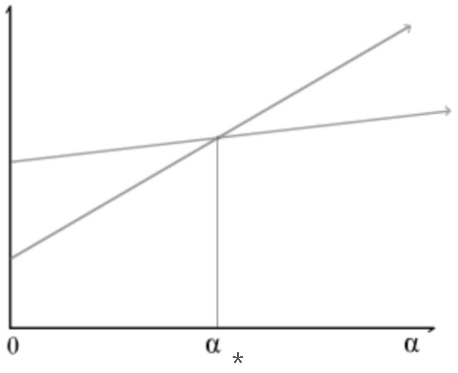
- Define T_t a branch rooted at a node t, and and $\tilde{T}_t$ is its set of terminal nodes.

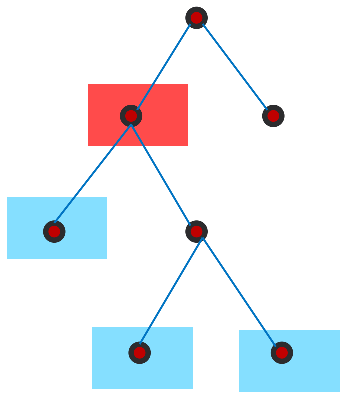
- Let α be a regularization parameter.

$$R_\alpha(t) = R(t) + \alpha * 1$$

$$R_\alpha(T_t) = \sum_{t' \text{ in } \tilde{T}_t} R(t') + \alpha * |\tilde{T}_t|$$

$$\alpha_* = \frac{R(t) - R(T_t)}{|\tilde{T}_t| - 1}$$





Cost complexity pruning (or Weakest link pruning) *

1. Construct a large tree T_0.

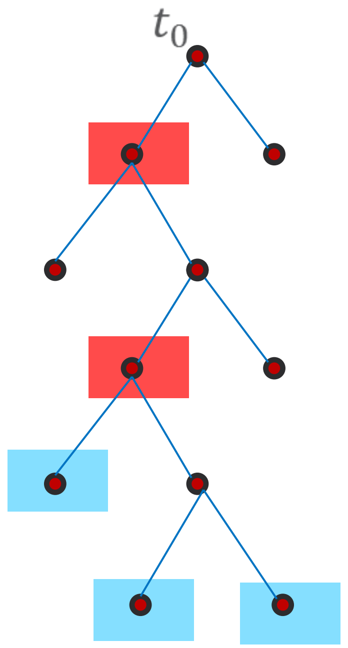
2. Find a node t that minimizes the function $g(\mathrm{t}) = \frac{R(t)-R(T_t)}{|\tilde{T}_t|-1}$. Let $\alpha_1 = \mathrm{g}(t)$ and remove all sub-nodes under t to produce T_1.

3. Repeat step 2 to find two sequences

$\alpha_1 < \alpha_2 < \alpha_3 < \cdots < \alpha_{|T|}$

$T_1 > T_2 > T_3 > \cdots > t_0$

Note: $R(t)$ can be stored.



Final Decision Tree

Algorithm 8.1 *Building a Regression Tree*

1. Use recursive binary splitting to grow a large tree on the training data, stopping only when each terminal node has fewer than some minimum number of observations.

2. Apply cost complexity pruning to the large tree in order to obtain a sequence of best subtrees, as a function of α.

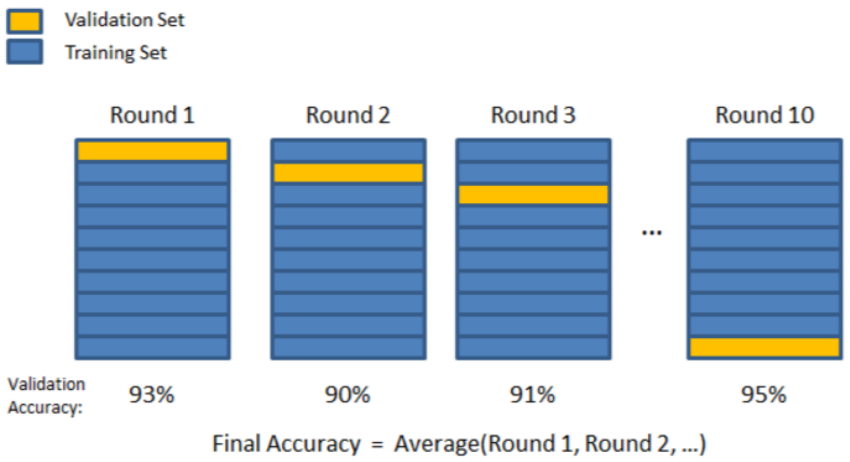
3. Use K-fold cross-validation to choose α. That is, divide the training observations into K folds. For each $k = 1, \ldots, K$:

   (a) Repeat Steps 1 and 2 on all but the kth fold of the training data.

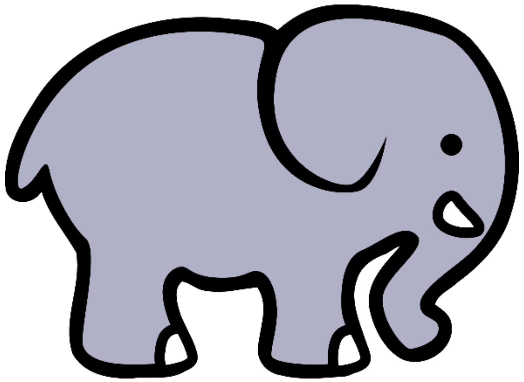
   (b) Evaluate the mean squared prediction error on the data in the left-out kth fold, as a function of α.

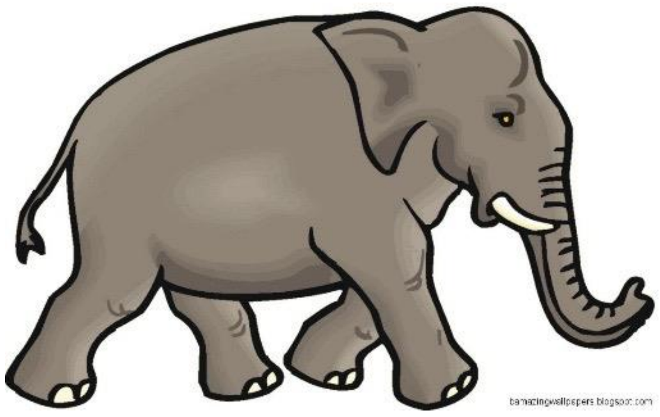
   Average the results for each value of α, and pick α to minimize the average error.

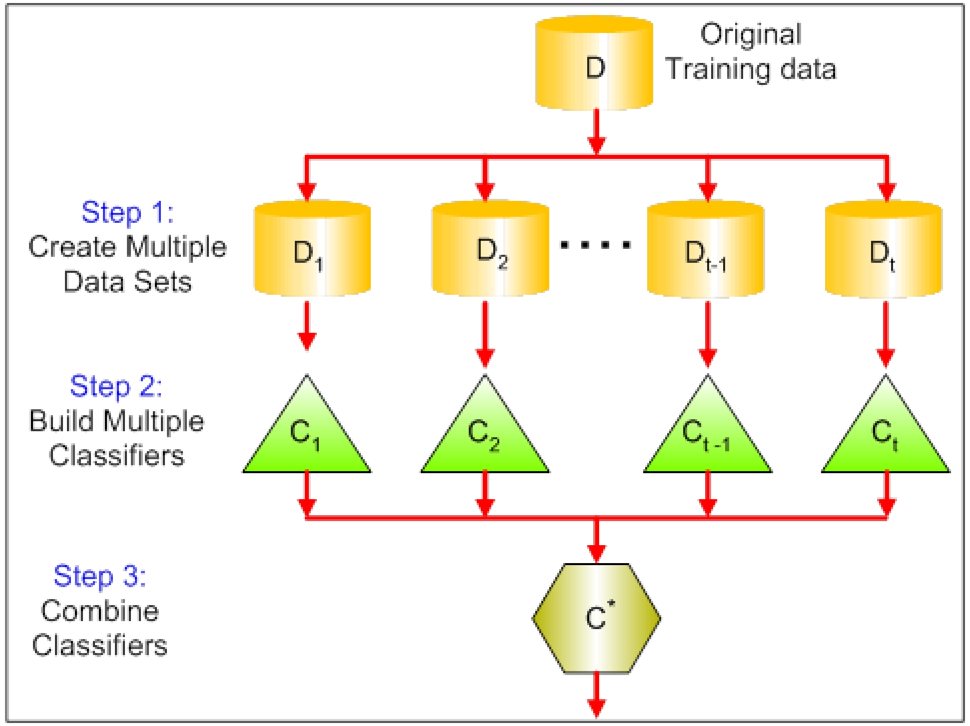
4. Return the subtree from Step 2 that corresponds to the chosen value of α.



Bagging





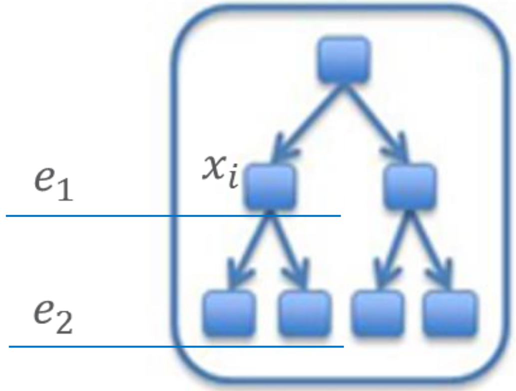

D
Original Training data
Step 1:
Create Multiple Data Sets
D_1
D_2
D_{t-1}
D_t
Step 2:
Build Multiple Classifiers
C_1
C_2
C_{t-1}
C_t
Step 3:
Combine Classifiers
C^*

Bagging – Prediction

- Averaging predictions from multiple trees, each is constructed on one part of a training data set. Famous method: Bootstrap (sampling with replacement).

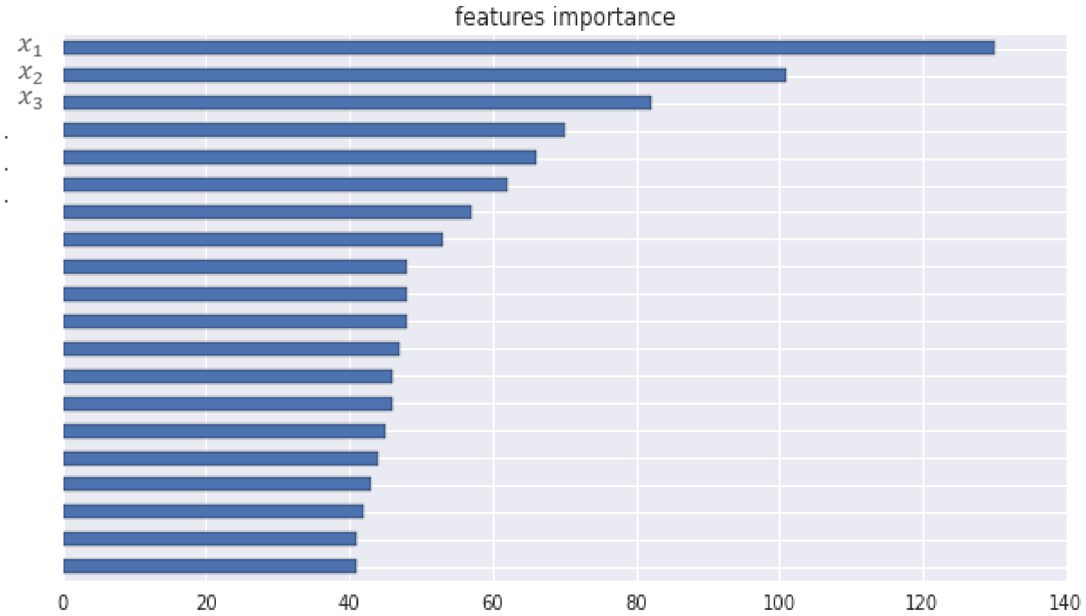
Bagging – Variable Importance

At each tree

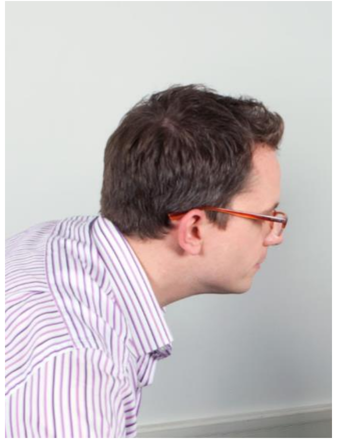


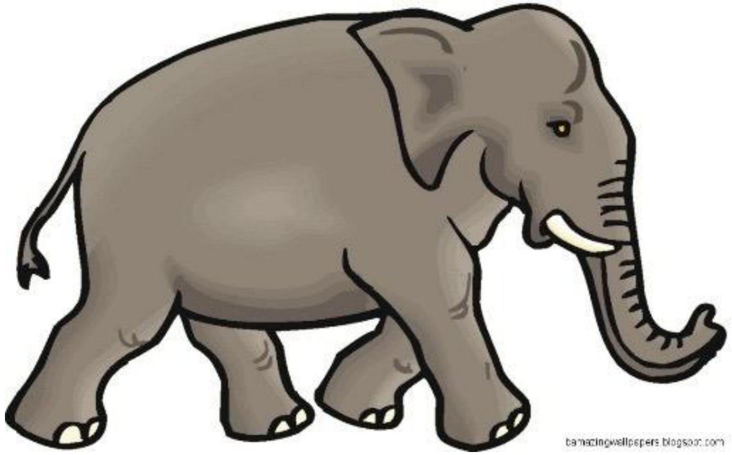
Importance of x_i
$= e_1 - e_2$

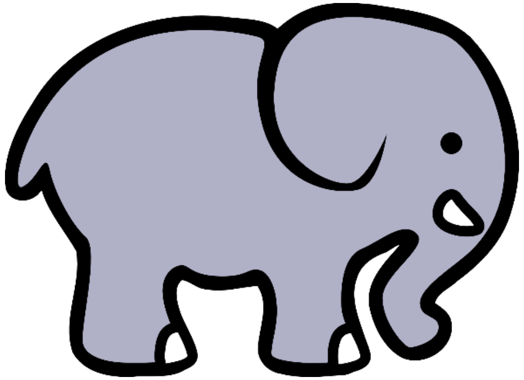
Sum-up



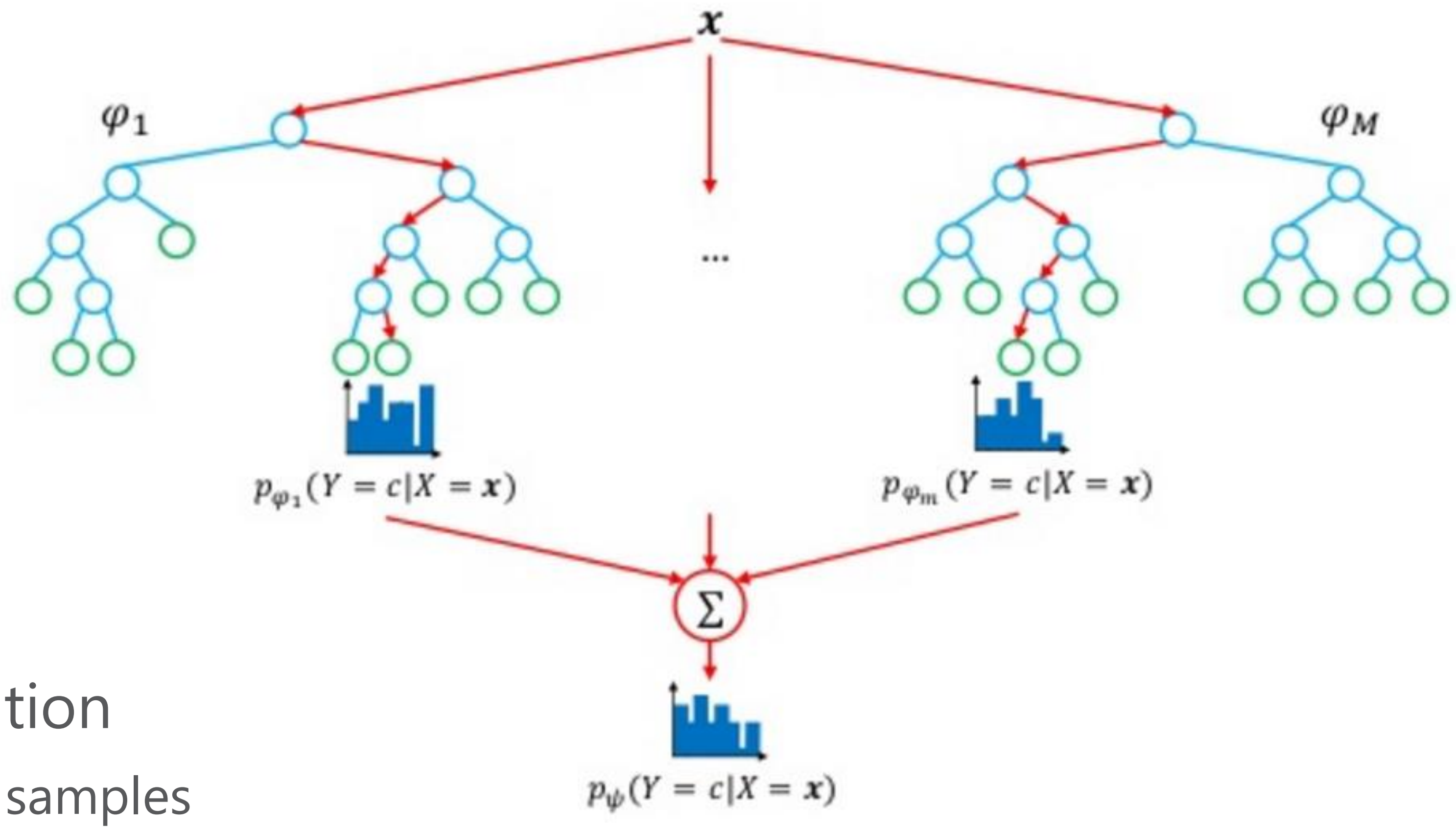
Random Forest







Random Forest



- Randomization
  - Bootstrap samples
  - Feature subsets are different for these trees