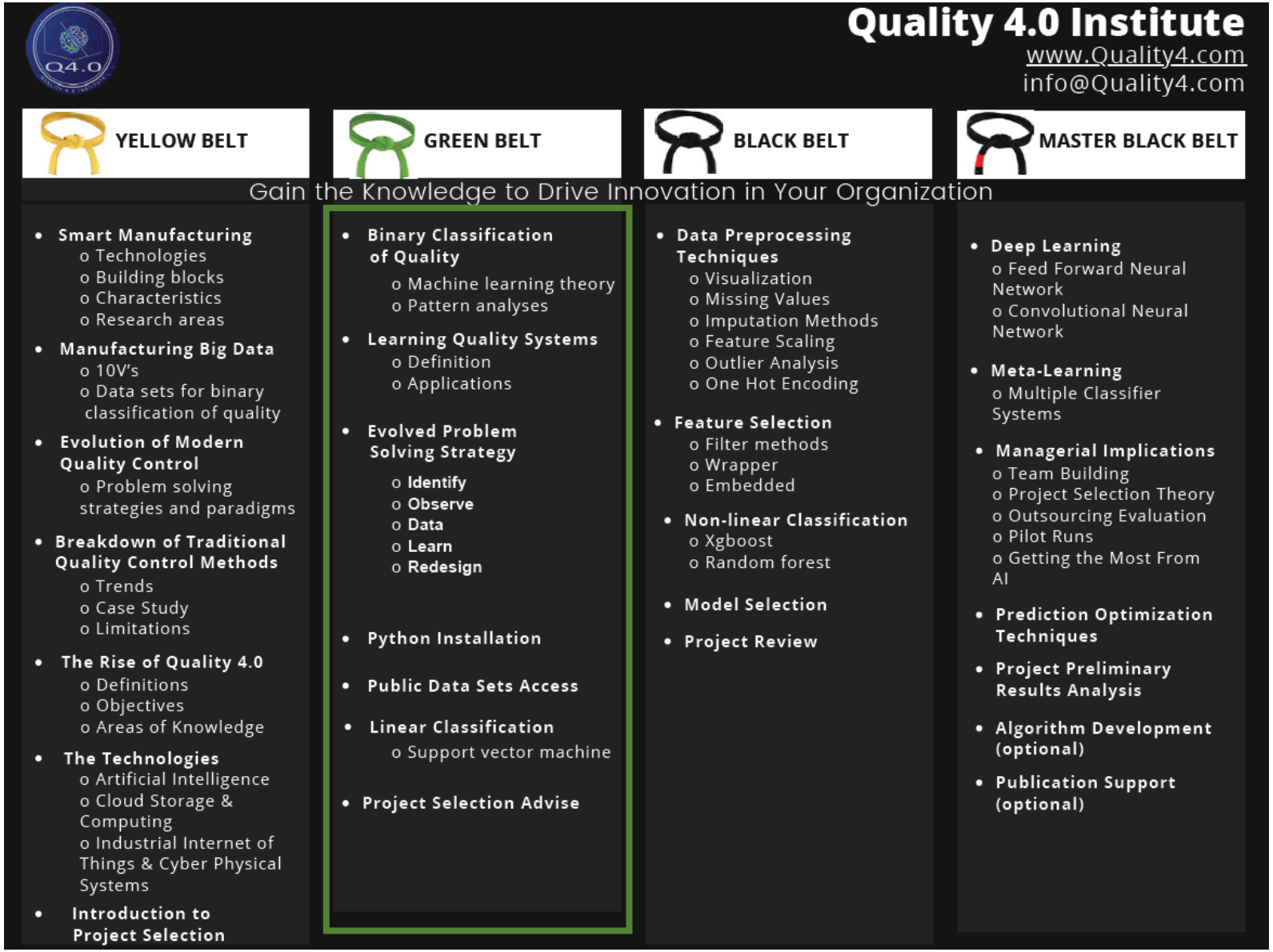
Green belt training

Carlos Escobar, PhD

Quality 4.0 Institute

Quality 4.0: Artificial intelligence for manufacturing innovation

About the program



Chapter 4: Classification

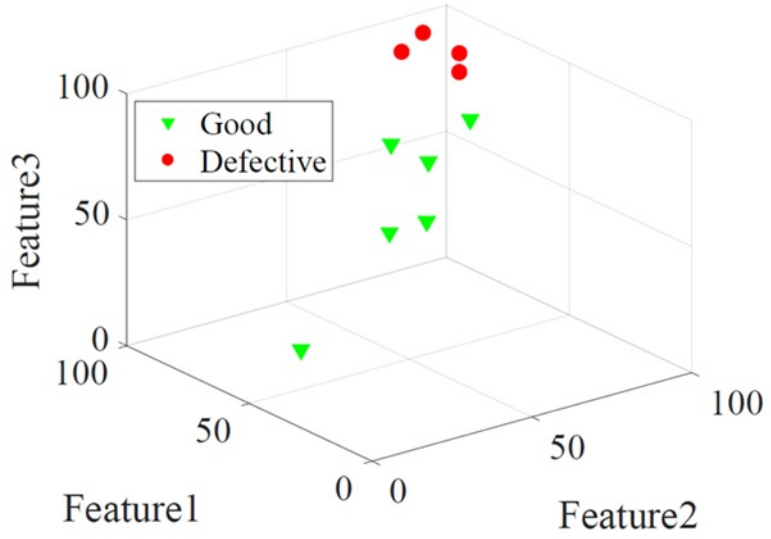
- Binary Classification
  - Classification
  - Binary Classification
  - Errors
    - Alpha and Beta Errors
    - Metrics
  - Case Study
  - Binary Patterns
    - Discriminative information

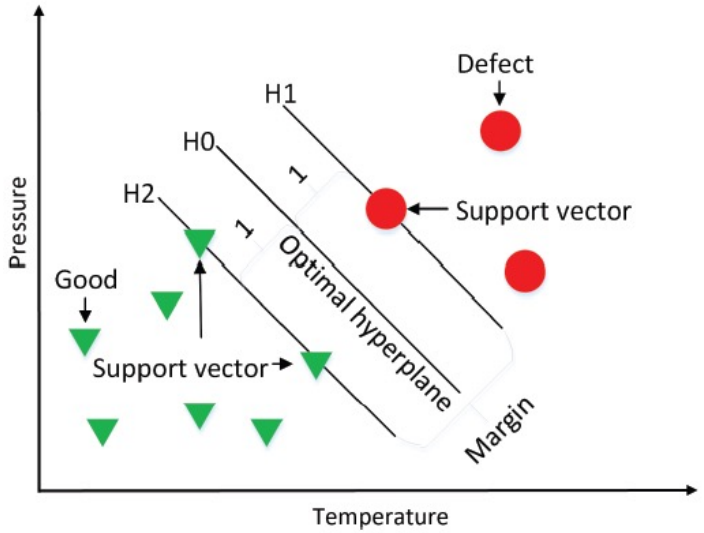
Case Study 1
Case Study 2

FIGURE 4.1 Contents of the Chapter 4

Classification

- Classification (pattern recognition), supervised learning task that uses MLAs to learn patterns from labeled data to automatically assign a class label to new samples.
- Patterns commonly exist in the hyper-dimensional space.

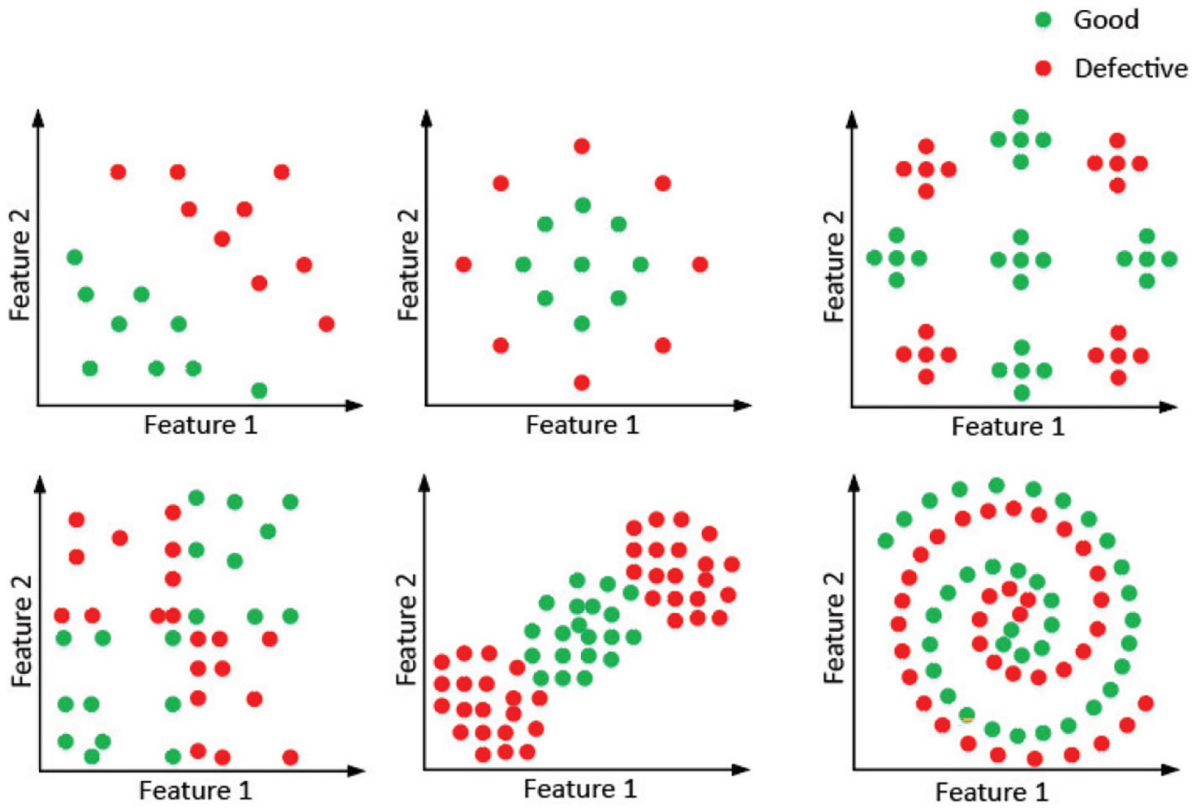




- Geometric problem, project items into the hyper-dimensional space where the classes separate.

Classification

- Patterns can take any form
- Pattern can exist in any dimensional space



Classification

Definitions

- Labeled data, supervised learning

$$l_i = \begin{cases} 1 & \text{if } i^{th} \text{ item is } defective\ (+) \\ 0 & \text{if } i^{th} \text{ item is } good\ (-) \end{cases}$$

- Model, developed based on historical samples (X, l) and MLAs.
  - Where X refers to the features, images, or signals used to train the MLA and l their associated quality label
  - Predicts $\hat{y}$, the probability of a positive result.

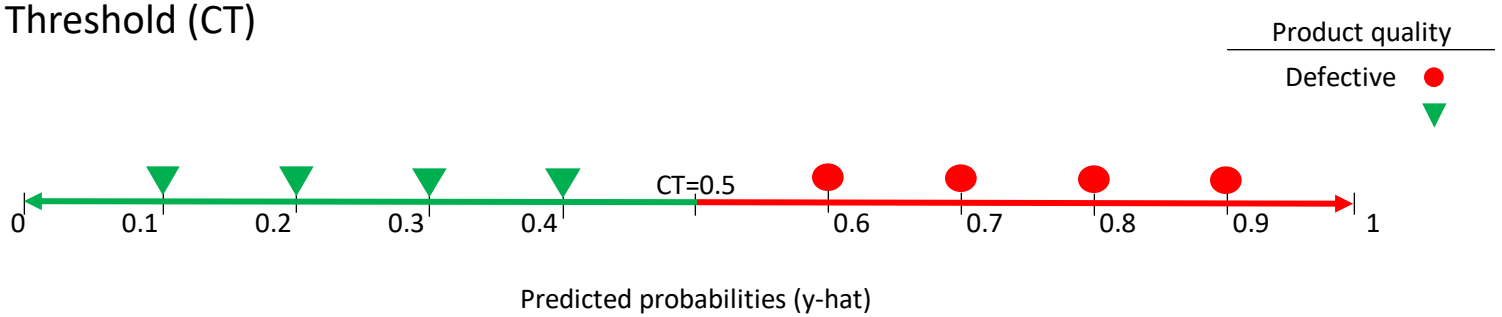
$$\hat{y} = f(X;\theta) = P(l = defective)$$

Classification

- CT, classification rule
  - Defined considering the business goals

$$l_i = \begin{cases} 1 & \text{if } \hat{y}_i => CT \text{ item is } defective\ (+) \\ 0 & \text{if } \hat{y}_i < CT \text{ item is } good\ (-) \end{cases}$$

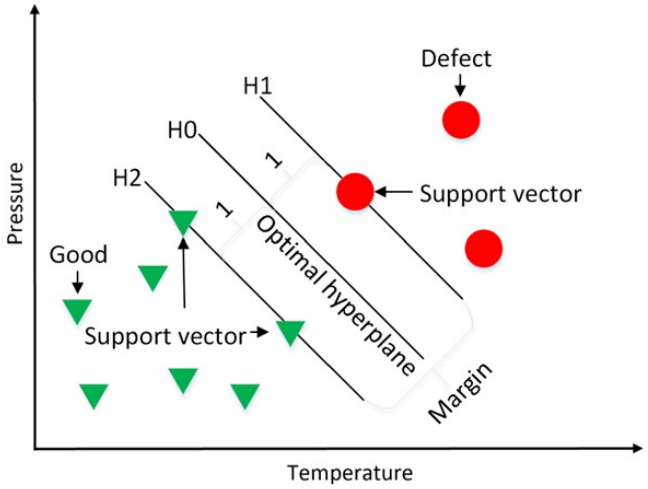
- Classifier, discrete-valued function that assigns a class label to a particular item based on the intrinsic patterns in the features.
  - Model
  - Classification Threshold (CT)

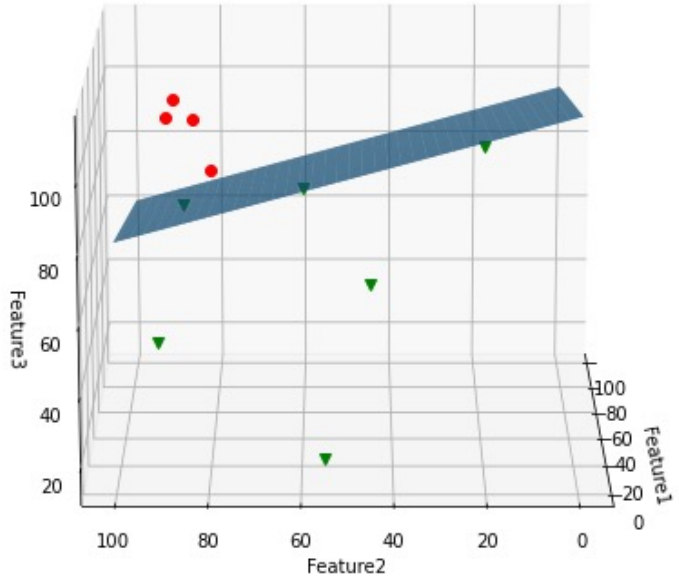


- Generalization, prediction ability on unseen data.

Classification

- Feature vector (independent variable), observable variable of the phenomenon being investigated.
- Separating hyperplane (classification boundary), separates the classes (good, defective)
  - Subspace whose dimension is one less than that of its ambient space.
  - E.g., the separating hyperplane of a 2-dimensional space is a line.
- Hyperparameters, user-defined parameter that control the learning process and highly affects the prediction performance (i.e., number of neurons, trees, neighbors)

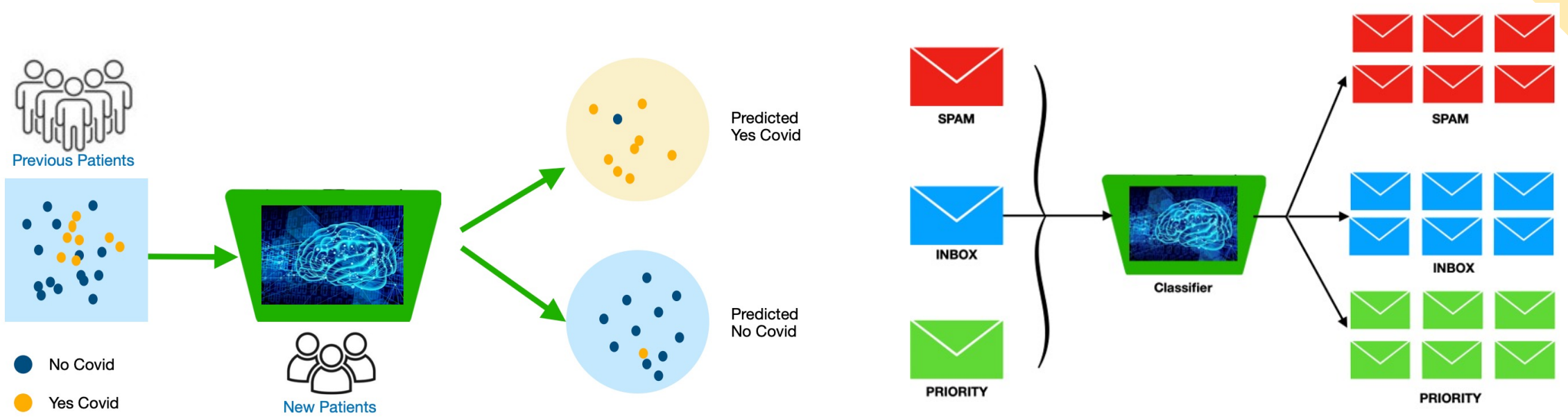




Classification

Applications

- Binary classification vs multi-class classification



- Classes must be well characterized for a MLA to learn the classification boundary

Classification

- Classification is suitable for modeling non-stochastic or deterministic relationships between inputs (features) and outputs (classes).
- Should not be used when two items with identical inputs can easily have different outcomes, in such cases a probabilistic study is better.

Classification

Neural networks (shallow)

- Type of machine learning model inspired by the structure and function of the human brain.
- Composed of interconnected nodes called neurons or "units" that work together to process and learn complex patterns in the training data.
- Organized into layers
  - Input layer that receives the initial data
  - One hidden layers that perform intermediate computations
  - Output layer that produces the prediction, $\hat{y}$.

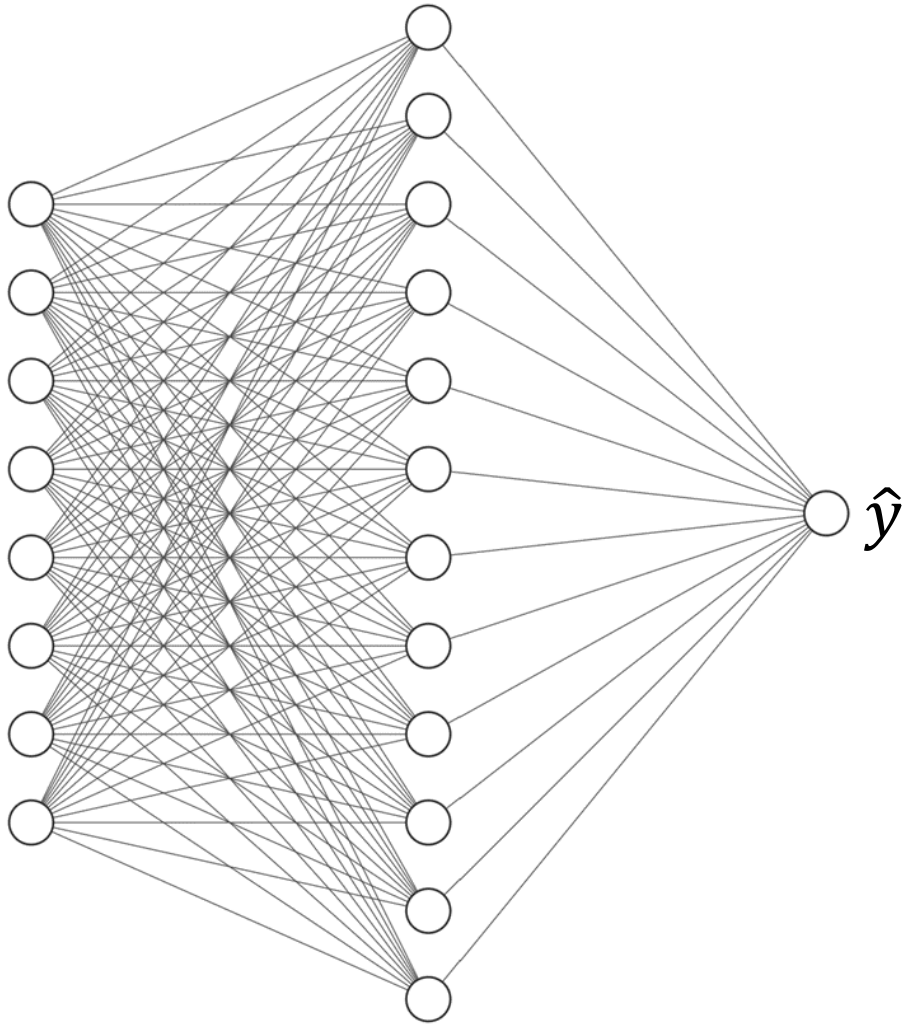


Image created at https://alexlenail.me/NN-SVG/

Classification

Neural networks (deep learning)

“subset of machine learning, which is essentially a neural network with three or more layers”. IBM

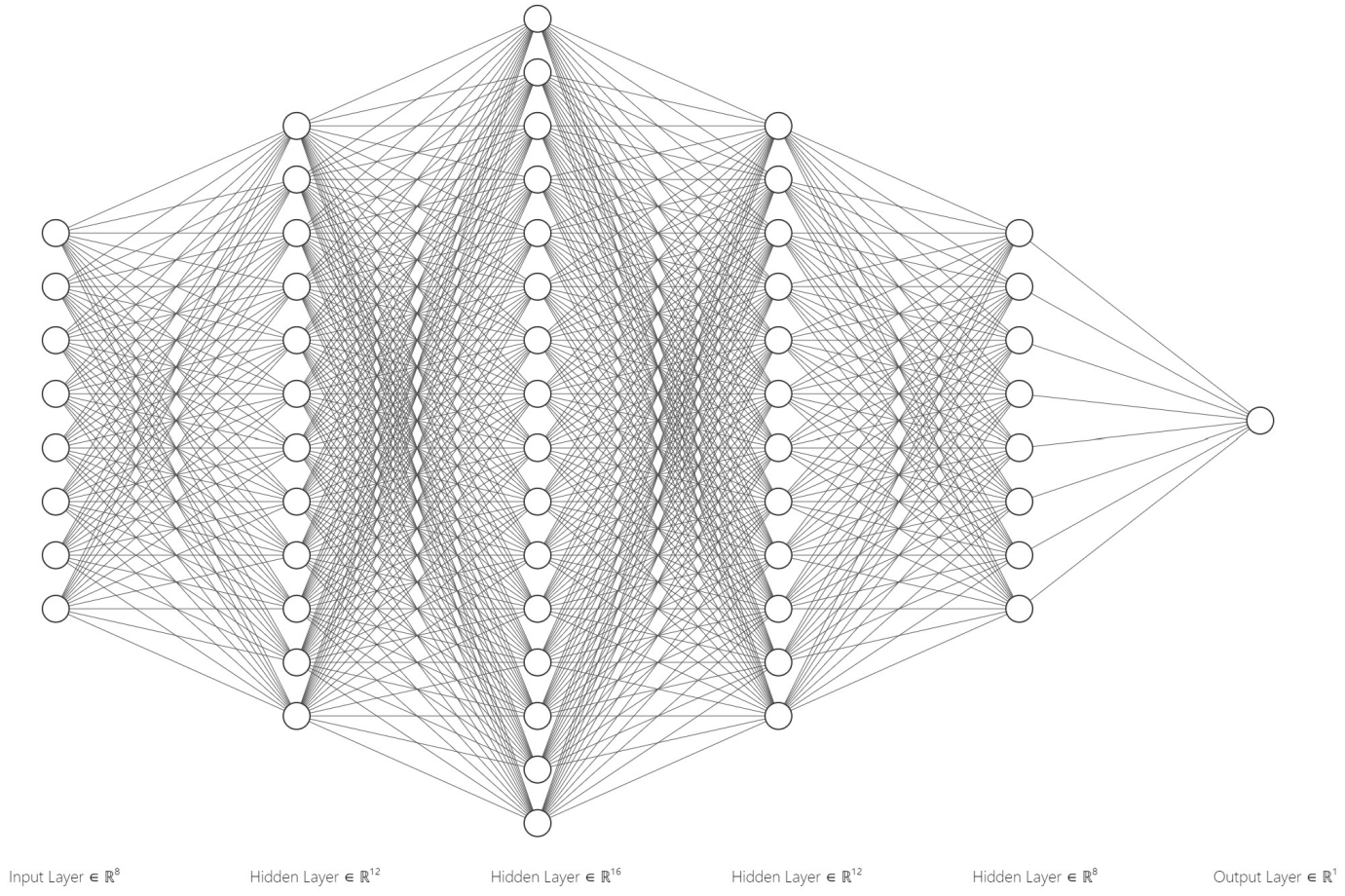


Image created at https://alexlenail.me/NN-SVG/

Classification

Deep Learning (DL)

- "Big data era" enables new innovations using DL
- Emergent technology is gaining much popularity due to its supremacy in terms of learning complex patterns when trained with huge amounts of data.

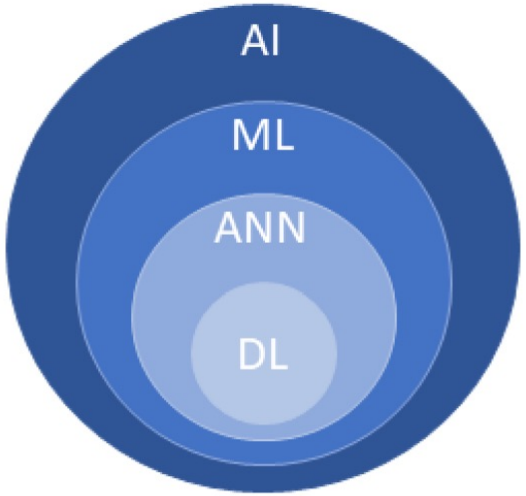
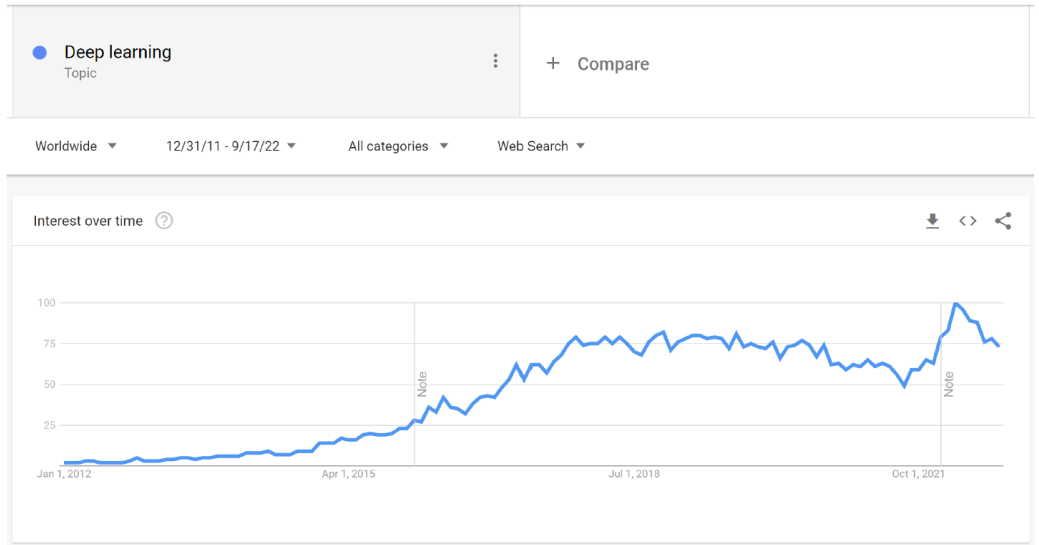


FIGURE 4.5 *AI*, *ML*, and *DL* in perspective.



Classification

- DL, specifically CNNs, can change the status quo of visual monitoring in manufacturing. This is very important, as human-based monitoring systems are still widely used in manufacturing.

Deep learning is covered in the master black belt certification

Classification

Traditional MLAs:

- Work better with simple problems with limited amount of data
- Need manual feature creation and more visible patterns

DL learns patterns from unstructured (i.e., images, sound) and structured data

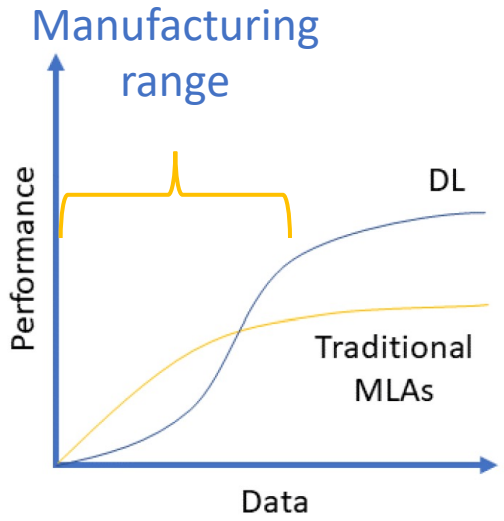


FIGURE 4.6 Performance of *DL* and traditional *MLAs* wrt the data size used to train them[3].

Classification

Important considerations

- Be aware of the law of the instrument, aka Maslow's hammer, a cognitive bias that involves an over-reliance on a familiar tool.
  - Study the problem before blindingly selecting a DL approach.
  - Most of today's problems are more effectively solved by traditional MLAs rather than DL.

Source: Yavar Bathaee. The artificial intelligence black box and the failure of intent and causation. Harv. JL & Tech., 31:889, 2017.

Classification

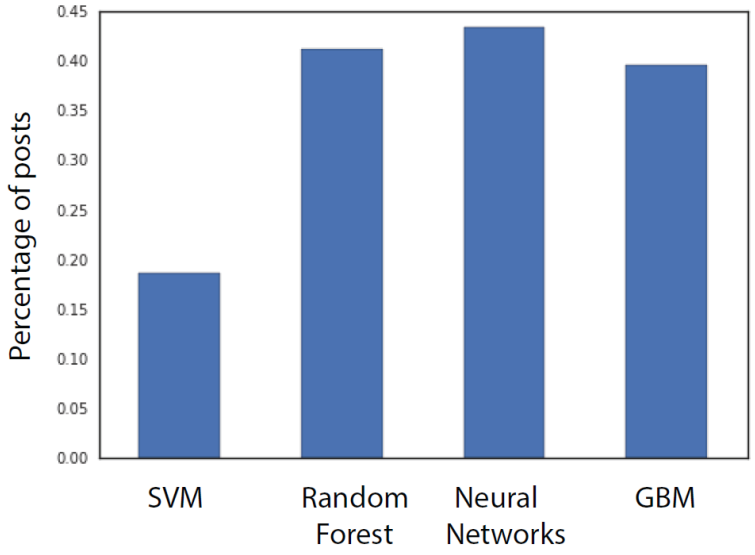
TABLE 4.1 Summary of considerations for selecting either *DL* or traditoinal *MLAs*.

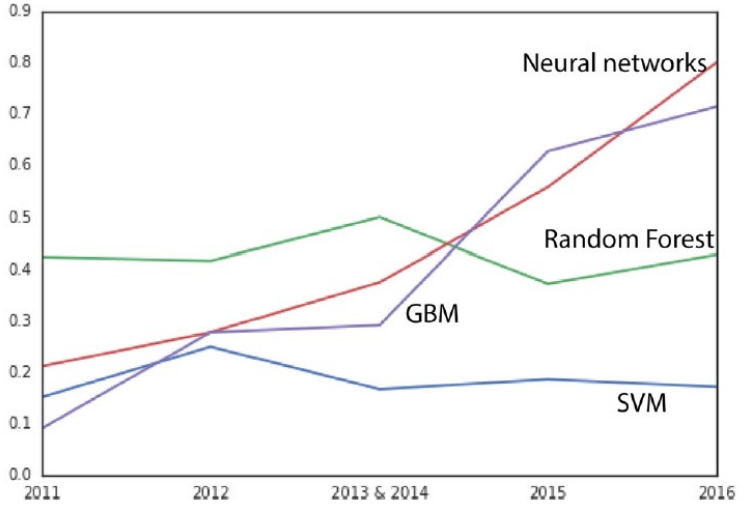
	Traditional MLAs	DL
Training data	Preferable for small data sets	Better for large data sets
Training time	Faster (seconds to hours)	Longer (days to weeks)
Computing power	Scalable computing	Computationally intensive
Interpretability	High (for some *MLAs*	Very low
Feature engineering	Manual process based on domain knowledge	Automatically through it's hidden layer architecture
Image classification	Not recommended	Convolutional neural networks are very accurate

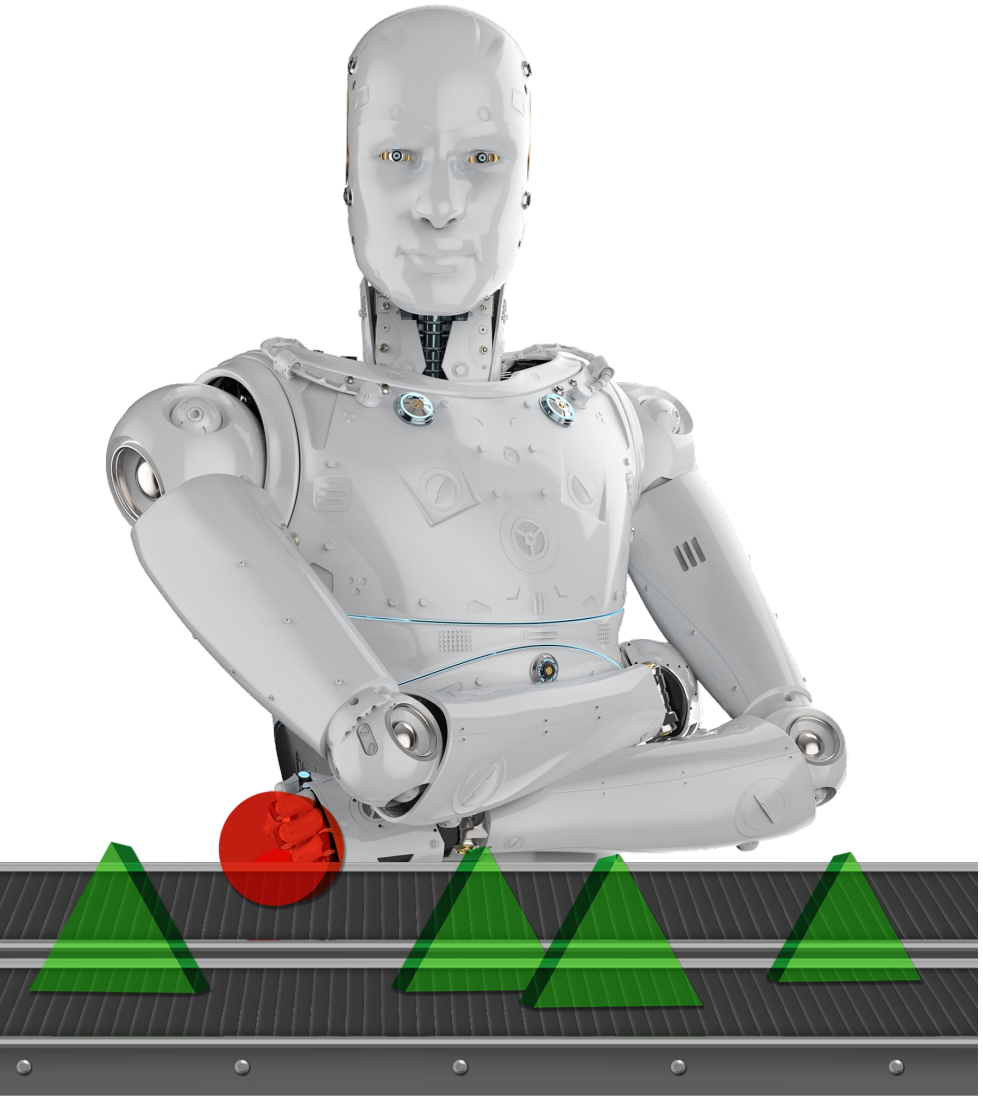
Classification

Top performer MLAs

- "No Free Lunch", no machine learning algorithm works best for every problem.
  - We need to test several MLAs
- Empirical results based on numerous models developed in Kaggle competitions, report the superiority of the Random Forest (*RF*), *SVM*, *XGBoost*, and *ANN* in solving real problems.
  - *XGBoost* is the top performer for problems involving structured data.
  - *DL* is for unstructured data.





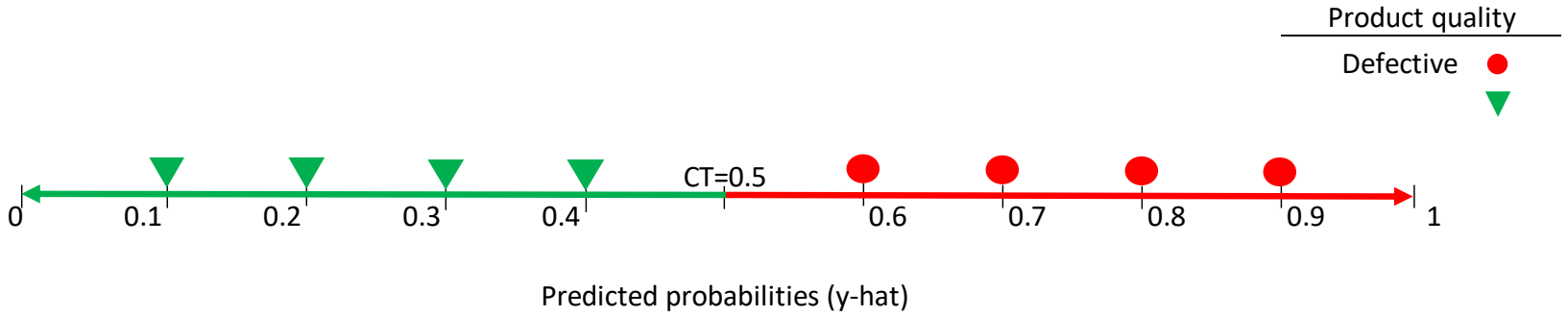


Classification errors

Classification errors

- Complete separation of classes
  - Eight samples
  - Four good quality
  - Four defective
  - All of them are correctly classified (based on the classification rule)

$$l_i = \begin{cases} 1 & \text{if } \hat{y}_i => CT \text{ item is } defective\ (+) \\ 0 & \text{if } \hat{y}_i < CT \text{ item is } good\ (-) \end{cases}$$



Classification errors

Confusion Matrix

- Prediction is performed under uncertainty
- Data-driven methods are not 100%, they commit errors
- Effect of these errors needs to be understood and quantified during the learning process
- Table with two rows and two columns that summarizes the prediction ability of a classifier

Table 3: Confusion matrix.

	Predicted good	Predicted defective
Good item	True Negative (TN)	False Positive (FP)
Defective item	False Negative (FN)	True Positive (TP)

Reduce efficiency
(reevaluation/rework – hidden factory)

Generate
warranty events

Main concern

Classification errors

- True Negative (TN), good quality item, correctly predicted good quality by the classifier.
- True Positive (TP), defective item, correctly predicted defective.
- False Positive (FP), good item, incorrectly predicted defective.
  - α (type-I) error
- False Negative (FN), defective item, incorrectly predicted good
  - β (type-II) error

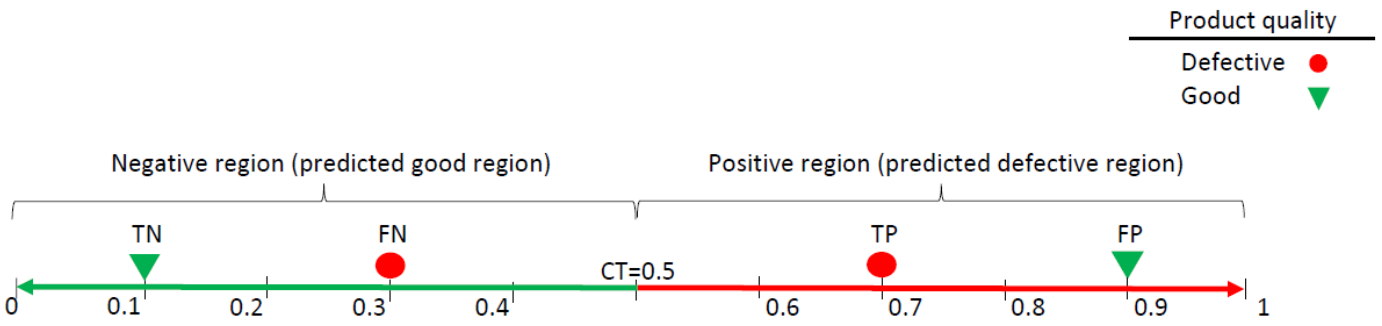


FIGURE 4.8 Concepts of the confusion matrix based on a $CT = 0.5$

Classification errors

Alpha and beta errors and their implications

- Both errors (α and β) are important.
- Their implications must be considered when defining the CT.
- β error or the false negative rate describes the detection ability of the classifier.
- α error or false positive rate describes how efficient is the classifier is at detecting the defective items.

$$\alpha = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}$$

$$\beta = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TP}}$$

Classification errors

- Statistically, the two types of error rates pose a trade-off.
  - Errors cannot be simultaneously minimized.
- FNs or failing to detect a defective item, results in a warranty event.
- FPs create the hidden factory effect (rework, scrap).

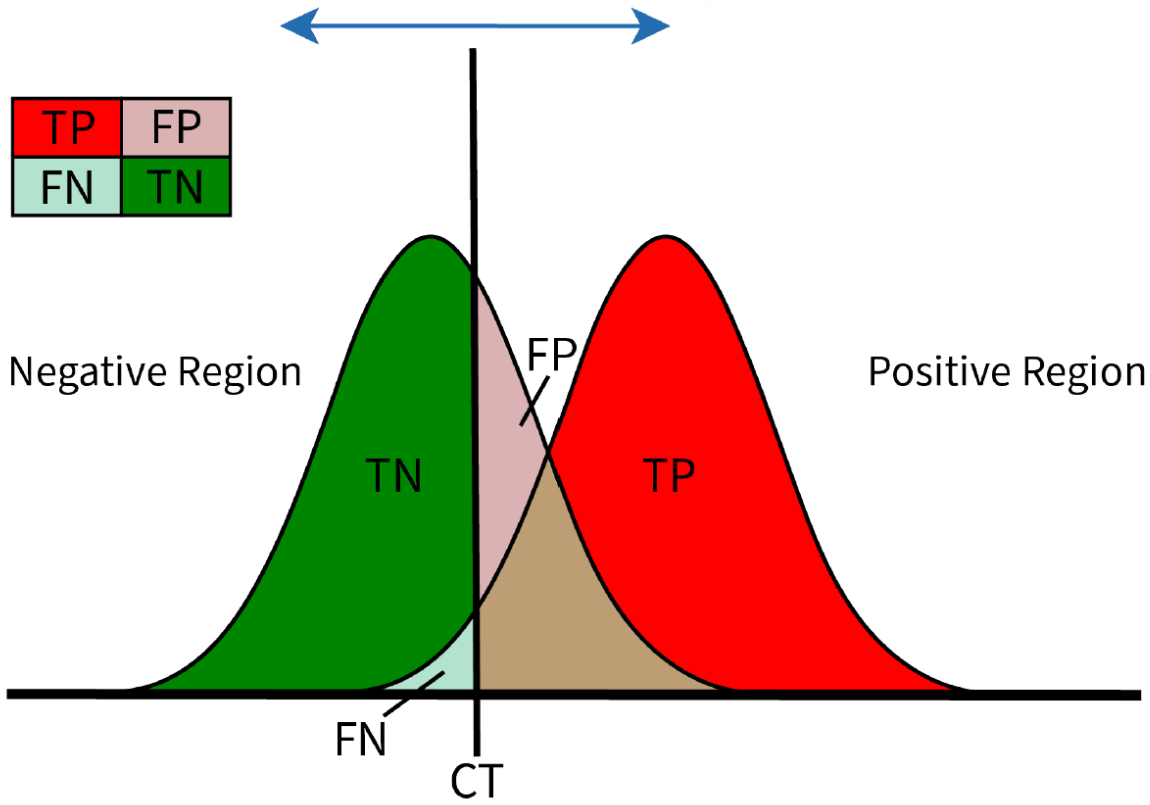


FIGURE 4.9 α and β trade-off

Classification errors

Metric of classification performance

G-means

- Based on the alpha and beta errors
- Solves the alpha and beta tradeoff by assigning higher importance FNs (beta error) when classes are highly unbalanced
- "Probabilistic−based measure“
- $G - means \, \epsilon \, [0,1]$

$$G - means = \sqrt{(1-\alpha) \times (1-\beta)}$$

- Value of 1 describes complete separation

- Value of 0 usually fails to detect all the defects

Classification errors

Accuracy

- Describes the percentage of correct classifications
  - Number of times the ML model was overall correct.
- Useful in the case of balanced dataset and equally important classes.
- Misleading performance metric for highly imbalanced data:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

- Value of 1 describes complete separation

Activity 1

Compute the MPCD of each scenario, 10mins

- Each scenario has 5,005 samples
  - 5000 good
  - 5 defects
- Red circles represent one sample
- Green triangles may represent multiple samples (e.g., x1000 means one thousand samples)

Scenario	TN	TP	FN	FP	α	β	Accuracy	G-means
1	5000	5	0	0				
2	5000	4	1	0				
3	4999	5	0	1				
4	4998	5	0	2				
5	5000	3	2	0				
6	4999	4	1	1				
7	4000	5	0	1000				
8	5000	0	5	0				

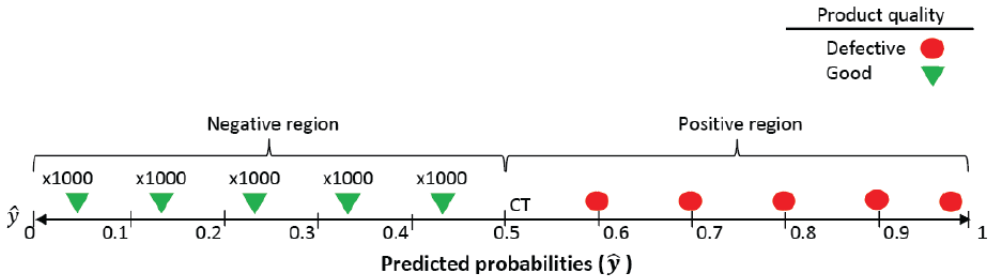


FIGURE 4.12 Scenario 1 - complete separation

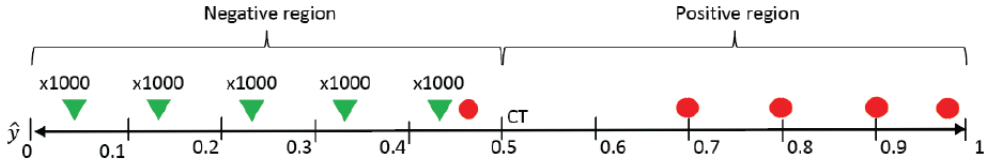


FIGURE 4.13 Scenario 2 - one *FN*

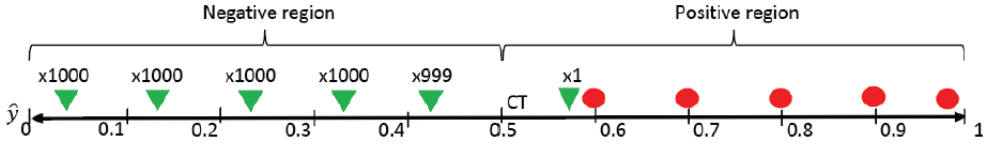


FIGURE 4.14 Scenario 3 - one *FP*

Activity 1

Scenario	TN	TP	FN	FP	α	β	Accuracy G-means
1	5000	5	0	0			
2	5000	4	1	0			
3	4999	5	0	1			
4	4998	5	0	2			
5	5000	3	2	0			
6	4999	4	1	1			
7	4000	5	0	1000			
8	5000	0	5	0			

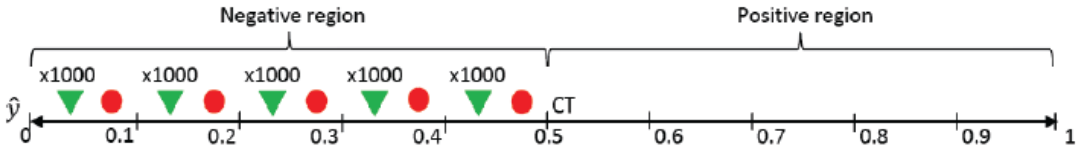


FIGURE 4.19 Scenario 8 - myope

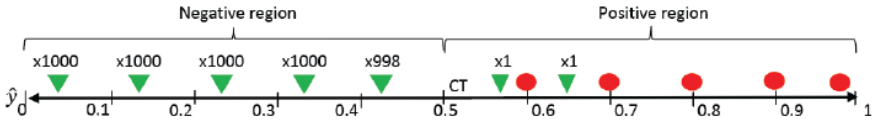


FIGURE 4.15 Scenario 4 - two *FPs*

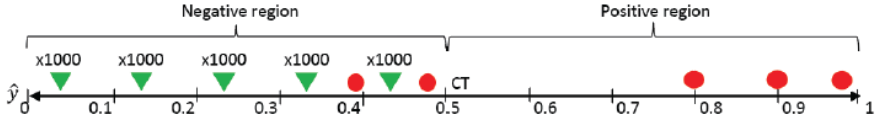


FIGURE 4.16 Scenario 5 - two *FNs*

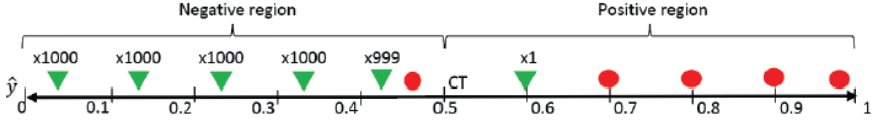


FIGURE 4.17 Scenario 6 - one *FP* and one *FN*

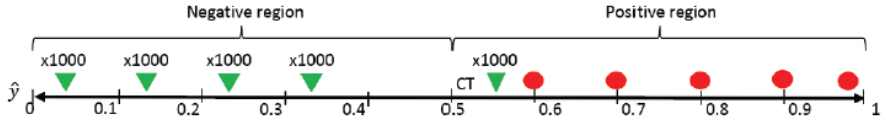


FIGURE 4.18 Scenario 7 - 1000 *FPs*

Activity 1

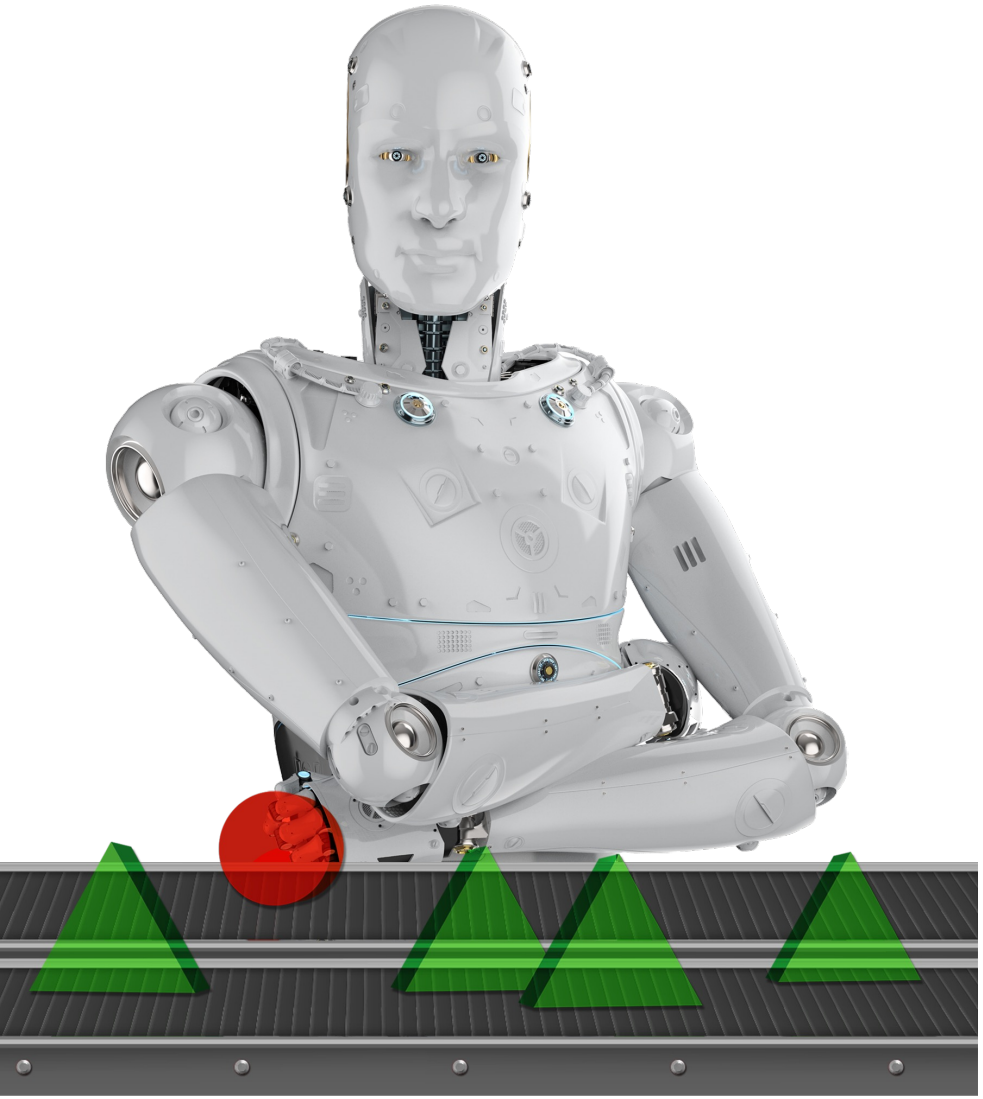
- Results

TABLE 4.3 Classification metrics.

Scenario	TN	TP	FN	FP	α	β	*Accuracy*	*G-mean*
1	5000	5	0	0	0	0	1	1
2	5000	4	1	0	0	0.2	0.999	0.894
3	4999	5	0	1	0.0002	0	0.999	0.999
4	4998	5	0	2	0.0004	0	0.999	0.999
5	5000	3	2	0	0	0.4	0.999	0.775
6	4999	4	1	1	0.0002	0.2	0.999	0.894
7	4000	5	0	1000	0.2	0	0.800	0.894
8	5000	0	5	0	0	1	0.999	0

Remarks

- G-means is very sensitive to FNs in highly unbalanced data
- One FN, scenario 2, resulted in a beta error of 0.20 ($G - means = 0.894$)
- One FP, scenario 3, resulted in a very small alpha error 0.0002 ($MPCD = 0.999$)



Binary patterns

Binary patterns

Two factors define the ability of a classifier to separate the classes:

1. Intrinsic discriminative information contained in the patterns of each feature.
2. Application of the right MLA.

Linear classifier cannot separate the classes

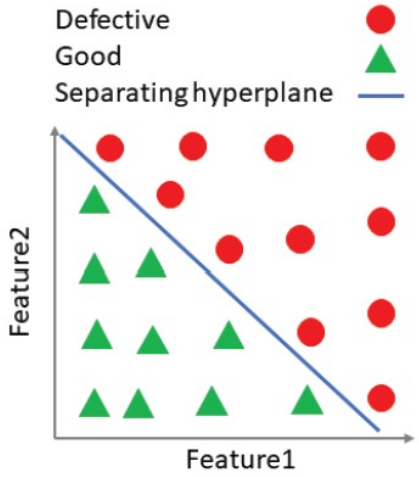


FIGURE 4.20 Linearly separable pattern.

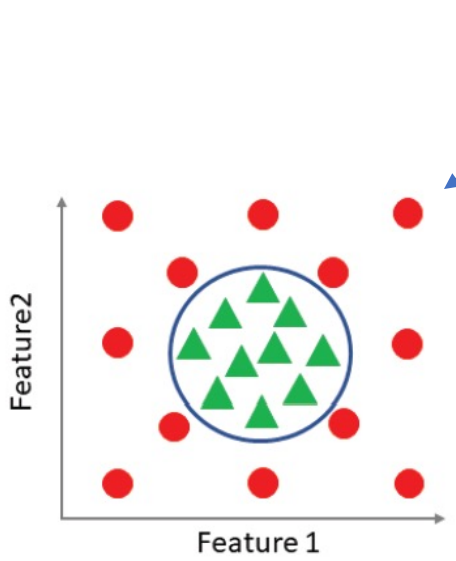


FIGURE 4.21 Non-linearly separable pattern.

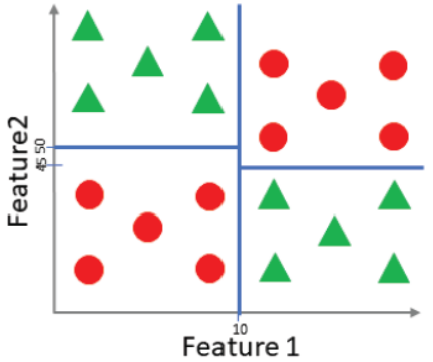


FIGURE 4.22 Non-linearly separable pattern.

Binary patterns

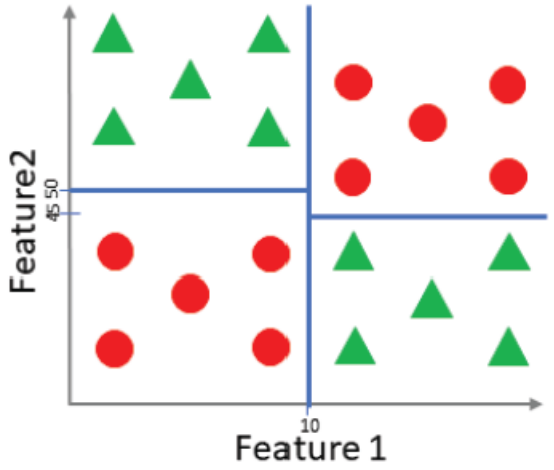


FIGURE 4.22 Non-linearly separable pattern.

Classification rules

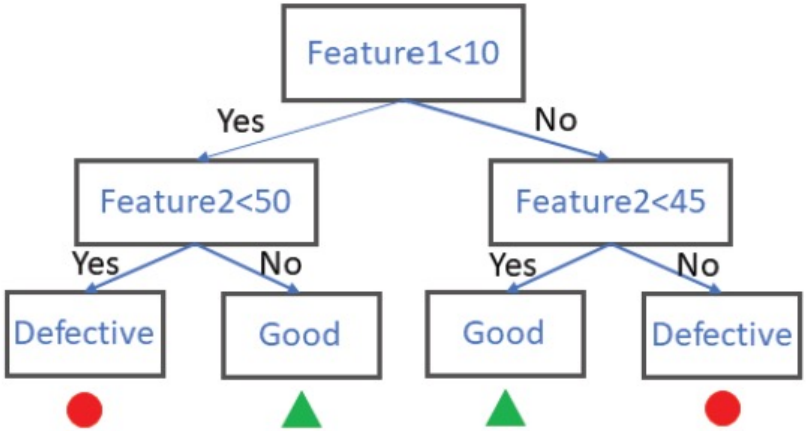


FIGURE 4.23 Classification tree.

Binary patterns

Discriminative information – virtual case 1

- MLA cannot separate the classes without discriminative information.
- Three features generate a clear discriminative pattern

TABLE 4.4 Virtual feature values and the associated quality status of each sample.

Index	Feature 1	Feature 2	Feature 3	Quality status
1	89	91	95	Defective
2	33	20	98	Good
3	95	93	88	Defective
4	40	45	56	Good
5	80	86	92	Defective
6	20	55	12	Good
7	52	60	80	Good
8	90	95	20	Good
9	99	89	60	Good
10	85	82	75	Defective

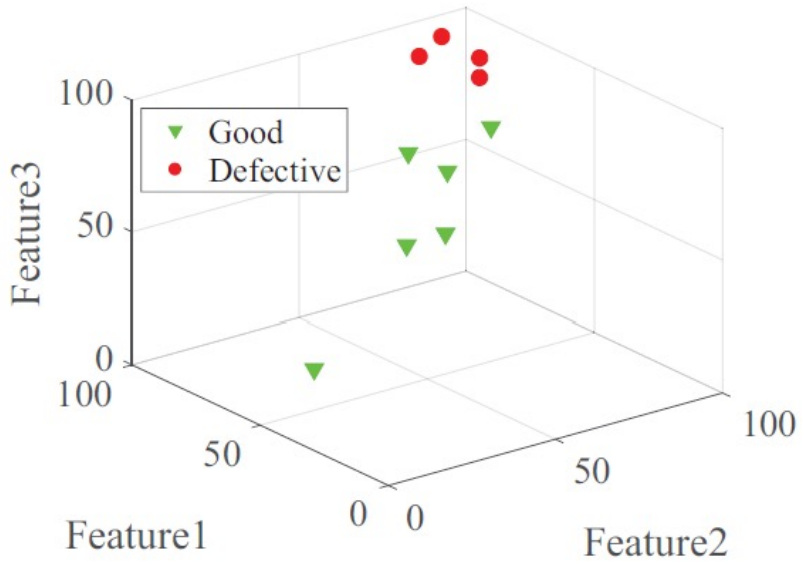


FIGURE 4.24 3*D* quality pattern. Complete separation.

Binary patterns

- Classes do not separate in one and two dimensions
- Right figure shows features 1 and 2, but features 1 and 3, and 2 and 3 do not separate the classes

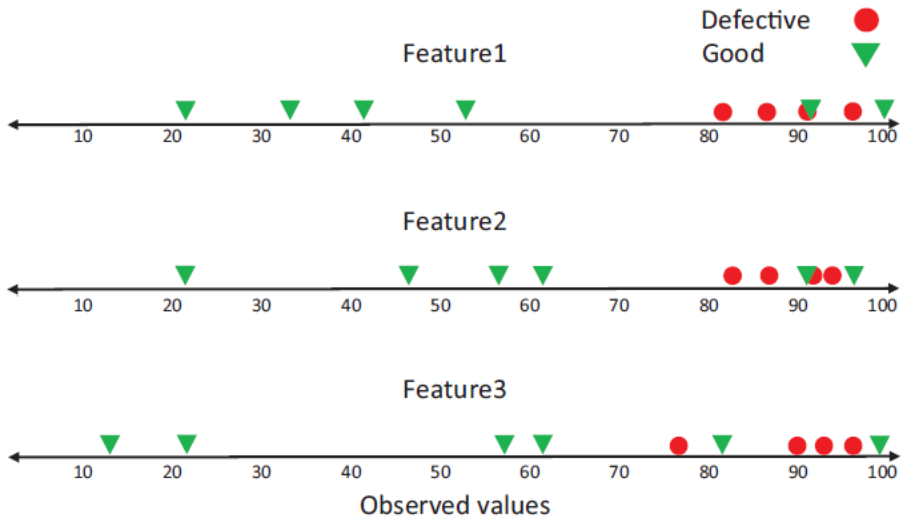


FIGURE 4.25 $1D$ quality pattern.

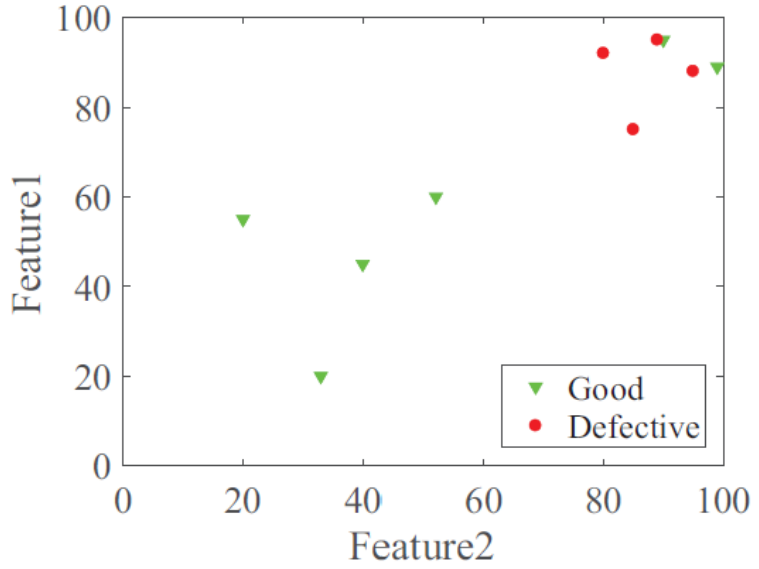


FIGURE 4.26 $2D$ quality pattern.

Binary patterns

- Solution

Sometimes patterns are generated when features are combined. They can exist in the hyperdimensional space. Therefore, they may not be seen.

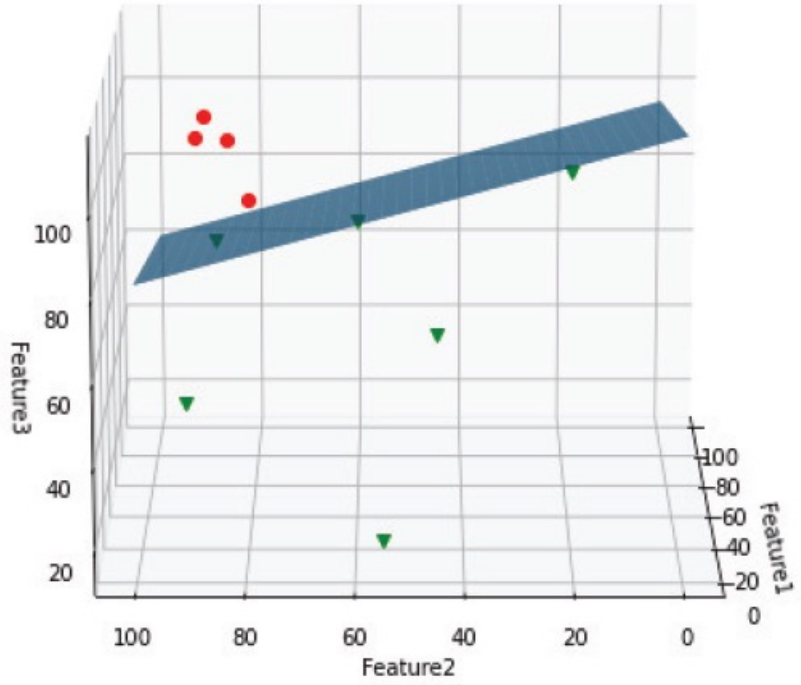


FIGURE 4.27 Linear separation scheme based on the *SVM*.

TABLE 4.5 Confusion matrix for this case study using SVM.

	Predicted good	Predicted bad
Good item	4	0
Defective item	0	6

Binary patterns

Discriminative information – virtual case 2

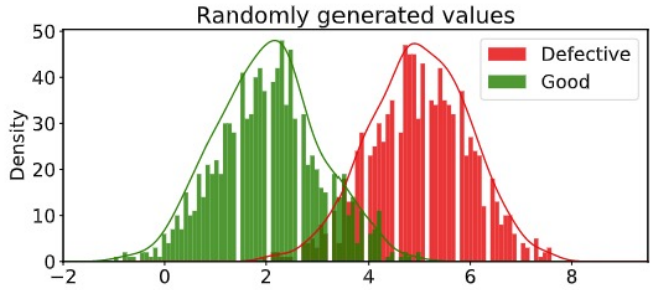
- Allows the algorithm separate the classes.
  - No discriminative information no solution
- Six Virtual Features (VF) with a binary label are created:
  - Using the normal distribution, 2000 random values are generated
  - Negative label (good) is assigned to the first 1,000 samples
  - Positive label (defective) to the rest
  - Negative and positive values have different statistical distributions
- Study is based on the margin principle
  - Generalization performance (i.e., prediction ability on unseen data) of a classifier is strongly correlated to the distribution of its margins.

Table 2: Parameters of the normal distribution used to generate the random values of each *VF*.

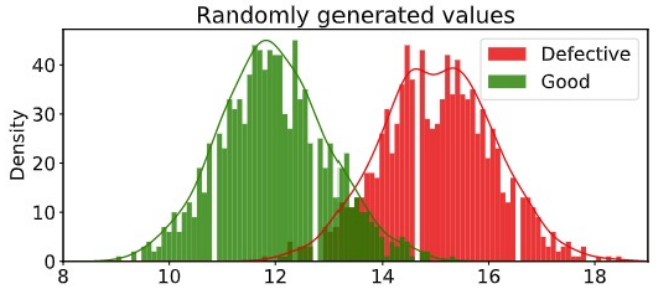
Virtual Feature	Good		Defective	
	μ	σ	μ	σ
1	2	1	5	1
2	12	1	15	1
3	22	1	25	1
4	2	1	3	1
5	2	1	6	1
6	2	0.5	3	0.5

Discriminative analysis

- SVM algorithm applied to learn the separating hyperplane.
- Accuracy (% of correct classifications) is used to evaluate the intrinsic discriminative information.



(a) Virtual feature 1.

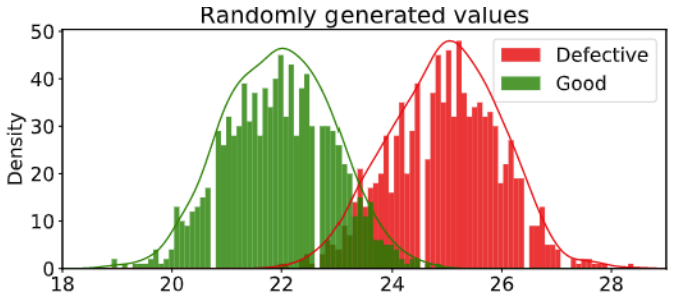


(b) Virtual feature 2.

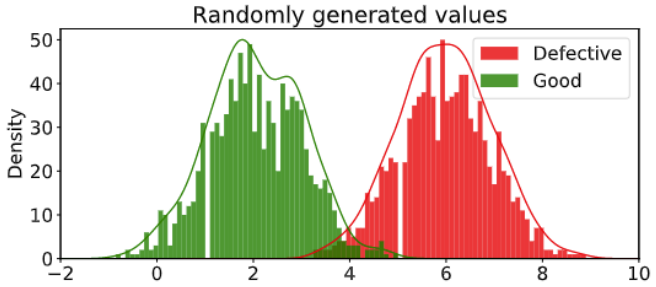
Table 2: Parameters of the normal distribution used to generate the random values of each *VF*.

Virtual Feature	Good μ	Good σ	Defective μ	Defective σ	Prediction Accuracy (%)
1	2	1	5	1	92.83
2	12	1	15	1	92.66
3	22	1	25	1	93.50
4	2	1	3	1	68.66
5	2	1	6	1	97.83
6	2	0.5	3	0.5	100

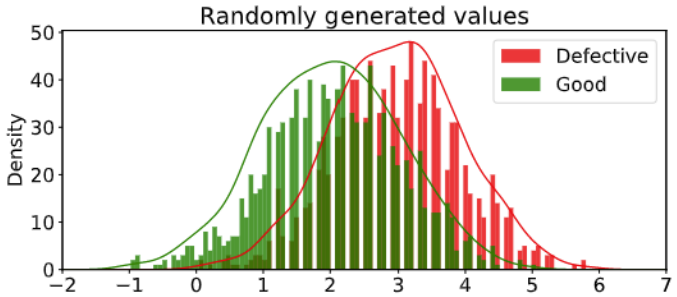
Discriminative analysis



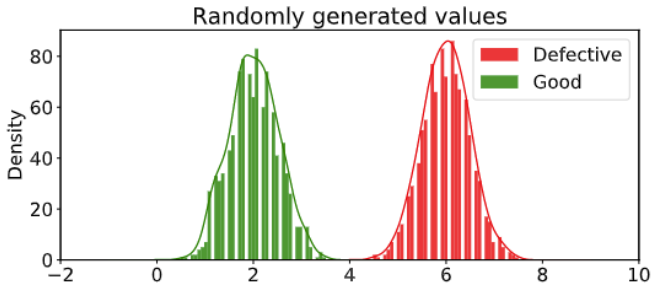
(c) Virtual feature 3.



(e) Virtual feature 5.



(d) Virtual feature 4.



(f) Virtual feature 6.

Table 2: Parameters of the normal distribution used to generate the random values of each *VF*.

Virtual Feature	Good μ	Good σ	Defective μ	Defective σ	Prediction Accuracy (%)
1	2	1	5	1	92.83
2	12	1	15	1	92.66
3	22	1	25	1	93.50
4	2	1	3	1	68.66
5	2	1	6	1	97.83
6	2	0.5	3	0.5	100

Discriminative analysis

- VF-1 and VF-2 are combined. Accuracy is improved from 92.83% and 92.66% to 98.5%.
- VF-1 and VF-4 are combined. Improvement from 92.83% and 68.66% to 94.16%.

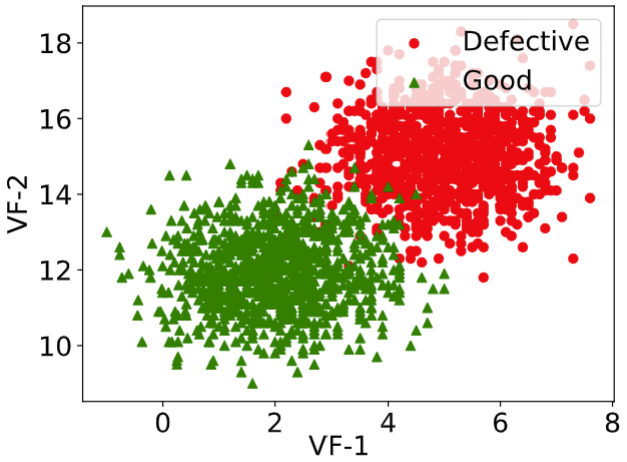


Figure 12: Combination of virtual features 1 and 2.

Table 2: Parameters of the normal distribution used to generate the random values of each *VF*.

Virtual Feature	Good μ	Good σ	Defective μ	Defective σ	Prediction Accuracy (%)
1	2	1	5	1	92.83
2	12	1	15	1	92.66
3	22	1	25	1	93.50
4	2	1	3	1	68.66
5	2	1	6	1	97.83
6	2	0.5	3	0.5	100

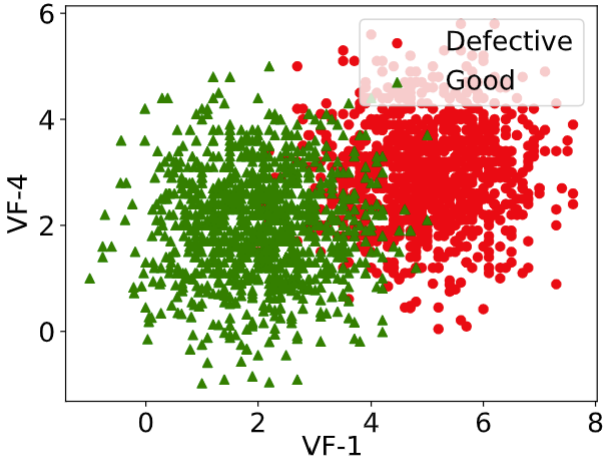


Figure 13: Combination virtual features 1 and 4.

Discriminative analysis

- VF-1, VF-2, and VF-3 almost separate the classes, from 92.83%, 92.66%, and 93.5 to 99.66%.

- Including all features with discriminative information in the model does not always improve prediction.
- Feature selection methods help to identify which feature combination is the optimal (or close to optimal).

Table 2: Parameters of the normal distribution used to generate the random values of each *VF*.

Virtual Feature	Good μ	Good σ	Defective μ	Defective σ	Prediction Accuracy (%)
1	2	1	5	1	92.83
2	12	1	15	1	92.66
3	22	1	25	1	93.50
4	2	1	3	1	68.66
5	2	1	6	1	97.83
6	2	0.5	3	0.5	100

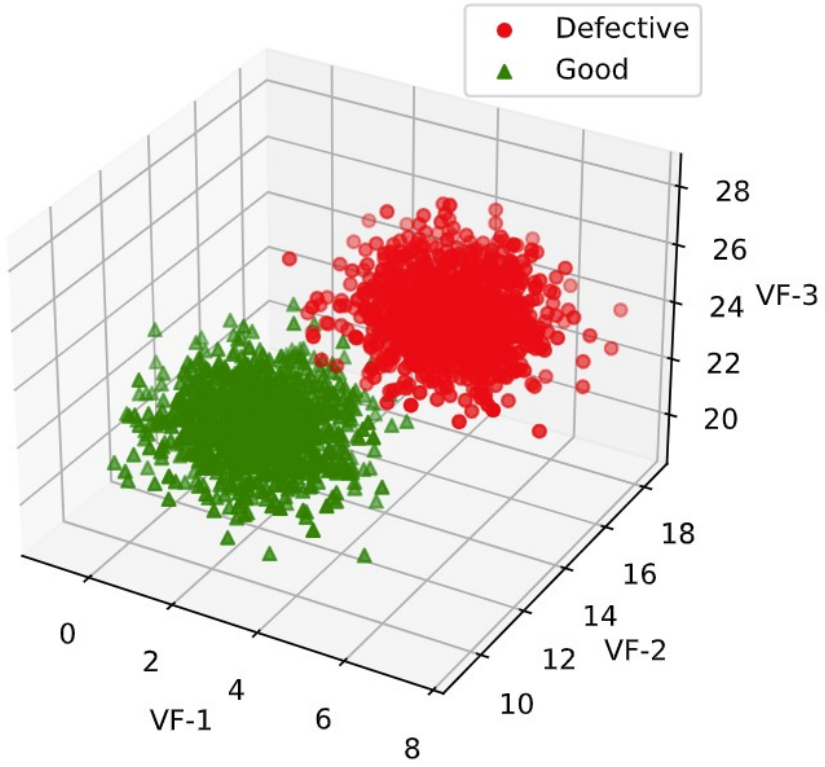


Figure 14: Combination of virtual features 1, 2 and 3.

Discriminative analysis

Remarks

- The largest the margin between classes the better
- The smallest the variance (e.g., noise) of the statistical distribution of classes the better

Activity 2

Access the following links and learn about the two of foundational concepts in machine learning, 10mins

https://towardsdatascience.com/bias-variance-and-how-they-are-related-to-underfitting-overfitting-4809aed98b79

Discussions, 5mins

Chapter 5: Machine Learning

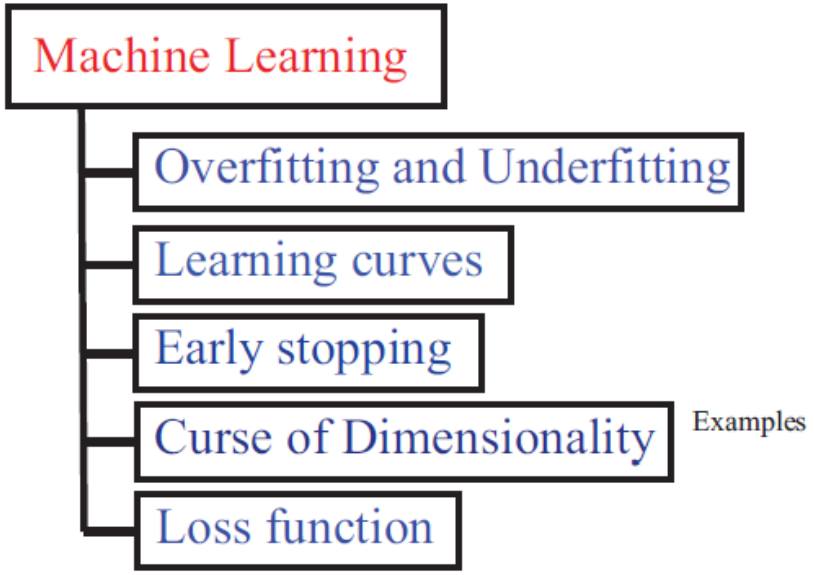
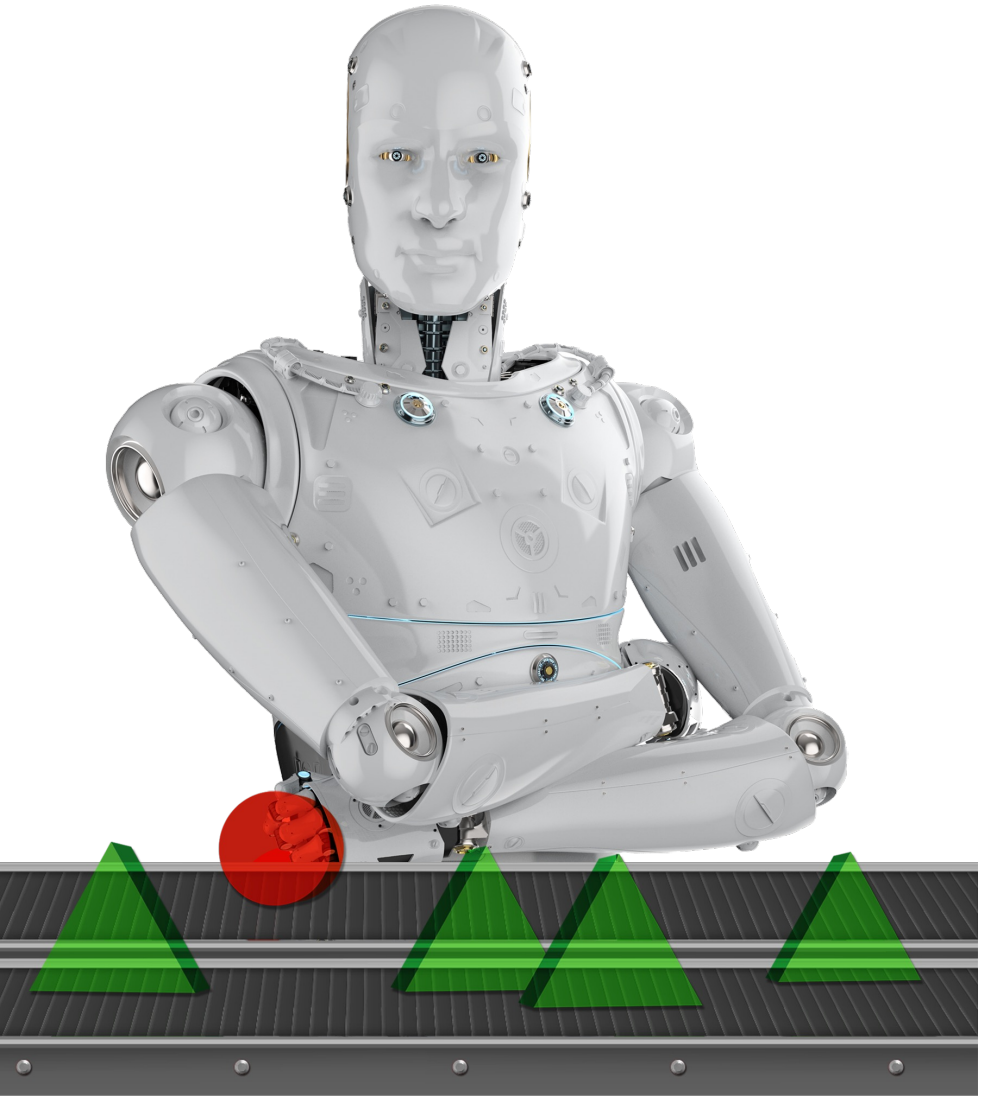


FIGURE 5.1 Contents of the Chapter 5



Overfitting and underfitting

Overfitting and underfitting

- Prediction error must kept as low as possible during the learning (i.e., training) process, for the model to make accurate predictions on unseen data.
- Errors are broken down into two different sources:

$$Error = ReducibleError + IrreducibleError$$

- Irreducible error cannot be reduced by creating good models.
- No matter how good the model is, the manufacturing systems generating the feature space (X) will have certain amount of noise or irreducible error that can not be removed.

Overfitting and underfitting

Key concept in machine learning

- Bias-variance trade-off refers to the problem of finding a balance between two main sources of errors that we can reduce when building a model: bias and variance.

$$Error = Bias^2 + Variance + IrreducibleError$$

Overfitting and underfitting

- **Bias**, error that arises when a model is too simplistic to capture the underlying patterns in the data.
- High-Bias model is likely to under-fit the data, meaning that it fails to capture the complexity of the relationships between the input features and the target variable.

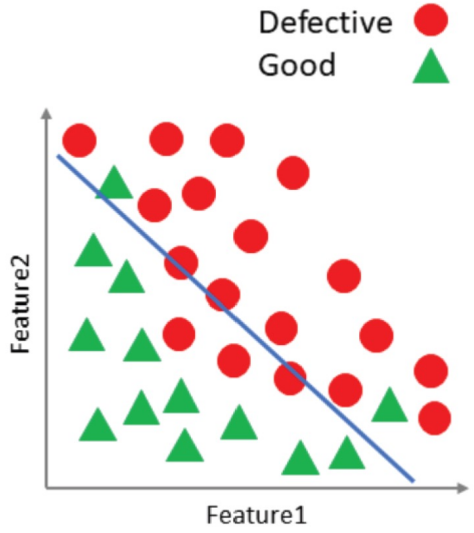


FIGURE 5.3 Under-fitting in the context of binary classification.

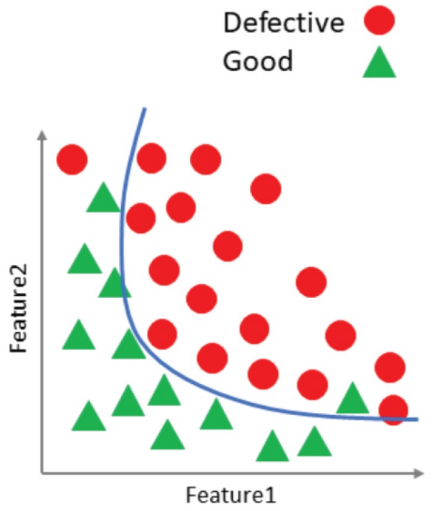


FIGURE 5.7 Right-fitting in the context of binary classification.

Overfitting and underfitting

Underfitting reasons (high bias)

- Insufficient model complexity
- Insufficient training
- Insufficient data
- Poor data quality
- Inappropriate feature selection
- Inappropriate model development or selection

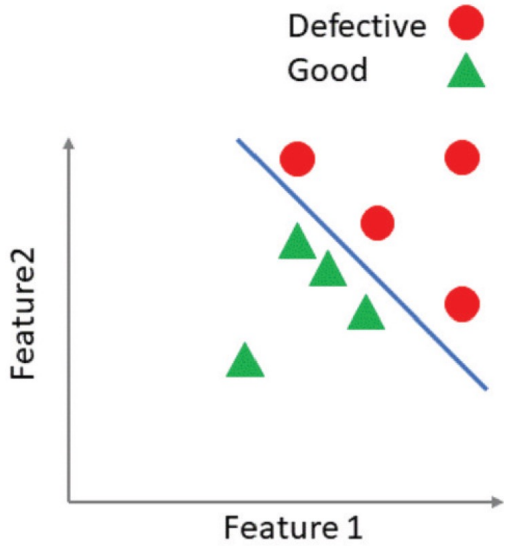


FIGURE 5.4 Small training set

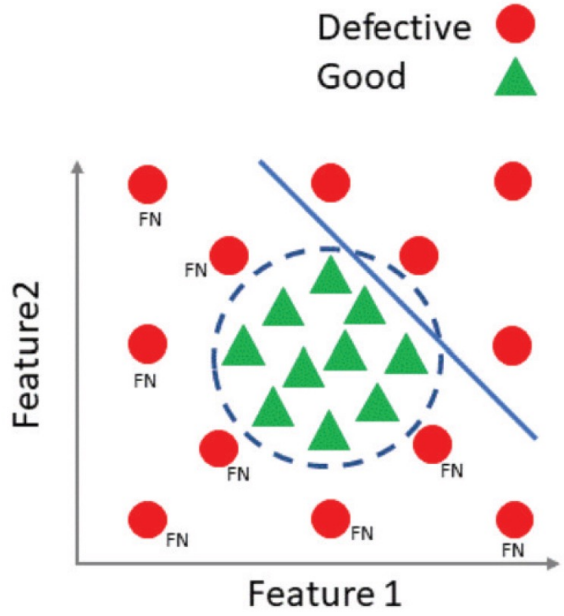


FIGURE 5.5 Full pattern

Overfitting and underfitting

Address the underfitting problem

- Increase the model complexity
- Add more features
- Increase the amount of training
- Improve the quality of the data, or choose a different model

Overfitting and underfitting

- **Variance**, error that arises when a model is too complex and captures noise in the data rather than the underlying patterns.
- High variance model is likely to overfit the data, fits the training data well but fails to generalize to new, unseen data.

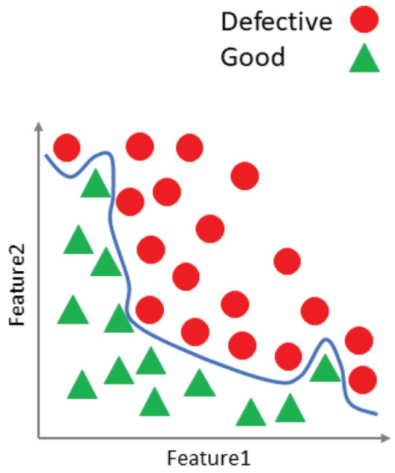


FIGURE 5.6 Over-fitting in the context of binary classification.

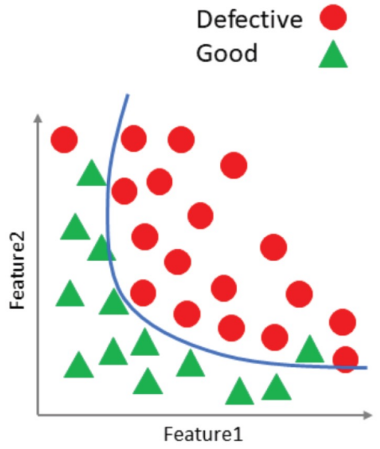


FIGURE 5.7 Right-fitting in the context of binary classification.

- Whereas this hyperplane completely separates the classes, it would not generalize well to unseen data.

Overfitting and underfitting

Overfitting reasons

- Model complexity
- Insufficient training data
- Data noise
- Hyperparameters
- Leaking information from test data

- To address overfitting
  - Simplify the model
  - Reduce the number of features
  - Increase the amount of training data
  - Regularization, early stopping, and dropout for artificial neural networks
- Large training data set prevents the MLA to fit all the samples, forcing them learn more generalizable patterns.

Overfitting and underfitting

- To achieve good generalization performance, a ML model must have the optimum model complexity
- Efficiently solving the bias-variance trade-off, where reducing one source of error increases the other
- For example, increasing the complexity of a model can reduce bias but increase variance, while simplifying a model can reduce variance but increase bias

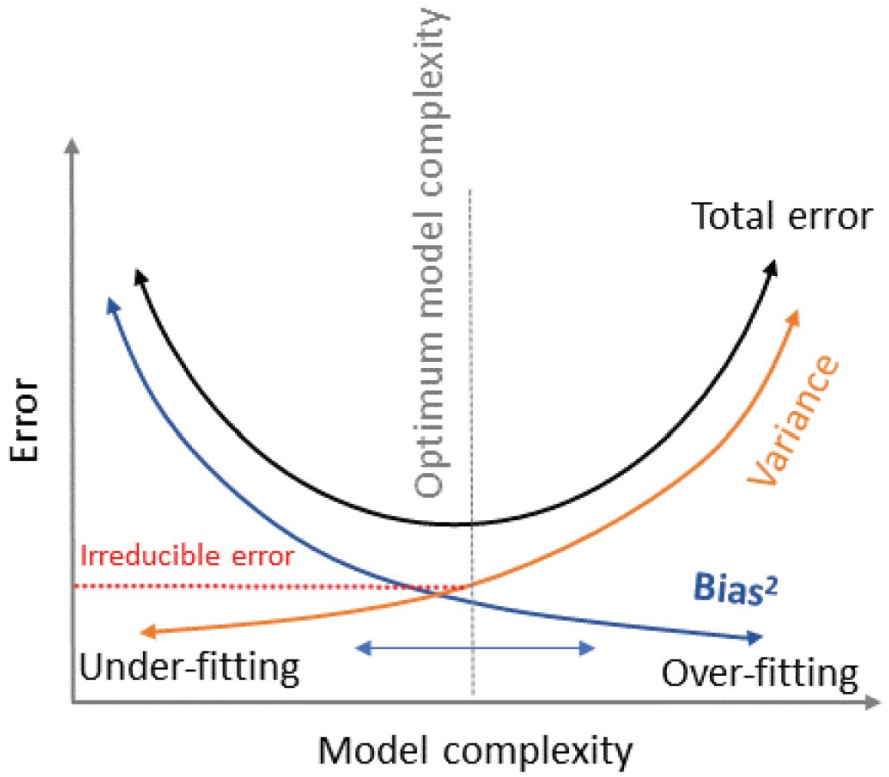
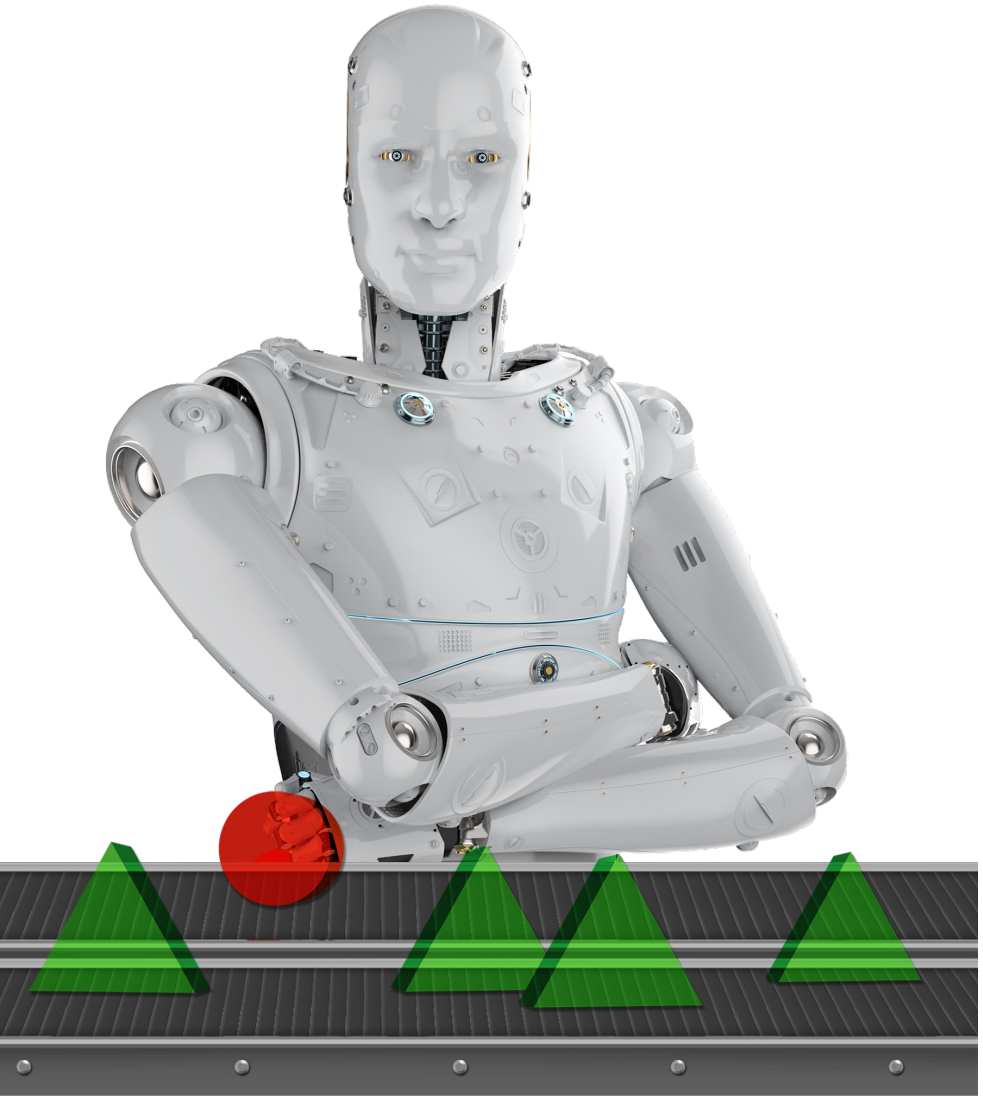


FIGURE 5.2 Bias and variance trade-off.

Model selection also helps to find the right complexity of the model



Learning curves

Quick reminder

- If the data set is split into two sub-data sets, they are usually called training and test sets
- In the following slides test set refers to the unseen data

FIGURE 3.8 Train and test scheme.

Learning curves

- Learning curve is a graphical representation of the performance of MLA as it learns from data over time
- Shows how the performance of the model changes as the amount of training data increases.
- There are two types of learning curves
  - Training
  - Testing
- Learning curves help to determine if:
  - More data is needed
  - More or less features are needed
  - Learning problem cannot be solved

Learning curves

High bias

- Underfitting manifests in a learning curves with high testing and training errors, small gap between the two error curves

FIGURE 5.11 Learning curves for high bias. *Irreducible error* in green dashed line.

Learning curves

High variance

- Overfitting results in learning curves that exhibit a high testing error and a low training error (large gap).
- Indicating that the model is not generalized and it only performs well on the training data.

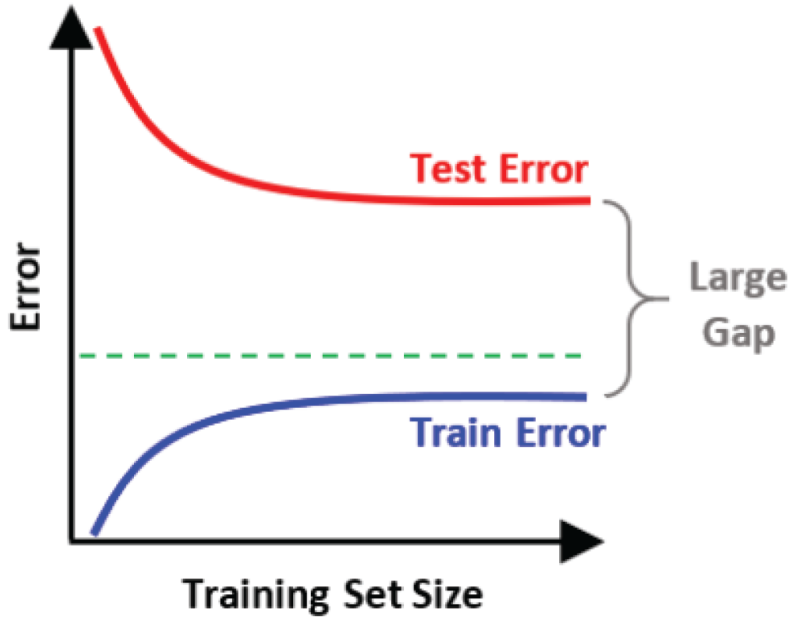


FIGURE 5.12 Learning curves for high variance. *Irreducible error* in green dashed line.

Learning curves

Right-fitting

- Quantifying the learning curves helps the Quality 4.0 engineers to properly diagnose high bias or variance errors
- Decide accordingly which steps to take to develop an ideal model, where the testing and training errors approximate to the irreducible error.

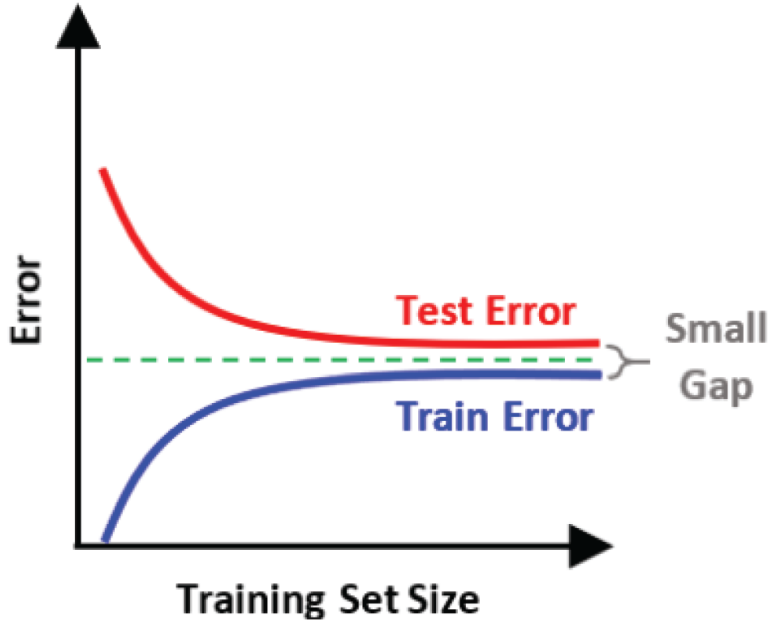


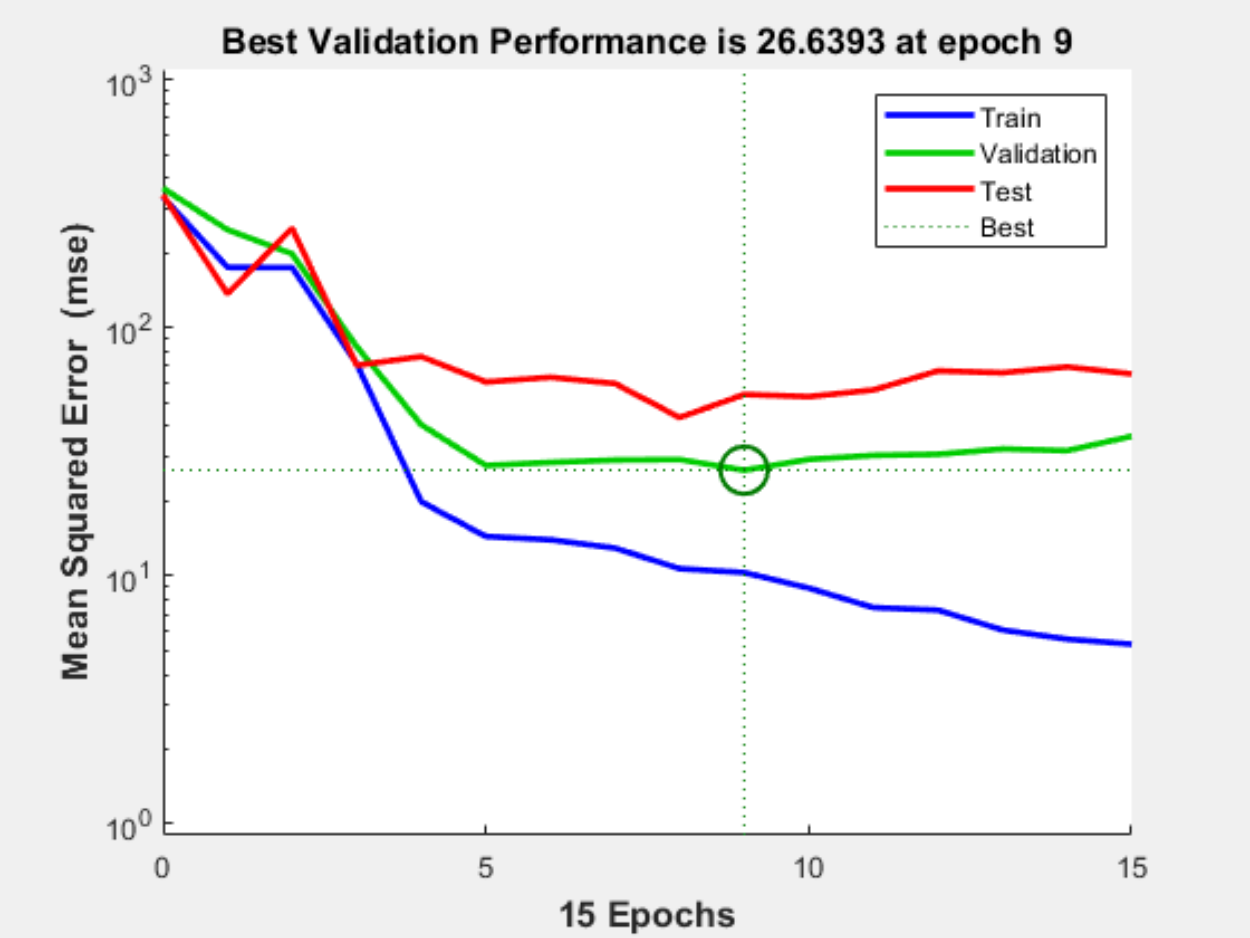
FIGURE 5.13 Learning curves for"ideal" model. *Irreducible error* in green dashed line.

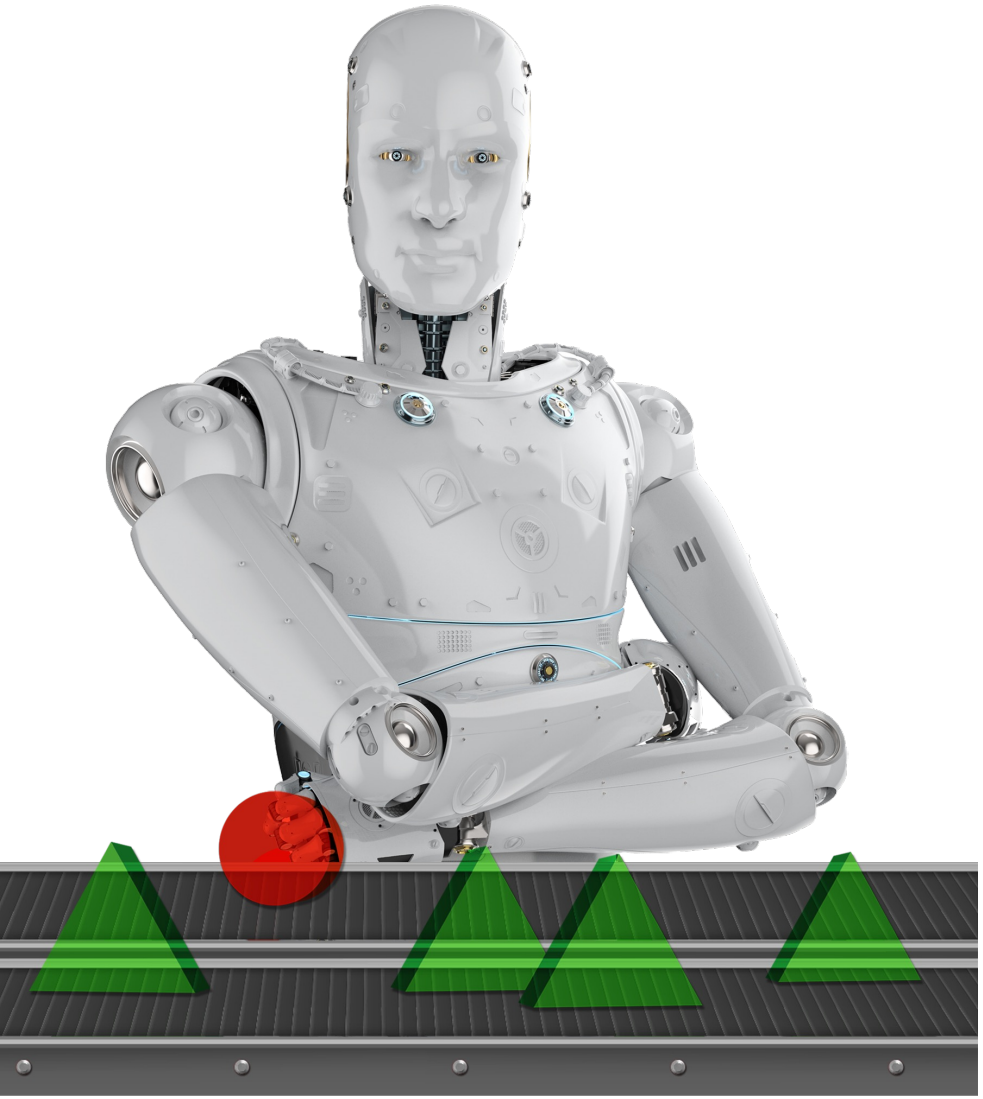
Learning curves

- Indiscriminately adding more samples may not improve generalization performance.
- Indiscriminately including large number of features increases the chances for MLA to learn coincidental or spurious patterns.

Quick discussion

- What happens after the circle?





Curse of dimensionality

Curse of dimensionality

- Machine learning excels at analyzing data with many dimensions (i.e., features), but it becomes more challenging to create a good model as the number of dimensions increases.
- High dimensional data refers to a data set in which the number of dimensions (D) is bigger than the number of observations (m), $D \gg m$.
- High dimensional data leads to the curse of dimensionality
- Many MLA degrade in performance as the number of features increases.
- When the number of features increase, the available samples in the feature space become exponentially sparser.
  - Making it easier for the MLA to separate the classes by learning spurious decision boundaries that do not generalize to unseen data

Curse of dimensionality

Example

six samples cover the 1D space, the same way, 36 samples are required for 2D, and 216 for 3D. On the other hand, if all the features in a dataset are binary, the total number of samples needed to cover all the possible combinations of three features is 2^3 = 8, for 30 features is 2^{30} = 1,073,741,824.

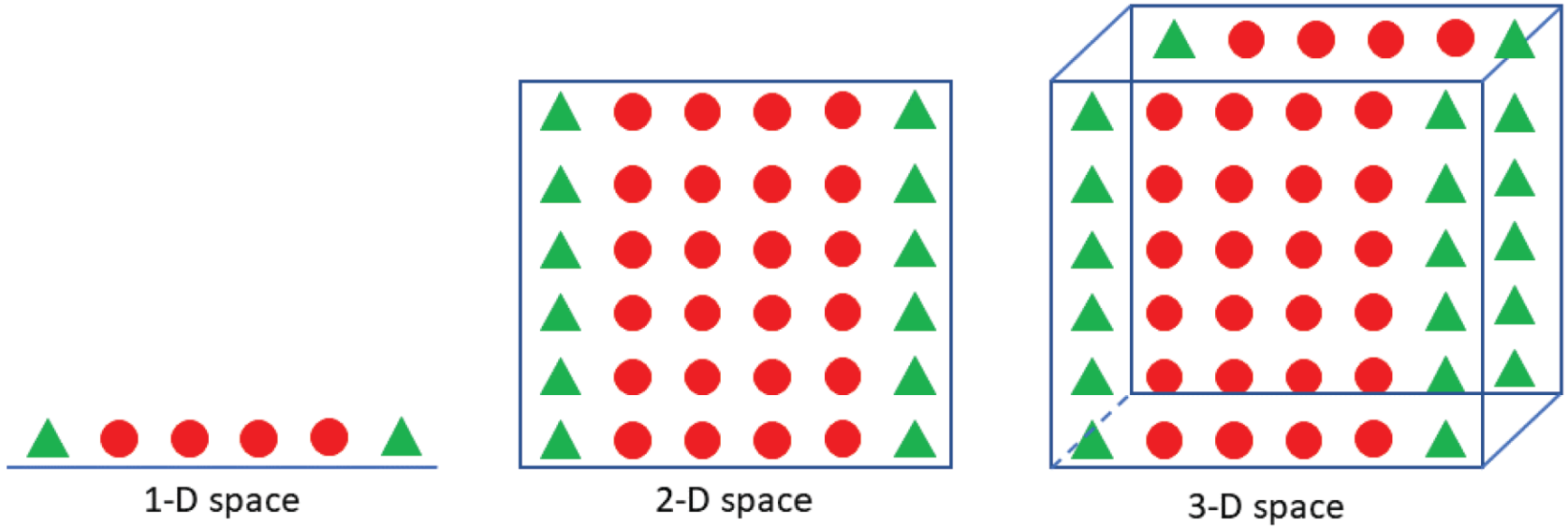


FIGURE 5.14 Number of samples required to cover one, two, and three dimensional spaces.

Curse of dimensionality

Hughes Phenomenon

- As the number of features or dimensions increases, the performance of a classifier initially improves, but then eventually degrades
- Model becomes more complex and start to over-fit the training data, capturing noise and spurious correlations rather than the true underlying patterns.
  - Therefore, adding more features beyond a certain point can reduce the predictive ability.

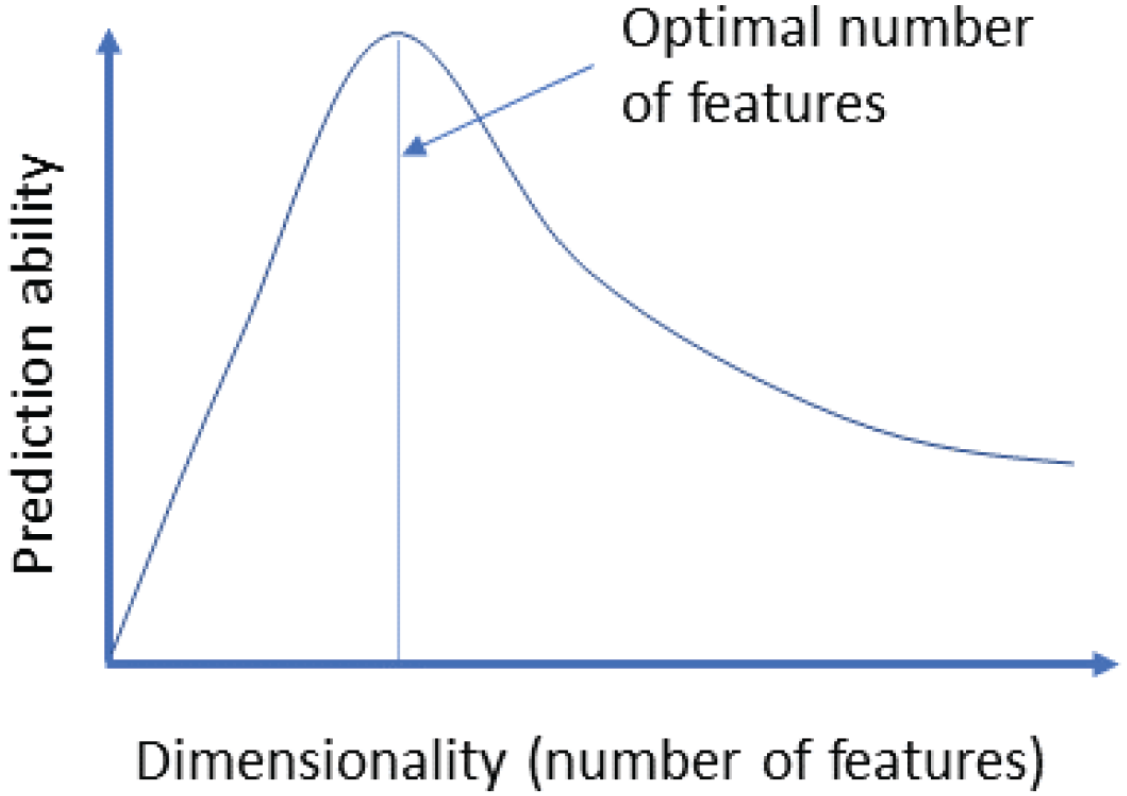


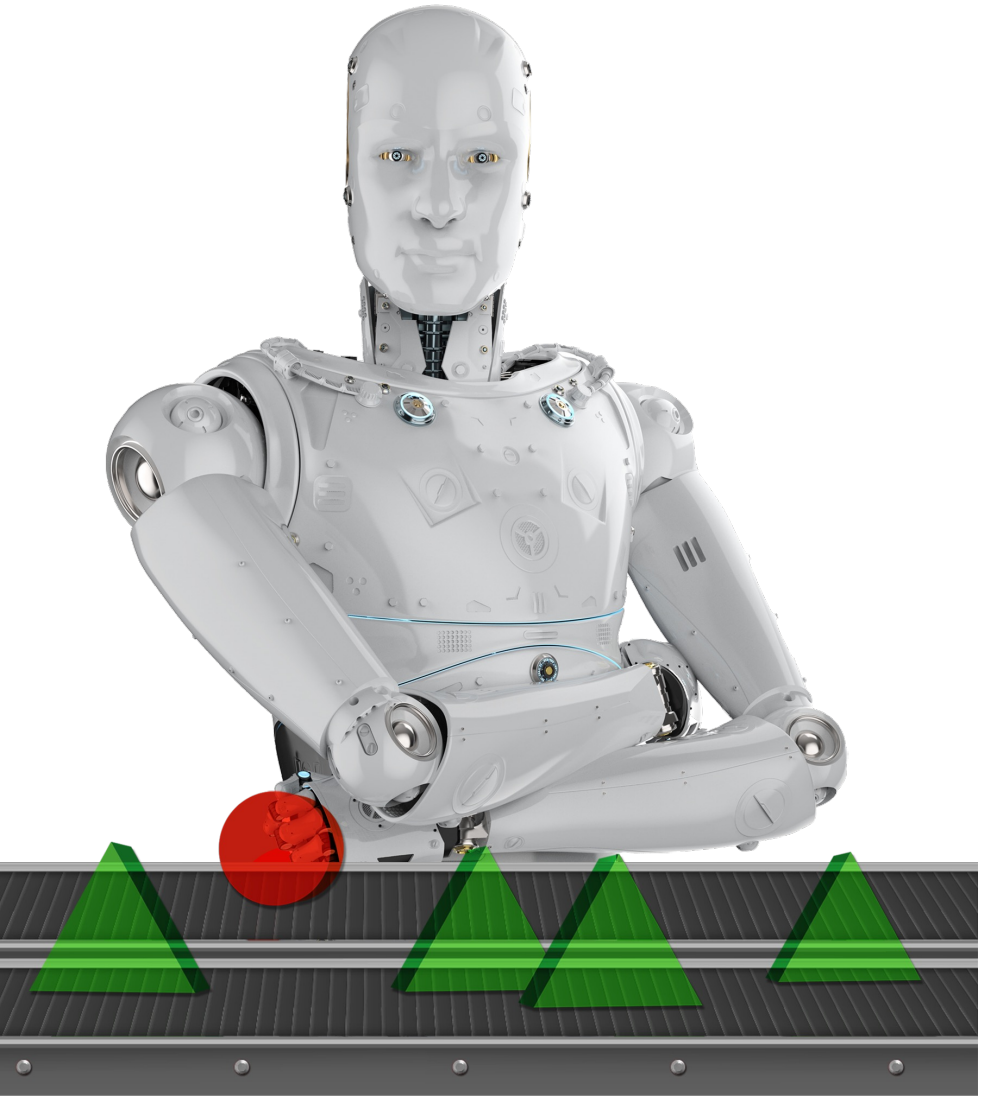
FIGURE 5.15 Hughes Phenomenon.

Curse of dimensionality

Example

- *Model-1*, includes only pressure (bar).
- *Model-2*, includes pressure and temperature (Fahrenheit).
- *Model-3*, includes pressure, temperature, and time (sec).
- *Model-4*, includes pressure, temperature, time, and machine cycles-to-failure (number of cycles).
- *Model-5*, includes pressure, temperature, time, machine cycles-to-failure, and the geographic location of the plant (city).
- *Model-6*, includes pressure, temperature, time, machine cycles-to-failure, the geographic location of the plant, and the number of machine cycles since last maintenance (number).
- *Model-7*, includes pressure, temperature, time, machine cycles-to-failure, the geographic location of the plant, the number of machine cycles since last maintenance, and pressure (psi).

Prediction ability of the first four models may constantly increase, but model-5 includes an irrelevant feature (geographic location), model-6 a trivial one, this information was mainly capture in model-4. Finally, model-7 a redundant one, it provides the same information as the model-2.



Loss function for binary classification

Loss function for binary classification

- Loss functions or objective functions, play an important role in the construction of MLA and the improvement of their performance.
- Indicate how well the MLA is fitting the data and define the best parameters.
- The binary cross-entropy (BCE) loss also known as Binary Log Loss or Binary Cross-Entropy Loss, is the most common loss function used in binary classification problems.

$$Loss_{BCE} = -\frac{1}{m}\sum_{i=1}^{m}(l_i \cdot log(\hat{y}_i) + (1 - l_i) \cdot log(1 - \hat{y}_i)) \qquad (5.3)$$

where m refers to the number of samples, $l_i \in \{0, 1\}$ the the true class label of sample i and $\hat{y}_i = f(X; \theta) = P(l_i = 1)$, its associated predicted probability. The *log loss* is measured as a number between 0 and 1, with 0 being a perfect model.

Loss function for binary classification

Example

Figures 5.17 and 5.18 show two virtual models, the labels are described using the same color convention and the predicted probabilities $\hat{y}$ are described at the top of each sample, (Table 5.1) shows the log *loss* of each model.

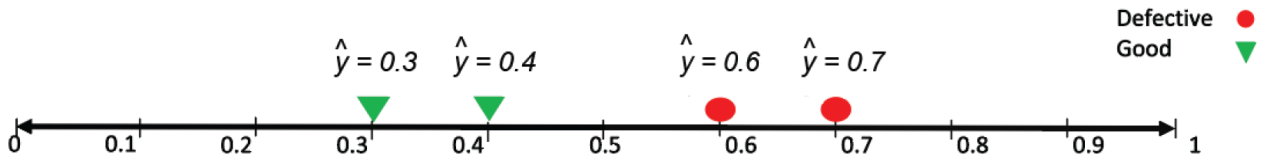


FIGURE 5.17 Virtual *model-1*.

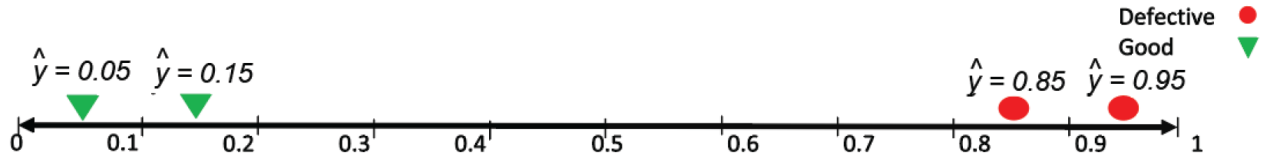


FIGURE 5.18 Virtual *model-2*.

Activity 3

First coding exercise (10mins)

- Download and install Anaconda, https://www.anaconda.com/
- Open a Jupyter notebook
- Compute the binary cross-entropy error of virtual model 1 and 2.

```
In [1]: #import libraries
        import numpy as np
        from sklearn.metrics import log_loss

In [2]: #array (vector) with the label of each sample
        y=np.array([0,0,1,1])

In [3]: #array (vector) with the predicted probability of each sample
        y_hat=np.array([.3,.4,.6,.7])

In [4]: #used a predefined function to compute the logloss
        BCE1=log_loss(y,y_hat)

In [5]: #printh the variable BCE1 to see the value
        print(BCE1)

        0.4337502838523616
```

Discussions, 5mins

Loss function for binary classification

TABLE 5.1 Binary cross entropy loss of virtual models.

Model	*BCE* (log loss)
Virtual model1	0.4337
Virtual model2	0.1069

As shown in (Table 5.1), the virtual model2 has a significantly smaller *BCE* loss (0.4337 vs 0.1069). This is due to the negative classes have predicted probabilities closer to zero and the positive classes have predicted probabilities closer to one. This example illustrates the basic concept behind the *BCE* loss.

Chapter 8: Learning quality control (LQC)

- Learning Quality Control
  - Definition
  - Applications
  - The Problem Solving Strategy
    - Identify
    - Observe
    - Data
    - Learn
    - Redesign
  - Boosting the Statistical Process Control
  - Managerial Implications
    - The strategy
    - The certifications
    - The team

FIGURE 8.3 Contents of the Chapter 8

Basic relevant concepts

- Learning quality control (LQC) is the evolution of statistical process control (SPC).
- SPC is founded on statistical methods.
- LQC is founded on statistical methods and boosted with ML and DL.
- These technologies allow us to solve a whole new range of engineering intractable problems.

Basic relevant concepts

Inspection

- Inspection involves examining the manufacturing items at various stages of the manufacturing process or after the product is completed.
- Identify defects, deviations from specifications, or other quality issues.
- Done by technician who visually and manually inspect the product, using inspection methods and measuring tools, to ensure the manufacturing item meets the engineering specifications.

FIGURE 8.1 Inspection.

Basic relevant concepts

SPC

- Methodology used in quality control to monitor and control a process to ensure that it operates at its full potential.
- Statistical techniques to measure and analyze data from the process and identify any variations that might occur.
- Identify and eliminate any special causes of variation that could result in defects or non-conformance to specifications.
- Control charts popular SPC tools

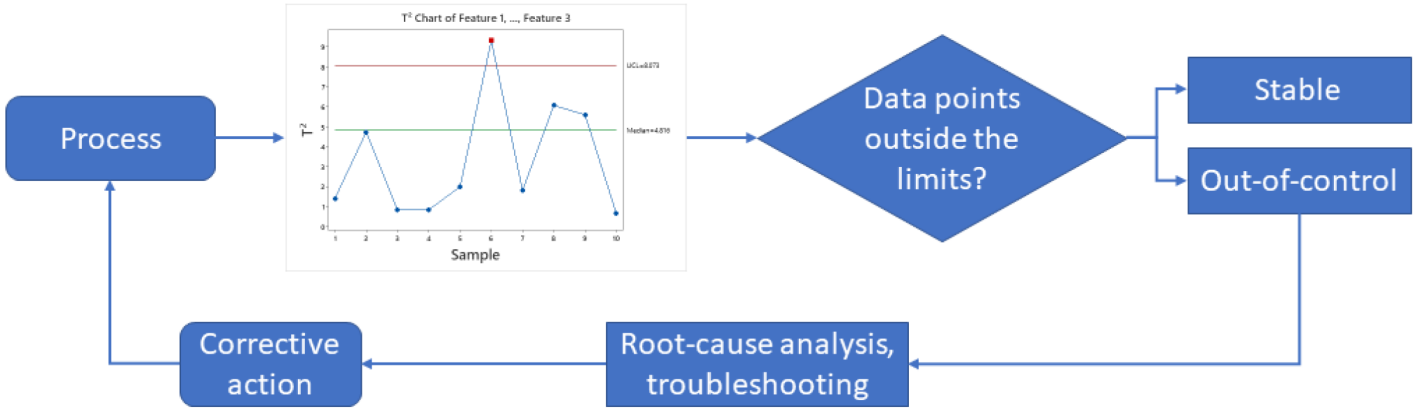
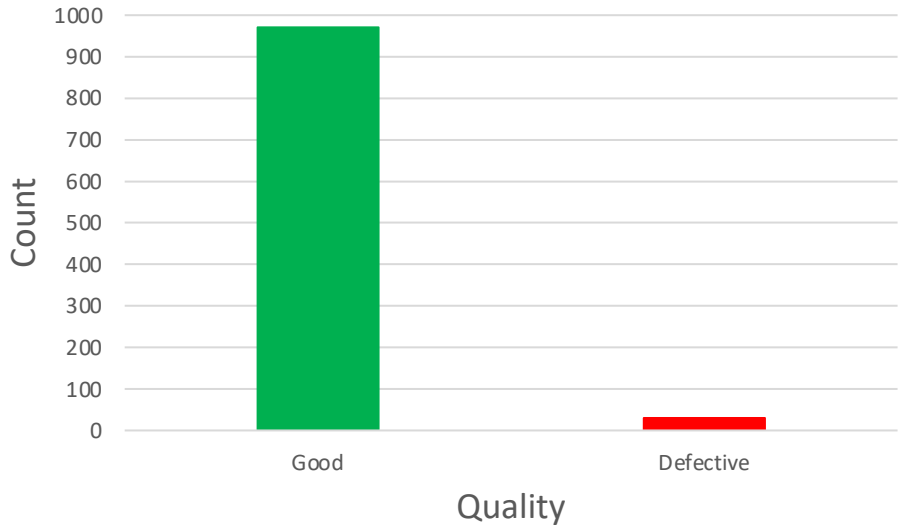


FIGURE 8.2 Statistical process control.

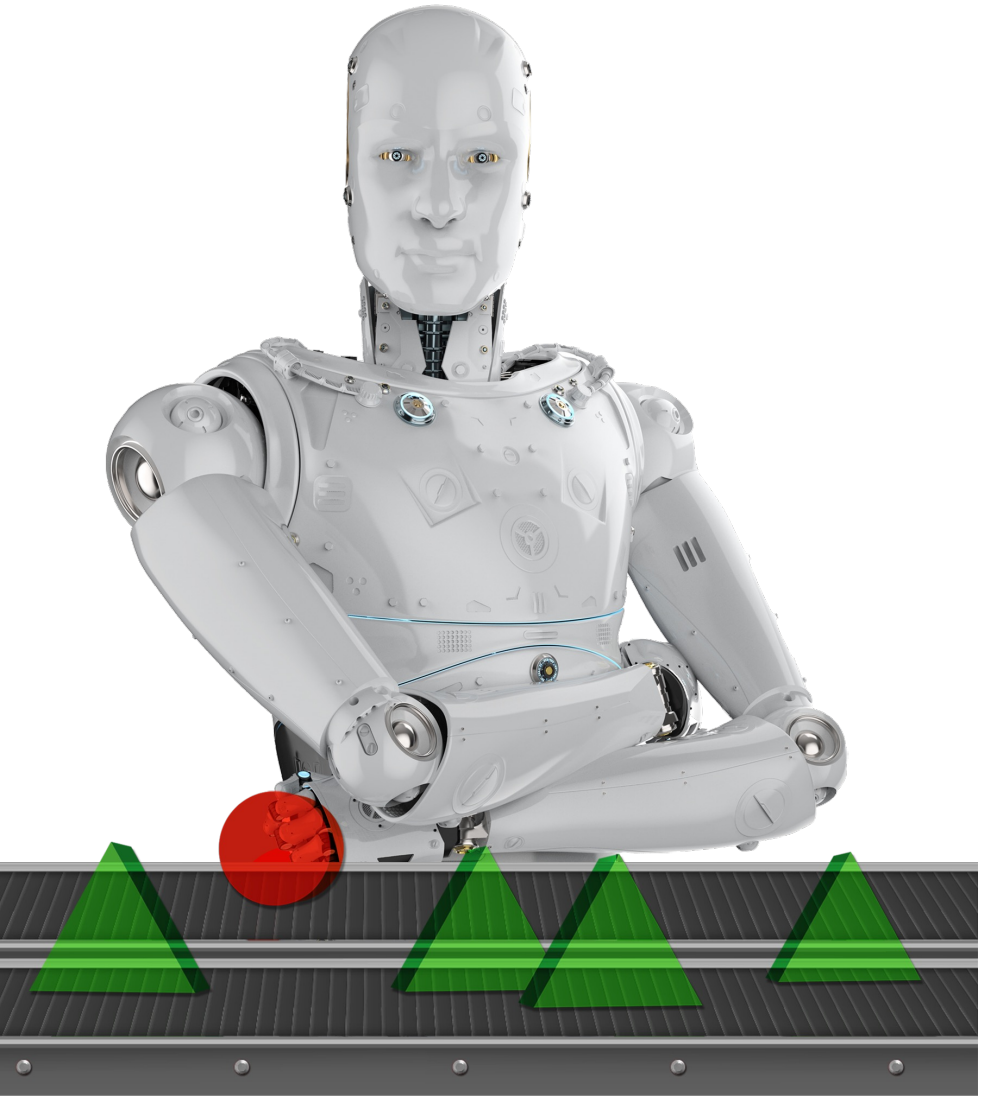
Basic relevant concepts

Binary classification of quality (BCoQ) data sets

- Application of ML or DL algorithms to process-derived data to develop a binary classifier aimed at defect detection.
- Data sets for BCoQ tend to be highly or ultra-imbalanced (minority class count<1%).
  - Rare quality event detection



- Ultra imbalanced data structures suffer from the tendency of MLAs to misclassify most minority classes (defective) as the majority class (good).



Definition

Definition

LQC is a real time online process-monitoring system for defect prediction and detection. Both applications are formulated as binary classification problems, as shown in equation (8.1). Historical samples (X, l), including features, images, or signals denoted by X and their associated quality label denoted by $l \in \{$ good, defective $\}$ are used to train a classifier in the general form $\hat{y} = f(X; \theta) = P(l = \text{defective})$, where $\hat{y}$ refers to the predicted probability that the manufactured item is defective. The classifier includes the predictive model and the classification threshold (CT), the classification of unseen data is performed based on equation (8.2).

$$l_i = \begin{cases} 1 & \text{if } i^{th} \text{ item is defective } (+) \\ 0 & \text{if } i^{\text{th}} \text{ item is good } (-) \end{cases} \tag{8.1}$$

$$l_i = \begin{cases} 1 & if\ \hat{y}_i \geq CT\ item\ is\ defective\ (+) \\ 0 & if\ \hat{y}_i < CT\ item\ is\ good\ (-) \end{cases} \tag{8.2}$$

A positive label indicates a defective item, and a negative label is related to a good-quality item. The classifier is developed offline, and it may require weeks or even months of R&D and engineering efforts. Once the classifier is ready, it is deployed into production (online) [1].

Definition

- Defect prediction, refers to the process of identifying quality patterns early in the process that may potentially result in downstream quality problems.
- Defect prediction allows to take proactive actions to prevent defects from occurring, such as scheduling maintenance, changing process parameters, or inspecting raw materials.

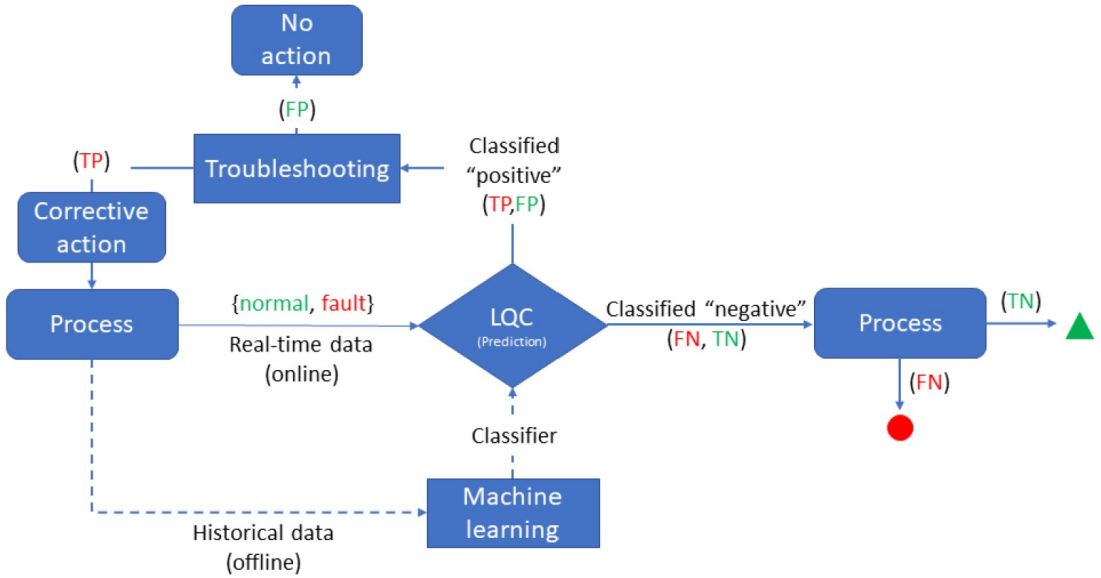


FIGURE 8.4 *LQC* for prediction application.

Definition

- Defect detection, refers to the process of identifying the defective items during the execution of the process.
- Defect detection allows to remove the manufactured items from the value adding process; they can be reworked or scrapped.

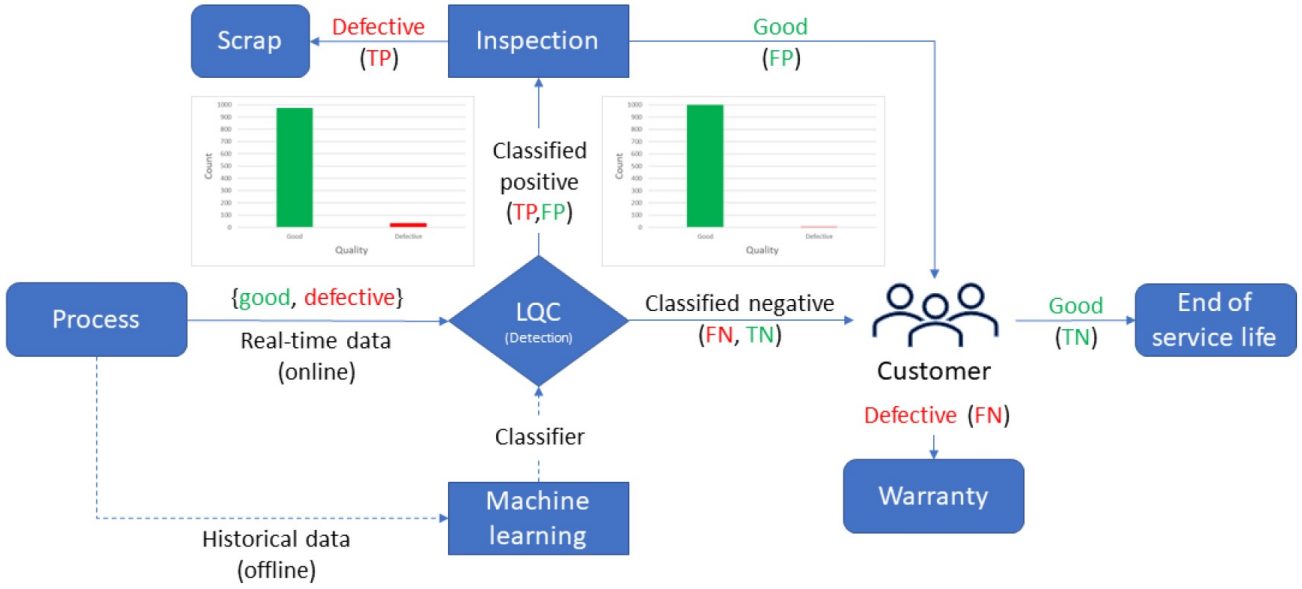


FIGURE 8.5 *LQC* for detection application.

Definition

- Figure 8.6 illustrates the probabilistic process monitoring component of LQC
- In contrast to other control charts (Fig. 8.2) that monitor means, variations, ranges, etc, LQC monitors probabilities which are compared to the previously defined CT to identify or predict quality issues.

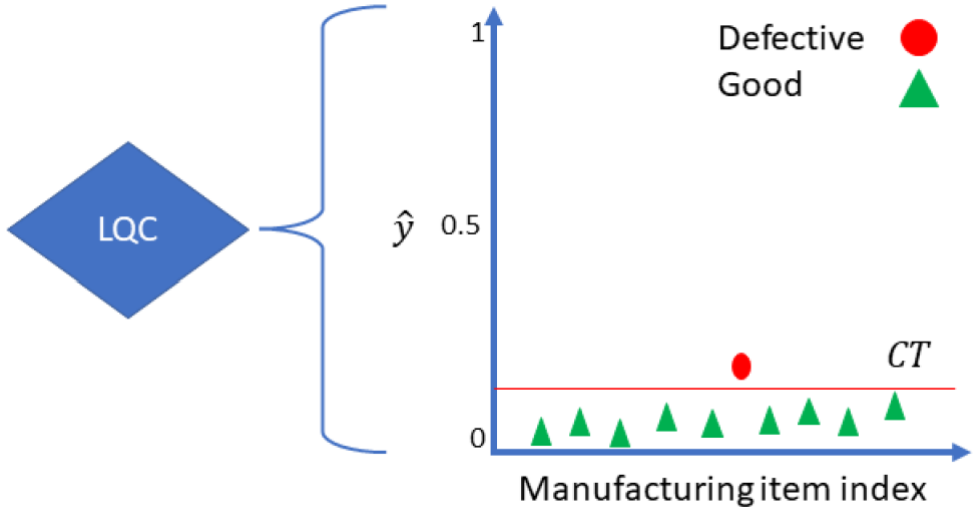
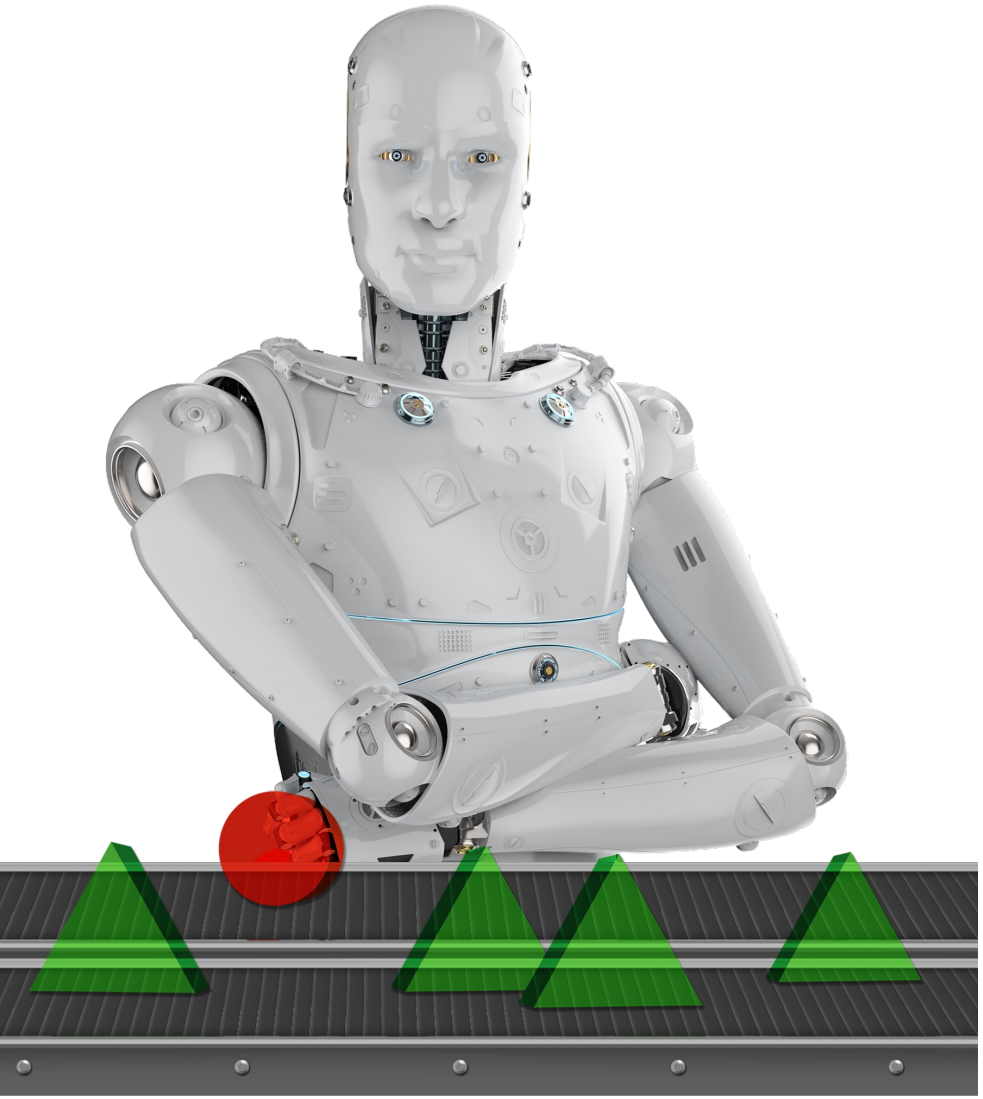


FIGURE 8.6 *LQC* monitoring.

Definition

Remarks

- Machine learning algorithms have the capacity to learn non-linear complex quality pattern that exist in the hyper-dimensional space.
- Because of this ability, oftentimes the classifiers surpassed traditional methods such as control charts in the detection of anomalies, deviation, outliers, or faulty operating conditions.



Applications

Applications

Process controlled by traditional methods such as SPC can be boosted or replaced by a LQC system

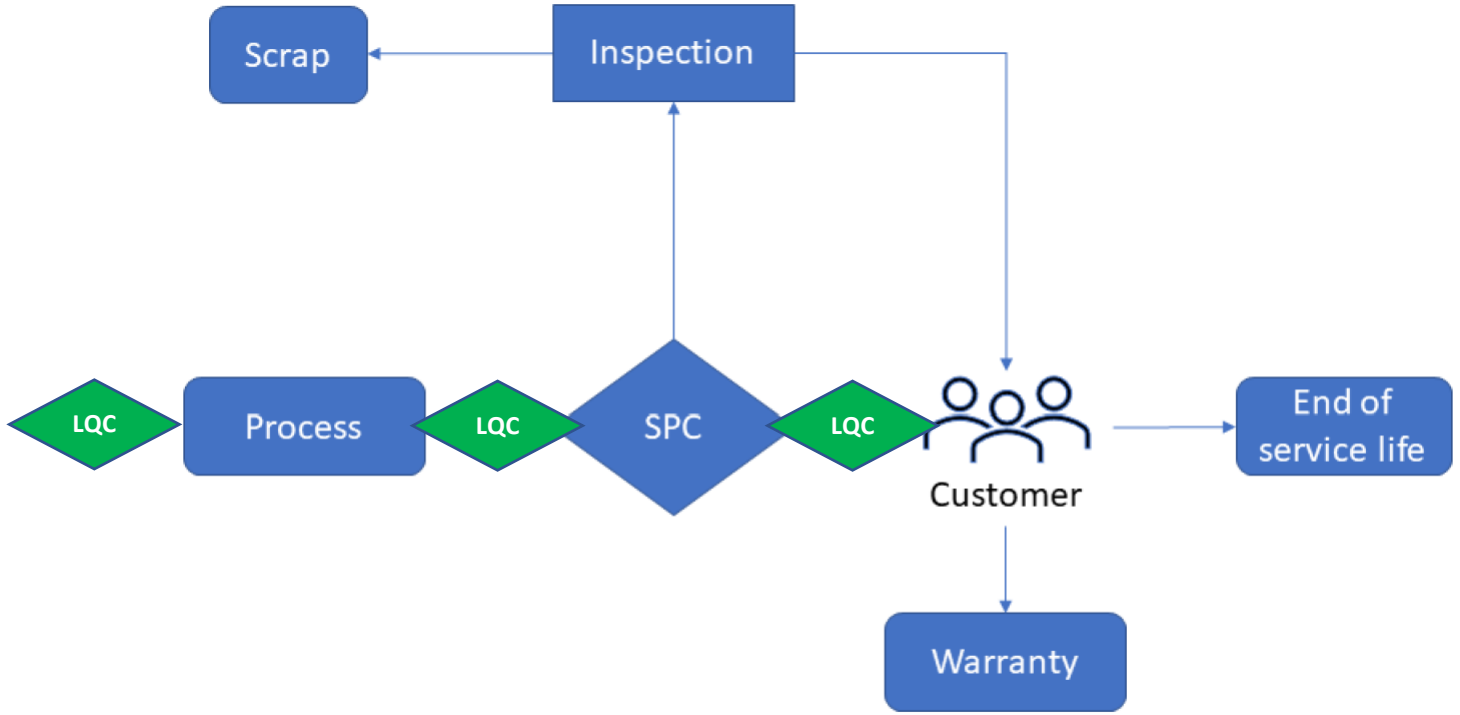


FIGURE 8.7 *SPC* monitoring system.

Applications

- Vision systems have the capacity to increase automation and efficiency.
  - Manual or visual monitoring approaches are historically 80% accurate

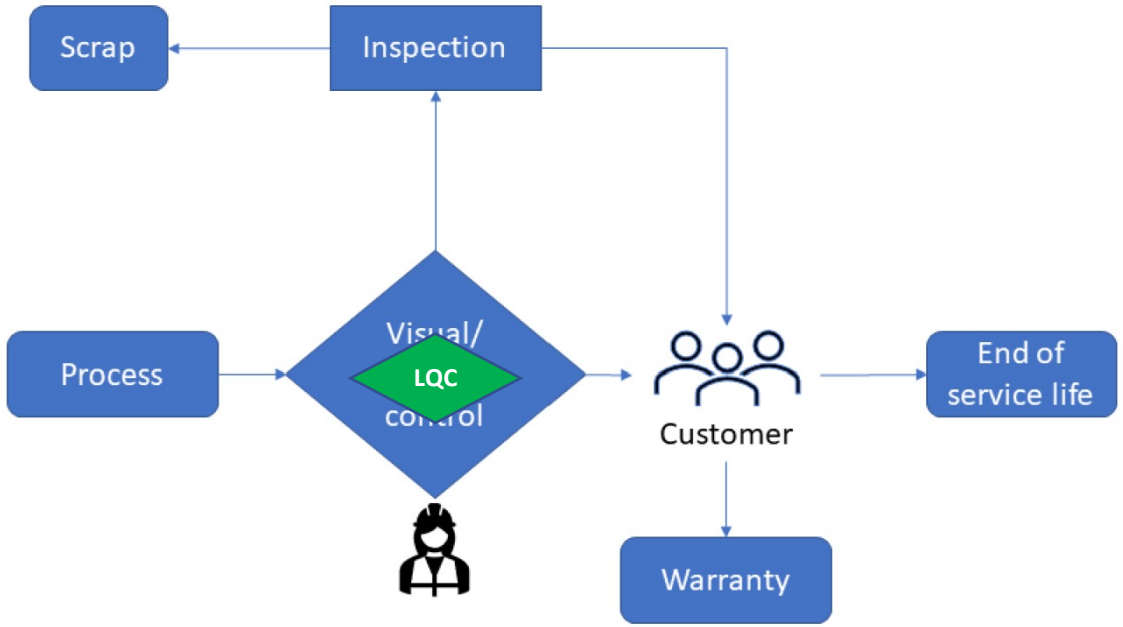


FIGURE 8.8 Visual or manual monitoring system.

Applications

- Extreme cases, empirical data is generated to develop LQC system; even if the physics of the process is not totally understood.
- Data-driven insights may speed up the process understanding from a physics perspective.

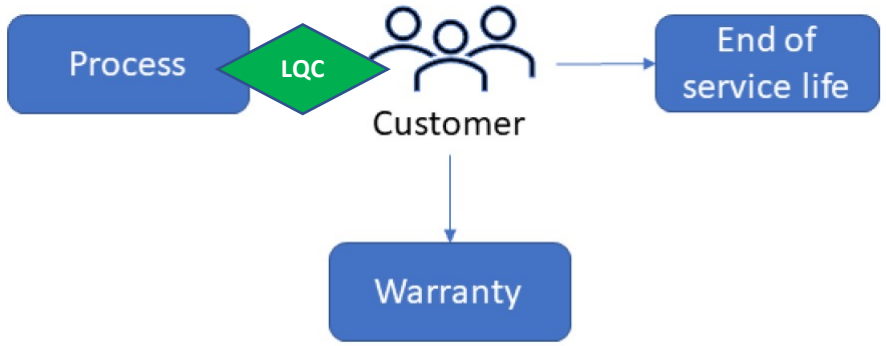
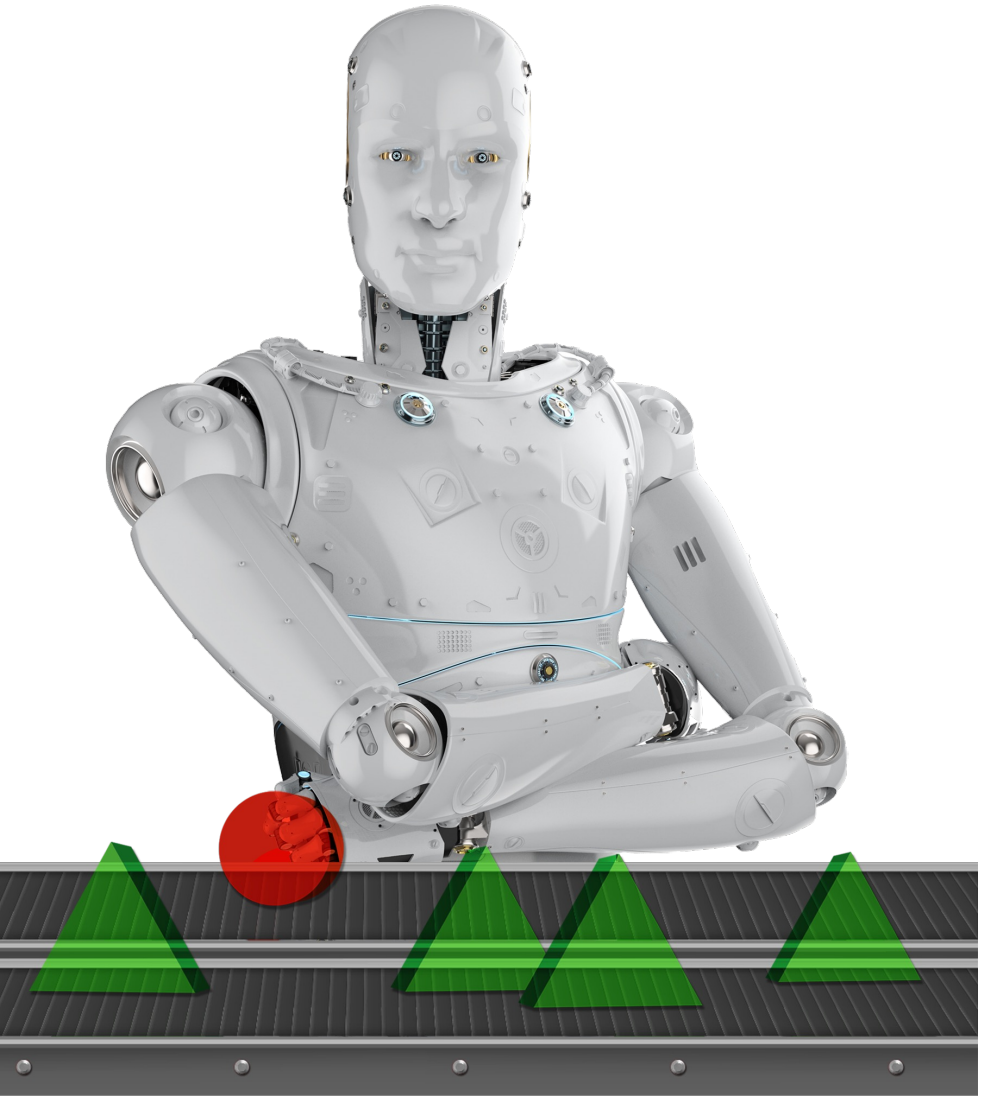


FIGURE 8.9 No *QC* monitoring system.

Applications

Remarks

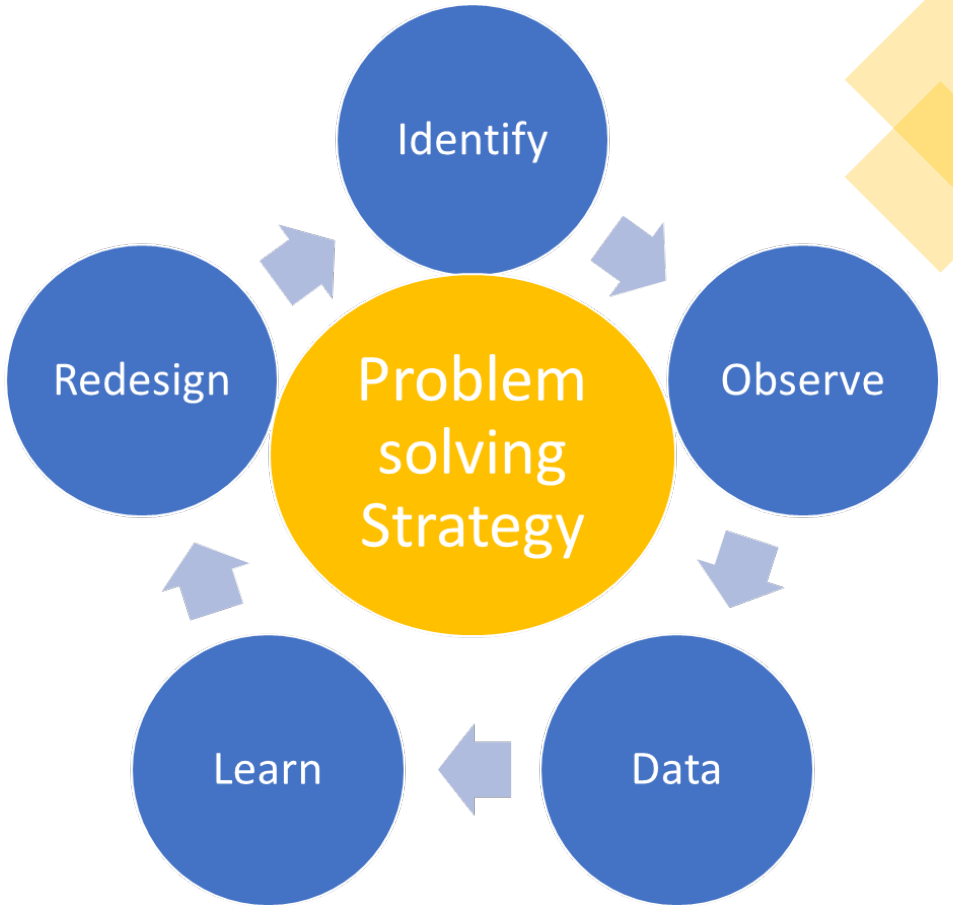
- LQC applications have the potential of moving a step forward the frontiers of manufacturing science, by either:
  - Boosting traditional methods.
  - Replacing visual inspections.
  - Developing a QC system, even if the process is not totally understood from a physics perspective.



The problem-solving strategy

The problem-solving strategy

- Five-step problem solving strategy (PSS) is proposed by the authors to guide the implementation and solution process of LQC.
- Based on theory, empirical evidence, and the experience of the authors solving complex manufacturing quality problems.
- Each step addresses the challenges posed by ML to the application of QC in manufacturing.
- Main objective of guiding the solution process, is to democratize artificial intelligence.



The problem-solving strategy

Identify - select the right problem, define learning targets, and expected benefits.

- Today, the hype and success stories of AI have influenced most organizations to have an interest in deploying an AI initiative.
- According to recent surveys, 80-87% of projects never make it into production.
- Improper project selection is the primary cause of this discouraging statistic, as many of these projects are ill-conditioned from the launch.
- Strong project selection approach is founded on deep technical discussions, and business value analysis is required to develop a prioritized portfolio of projects.

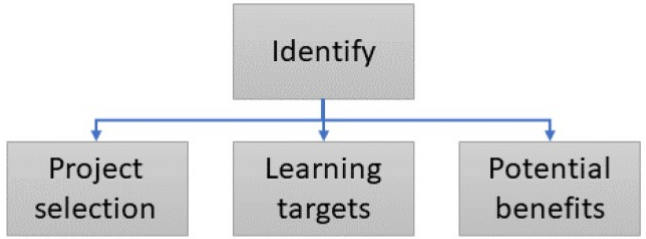


FIGURE 8.11 Step1 - Identify.

The problem-solving strategy

Pick chart borrowed from lean six sigma is useful and widely used, but it fails to provide enough arguments to find the winners.

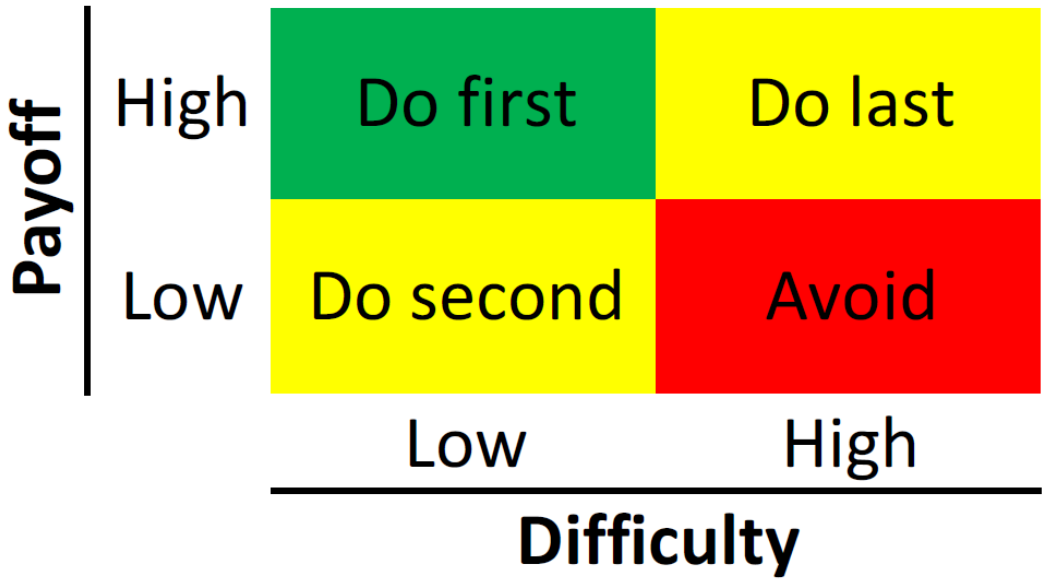


FIGURE 8.12 Pick chart.

The problem-solving strategy

1. Can the problem be formulated as binary classification problem?
2. What is the profitability?
3. Are predictor features available?
4. Are all sources of variation captured by the predictor features?
5. Are the predictor features strongly related to the underlying physics of the project?
6. Is the training data (i.e., sample size) big enough?
7. Does the system exhibit chaotic behavior?
8. Are the quality labels (i.e., good, defective) available?
9. If there are no quality labels, how long it would take to generate labels?
10. Can warranty data be used?
11. Does the project align with the overall team strategy and expertise?
12. In terms of plant dynamics and restrictions (e.g, cycle times), how difficult it would be to implement the solution?
13. In terms of the plant requirements (i.e., α, β errors), what is the probability to solve the problem?
14. How long it would take?
15. Are machine learning methods more suitable than other conventional statistical quality control methods?
16. Since prediction may not provide causation insights, is there any value in predicting without causation?
17. Can this project leverage/enable more projects?
18. Considering the nature of information technology, business intelligence and descriptive statistics projects, does this project promote learning, support automatic decisions and control?

Question	q_1	q_2	q_3	.	.	.	q_{18}	Total
Weights	$w_1 \in \{0, 1,3,9\}$	w_2	w_3	.	.	.	w_{18}	
Project 1	$p_1 \in \{0,1,3,9\}$	p_2	p_3				p_{18}	
Project 2								
Project 3								
.								
.								
.								
Project n								

FIGURE 8.12 Weighted project decision matrix for *Q4.0*.

The problem-solving strategy

Remarks

- Developing a sustainable solution in manufacturing is very complicated.
  - Highly ambitious “moon shots” are less likely to be successful than “low hanging fruit” projects that enhance business processes.
- Projects with data already available can be a good place to start since data generation is manual and time consuming.
- Overoptimistic bias induced by vendors and the fallacy of planning should be avoided when selecting and defining a project.
- Data science projects are not independent from one other.
  - Once the first successful project is implemented, it will inspire new projects and ignite new ideas towards the development of the mid/long term visions.
- Companies with a well-defined project selection strategy will be more likely to succeed in the era of Q4.0.
- Project portfolio is a living document that needs to be evaluated and redefined often.

The problem-solving strategy

Observe - determine which devices will be used to generate the data and define the communication protocols for data transmission

- Process domain knowledge and communications engineering are required in this step.

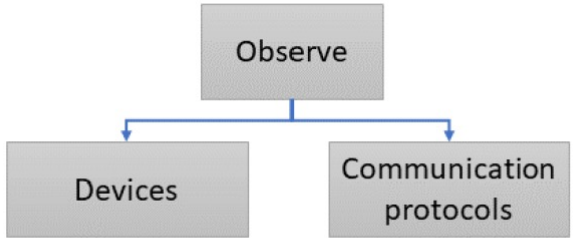


FIGURE 8.13 Step2 - Observe

The problem-solving strategy

Data - generate the learning data

- Generating either the features, signals, or images, and labeling each sample.
- CSC technologies are also defined in this step.

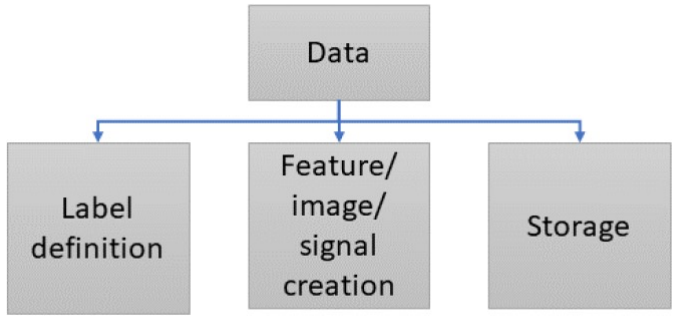


FIGURE 8.14 Step3 - Data.

The problem-solving strategy

Learn - develop the classifier

- Identifying the driving features support the engineering efforts
- Training and testing several MLA
- Exploring different fusion rules of the classifiers using a meta-learning algorithm to determine if the predictive system of LQC should include a single classifier or an MCS.
- Solution must have the capacity to learn new patterns and forget the old ones (i.e., relearning strategy).
    - This feature allows the solution to sustain over time.

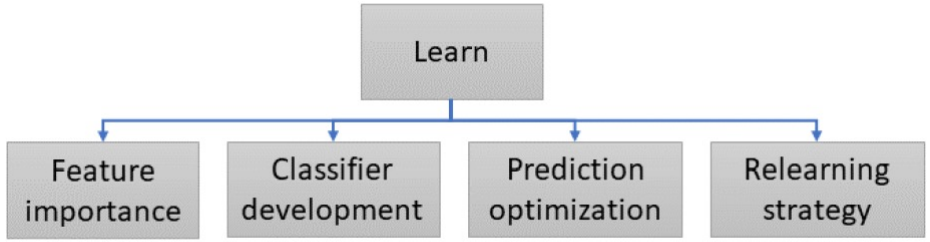


FIGURE 8.15 Step4 - Learn.

The problem-solving strategy

Redesign - leverage the data-driven insights to guide troubleshooting, root-cause analysis, and engineering knowledge discovery efforts aimed at process redesign, so that defects are prevented from a physics perspective.

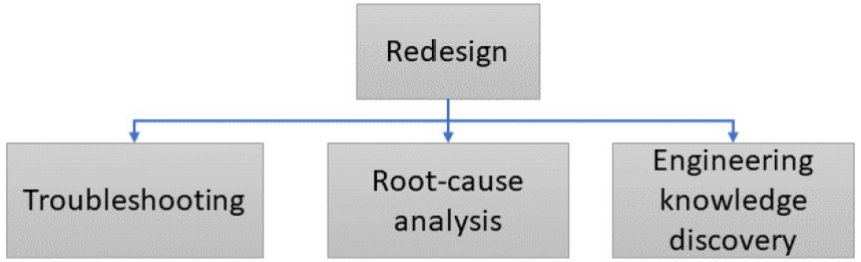
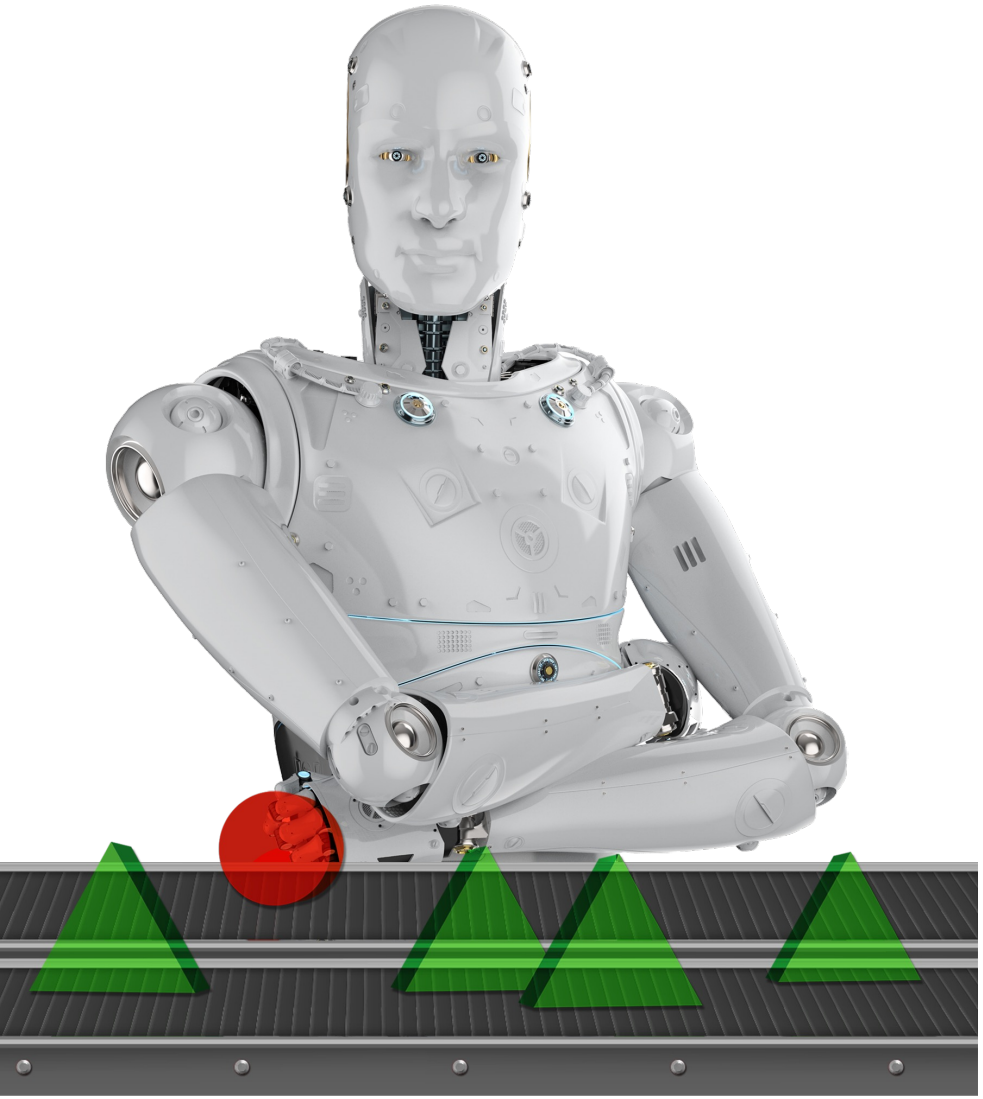


FIGURE 8.19 Step5 - Redesign.



Boosting SPC

Boosting SPC

Virtual case study is presented to illustrate how SPC charts, in some situations, may not detect patterns of concern that exist in the hyper-dimensional space.

- Pattern that exists in a 3D space is developed; three virtual features of sample size ten are created (including four defective items)

TABLE 4.4 Virtual feature values and the associated quality status of each sample.

Index	Feature 1	Feature 2	Feature 3	Quality status
1	89	91	95	Defective
2	33	20	98	Good
3	95	93	88	Defective
4	40	45	56	Good
5	80	86	92	Defective
6	20	55	12	Good
7	52	60	80	Good
8	90	95	20	Good
9	99	89	60	Good
10	85	82	75	Defective

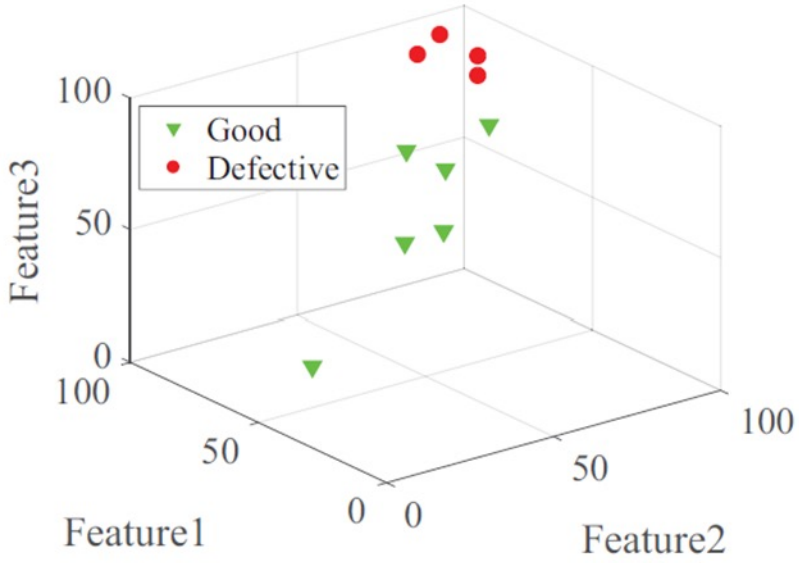


FIGURE 4.24 3*D* quality pattern. Complete separation.

Boosting SPC

- Solution – classifier (machine learning)

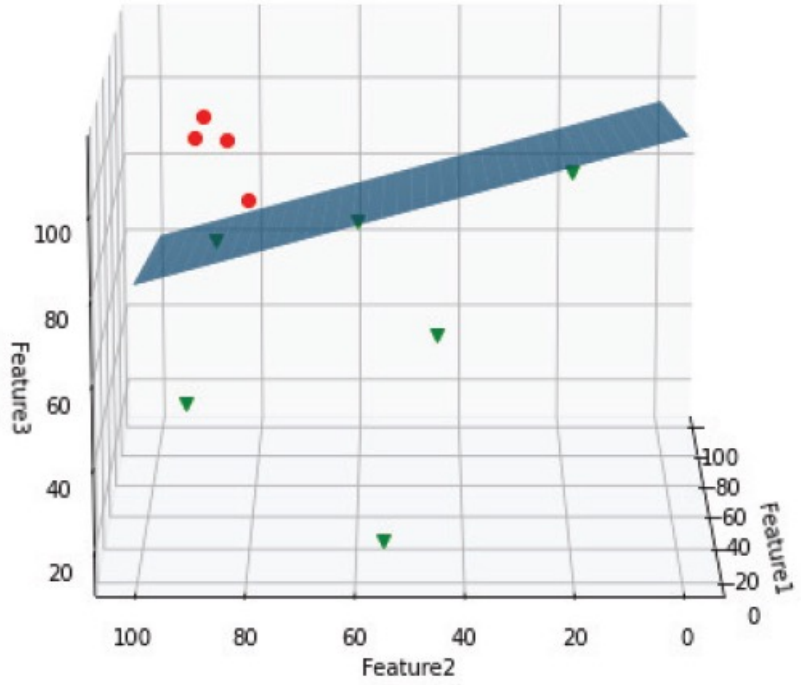


FIGURE 4.27 Linear separation scheme based on the *SVM*.

TABLE 4.5 Confusion matrix for this case study using SVM.

	Predicted good	Predicted bad
Good item	4	0
Defective item	0	6

Boosting SPC

Solution – univariate control charts (SPC)

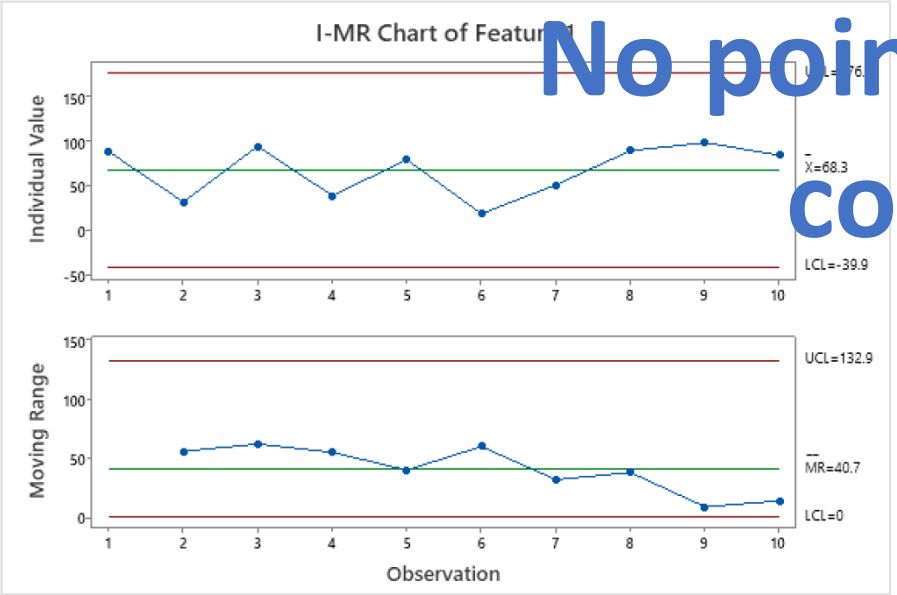


FIGURE 8.22 I-MR univariate control chart - Feature 1.

No point is out of control

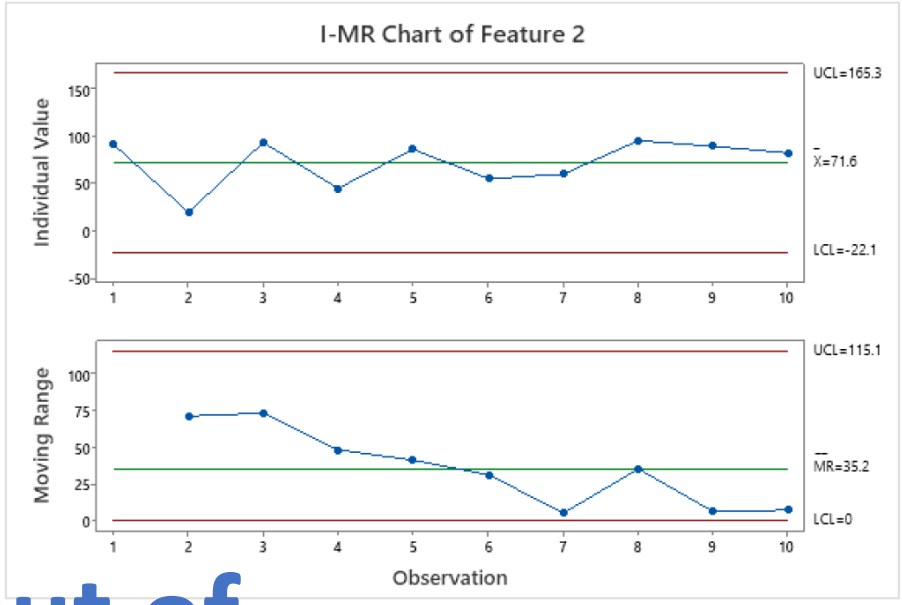


FIGURE 8.23 I-MR univariate control chart - Feature 2.

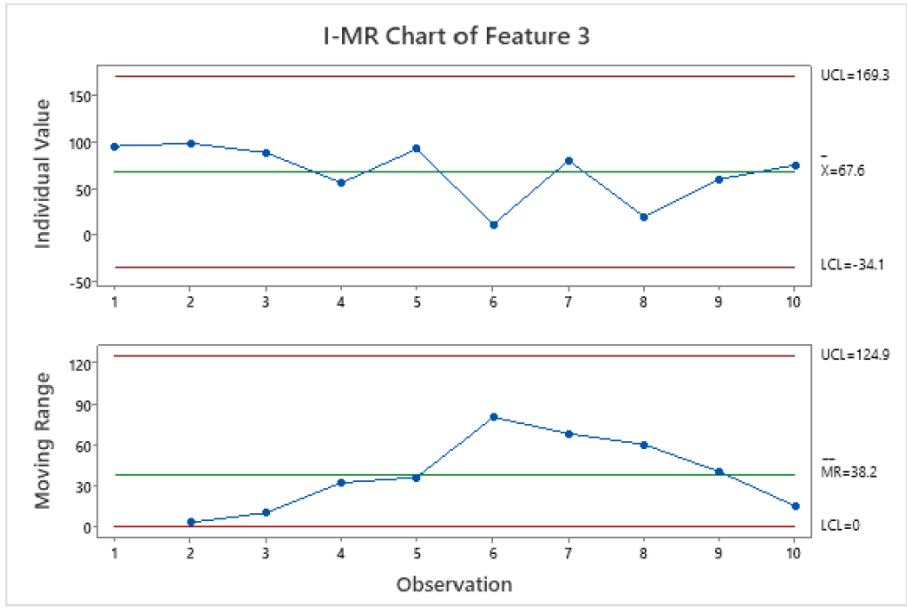


FIGURE 8.24 I-MR univariate control chart - Feature 3.

Boosting SPC

Solution – multivariate control charts (SPC)

Table 4. Virtual feature values and the associated quality status of each sample.

Index	Feature 1	Feature 2	Feature 3	Quality status	
1	89	91	95	Defective	
2	33	20	98	Good	FP
3	95	93	88	Defective	
4	40	45	56	Good	
5	80	86	92	Defective	
6	20	55	12	Good	FP
7	52	60	80	Good	
8	90	95	20	Good	
9	99	89	60	Good	
10	85	82	75	Defective	

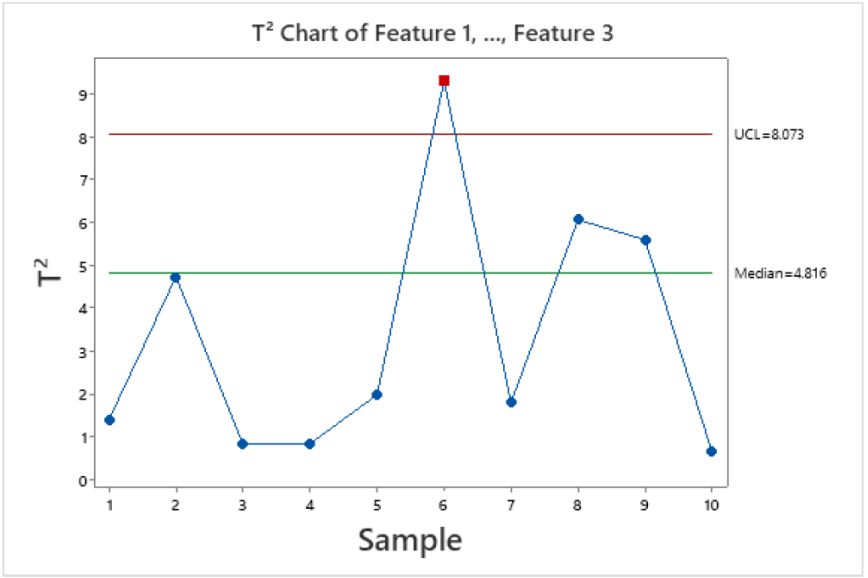


FIGURE 8.25 Multivariate Hotelling's T^2 - Feature 1-3.

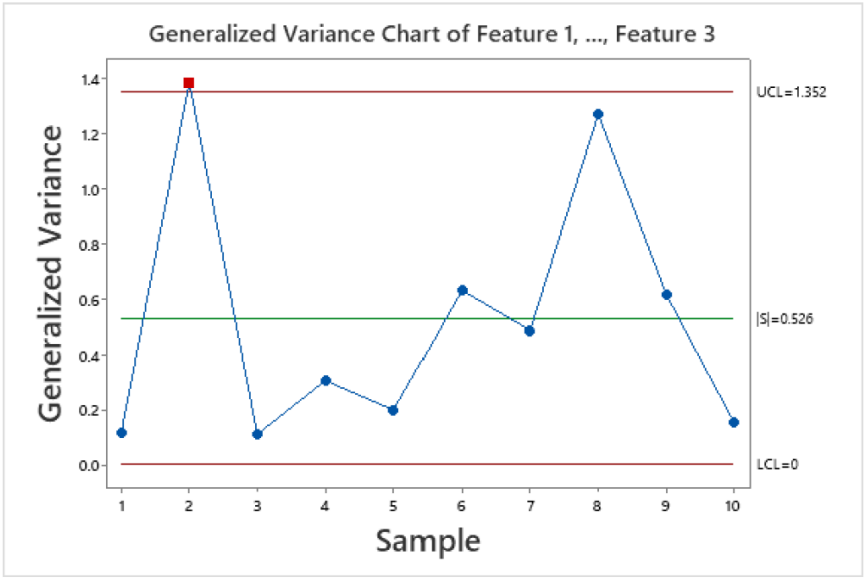


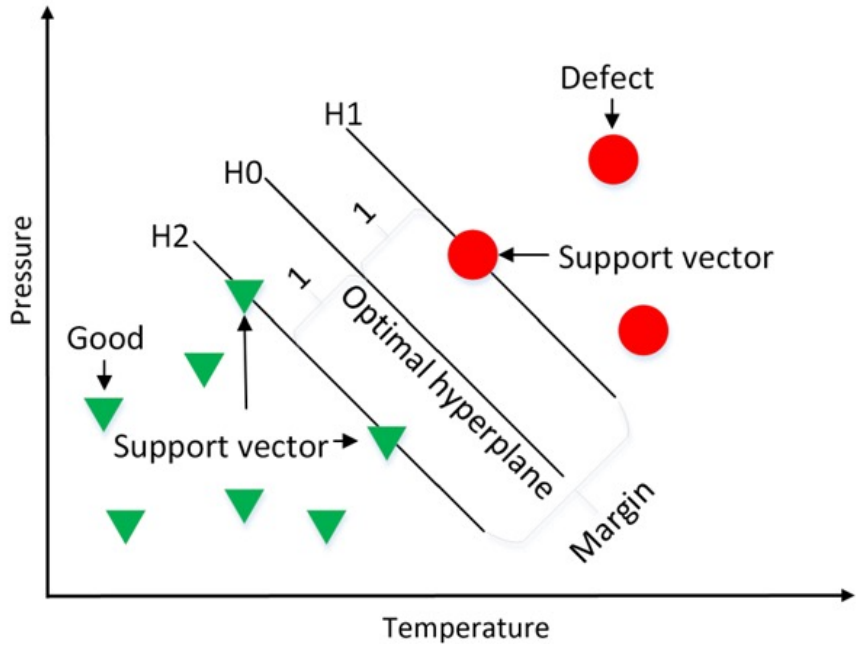
FIGURE 8.26 Multivariate generalized variance control charts - Feature 1-3.

Boosting SPC

- Although the 3D defective quality pattern is clearly observed, the control charts did not detect any defective items.
- Instead, the multidimensional charts created two false positives; two good-quality items were flagged as defectives during the manufacturing process.
- Virtual case study, demonstrated how widely used traditional quality may be boosted by LQC.

Learning the pattern

- SVM finds the most optimal decision boundary
- Optimal decision boundary maximizes the margin from the nearest points of all the classes
- Nearest points from the decision boundary are called support vectors
- Decision boundary in case of support vector machines is called the optimal hyperplane



For the mathematical description refer to: A. Ben-Hur and J. Weston, "A Users Guide to Support Vector Machines," in Data mining techniques for the life sciences. Springer, 2010, pp. 223–239.

Activity 4

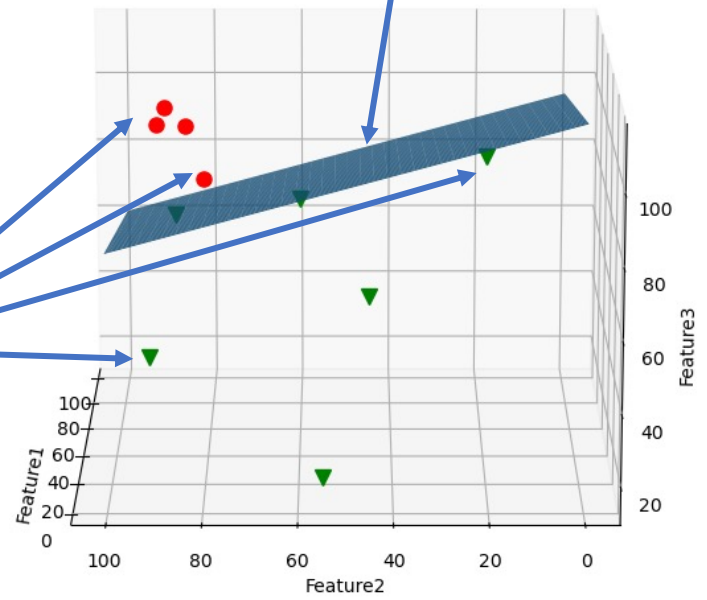
Training a classifier(10mins)

Table 4. Virtual feature values and the associated quality status of each sample.

Index	Feature 1	Feature 2	Feature 3	Quality status
1	89	91	95	Defective
2	33	20	98	Good
3	95	93	88	Defective
4	40	45	56	Good
5	80	86	92	Defective
6	20	55	12	Good
7	52	60	80	Good
8	90	95	20	Good
9	99	89	60	Good
10	85	82	75	Defective

Learn the separating hyperplane using the SVM

Training data

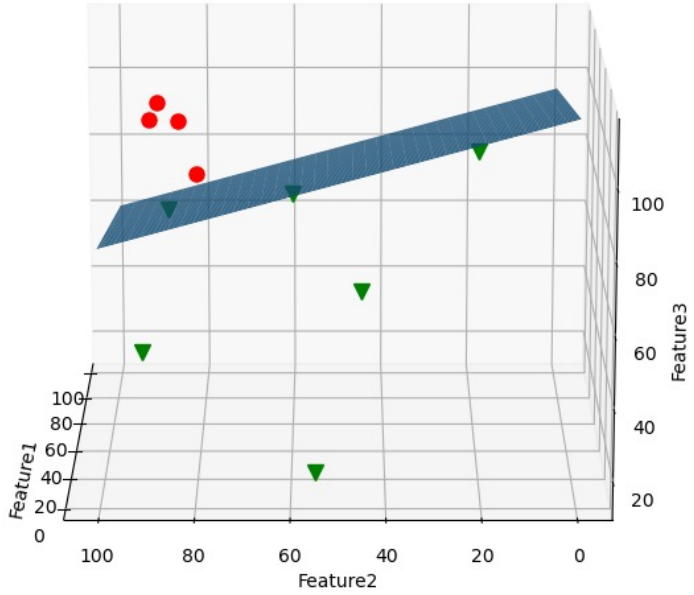


Discussions, 5mins

Activity 4

Training a classifier(10mins**)**

SVM_training_virtual_case

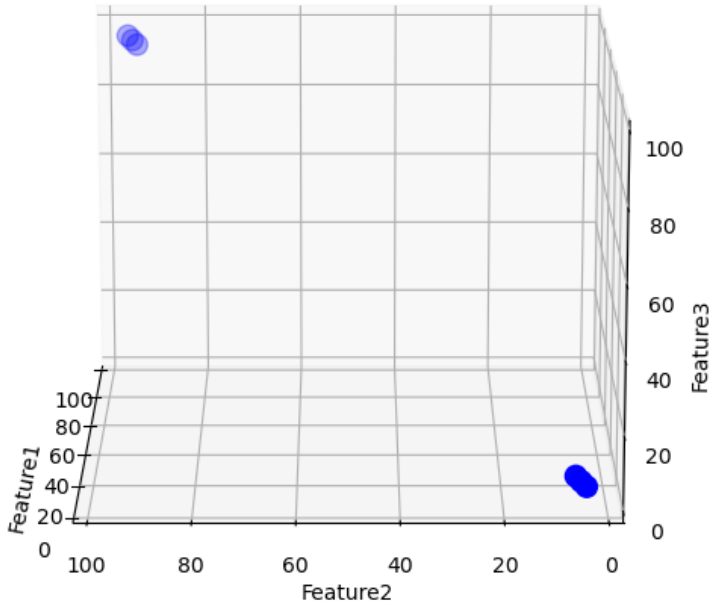


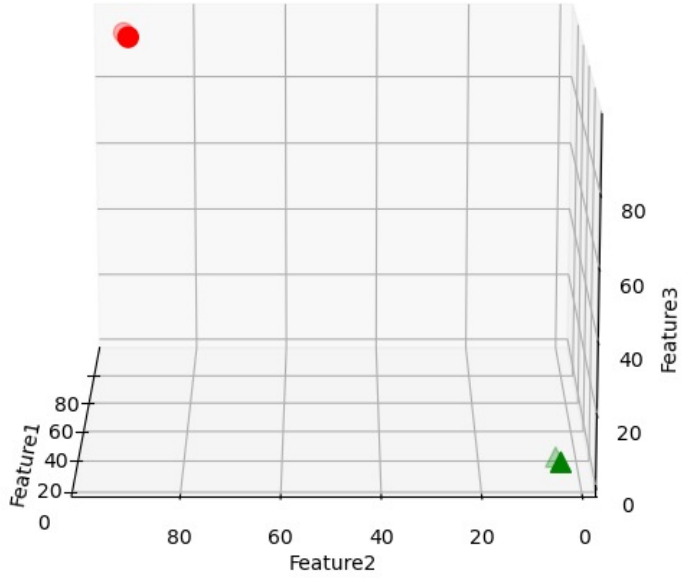
Discussions, 5mins

Activity 4

Training a classifier(10mins**)**

SVM_virtual_case_solution_plot





Discussions, 5mins


The End