제조 빅데이터 전문가 과정

빅데이터 기반의 생산성 효율

(주)에스투비즈
대표컨설턴트
이 이 백

Content

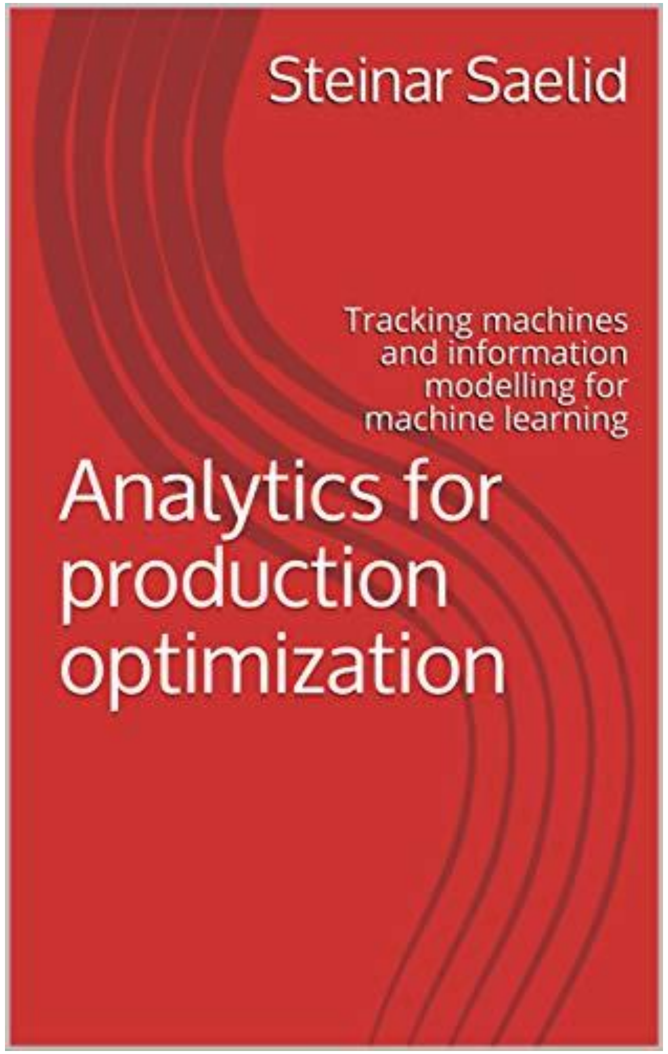
03

설비예지보전 및 생산 최적화

생산 최적화 프로세스

생산 최적화

- 실질 생산량과 기대 생산량 간의 Gap 관리 통제
- 기대 생산 성과를 달성하기 위한 자산 관리
- 가용 자원의 효율성 제고



PREDICTIVE ANALYTICS TO DRIVE REAL TIME PRODUCTION OPTIMIZATION

Why ConocoPhillips is taking the next step toward digital transformation

Mark Hutcherson, Director, Operations Excellence, ConocoPhillips.

Mark began his career with Exxon-Mobil in 2001 in facilities and project engineering roles. He joined Burlington Resources in 2005 and after the acquisition by ConocoPhillips in 2006, he transitioned to roles in production and completions engineering, supporting conventional oil and gas assets.

Mark assumed his current role in mid-2017. He also holds a B.S. in Civil Engineering from Kansas State University.

Mark, tell us about your background and current role at ConocoPhillips.

I've been in the oil industry since 2001 and joined ConocoPhillips in 2006. I've worked my way through various roles in facilities and production engineering and project management. I also have experience managing teams and most recently became the Director of Operations Excellence.

Since 2015, our Operational Excellence (OE) program has changed vastly. We went from a large centralized group of subject matter experts to only a few in our corporate center. OE remains in our global production organization, but where it used to be an organization of corporate oversight and operational governance, we've evolved to a supporting role that assists business units in continuing their OE journey in their own fit-for-purpose way.

OE has helped us push our independent business units and overall organization to a much higher standard, and that continues today.

Your OE function decreased significantly in 2015. Was that prompted by the oil price downturn?

There were lots of changes taking place in the industry at that time – we needed to adapt to the lower commodity price, so the timing was necessary but convenient. I think the timing was very appropriate for us because the OE concept and what we were trying to push in our framework had already been embedded.

Subject matter experts in OE were embedded back into the business units to directly contribute to the work, but our central OE group can pull on those resources as needed when business units

Production Optimization

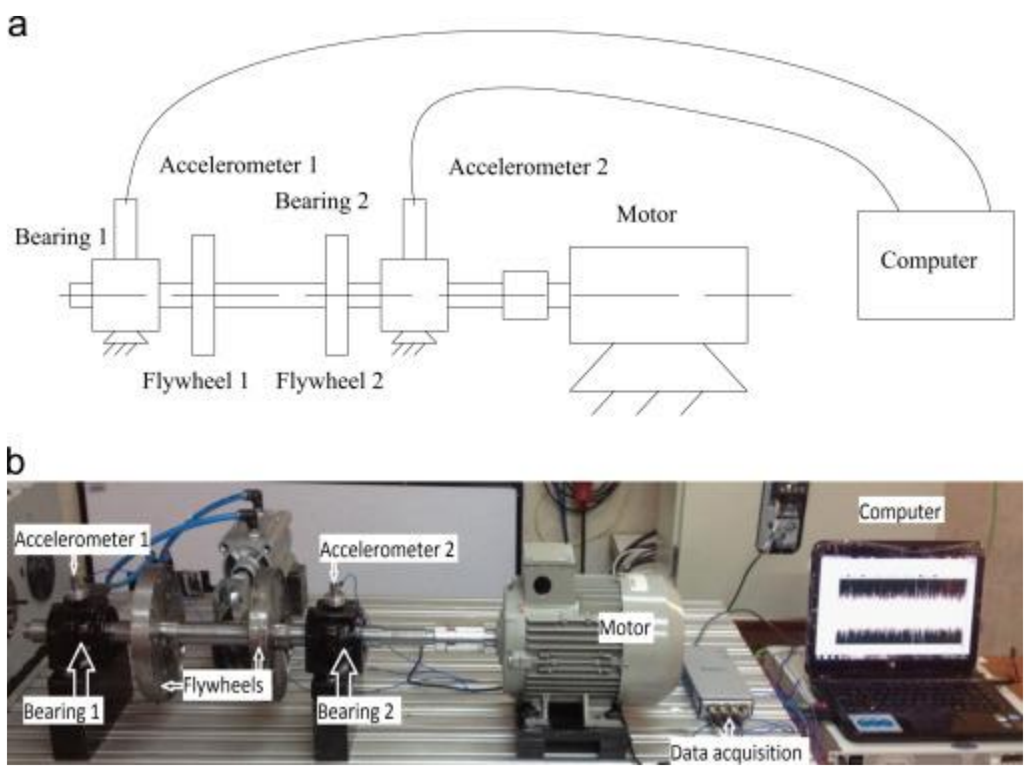
- Equipment Failure & Downtime
- Problem Diagnosis
- Event Detection
- 생산 성능 유지
- 자원 가용 효율성

- Optimization Priblem Solution
  - Lunear Programming
  - Non-Linear Programmig
  - Reinforcement Learning
  - Generative Adversarial Network
- Preventative Maintenance
  - Stratigy - Genetic Algorithm
  - Diagnosis – Decision Tree, Random Forest
- Yield/Demand Forecasting
  - Timeseries Model
  - ML - Support Vector Machine
  - DL – RNN/LSTM

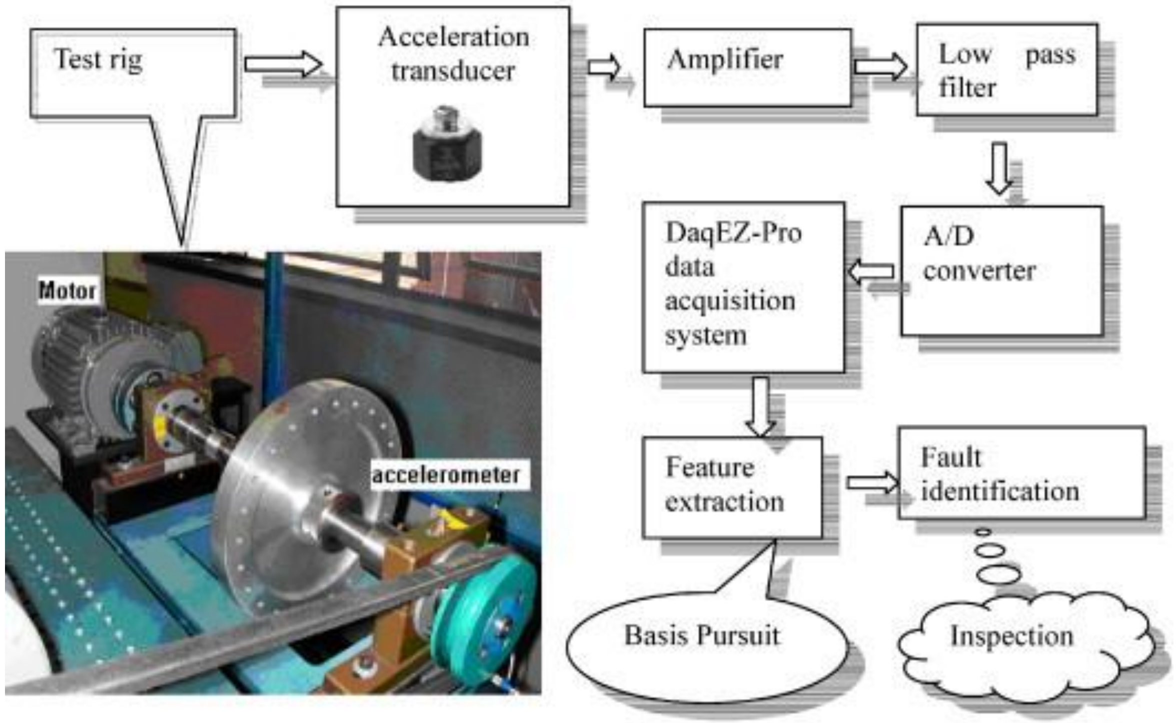
생산 최적화 프로세스

Root Cause Analysis → Preventative Maintenance

▣ Condition Monitoring - Failure Mode & Fault Cases



https://www.sciencedirect.com/science/article/abs/pii/S0952197616000427

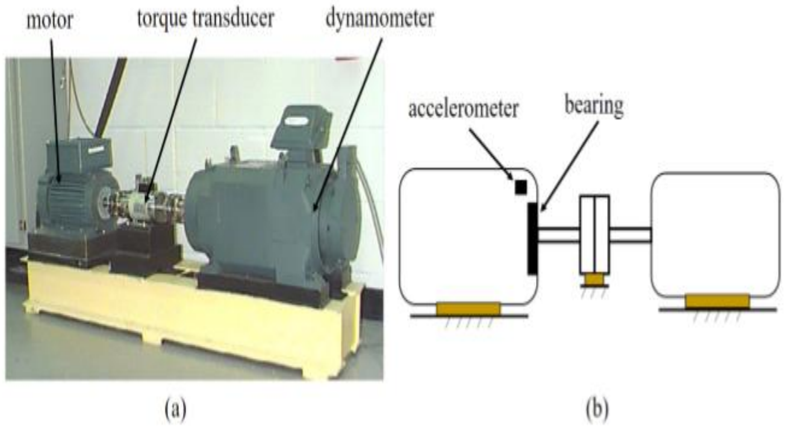


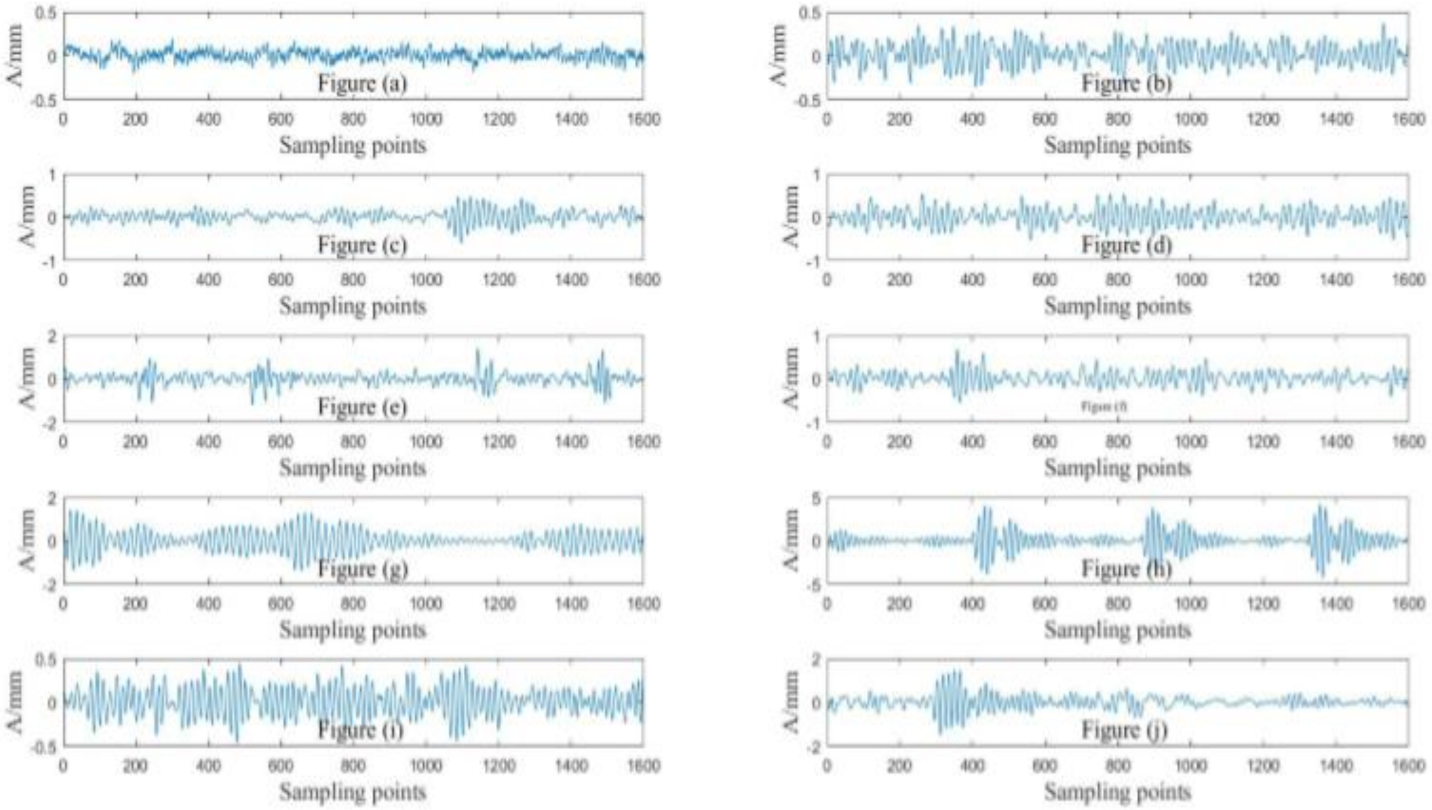
https://www.sciencedirect.com/science/article/pii/S0888327004000354

03. 설비 예지보전 및 생산 최적화

Root Cause Analysis → Preventative Maintenance

▣ 고장 진단 – FDC Parameters





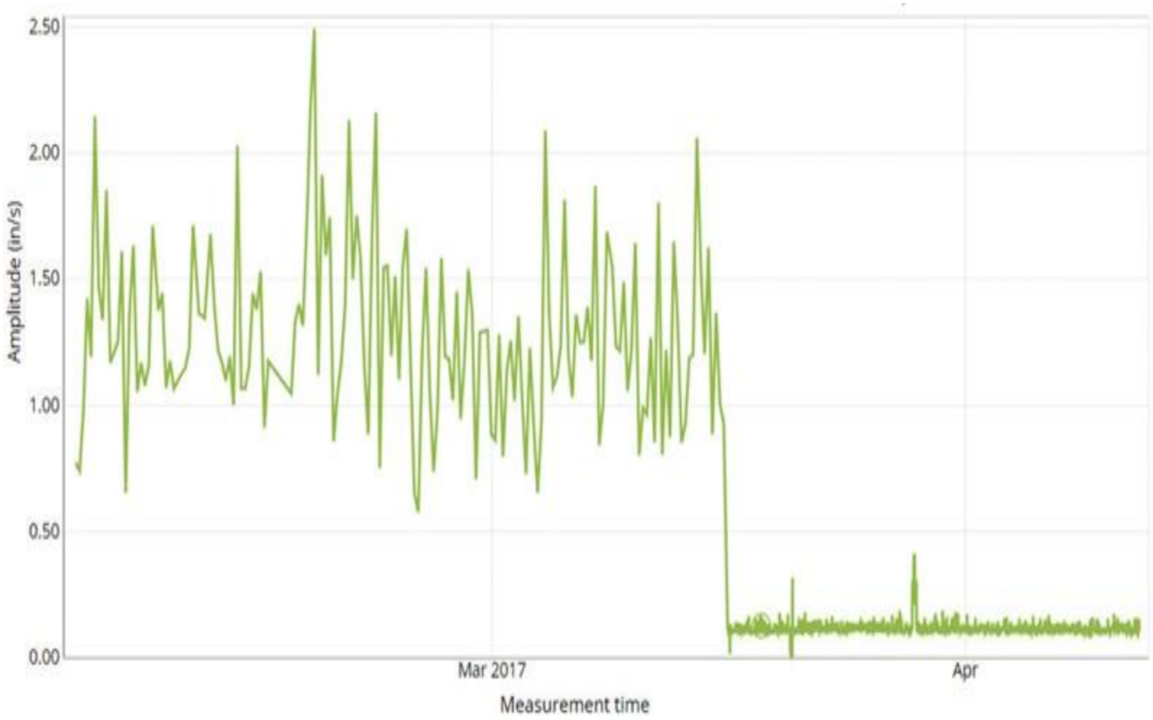
https://www.sv-jme.eu/?ns_articles_pdf=/ns_articles/files/ojs/5249/public/5249-28686-1-PB.pdf&id=6130

03. 설비 예지보전 및 생산 최적화

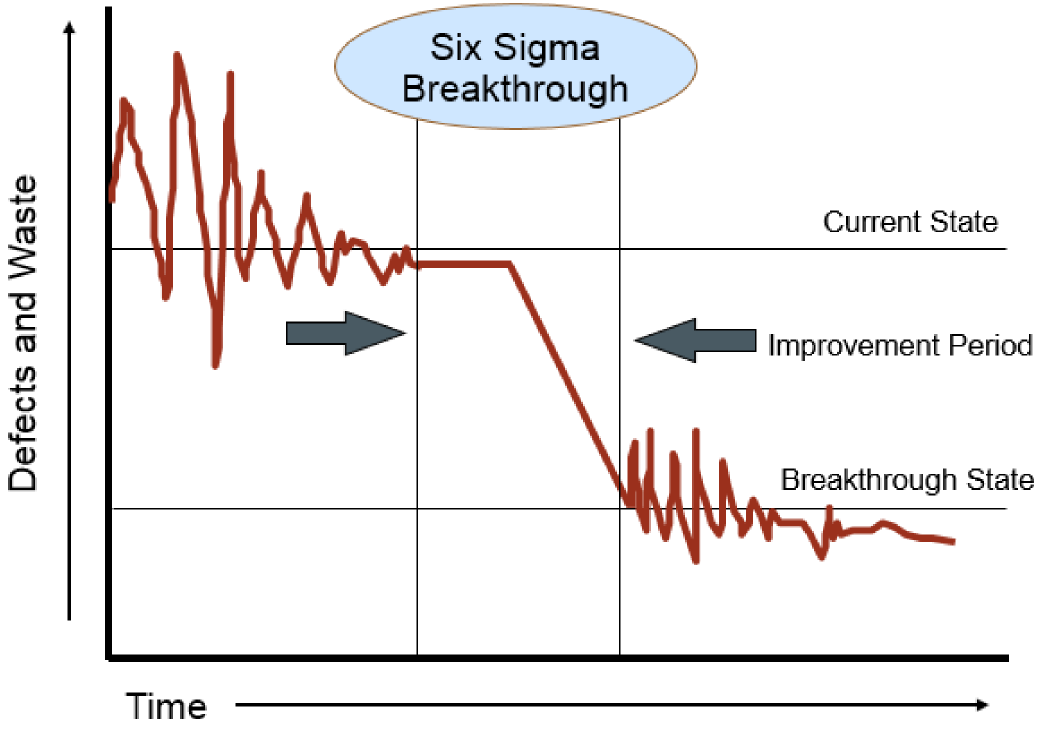
Root Cause Analysis → Preventative Maintenance

▣ FDC Parameters – Threshold disable

▶ Sudden Stop(돌발적 멈춤)



▶ failure symption(고장 징후)

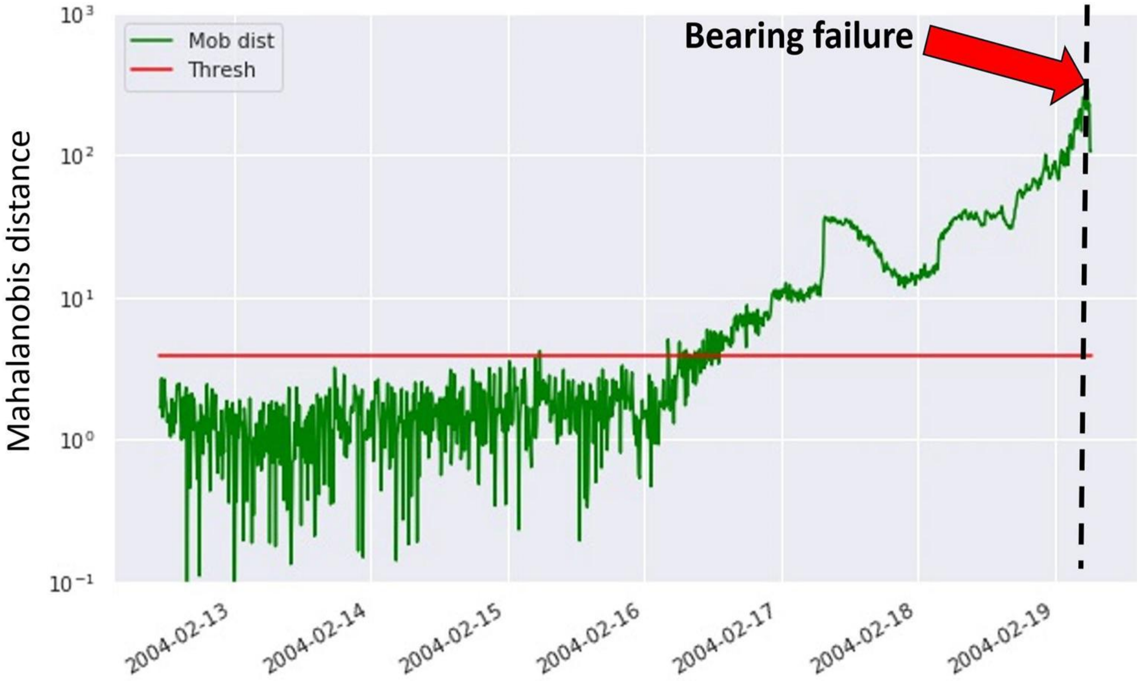


03. 설비 예지보전 및 생산 최적화

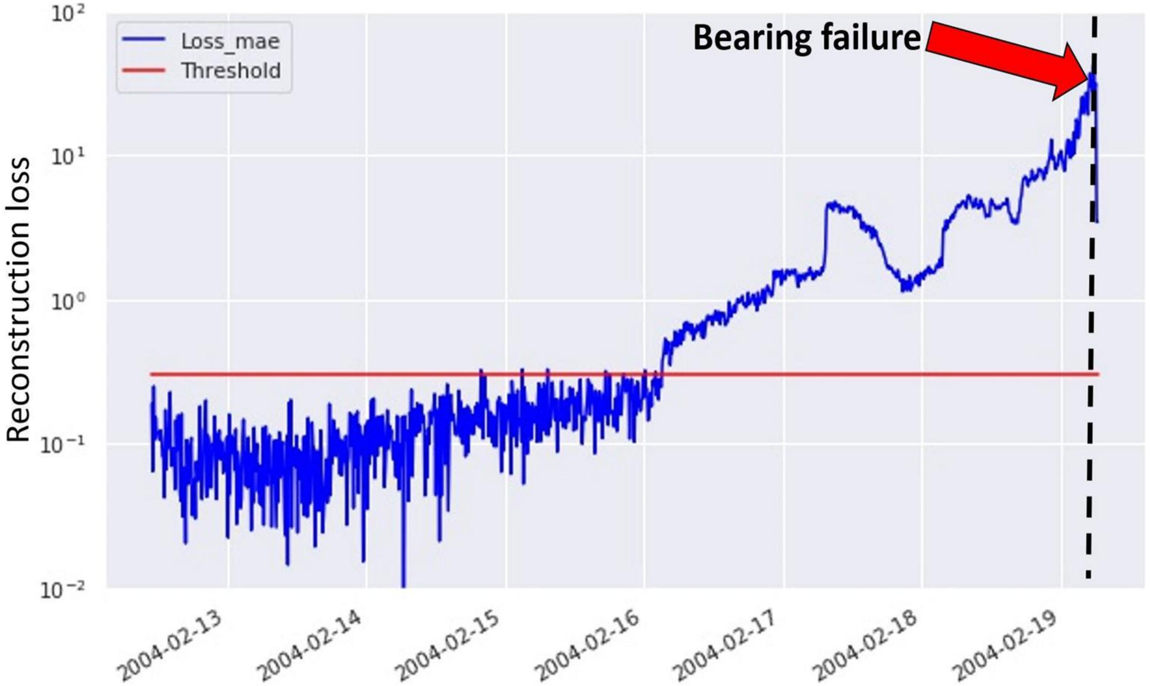
Root Cause Analysis → Preventative Maintenance

▣ FDC Parameters – Anomaly Detection using Mahalanobis and Autoencoder

▶ Mahalanobis Distance



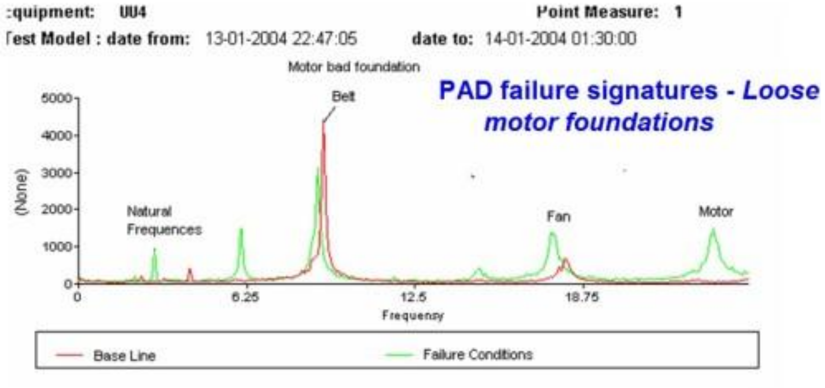
▶ Reconstruction Loss

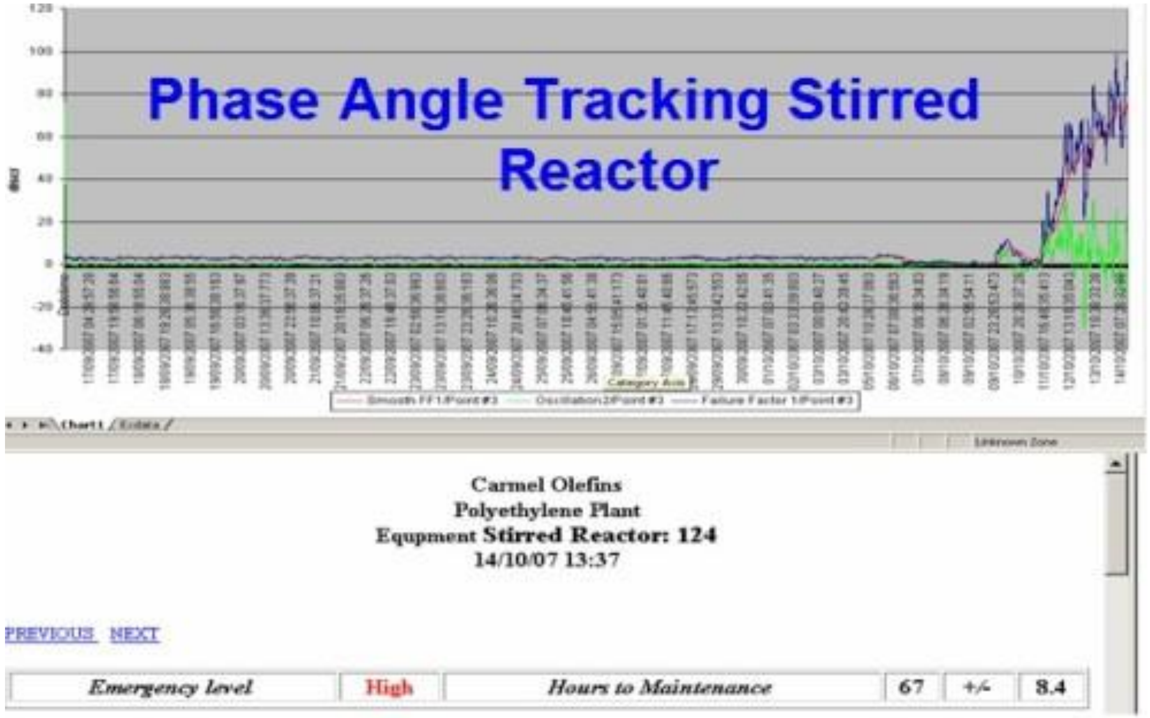


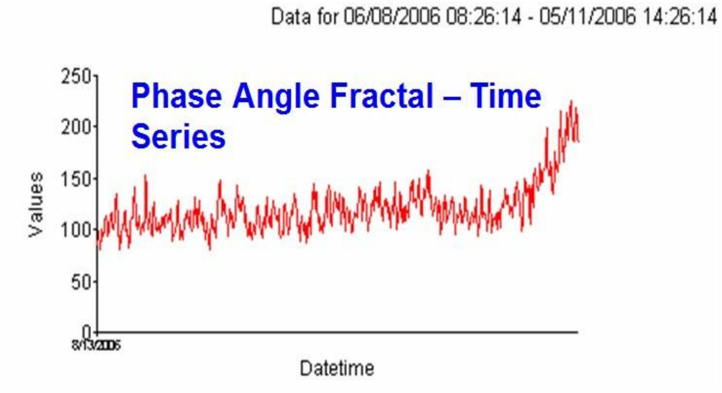
https://towardsdatascience.com/machine-learning-for-anomaly-detection-and-condition-monitoring-d4614e7de770

Root Cause Analysis → Preventative Maintenance

▣ Predictive Monitoring





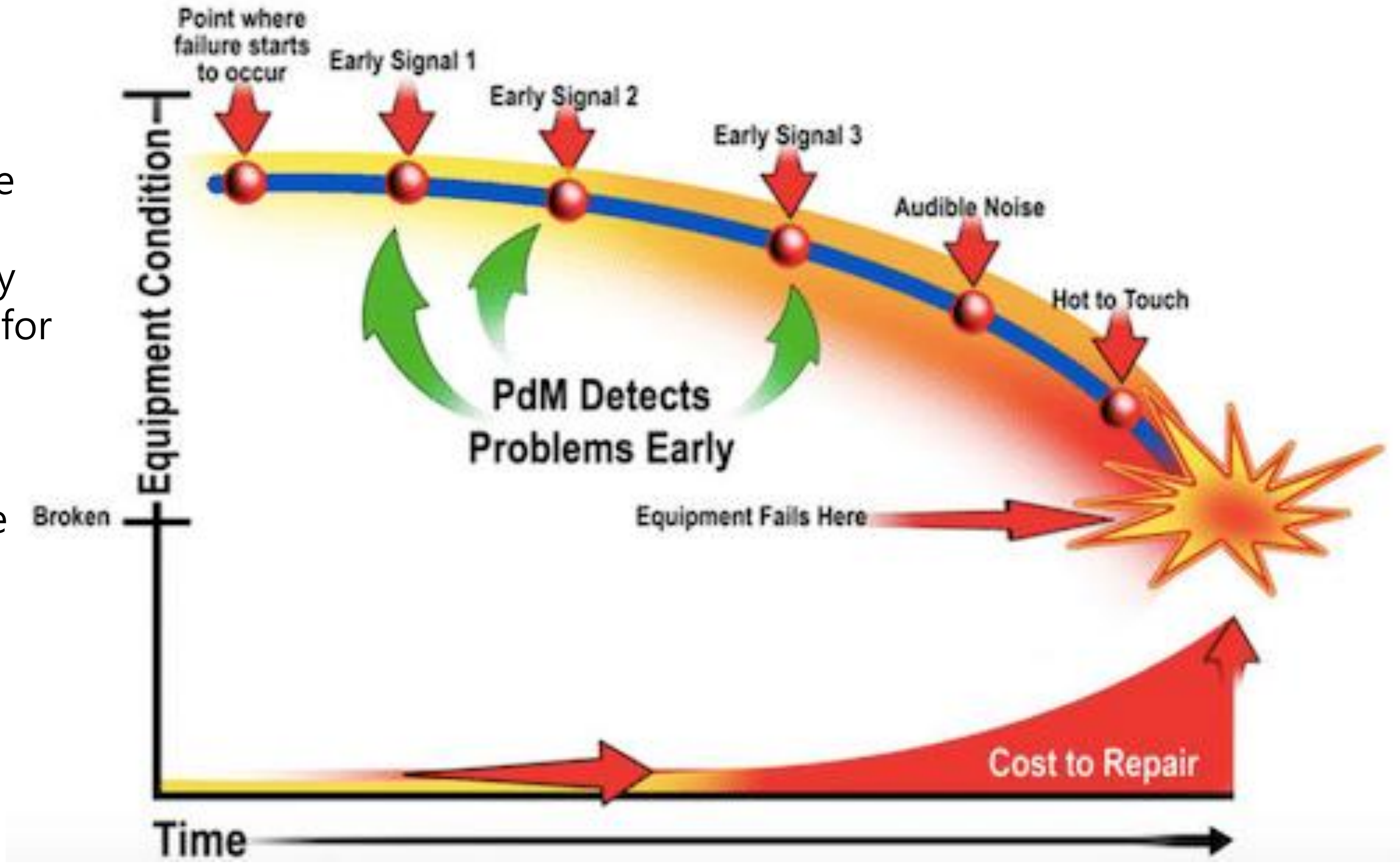


[source] https://www.martecassetsolutions.com.au/product/on-line-predictive-monitoring-of-rotating-machinery/

Root Cause Analysis → Preventative Maintenance

▣ Predictive maintenance의 Benefit

- Maximized performance time
- Improved rhythm and continuity of productive activities
- Increased productivity of workers both directly related to the machines in question and also for those who depend on their success
- Increased time available to plan and organize maintenance and repair activities
- Better planning of interventions and therefore better team preparation
- Better relations between production and maintenance services
- Effective spare parts management
- Reduced energy consumptionl



[source] https://appliedai.com/operations/predictive-maintenance

03. 설비 예지보전 및 생산 최적화

Root Cause Analysis → Preventative Maintenance

▣ Fault Detection using DT

Early Detection of Bearing Damage by Means of Decision Trees

Bovic Kilundu *, Christophe Letot *, Pierre Dehombreux *, Xavier Chiementin **

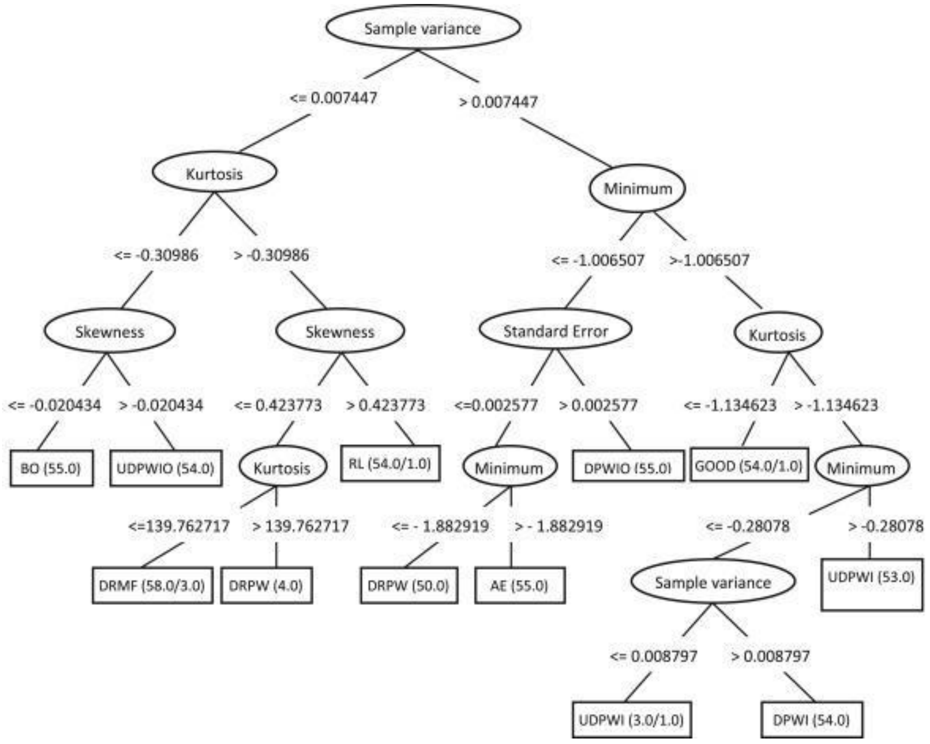
⊞ Show more

https://doi.org/10.3182/20081205-2-CL-4009.00038

Abstract

This paper presents a procedure for early detection of rolling bearing damages on the basis of vibration measurements. First, an envelope analysis is performed on bandpass filtered signals. For each frequency range, a feature indicator is defined as sum of spectral lines. These features are passed through a principal component model to generate a single variable which allows to track change in the bearing health. Thresholds and rules for early detection are learned thanks to decision trees. Experimental results demonstrate that this procedure enables early detection of bearing defects.

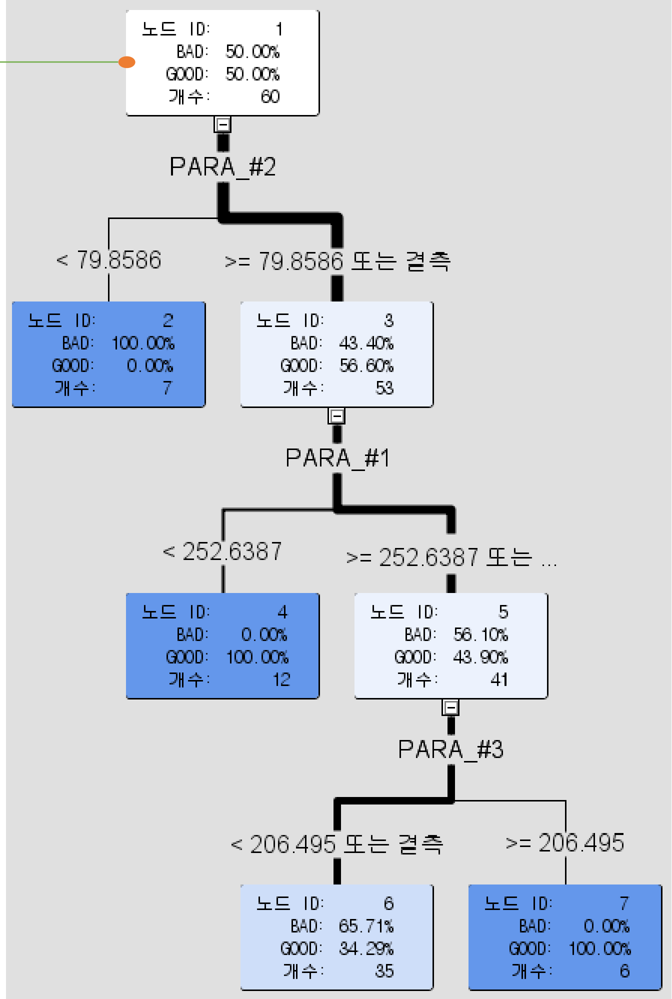
https://www.sciencedirect.com/science/article/pii/S1474667015355671



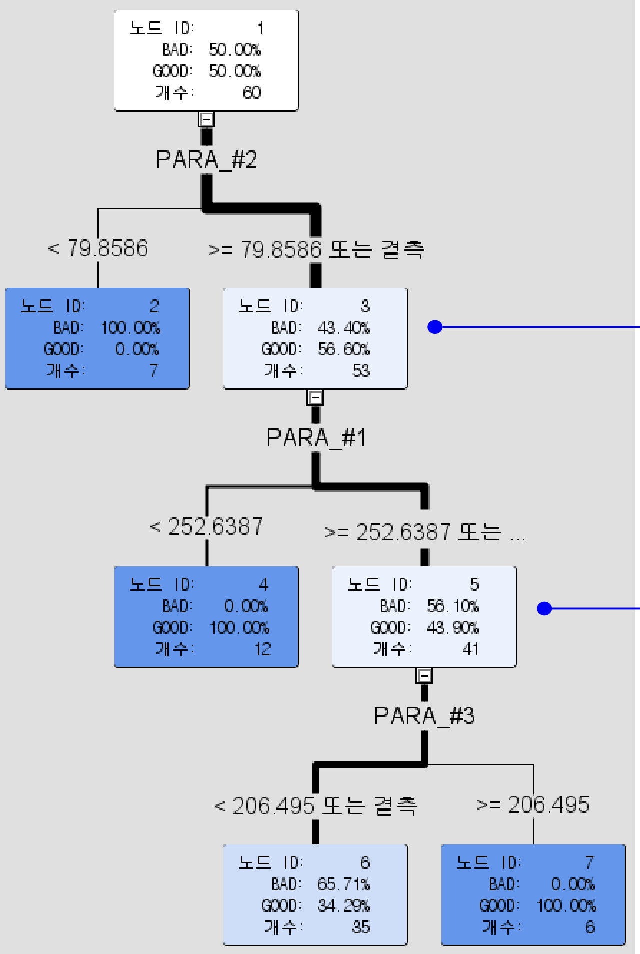
https://www.sciencedirect.com/science/article/pii/S2215098614000780

Root Cause Analysis → Preventative Maintenance → Decision Tree

- Tree의 구성
  - Node : Root Node, Intermediate Node, Terminal Node
  - Leaf : Terminal Node
  - Branch : 관찰치 전달 경로, Node Split
  - Depth : 상위 Node(Parent or Ancestor)로부터 하위 Node(Child or Descendant) 까지의 Length
- Node
  - Root Node
  - 범주형 또는 연속형 Label이 있는 모든 관찰치 집합의 특징(feature)을 학습을 시작하기 위한 시작점
  - 발상
    - ✓ 뿌리에서 영양분을 공급받아 과실 생산
    - ✓ 열매의 수(=학습 집합 관찰치수) : 400개
    - ✓ 열매의 품질
      - ➢ 우수 : 불량 = 300 : 100



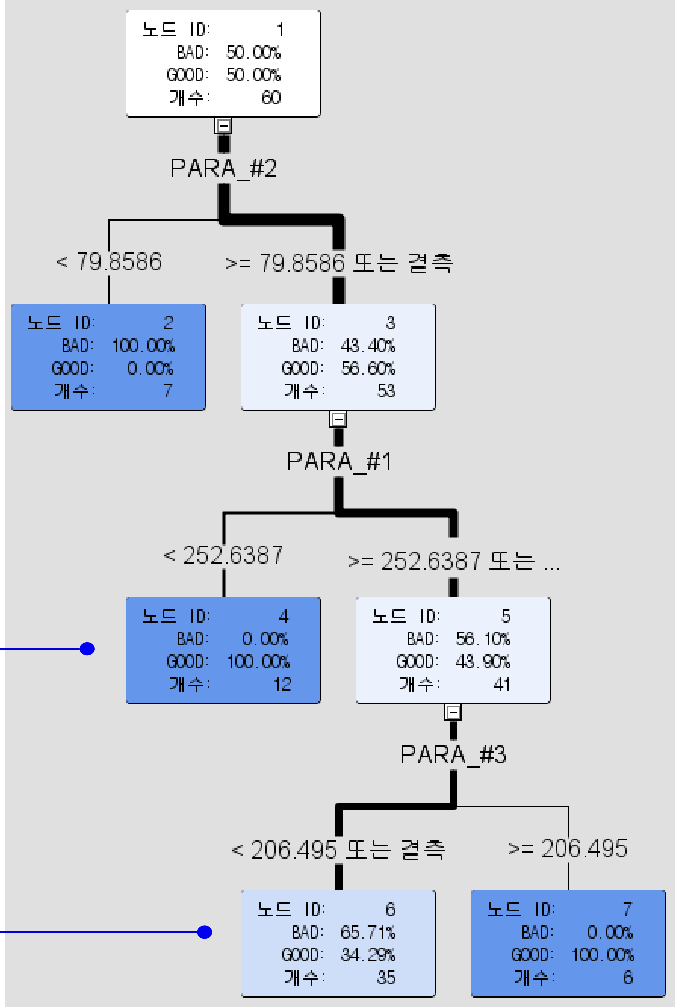
Root Cause Analysis → Preventative Maintenance → Decision Tree



- Intermediate Node
  - 상위 중간 노드와 하위 중간 노드는 가지로 연결
  - 가지는 관찰치를 특징 학습 과정을 통해 하위 노드로 전달하는 배송로 역할
  - 일반적으로는 좌우 두개의 가지를 통해 노드가 연결됨

Root Cause Analysis → Preventative Maintenance → Decision Tree

- Node
  - Terminal Node
  - 특정 관찰치(observation)가 속하는 최종 그룹 단위
  - 발상
    - ✓ 태양 빛을 많이 받은 쪽(Node)의 과실들이 그렇지 않은 쪽(Node)의 과실들보다 품질이 우수함 → 하나의 규칙
    - ✓ 빛을 많이 받은 쪽
      - ➢ 우수 : 불량 = 95 : 5
    - ✓ 빛을 가장 적게 받은 쪽
      - ➢ 우수 : 불량 = 15 : 30

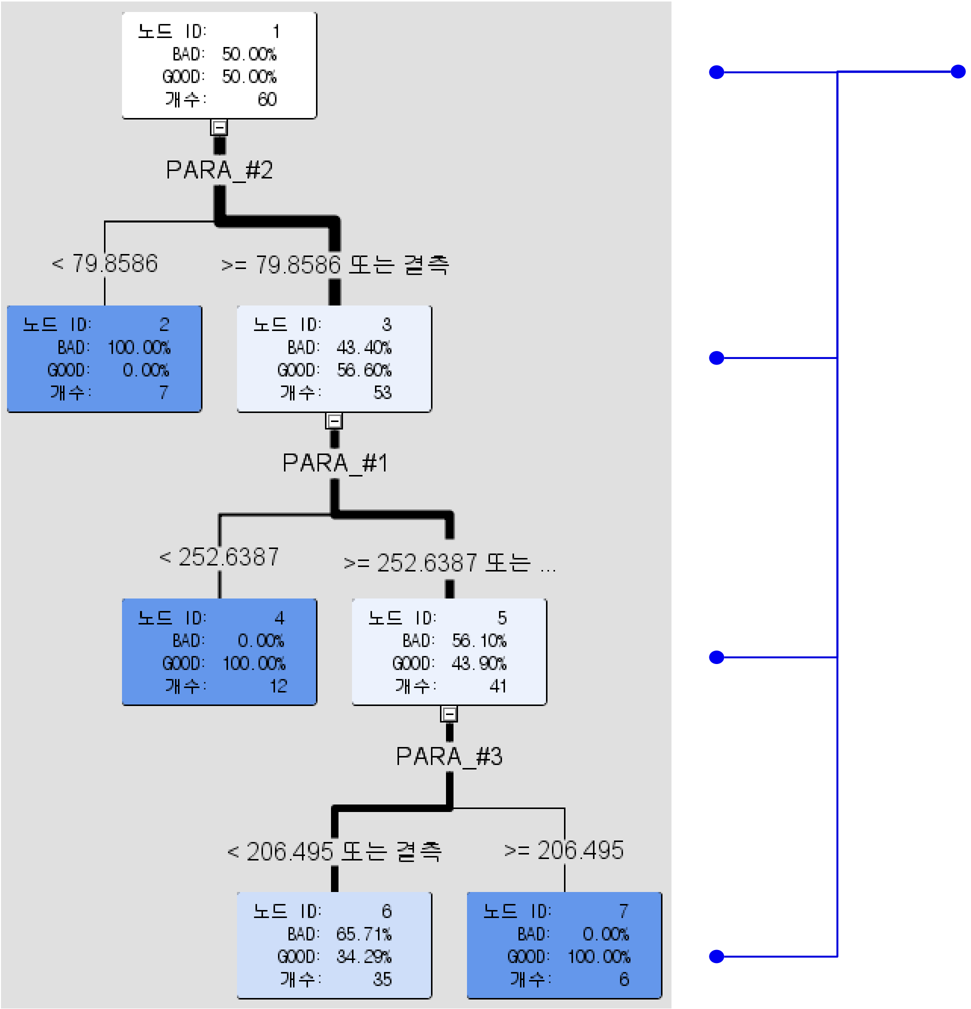


Root Cause Analysis → Preventative Maintenance → Decision Tree

Good & Bad by spray

Peach fruit from tree not sprayed with fungicides (left) and from tree sprayed with fungicides (right). (Courtesy D.F. Ritchie)

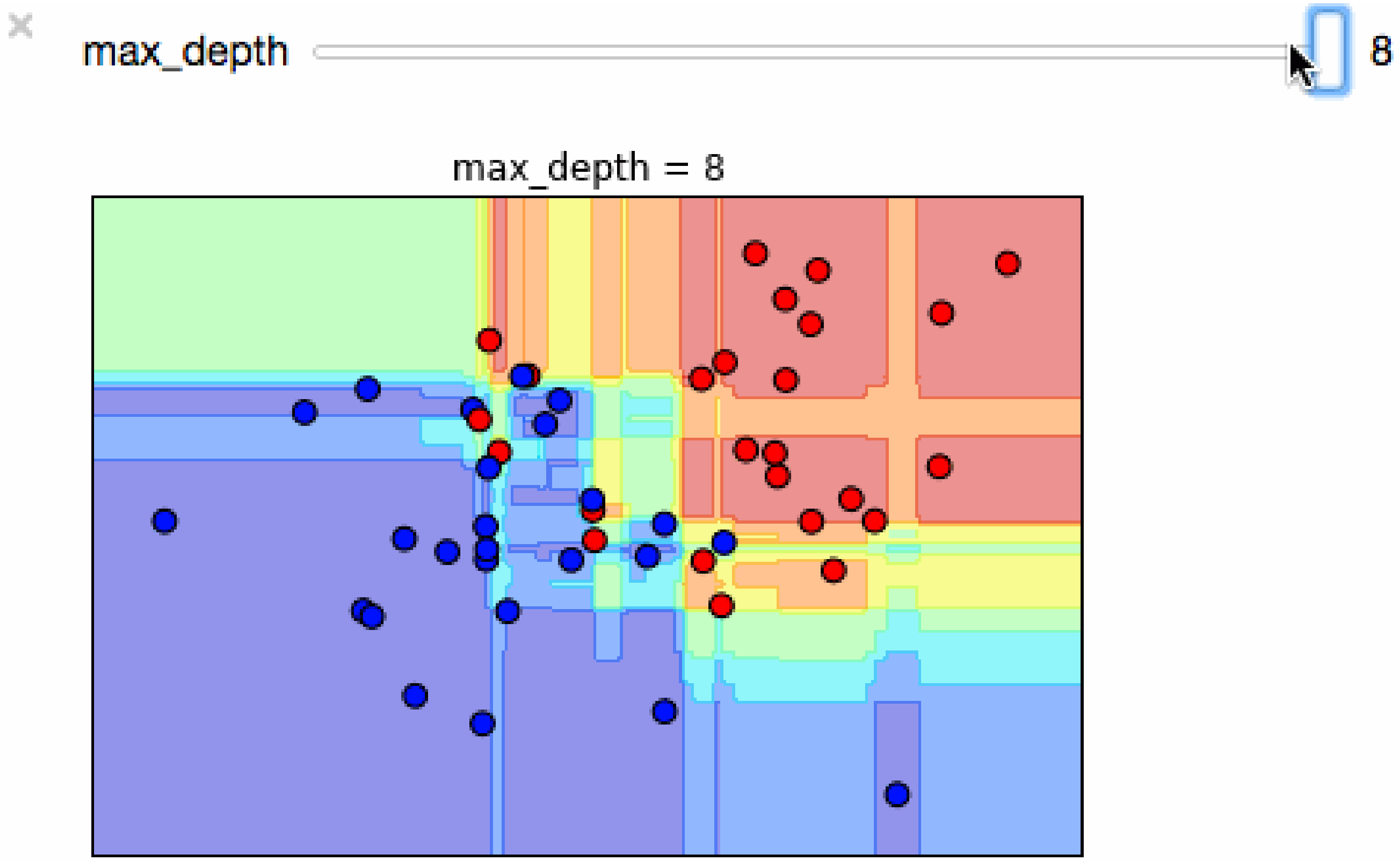
Root Cause Analysis → Preventative Maintenance → Decision Tree



- Depth
  - ✓ depth 0 : Root Node
  - ✓ depth 1 : Root와 연결된 Intermediate Node

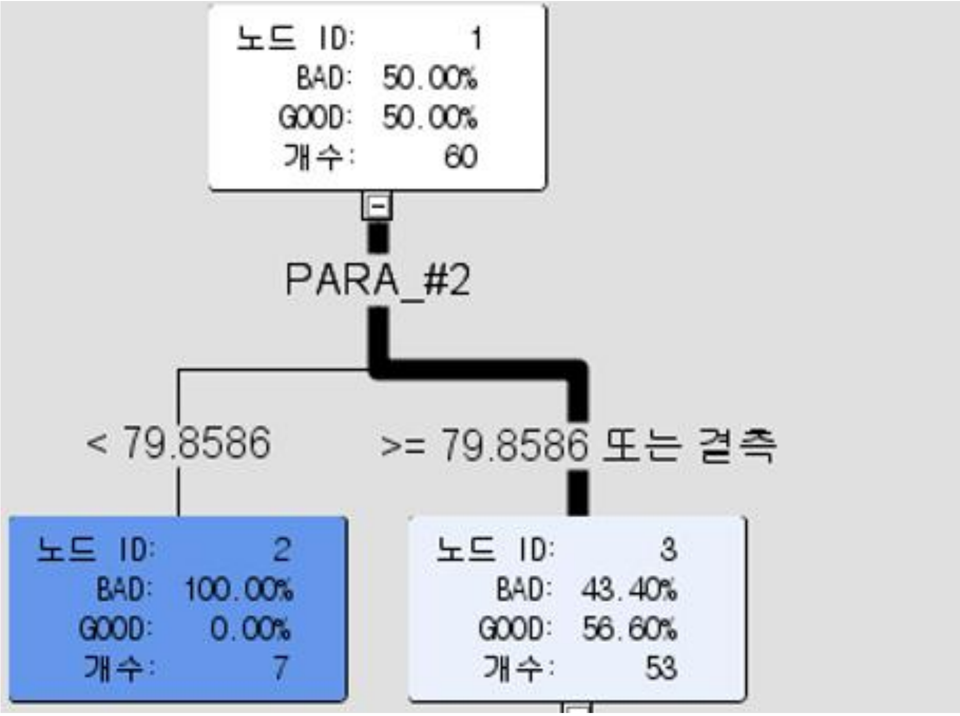
03. 설비 예지보전 및 생산 최적화

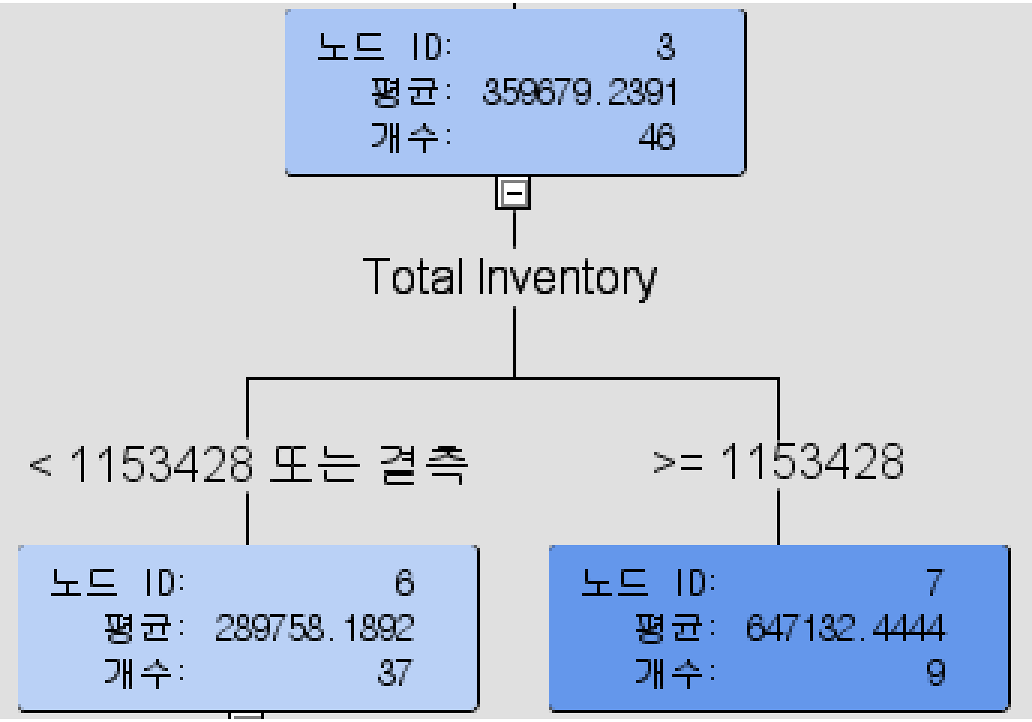
Root Cause Analysis → Preventative Maintenance → Decision Tree



Root Cause Analysis → Preventative Maintenance → Decision Tree

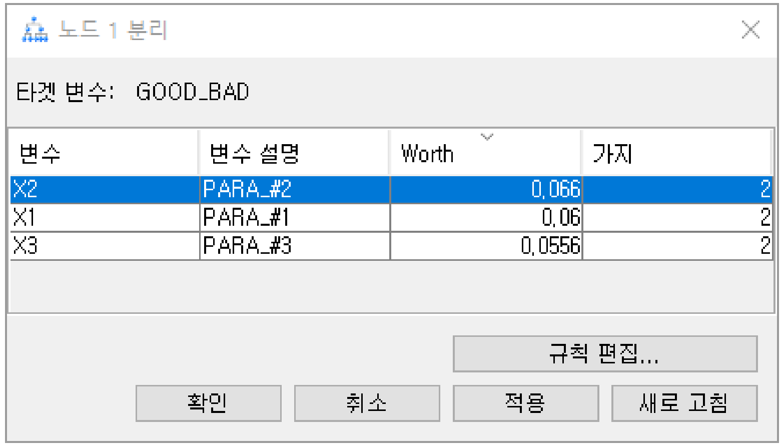
- 유형
  - Decision Tree Classification : Label type = Categorical(범주형)
  - Decision Tree Regression : Label type = Continuous(연속형)



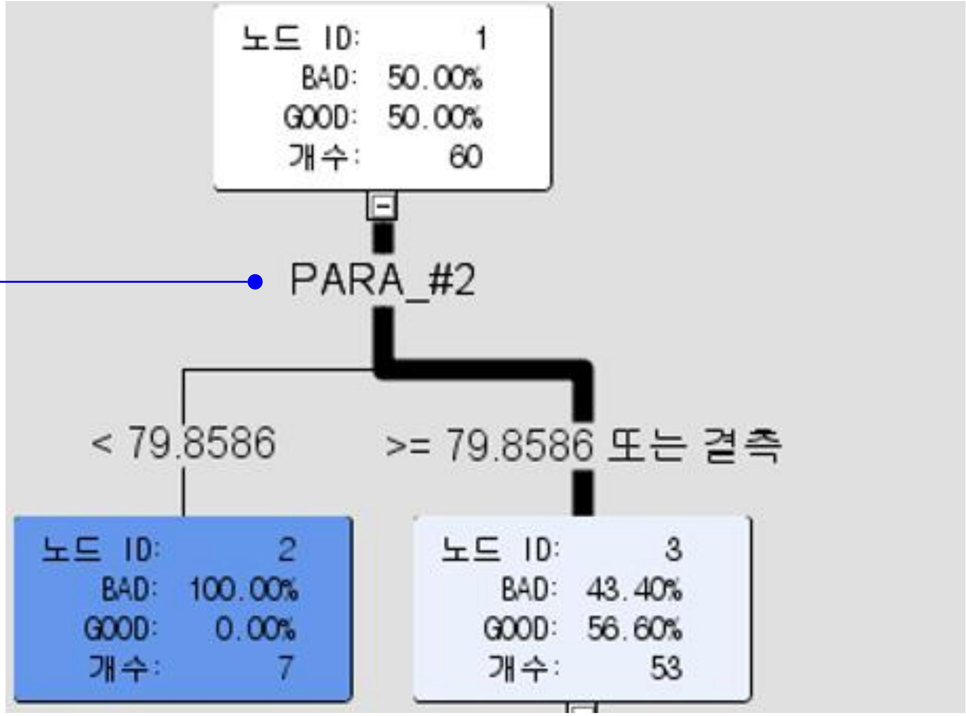


Root Cause Analysis → Preventative Maintenance → Decision Tree

- 학습 과정
  - Decision Tree의 학습 과정
    - ✓ 관찰치의 Label을 추정하기 위한 특징(Features)의 추출 하는 절차
    - ✓ 관찰치를 어느 Node로 보내는 것이 모델 향상에 기여하는 지를 결정(Decision)

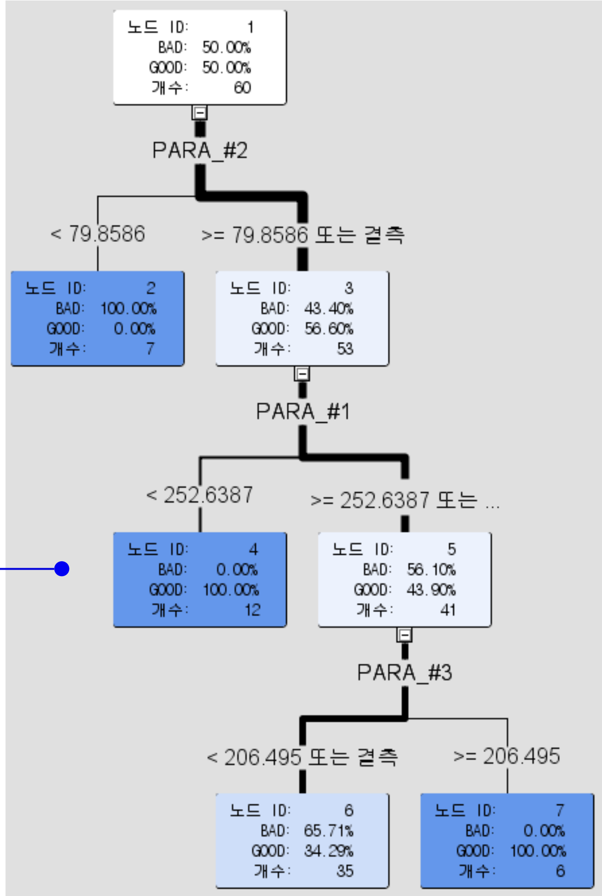


변수	변수 설명	Worth	가지
X2	PARA_#2	0.066	2
X1	PARA_#1	0.06	2
X3	PARA_#3	0.0556	2



Root Cause Analysis → Preventative Maintenance → Decision Tree

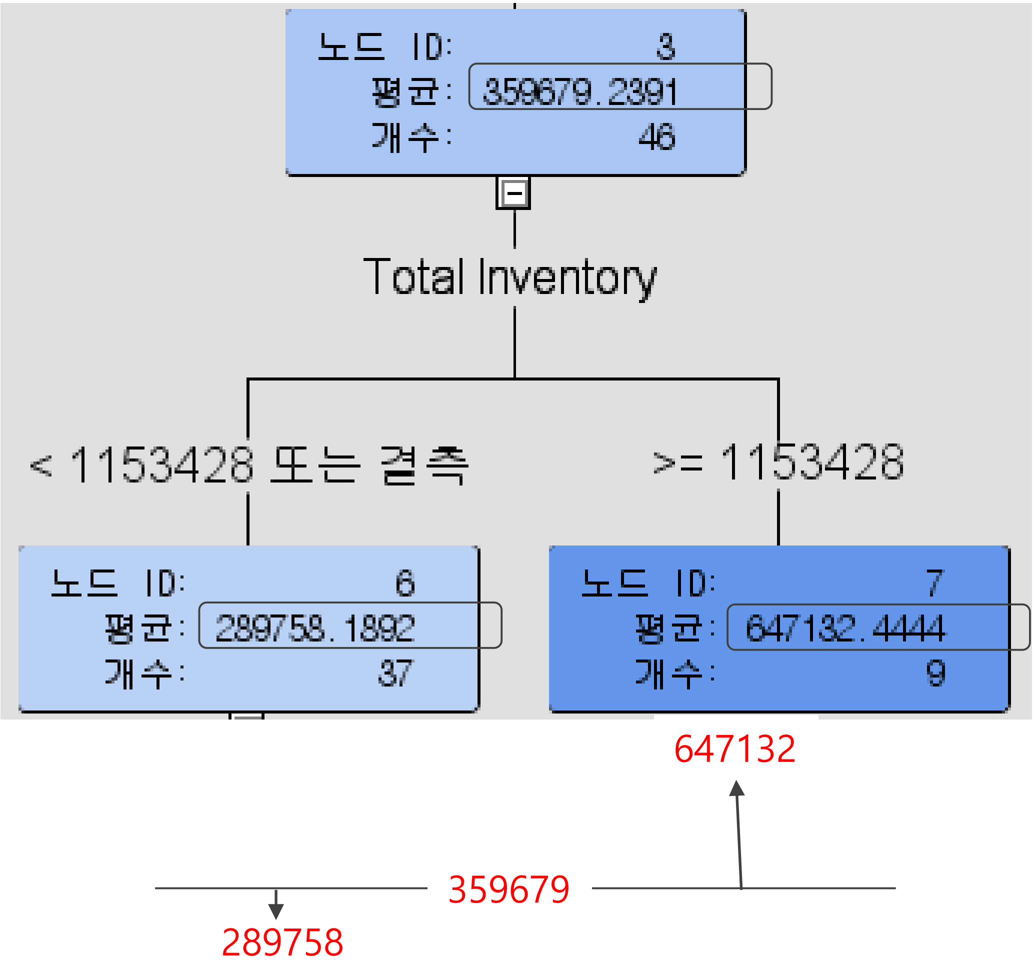
- 모델 검토
  - 신규 Labeled 관찰치의 예측 정확도
  - Classification
    - ✓ 새로운 Labeled 관찰치의 features(특징)을 예측식에 대입하여 최종 경로로 지정된 teminal node의 Major 집단군의 Label의 범주값과 일치 여부



04. 품질 최적화

Root Cause Analysis → Preventative Maintenance → Decision Tree

- 모델 검토
  - 신규 Labeled 관찰치의 예측 정확도
  - Regression
    - ✓ 새로운 Labeled 관찰치의 features(특징)을 예측식에 대입하여 최종 경로로 지정된 terminal node의 Major 집단군의 Label Value와의 차이 정도
    - ✓ 차이값 허용 범위는 비즈니스 적용 조건에 따라 사용자가 정함

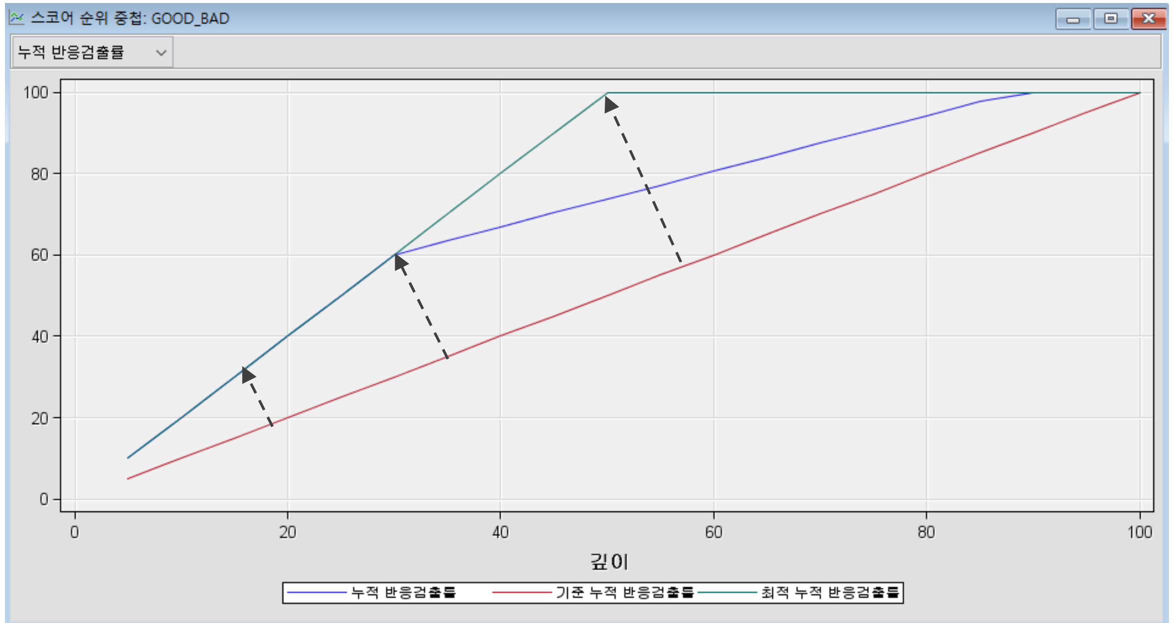


Root Cause Analysis → Preventative Maintenance → Decision Tree

- 모델 평가 통계량
  - Classification
    - ✓ Misclassification Rate (↔ Accuracy)
    - ✓ ROC
    - ✓ AUC(Area Under the

적합통계량

타겟	타겟 레이블	적합통계량	통계량 레이블	분석	평가	검증
GOOD_BAD	GOOD_BAD	_NOBS_	Sum of Frequencies	60	.	.
GOOD_BAD	GOOD_BAD	_MISC_	Misclassification Rate	0.2	.	.
GOOD_BAD	GOOD_BAD	_MAX_	Maximum Absolute Error	0.657143	.	.
GOOD_BAD	GOOD_BAD	_SSE_	Sum of Squared Errors	15.77143	.	.
GOOD_BAD	GOOD_BAD	_ASE_	Average Squared Error	0.131429	.	.
GOOD_BAD	GOOD_BAD	_RASE_	Root Average Squared Error	0.362531	.	.
GOOD_BAD	GOOD_BAD	_DIV_	Divisor for ASE	120	.	.
GOOD_BAD	GOOD_BAD	_DFT_	Total Degrees of Freedom	60	.	.



04. 품질 최적화

Root Cause Analysis → Preventative Maintenance → Decision Tree

- 모델 평가 통계량
  - Regression
    - ✓ MAE
    - ✓ MSE
    - ✓ RMSE
    - ✓ MAPE
    - ✓ etc

Mean squared error	$\mathrm{MSE} = \frac{1}{n}\sum_{t=1}^{n} e_t^2$
Root mean squared error	$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n} e_t^2}$
Mean absolute error	$\mathrm{MAE} = \frac{1}{n}\sum_{t=1}^{n} \lvert e_t \rvert$
Mean absolute percentage error	$\mathrm{MAPE} = \frac{100\%}{n}\sum_{t=1}^{n} \left\lvert \frac{e_t}{y_t} \right\rvert$

적합통계량

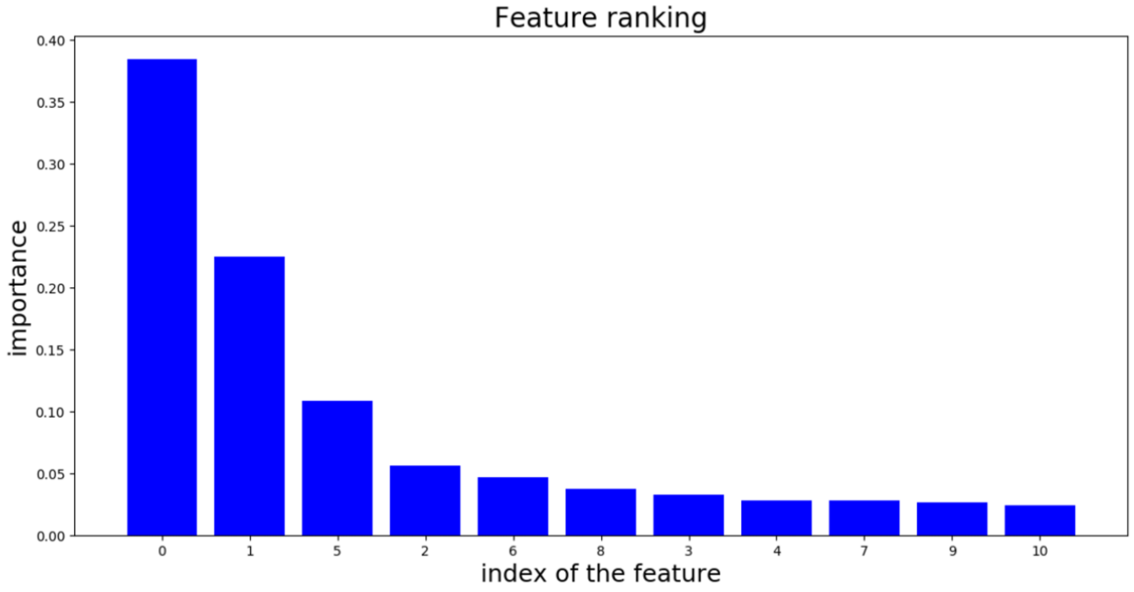
선택된 모델	선행 노드	모델 노드	모델 설명	타겟 변수	타겟 레이블	선택 기준: Train: Average Squared Error	Train: Sum of Frequencies	Train: Maximum Absolute Error	Train: Sum of Squared Errors	Train: Average Squared Error	Train: Root Average Squared Error	Train: Divisor for ASE	Train: Total Degrees of Freedom
Y	Tree2	Tree2	의사결정트...	Sales	Total Sales	1.8142E9	395	651584.6	7.166E11	1.8142E9	42593.74	395	395
	Tree	Tree	의사결정트...	Sales	Total Sales	1.8621E9	395	651584.6	7.355E11	1.8621E9	43151.58	395	395

Root Cause Analysis → Preventative Maintenance → Decision Tree

- Feature Importance(특징중요도)
  - Variable Importance(변수중요도)라고도 함
  - Tree Model의 Accuracy(정확도)와 Impurity(불순도) 개선에 기여하는 정도

변수 중요도

변수 이름	레이블	분리 규칙 개수	중요도
X3	PARA_#3	3	1.0000
X1	PARA_#1	1	0.7038
X2	PARA_#2	1	0.5796



Root Cause Analysis → Preventative Maintenance → Decision Tree

- Normal Target의 분리 규칙(Splitting Rule)
  - Normal Target(예: Good or Bad)를 Features에 대응할 때 Branch Node를 최적으로 분리하는 3가지 방법 지정

Entropy	Entropy 감소(reduction) 정보량 적용
Gini	Gini Index의 감소(reduction) 정보량 적용

Root Cause Analysis → Preventative Maintenance → Decision Tree

- Normal Target의 분리 규칙(Splitting Rule)

Gini

$$1-\sum_{j=1}^{r} p_j^2 = 2\sum_{j<k} p_j p_k$$

C1	0
C2	6

P(C1) = 0/6 = 0   P(C2) = 6/6 = 1

Gini = $1 - P(C1)^2 - P(C2)^2 = 1 - 0 - 1 = 0$

C1	1
C2	5

P(C1) = 1/6   P(C2) = 5/6

Gini = $1 - (1/6)^2 - (5/6)^2 = 0.278$

C1	2
C2	4

P(C1) = 2/6   P(C2) = 4/6

Gini = $1 - (2/6)^2 - (4/6)^2 = 0.444$

[source] http://slideplayer.com/slide/7696713/

Root Cause Analysis → Preventative Maintenance → Decision Tree

- Normal Target의 분리 규칙(Splitting Rule)

Gini

$$1-\sum_{j=1}^{r} p_j^2 = 2\sum_{j<k} p_j p_k$$

Pr(interspecific encounter) = 1-2(3/8)^2-2(1/8)^2 = .69

Pr(interspecific encounter) = 1-(6/7)^2-(1/7)^2 = .24

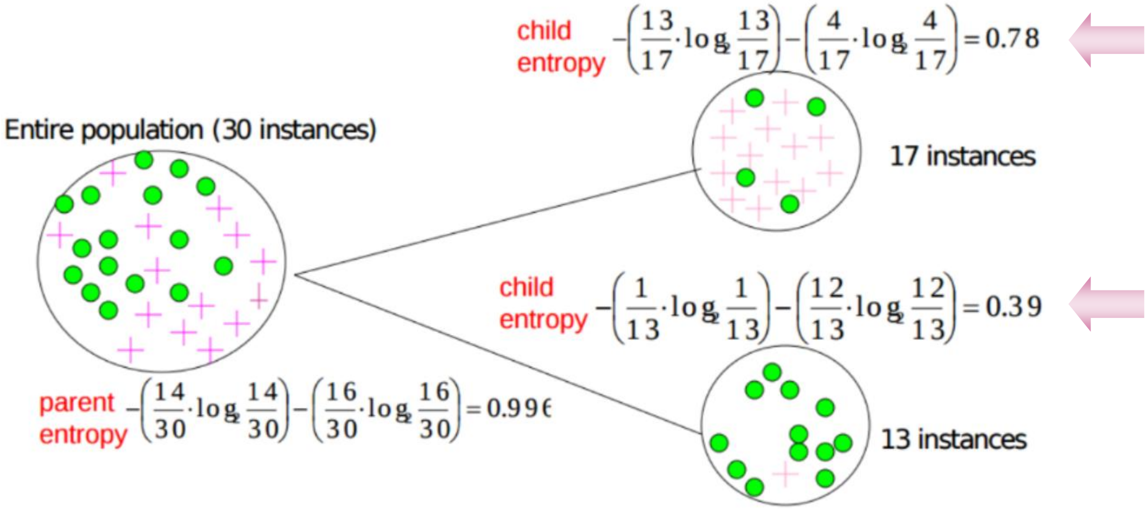
Root Cause Analysis → Preventative Maintenance → Decision Tree

- Normal Target의 분리 규칙(Splitting Rule)

Entropy

$$E(S) = \sum_{i=1}^{c} -p_i \log_2 p_i$$

Information Gain = entropy(parent) – [average entropy(children)]



(Weighted) Average Entropy of Children = $\left(\frac{17}{30}\cdot 0.787\right)+\left(\frac{13}{30}\cdot 0.391\right)=0.615$

Information Gain= 0.996 - 0.615 = 0.38 for this split

[source] http://www.grroups.com/blog/an-information-gain-based-feature-ranking-function-for-xgboost

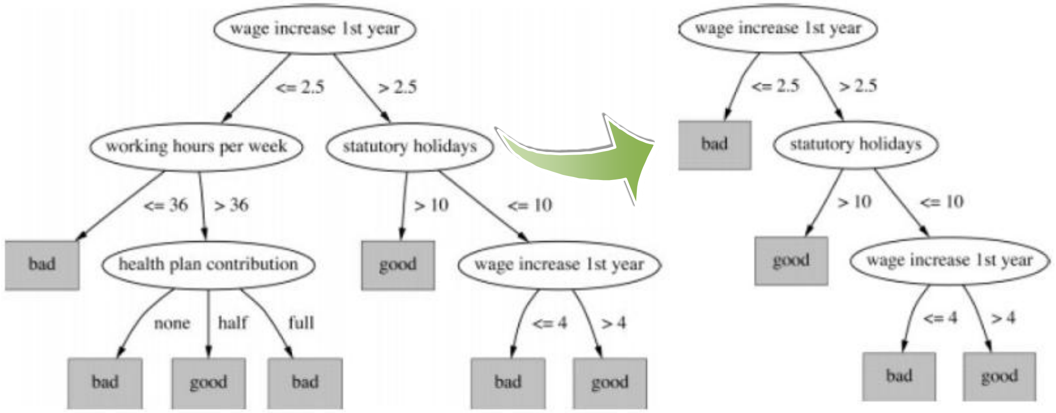
Root Cause Analysis → Preventative Maintenance → Decision Tree

- Normal Target의 분리 규칙(Splitting Rule) > 가지, 깊이, 크기
  - 분리 규칙을 적용할 Tree의 가지수, 깊이, 및 Node의 Size적용

Maximum Branch	분리할 가지의 수 지정(default : 2)
Maximum Depth	지정한 최대 깊이 내에서 분리 규칙 적용
Minimum Categorical Size	하나의 Node가 포함할 최소의 Observation 수 지정

Root Cause Analysis → Preventative Maintenance → Decision Tree

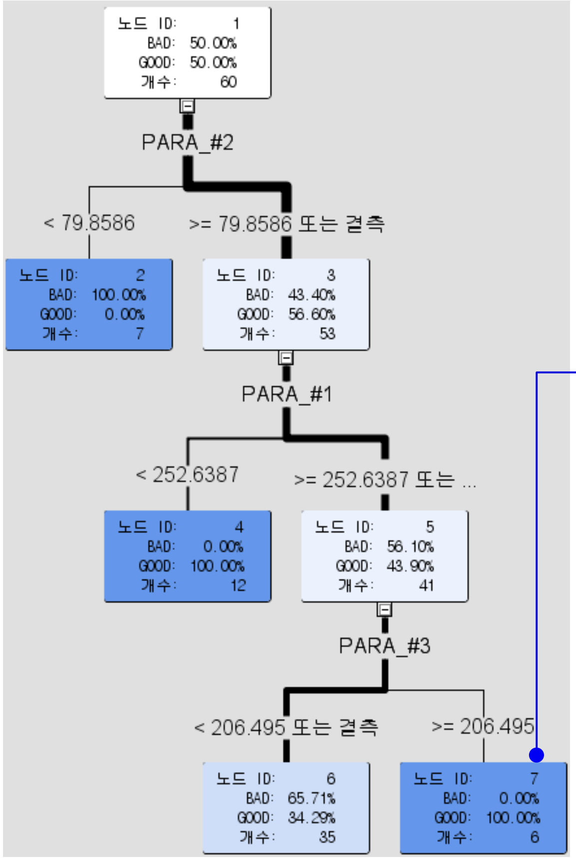
- 서브 트리(Subtree)



Method	▪ Assessment : 평가값이 선택된 가장 작은 subtree ▪ Largest : 전체 트리 선택 ▪ n : 최대 n leaf를 가지는 subtree 선택
Number of Leaves	Method가 n일때 leaf의 갯수
Assessment Fraction	Assessment Method가 Lift일 때 관찰치 비율

Root Cause Analysis → Preventative Maintenance → Decision Tree

- Analytic Modeling을 위한 Diagram 작업 수행
  - Decision Tree Modeling > Model Output
    - ✓ 노드 규칙 > Node 7



Node 7

if PARA_#3 >= 206.495

AND PARA_#2 >= 79.8586 or MISSING

AND PARA_#1 >= 252.639 or MISSING

then

Tree Node Identifier = 7

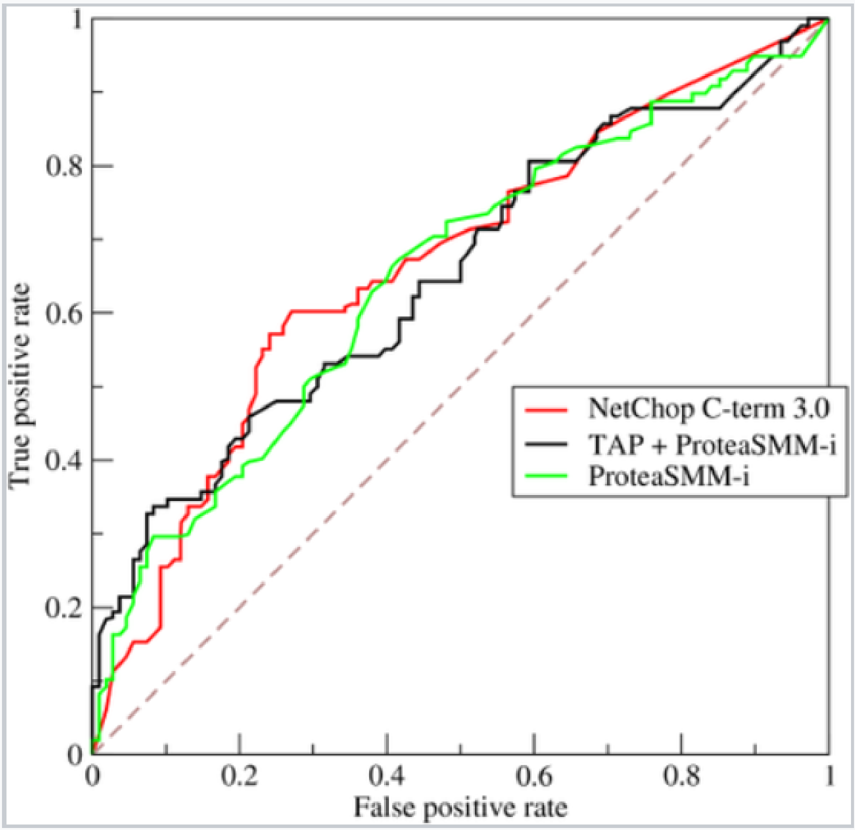
Number of Observations = 6

Predicted: GOOD_BAD=GOOD = 1.00

Predicted: GOOD_BAD=BAD = 0.00

Root Cause Analysis → Preventative Maintenance → Decision Tree

- Analytic Modeling을 위한 Diagram 작업 수행
  - 모델 비교 [Entropy, Gini] > ROC
    - ✓ Receiver Operating Characteristic



[source] https://en.wikipedia.org/wiki/Receiver_operating_characteristic

True positive rate (TPR), Recall, Sensitivity, probability of detection

$$= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$$

False positive rate (FPR), Fall-out, probability of false alarm

$$= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$$

Root Cause Analysis → Preventative Maintenance → Decision Tree

- Analytic Modeling을 위한 Diagram 작업 수행
  - 모델 비교 [Entropy, Gini] > ROC
    - ✓ Receiver Operating Characteristic

<table>
<tr><td></td><td></td><td colspan="2">True condition</td><td></td><td colspan="2"></td></tr>
<tr><td></td><td>Total population</td><td>Condition positive</td><td>Condition negative</td><td>Prevalence $= \frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$</td><td colspan="2">Accuracy (ACC) = $\frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$</td></tr>
<tr><td rowspan="2">Predicted condition</td><td>Predicted condition positive</td><td>True positive, Power</td><td>False positive, Type I error</td><td>Positive predictive value (PPV), Precision = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$</td><td colspan="2">False discovery rate (FDR) = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$</td></tr>
<tr><td>Predicted condition negative</td><td>False negative, Type II error</td><td>True negative</td><td>False omission rate (FOR) = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$</td><td colspan="2">Negative predictive value (NPV) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$</td></tr>
<tr><td></td><td></td><td>True positive rate (TPR), Recall, Sensitivity, probability of detection $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$</td><td>False positive rate (FPR), Fall-out, probability of false alarm $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$</td><td>Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$</td><td rowspan="2">Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR−}}$</td><td rowspan="2">$F_1$ score = $\frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$</td></tr>
<tr><td></td><td></td><td>False negative rate (FNR), Miss rate $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$</td><td>True negative rate (TNR), Specificity (SPC) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$</td><td>Negative likelihood ratio (LR−) $= \frac{\text{FNR}}{\text{TNR}}$</td></tr>
</table>

[source] https://en.wikipedia.org/wiki/Receiver_operating_characteristic

04. 품질 최적화

Root Cause Analysis → Preventative Maintenance → Decision Tree

- Library
  - Python
    - ✓ sklearn.tree.DecisionTreeClassifier
    - ✓ sklearn.tree.DecisionTreeRegressor
  - R
    - ✓ library(tree)
    - ✓ library(rpart)
    - ✓ library(party)

Root Cause Analysis → Preventative Maintenance → Decision Tree

sklearn.tree.DecisionTreeClassifier

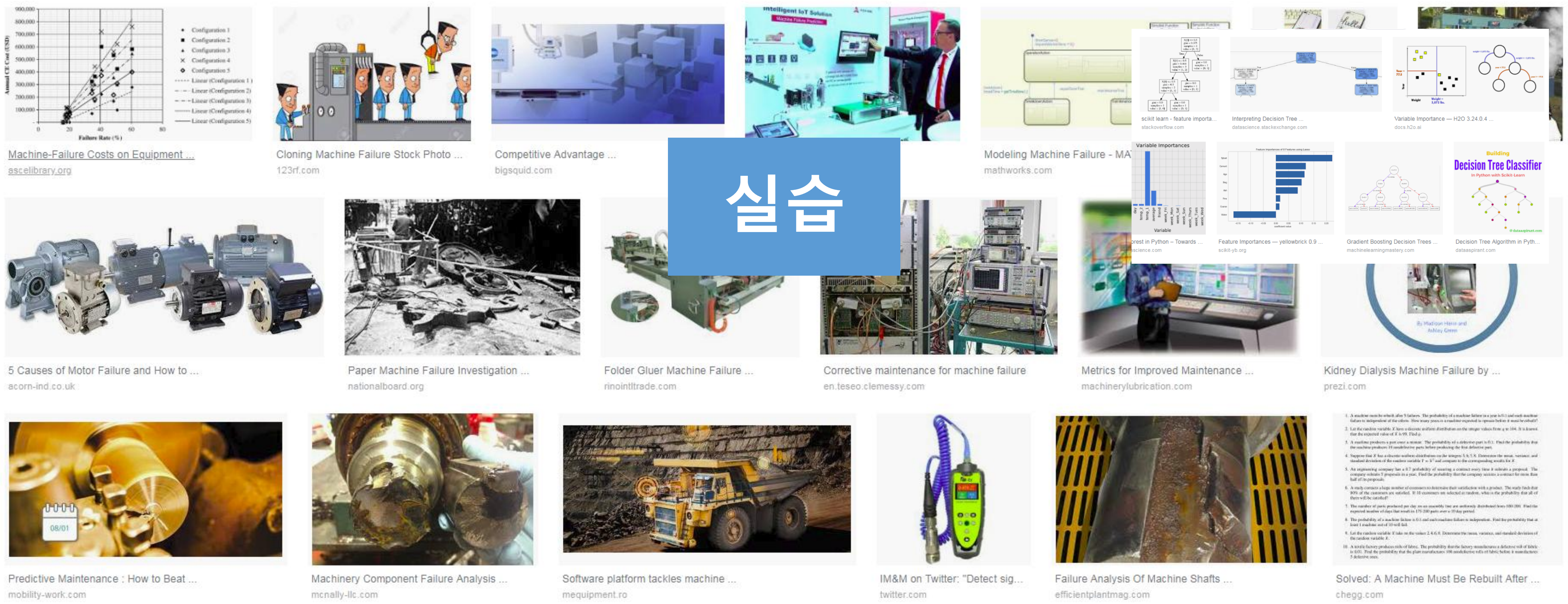
```
DecisionTreeClassifier(
    class_weight=None       # label별 가중치 [[None, 'balanced'], [{class_label:weight}]]
  , criterion='gini'        # split 기준 - ['gini', 'entropy']
  , max_depth=None          # tree의 최대 깊이
  , max_features=None       # best split 특징 수 - [[1, 2,...], [0.1, 0.2, ...], [None, 'auto', 'sqrt', 'log2']]
  , max_leaf_nodes=None     # 최대 leaf node 수
  , min_impurity_split=1e-07    # 불순도 결정 임계치
  , min_samples_leaf=1      # Leaf의 최소 Sample 수
  , min_samples_split=2     # split 최소 수
  , min_weight_fraction_leaf=0.0  # 전체 가중 합 중 leaf의 최소 가중치 비율
  , presort=False           # 데이터의 사전 정렬 여부
  , random_state=0          # random number 생성 seed 값
  , splitter='best'         # Node split 선택 전략 - [best, random]
  )
```

Root Cause Analysis → Preventative Maintenance → Decision Tree

sklearn.tree.DecisionTreeRegressor

```
DecisionTreeRegressor(
    criterion='mse'          # split 기준 - ['mse', 'friedman_mse', 'mae']
  , max_depth=None           # 최대 깊이
  , max_features=None        # best split 특징 수 - [[1, 2,...], [0.1, 0.2, ...], [None, 'auto', 'sqrt', 'log2']]
  , max_leaf_nodes=None      # 최대 leaf node 수
  , min_impurity_split=1e-07 # 불순도 결정 임계치
  , min_samples_leaf=1       # Leaf의 최소 Sample 수
  , min_samples_split=2      # split 최소 수
  , min_weight_fraction_leaf=0.0  # 전체 가중 합 중 leaf의 최소 가중치 비율
  , presort=False            # 데이터의 사전 정렬 여부
  , random_state=None        # random number 생성 seed 값
  , splitter='best'          # Node split 선택 전략 - [best, random]
  )
```

Root Cause Analysis → Preventative Maintenance → Decision Tree



Prediction → Demand Forecasting

Study of SVM-Based Air-Cargo Demand Forecast Model

Hong-jun Heng, Bing-zhong Zheng, Ya-jing Li •

Published in International Conference on Computational... 2009 • DOI: 10.1109/CIS.2009.180

This paper analyzed some existing problems of the present air-cargo forecast methods. Then it established the SVM (support vector machine) model for air-cargo demand forecasting. Taking the historical statistical data of Beijing to Shanghai cargo volumes from Jan-2005 to Mar-2006 as fitting and forecasting specimens, we can obtain the prediction model to optimize, which was compared with that of Brown cubic exponential smoothing, by analyzing fitting and forecasting effect of model for... CONTINUE READING

	MAPE	MXE	RWSE	EC
SVM	0.102499895	0.247880429	31092.48716	0.947636381
Brown	0.259869908	0.428480366	76391.85697	0.883326088

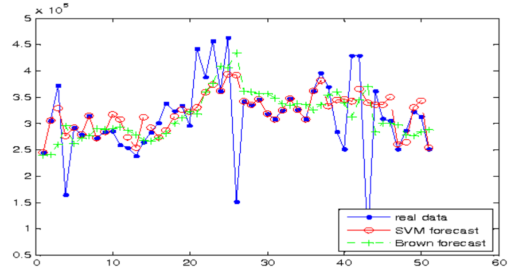
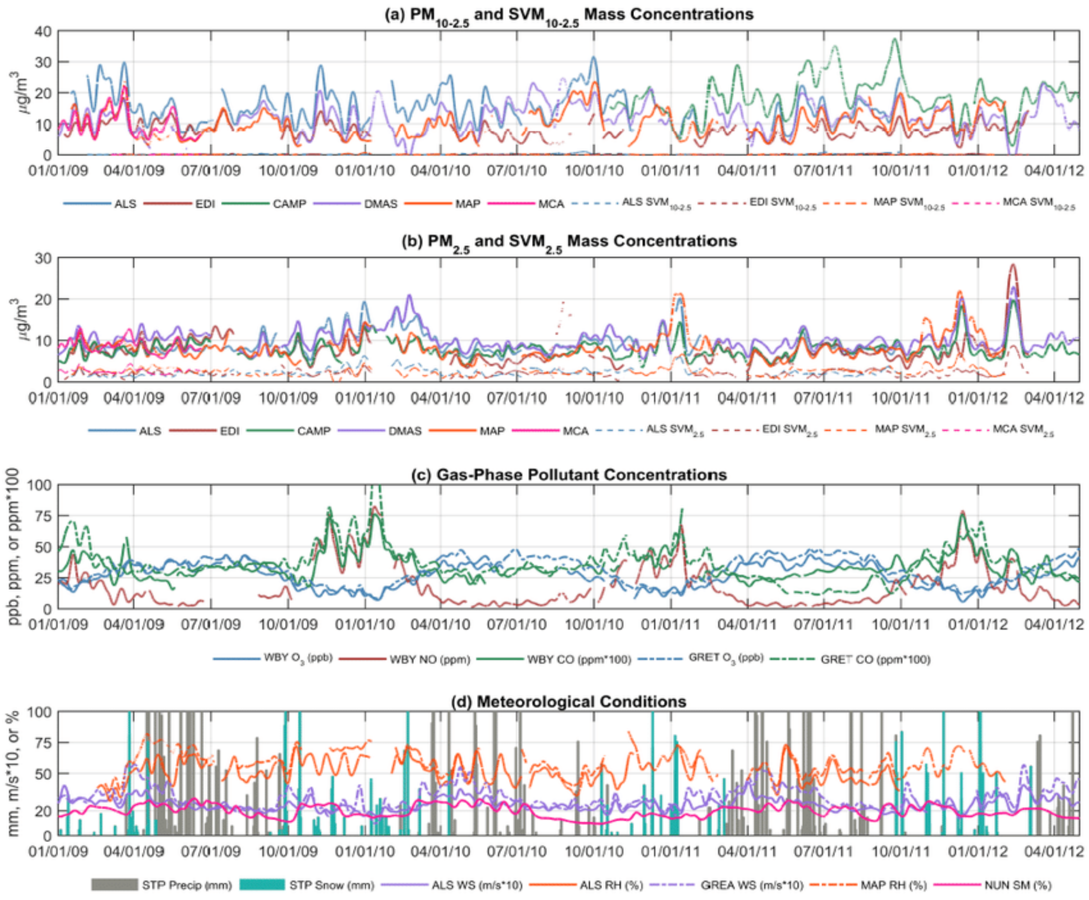


Figure 2. The curve of SVM and Brown forecast result compared with real data

https://www.researchgate.net/figure/Most-important-and-quality-articles-using-Support-Vector-Machines-for-Energy-demand_tbl6_301665131



https://www.researchgate.net/figure/Smoothed-th-3-h-time-series-of-hourly-average-a-PM-10-25-and-SVM-10-25-mass_fig1_282966991

Prediction → Demand Forecasting → Support Vector Machine

Most important and quality articles using Support Vector Machines for Energy demand forecasting

Authors	Purpose	Demand type and period	Determinants	Case study/Location	Modeling	Prediction accuracy (Some based on error)
Wang et al. [120]	Forecasting the electricity demand by preprocessed data	Monthly electricity consumption	Historical data of monthly electricity demand	Yes / China	Trend fixed ε-SVR approach	MAPE = 3.79%
Setiawan et al. [121]	Forecasting very short-term electricity demand	Electricity consumption in 5 minutes intervals	Actual load historic data by 5 minutes intervals	Yes / New South Wales, Australia	Support vector regression	MAPE = lower than 1% Mean absolute error (MAE) = lower than 1% Relative absolute error (RAE) = 4% to 5%
Hong et al. [125]	Load demand forecasting	Monthly electric loads	Historical monthly electric load data	Yes / Northeastern China	Chaotic immune algorithm / Support vector regression	MAPE = 1.76%
Fattaheian-Dehkordi et al. [127]	Forecasting the short-term electricity consumption	Hour-ahead electricity demand	Historical load data / Temperature	Yes / Tehran, Iran	Support vector regression	MAPE = 0.75% to 1.46%
Xiong et al. [128]	Interval forecasting of electricity demand	Hourly electricity consumption	Daily electricity demand / Hourly electricity demand	Yes / USA	Support vector regression / Bivariate empirical mode decomposition	NA

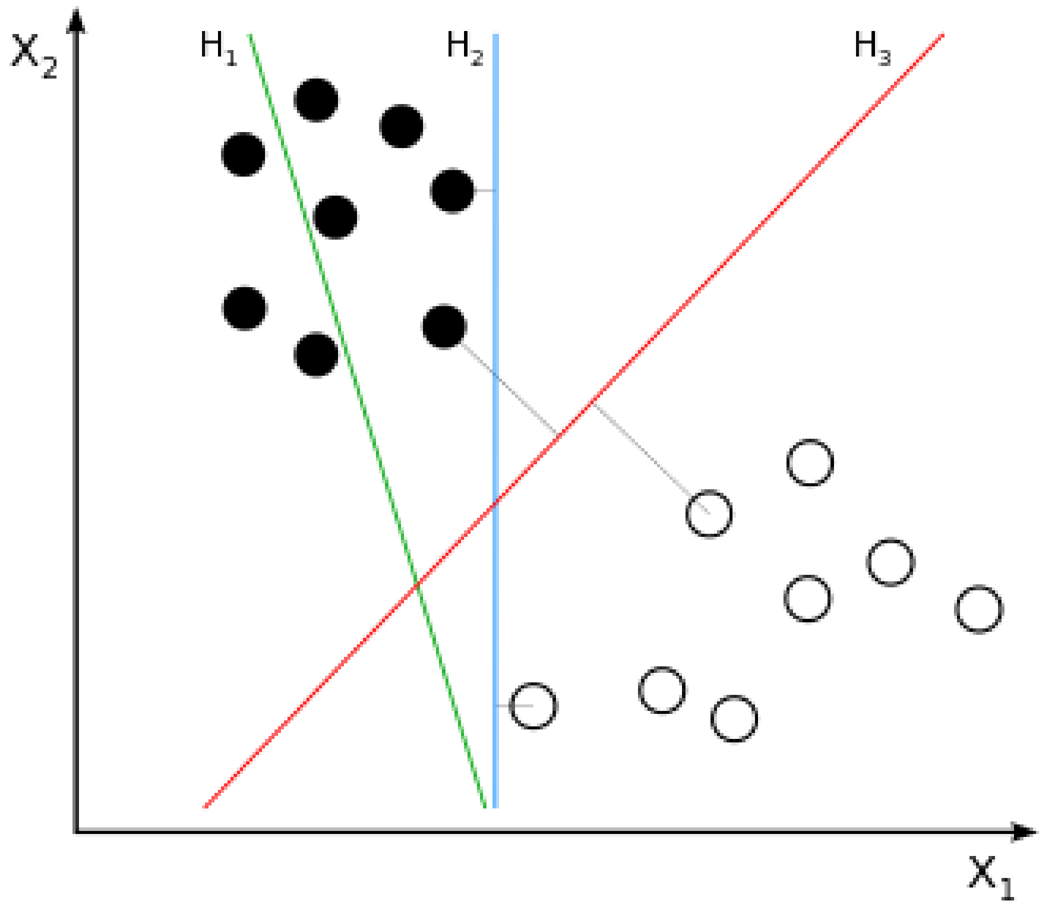
https://www.researchgate.net/figure/Most-important-and-quality-articles-using-Support-Vector-Machines-for-Energy-demand_tbl6_301665131

Prediction → Demand Forecasting → Support Vector Machine

- Support Vector Machine의 기본 개념
  - 생각의 고리
    - 다른 특징에 편향된 밀도를 가진 그룹
    - 예: 숲속의 군락
  - 두개의 특징 차원으로 Labeling이 가능한 경우를 생각해보자
  - 서로 다른 위치에 밀집되어 있고, 서로다른 Label을 가지고 있다면, 그 사이에 경계선(linear line)을 그어서 식별할 수 있음.
  - 만약, 경계선 부근에서 일부 관찰치들이 다근 그룹의 Label을 가지고 있다면 어떻게 최적의 경계선을 그을 수 있을까?

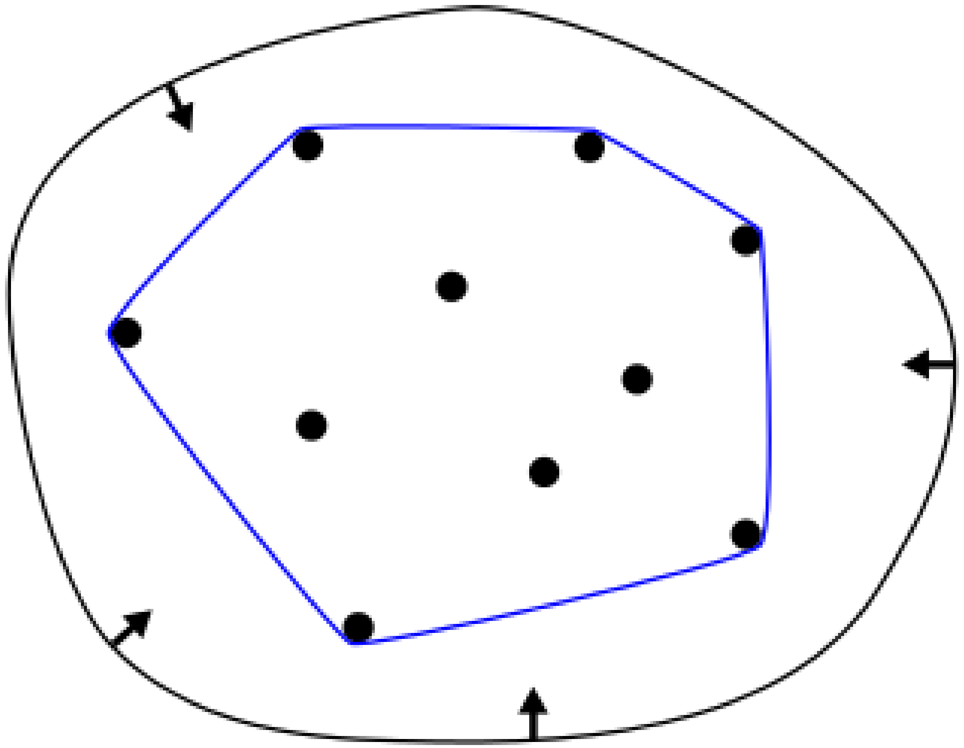
Prediction → Demand Forecasting → Support Vector Machine

- Support Vector Machine의 기본 개념
  - 그룹간 경계를 최대화하는 해(solution)의 도출
    - 특징 차원(features)들이 투영된 공간에서 관찰치 집합의 밀집을 경계하는 선간의 거리가 최대가 되게 하는 알고리즘
    - 경계 최대화로 인해 학습 정확도가 떨어질 수 있으나, 이는 과적합 방지와 Trade-Off
    - 즉, 모델의 실세계 적용(일반화)에 더 근접할 수 있어 예측 유리함



Prediction → Demand Forecasting → Support Vector Machine

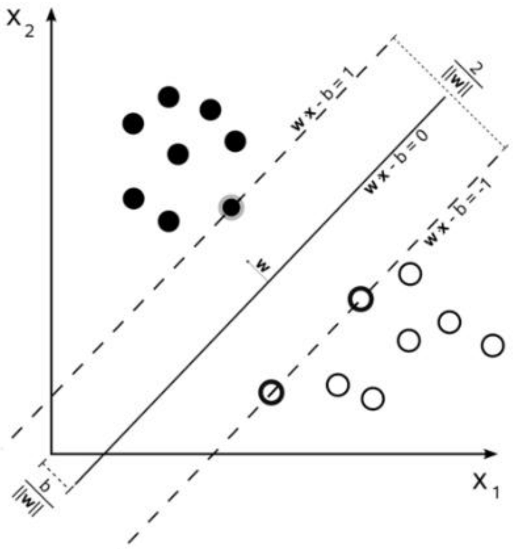
- Support Vector Machine의 기본 개념
  - Convex Hull(볼록 폐포) : 같은 Label의 관찰치 집단의 경계선에 걸쳐있는 관찰치들을 연결한 다각형 영역
  - hyperplane과 convex hull 선상의 벡터(관찰치)를 수직으로 연결하는 거리로 label을 분리하는데 기회를 가지게 됨 → 알고리즘의 학습 대상



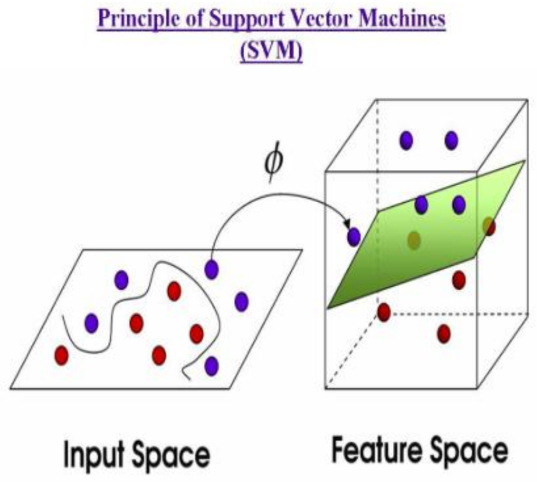
Prediction → Demand Forecasting → Support Vector Machine

- Support Vector Machine의 기본 개념
  - Hyperplane
  - 집단간 구분에 사용되는 Line 또는 Plane
  - feature 차원이 N일때 hyperplane의 차원은 N-1차원의 subspace임

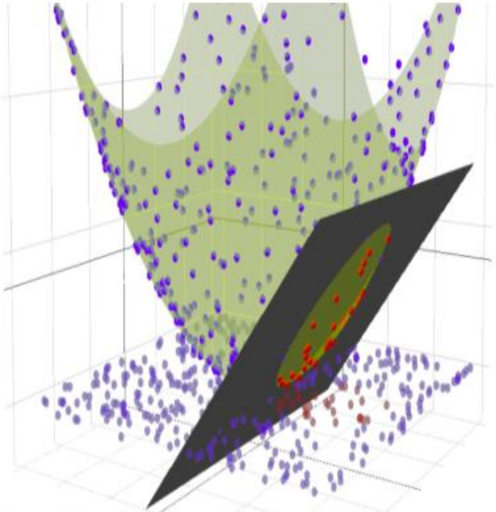
- feature 차원수 = 2 → hyplane의 차원수 = 1, 즉 line
- feature 차원수 = 3 → hyplane의 차원수 = 2, 즉 plane
- feature 차원수 = 4 → hyplane의 차원수 = 3, 즉 square



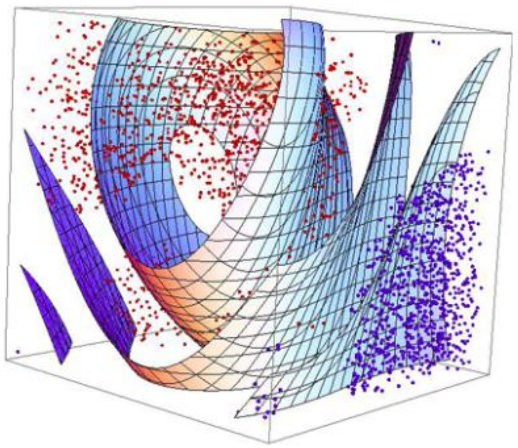
Source : https://ko.wikipedia.org/wiki/%EC%84%9C%EC%9B%8C%ED%84%B0_%EB%B2%A1%ED%84%B0_%EB%A8%B8%EC%8B%A0



Source : http://www.sbaban.org/tag/ml/



Source : http://efavdb.com/svm-classification/



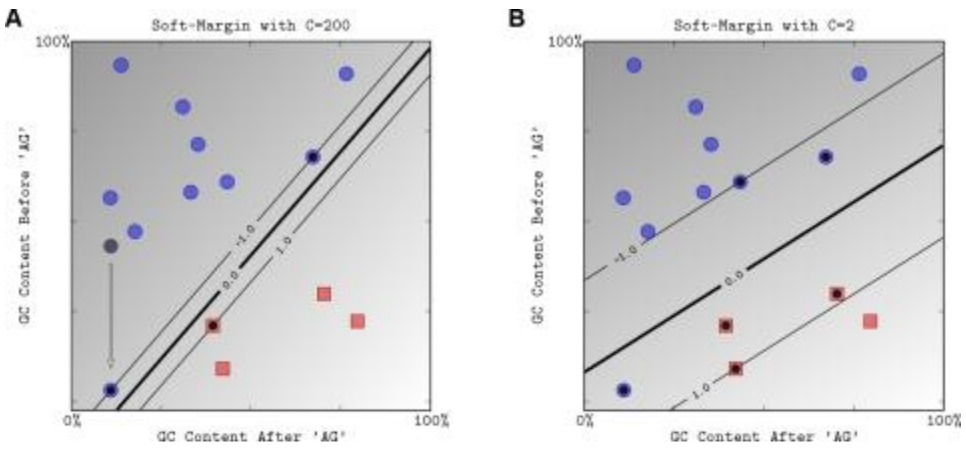
Source : https://www.quora.com/Support-Vector-Machines-How-does-going-to-higher-dimension-help-data-get-linearly-separable-which-was-non-linearly-separable-in-actual-dimension

Prediction → Demand Forecasting → Support Vector Machine

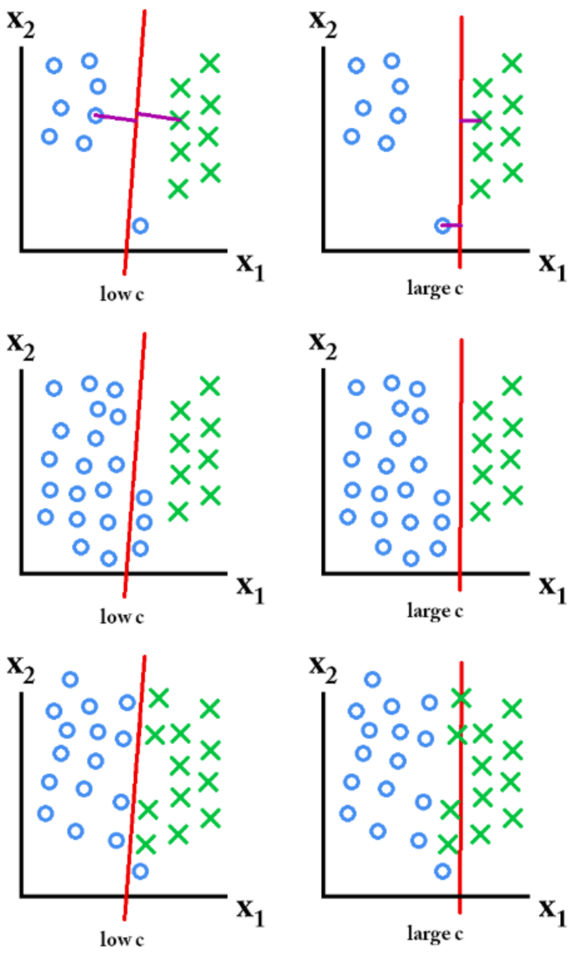
- Support Vector Machine의 기본 개념
  - Support Vector : Convex Hull 중 서로 다른 label을 효과적으로 분리하는데 기여하는 edge vector로서 Hyperplane으로 부터 가장 가까운 vector
  - Maximum-margin Hyperplane
  - hyperplan으로 부터 구별되는 서로 다른 Label의 support vector와의 거리를 최대로 하는 hyperplane을 말함
  - Hard Margin
    - 경계 구간의 Label Class를 엄격하게 구분
    - Learning Data에 대입한 결과는 Soft Margin보다 우수
    - 하지만 Test Data를 적용한 결과는 Soft Margin보다 우수하다고 할 수 없음
      - ✓ Regularization error 문제 발생
      - ✓ Hyper Plan에 근접 지역에서의 Noise까지도 Learning하여 Overfitting 발생
  - Soft Margin
    - Decision Hyperplane이 잘못 분류되는 관찰치 허용
    - Corinna Cortes와 Vladimir N. Vapnik가 제안
    - 경계선내에 관찰치들 중 Label값이 다른 관찰치가 일부 존재하더라도 페널티를 부과하지 않음
    - Slack Variable을 도입하여 오차 허용

Prediction → Demand Forecasting → Support Vector Machine

- Support Vector Machine의 기본 개념
  - 주요 Parameter > C - Complexity parameter
    - 오류 관찰치 허용 조건(error term)에 대한 Penalty 부여 정도
    - C값에 따라서 정규화(regularization : L1 or L2) 수준이 결정됨
    - a squared l2 penalty : L2-Loss(Hinge Loss의 제곱값)
    - C값이 커지면 margin이 더 hard 해지고
    - C값이 작아지면 margin이 더 soft 해짐



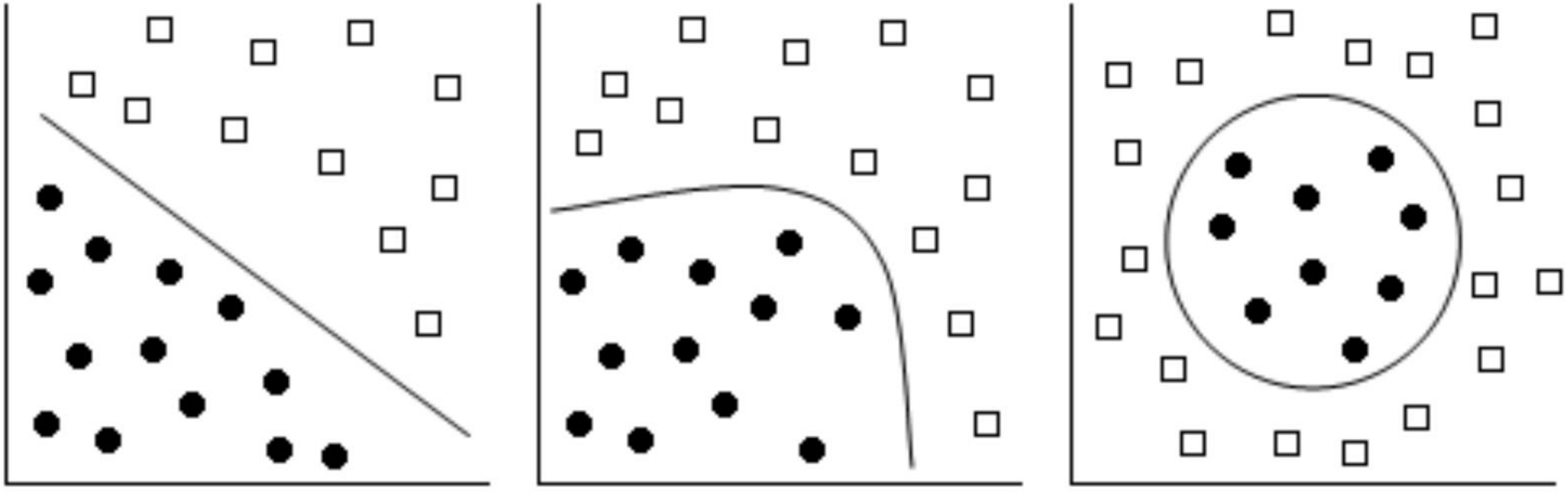
Source : https://openi.nlm.nih.gov/detailedresult.php?img=PMC2547983_pcbi.1000173.g003&req=4



04. 품질 최적화

Prediction → Demand Forecasting → Support Vector Machine

- Support Vector Machine의 기본 개념
  - 주요 Parameter > Kernel (Trick)



Linear

Polynomial

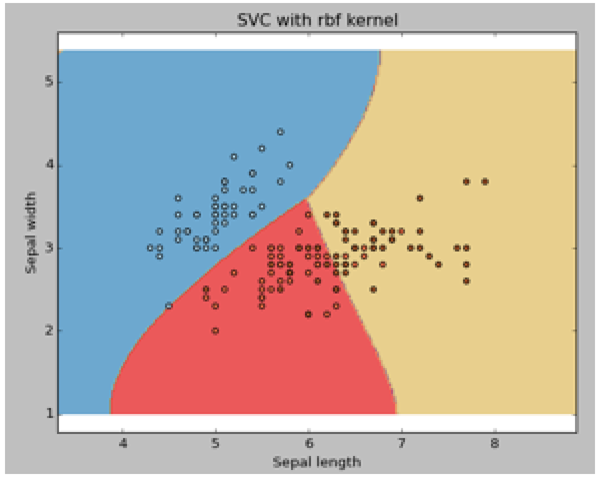
RBF(Radio Basis Function= Gaussian)

Prediction → Demand Forecasting → Support Vector Machine

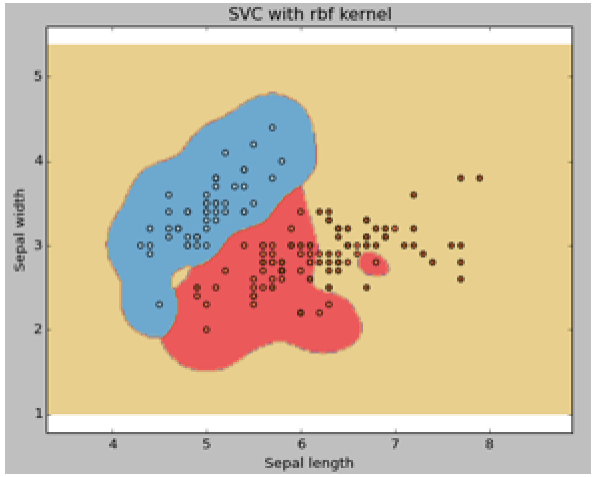
- Support Vector Machine의 기본 개념
  - 주요 Parameter > gamma
    - Gaussian Kernel의 Circle Curve 조절
    - Gamma ↑ Grouping Buundray Circle Curve ↑
    - Gamma ↓ Grouping Buundray Circle Curve ↓

- Parameter의 Tunning은 Regularization(정규화)와 과적합(overfitting)의 Trade-off를 고려하여야 함

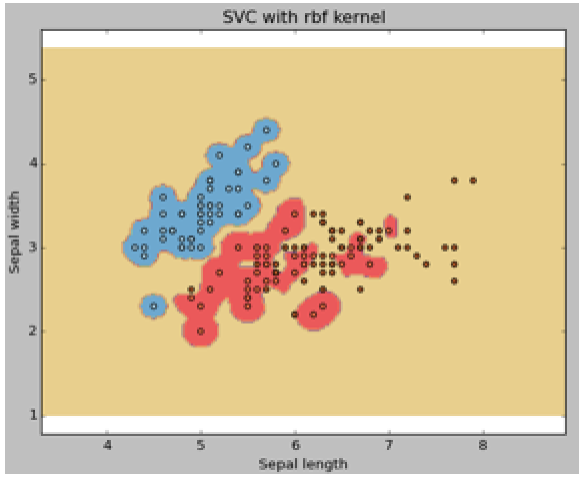
gamma =0



gamma =10



gamma =100



Source : https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

04. 품질 최적화

Prediction → Demand Forecasting → Support Vector Machine

- Support Vector Machine과 Neural Network
  - SVM은 Neural Network 계열의 Perceptron 방법을 응용
  - SVM의 우수성으로 Neural Network은 매우 오랜 기간 암흑기(AI Winter)를 가짐
  - SVM의 비확률적 분류 알고리즘으로 발전
  - 반면에, Neural Network 계열은 분류 오류를 최소화하는 확률 모델 패러다임내에서 해를 연구 진행 과정에서 수많은 시행 착오를 겪음

04. 품질 최적화

Prediction → Demand Forecasting → Support Vector Machine

- Library
  - Python
    - ✓ sklearn.svm
  - R
    - ✓ library("e1071")
    - ✓ svm()

Prediction → Demand Forecasting → Support Vector Machine

sklearn.svm.LinearSVC

```
LinearSVC(
  C=1.0          # float, optional (default=1.0)
          # Penalty parameter C of the error term.
, class_weight=None   # {dict, 'balanced'}, optional
          # Set the parameter C of class i to class_weight[i]*C for SVC.
          # If not given, all classes are supposed to have weight one.
          # The "balanced" mode uses the values of y to automatically adjust weights inversely proportional
          #   to class frequencies in the input data as n_samples / (n_classes * np.bincount(y))
, dual=True        # bool, (default=True)
          # Select the algorithm to either solve the dual or primal optimization problem.
          # Prefer dual=False when n_samples > n_features.
, fit_intercept=True  # boolean, optional (default=True)
          # Whether to calculate the intercept for this model. If set to false, no intercept will be used in calculations
          #   (i.e. data is expected to be already centered).
, intercept_scaling=1   # float, optional (default=1)
          # When self.fit_intercept is True, instance vector x becomes [x, self.intercept_scaling],
          #   i.e. a "synthetic" feature with constant value equals to intercept_scaling is appended to the instance vector.
          # The intercept becomes intercept_scaling * synthetic feature weight
          #   Note! the synthetic feature weight is subject to l1/l2 regularization as all other features.
          # To lessen the effect of regularization on synthetic feature weight (and therefore on the intercept)
          #   intercept_scaling has to be increased.
, loss='squared_hinge'  # string, 'hinge' or 'squared_hinge' (default='squared_hinge')
          # Specifies the loss function. 'hinge' is the standard SVM loss (used e.g. by the SVC class) while 'squared_hinge' is the square of the hinge loss.
, max_iter=1000      # int, (default=1000)
          # The maximum number of iterations to be run.
, multi_class='ovr'     # string, 'ovr' or 'crammer_singer' (default='ovr')
            # Determines the multi-class strategy if y contains more than two classes. "ovr" trains n_classes one-vs-rest classifiers, while "crammer_singer" optimizes a joint objective over all classes.
            # While crammer_singer is interesting from a theoretical perspective as it is consistent, it is seldom used in practice as it rarely leads to better accuracy and is more expensive to compute.
            # If "crammer_singer" is chosen, the options loss, penalty and dual will be ignored.
, penalty='l2'     #  string, 'l1' or 'l2' (default='l2')
          # Specifies the norm used in the penalization. The 'l2' penalty is the standard used in SVC. The 'l1' leads to coef_ vectors that are sparse.
, random_state=None   # int, RandomState instance or None, optional (default=None)
          # The seed of the pseudo random number generator to use when shuffling the data.
          # If int, random_state is the seed used by the random number generator;
          # If RandomState instance, random_state is the random number generator;
          # If None, the random number generator is the RandomState instance used by np.random.
, tol=0.0001       # float, optional (default=1e-4) Tolerance for stopping criteria.
, verbose=0        # int, (default=0)
          # Enable verbose output. Note that this setting takes advantage of a per-process runtime setting in liblinear that, if enabled, may not work properly in a multithreaded context.
)
```

04. 품질 최적화

Prediction → Demand Forecasting → Support Vector Machine

sklearn.svm.SVC

```
SVC(
C=1.0
, cache_size=200
, class_weight=None
, coef0=0.0
, decision_function_shape=None
, degree=3, gamma='auto'
, kernel='rbf'
, max_iter=-1
, probability=False
, random_state=None
, shrinking=True
, tol=0.001
, verbose=False
)
```

Prediction → Demand Forecasting → Support Vector Machine

sklearn.svm.LinearSVR

```
LinearSVR(
    C=1.0              # float, optional (default=1.0)
                # Penalty parameter C of the error term. The penalty is a squared l2 penalty. The bigger this parameter, the less regularization is used.
  , dual=True            # bool, (default=True)
                # Select the algorithm to either solve the dual or primal optimization problem. Prefer dual=False when n_samples > n_features.
  , epsilon=0.0           # float, optional (default=0.1)
                # Epsilon parameter in the epsilon-insensitive loss function. Note that the value of this parameter depends on the scale of the target variable y.
                # If unsure, set epsilon=0.
  , fit_intercept=True       # boolean, optional (default=True)
                # Whether to calculate the intercept for this model. If set to false, no intercept will be used in calculations (i.e. data is expected to be already centered).
  , intercept_scaling=1.0      # float, optional (default=1)
                # When self.fit_intercept is True, instance vector x becomes [x, self.intercept_scaling], i.e. a "synthetic" feature
                #   with constant value equals to intercept_scaling is appended to the instance vector.
                # The intercept becomes intercept_scaling * synthetic feature weight Note! the synthetic feature weight is subject to l1/l2 regularization
                #   as all other features.
                # To lessen the effect of regularization on synthetic feature weight (and therefore on the intercept) intercept_scaling has to be increased.
  , loss='epsilon_insensitive'  # string, 'epsilon_insensitive' or 'squared_epsilon_insensitive' (default='epsilon_insensitive')
                # Specifies the loss function. 'l1' is the epsilon-insensitive loss (standard SVR) while 'l2' is the squared epsilon-insensitive loss.
  , max_iter=1000          # int, (default=1000)
                # The maximum number of iterations to be run.
  , random_state=None        # int, RandomState instance or None, optional (default=None)
                # The seed of the pseudo random number generator to use when shuffling the data.
                # If int, random_state is the seed used by the random number generator; If RandomState instance, random_state is the random number generator;
                # If None, the random number generator is the RandomState instance used by np.random.
  , tol=0.0001           # float, optional (default=1e-4)
                # Tolerance for stopping criteria.
  , verbose=0            # int, (default=0)
                # Enable verbose output. Note that this setting takes advantage of a per-process runtime setting in liblinear that, if enabled, may not work properly in a multi
threaded context.

)
```

04. 품질 최적화

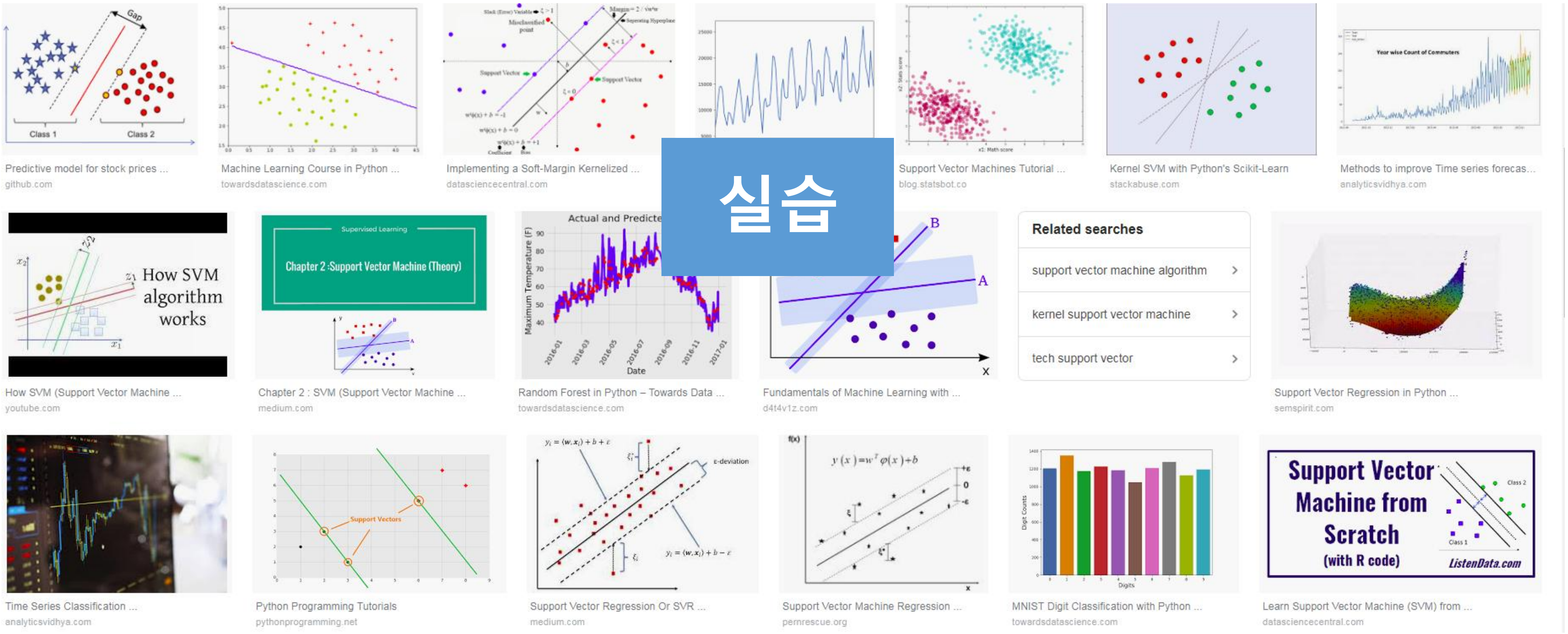
Prediction → Demand Forecasting → Support Vector Machine

sklearn.svm.SVR

```
SVR(
C=1.0
, cache_size=200
, coef0=0.0
, degree=3
, epsilon=0.1
, gamma='auto'
, kernel='rbf'
, max_iter=-1
, shrinking=True
, tol=0.001
, verbose=False
)
```

04. 품질 최적화

Prediction → Demand Forecasting → Support Vector Machine



실습

Optimization Problem

- 한정 자원 범위내에서의 제조상의 제반 목적의 최적화 문제의 Solution 찾기
  - 생산 최대화 – 주어진 자원하의 최대생산 조합
  - 물류 효율성 – 수송 경로, 시간 최소화, 비용 최소화
  - ,비용 절감 – 효율적 자원 배분, 부품 조달 비용
- 한정 자원
  - 인적자원, 원재료/부품, 설비, 가용시간 등

선형 계획(Linear Programming)

- 복수의 가변 요소들의 1차 방정식의 제약조건

$$
\begin{aligned}
&\text{minimize } f(x) \\
&\text{subject to } g_i(x) \leq 0 \text{ for each } i \in \{1, \ldots, m\} \\
&\qquad h_j(x) = 0 \text{ for each } j \in \{1, \ldots, p\} \\
&\qquad x \in X.
\end{aligned}
$$

비선형 계획(Non-Linear Programming)

- 복수의 가변 요소들의 다차원 방정식의 제약조건을 포함하는 경우

maximize

$$f(\mathbf{x}) = x_1 + x_2$$

where $\mathbf{x} = (x_1, x_2)$.

subject to

$$x_1 \geq 0$$
$$x_2 \geq 0$$
$$x_1^2 + x_2^2 \geq 1$$
$$x_1^2 + x_2^2 \leq 2$$

Optimization Problem

2016 3rd International Conference on Mechanical, Industrial, and Manufacturing Engineering (MIME 2016)
ISBN: 978-1-60595-313-7

Optimization of Production Plan Based on Linear Multi-objective Programming

$$\max\begin{cases} z_1 = c_{11}x_1 + c_{12}x_2 + \ldots + c_{1n}x_n \\ z_2 = c_{21}x_1 + c_{22}x_2 + \ldots + c_{2n}x_n \\ \ldots \\ z_r = c_{r1}x_1 + c_{r2}x_2 + \ldots + c_{rn}x_n \end{cases} \quad (1)$$

Restrictions:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \le b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \le b_2 \\ \ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \le b_m \\ x_1, x_2, \ldots, x_n \ge 0 \end{cases} \quad (2)$$

Multi-objective linear programming in the form of the matrix is as follows:

The objective function: $\max Z = cx$

Restrictions: $$\begin{cases} Ax \le b \\ x \ge 0 \end{cases} \quad (3)$$

An algorithm for nonlinear optimization problems with binary variables

Walter Murray, Kien Ming Ng • Published in Comp. Opt. and Appl. 2010 • DOI: 10.1007/s10589-008-9218-1

One of the challenging optimization problems is determining the minimizer of a nonlinear programming problem that has binary variables. A vexing difficulty is the rate the work to solve such problems increases as the number of discrete variables increases. Any such problem with bounded discrete variables, especially binary variables, may be transformed to that of finding a global optimum of a problem in continuous variables. However, the transformed problems usually have astronomically

Gradient-based Methods for Production Optimization of Oil Reservoirs

PROFIT MAXIMIZATION SOLID TRANSPORTATION PROBLEM UNDER BUDGET CONSTRAINT USING FUZZY MEASURES

03. 설비 예지보전 및 생산 최적화

Optimization Problem

실습

THANK YOU

감사합니다