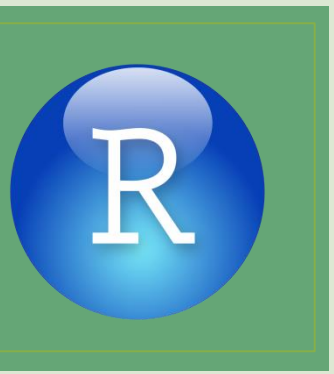
'가성비' 좋은 노트북의 예측과 그 활용

통계학과 오희준
독어독문학과 이민주

CONTENTS



01

주제 소개

주제 선정 배경

* 발표, 과제, 팀플 등

R

sas
THE POWER TO KNOW.

* 프로그래밍

* 휴대성

주제 선정 배경

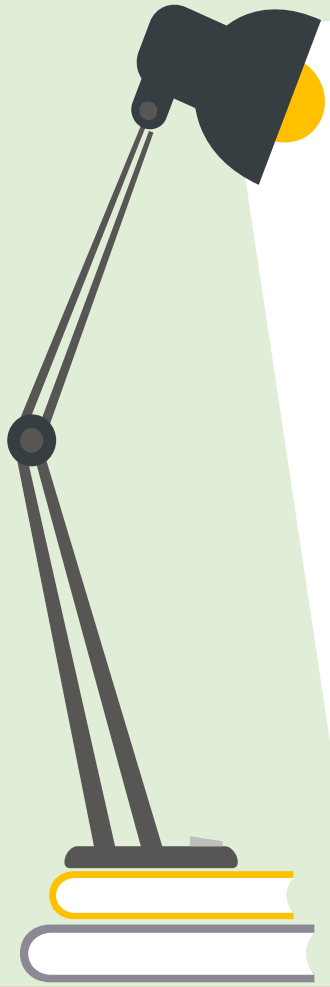
NAVER 노트북추천해주세요

통합검색 블로그 카페 지식iN 이미지 동영상 어학사전 뉴스 더보기 검색옵션

정렬 기간 영역 옵션유지 꺼짐 켜짐 상세검색

연관검색어 성능좋은노트북추천 삼성 노트북 추천 대학생 노트북 추천 미니노트북추천 신고
여대생 노트북 추천 싼 노트북 추천 인강용 노트북 추천 여자 노트북 추천
엘지 노트북 추천 디자인과 노트북 추천 쓸만한 노트북 업무용 노트북 추천
13인치 노트북 추천 울트라씬노트북추천 사무용 노트북 추천
신입생 노트북 추천 디자인 노트북 추천 hp 노트북 추천 디자인용노트북추천 접기

컴퓨터를 잘 알지 못하는 사람들도
합리적으로 노트북을 구매할 수 있게 도와 주기 위함



노트북 가격과 구성요소 데이터를 이용해

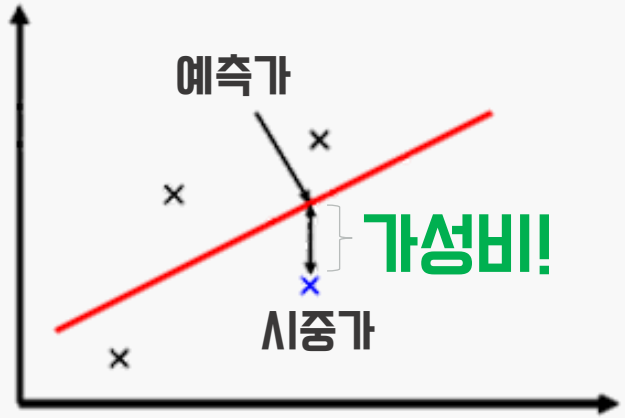
가성비 좋은 노트북을 찾기 위한

모델을 만들어보고 더 나아가 활용해보자!

주제 구체화

가성비의 개념 정의

* 예측가 : 노트북 시중가격의 부품 성능 별 가중치를 고려한 모델 예측값



시중가 < 예측가	가성비가 좋다
시중가 > 예측가	가성비가 좋지 않다

분석 활용

가성비 좋은 노트북 추천!

가성비가 좋은 노트북 중
개인의 선호에 맞춘 기능으로
필터링하여 노트북 추천!

분석 활용

NEW

새롭게 출시될 노트북의 적정 가격 제시

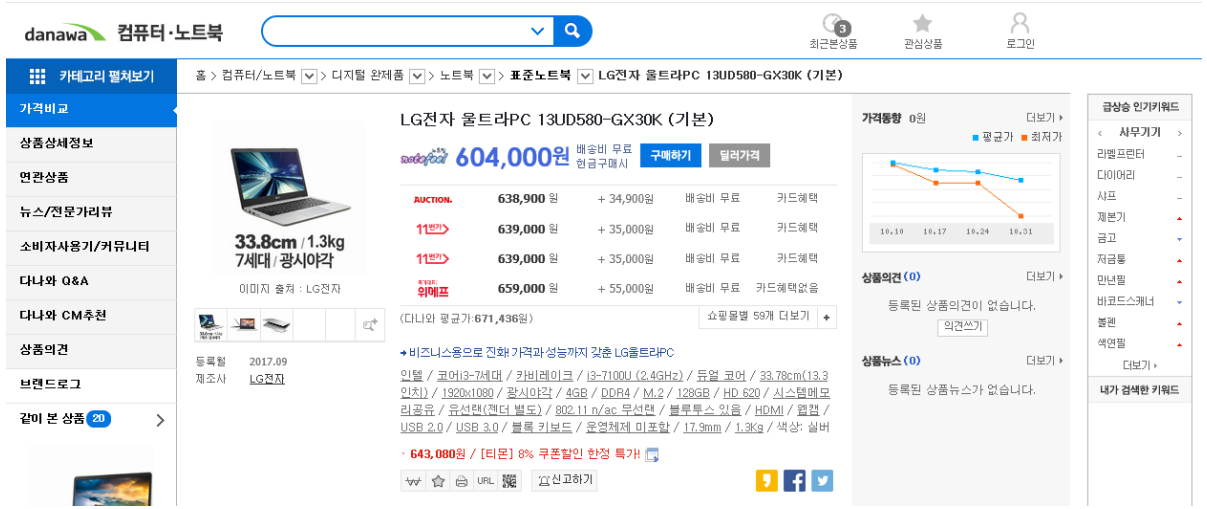
새롭게 출시된 혹은 출시 될 모델의
성능을 바탕으로 모델에 적합하여
적정 가격 제시!

02

데이터 소개

www. DANAWA .com 노트북 가격 비교 사이트 <다나와>

1 <다나와> 노트북 정보 크롤링



제조회사	LG전자 (제조사 웹사이트 바로가기)	등록년월	2017년 09월
CPU 제조사	인텔	CPU 종류	코어i3-7세대
CPU 코드명	카비레이크	CPU 넘버	i3-7100U (2.4GHz)
코어 형태	듀얼 코어		
디스플레이			
화면 크기	33.78cm(13.3인치)	화면 비율	와이드 16:9
해상도	1920x1080		
디스플레이 특징			
눈부심방지		광시야각	○
터치스크린		밝기자동조절	
회전LCD		베젤없음	
G-Sync		슬림형 베젤	
120Hz 지원		144Hz 지원	
와이드뷰			
메모리 특징			
메모리 용량	4GB	메모리 타입	DDR4
HDD 특징			
7200 rpm		HDD 충격 보호	
HDD 용량			
SSD 형태			
SSD		mSATA	
M.2	○	eMMC	

3000여개 노트북별 성능 및 가격 정보 데이터 구축!

* 2016년 1월 이후 등록된 노트북만 크롤링 (2017.11.2~4)

www. DANAWA .com 노트북 가격 비교 사이트 <다나와>

2 데이터 전처리

1. 특정 변수(배터리, 어댑터 등)에서 나타난 결측값 채우기

   그래도 채워지지 않은 결측값은 knnImputation 사용

2. 중복으로 크롤링된 observation 삭제

3. 외장 그래픽 카드가 특정 모델이 아닌 observation 삭제 (∵ 점수 부여 X)

4. 해상도와 모니터 크기(inch)로 DPI 변수 생성

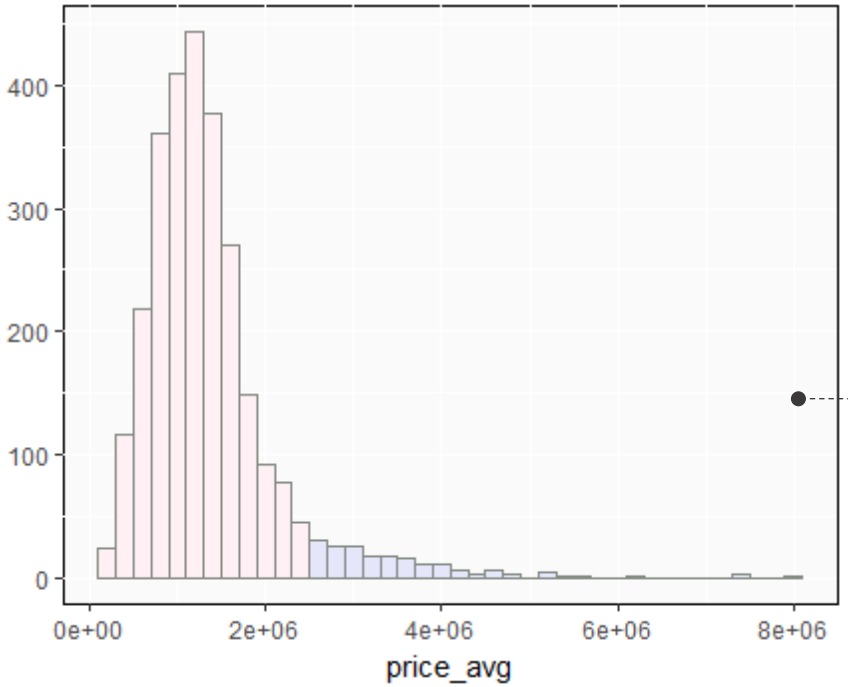
   * DPI : dots per inch (단위 해상도)

총 2766개 관측값!

www. DANAWA .com 노트북 가격 비교 사이트 <다나와>

3 변수 탐색 - 반응변수

price_avg : 다나와 사이트에 기재된 각종 쇼핑몰의 평균 가격




```
> summary(dat$price_avg)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 161006  888000 1219832 1348516 1573872 7947050
```

전체 데이터의 **93%**를
차지하는
250만원 이하 가격의
노트북만 분석 진행!

※ 실제 분석에서는 가격의 범위를 고려하여 log 취함

www. DANAWA .com 노트북 가격 비교 사이트 <다나와>

3 변수 탐색 - 설명변수

<기본적인 정보>

name	모델명	company	제조회사
release_month	등록 후 개월 수	OS	운영체제
weight	무게 (kg)	world_warranty	전세계 a/s 서비스
battery	배터리 (Wh)	adapter	어댑터 (Wh)

<처리 능력>

CPU_company	CPU 제조회사	CPU_score	CPU 점수
core_num	프로세서 수		

산업 표준화 지수를 차용하여 만든 CPU 성능진단 점수 (Passmark)

www. DANAWA .com 노트북 가격 비교 사이트 <다나와>

3 변수 탐색 - 설명변수

<기억장치>

memory	DDR4(L) 여부	memory_vol	메모리 용량 (RAM)	주기억장치
faster_HDD	7200rpm 이상 여부	HDD_vol	하드디스크 용량	보조기억장치
SSD_vol	SSD 용량	multireader	SD카드 삽입할 수 있는 매체 존재 유무	

<그래픽>

Graphic_inner	내장 그래픽카드 유무	Graphic_outer	외장 그래픽카드 유무
outerGPU_3D	외장 그래픽 카드 3D 성능 상대 점수	outerGPU_Compute	외장 그래픽 카드 연산 기능 상대 점수

www. DANAWA .com 노트북 가격 비교 사이트 <다나와>

3 변수 탐색 - 설명변수

<모니터>

Monitor	모니터 크기 (inch)	Anti_glare	눈부심방지
Wideview	와이드뷰	Touchscreen	터치스크린
Brightness_auto	밝기 자동 조절	RotateLCD	회전LCD
Refresh	화면 프레임 속도 120Hz 이상	Dpi	화면 1인치 당 dots 개수

www. DANAWA .com 노트북 가격 비교 사이트 <다나와>

3 변수 탐색 - 설명변수

<멀티미디어>

DVDrecorder	DVD 쓰기 가능	BlueRayrecorder	Blueray 쓰기 가능
multiboost	HDD, SSD 포함 CD, DVD, Blueray 삽입 가능 매체 유무	bluetooth	블루투스
USB_type	USB 단자 종류	type_C	USB Type c 존재
display_out_num	화면 출력 종류 개수	webcam	웹캠 존재

www. DANAWA .com 노트북 가격 비교 사이트 <다나와>

3 변수 탐색 - 설명변수

<키보드>

number_key	숫자 키보드	convex_key	볼록 키보드
mechanical_key	기계식 키보드	antiwater_key	침수방지 키보드
light_key	키보드 라이트 (백색)	RGBlight_key	RGB 라이트

<기타>

wired_lan	유선랜	fingerscan	지문 인식

총 48개 설명변수!

03

데이터 분석

DATA ANALYSIS

파일 홈 공유 보기 관리

분석 목차

- 성균관대학교
  - 통계학과
  - 과제


Cluster Analysis


Linear Regression


Tree


Random Forest


Boosting


GAM

DATA ANALYSIS

파일 홈 공유 보기 관리

군집분석

- 성균관대학교
  - 통계학과
  - 과제


군집분석


Linear Regression


Tree


Random Forest


Boosting


GAM

cafe.naver.com/powersat

0 왜 군집분석?

성능 별로 노트북을 구분해보자!

1 비계층적 군집분석

K-means

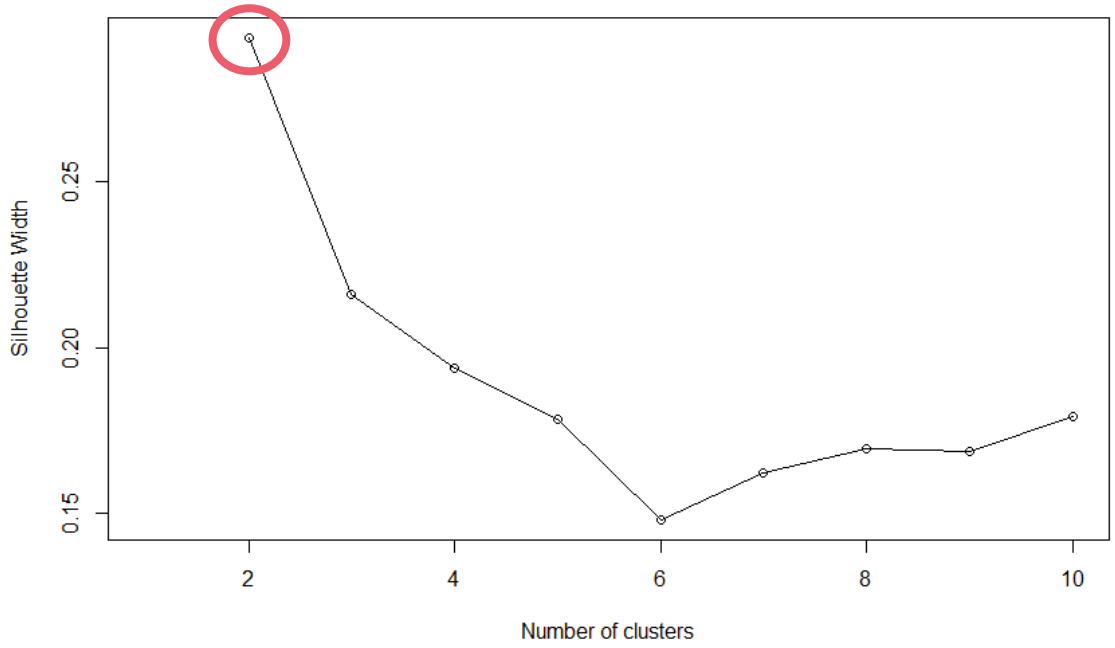
- ✓ 관측값들의 **평균값**을 사용해 중심점(Centroid) 계산
- ✓ 평균을 사용하기 때문에 이상치에 민감
- ✓ **연속형 변수**들만 사용 가능
- ✓ 변수들의 scale에 민감함

K-medoids

- ✓ 1개 이상의 관측값을 medoids로 설정하고 군집화
- ✓ k-means보다 이상치에 robust
- ✓ 연속형, 범주형, 혼합형 유사도 척도를 사용하여 **모든 데이터 유형**의 유사성을 계산 가능

* 수치형 데이터는 변수간 단위 차이에 영향을 많이 받으므로 변수에 가중치를 줄 것이 아니라면 반드시 정규화(표준화)필요!

2 군집 개수에 따른 실루엣 값 그래프



군집이 2개일 때 최적!

3 Group 1 – 군집 분석 결과

```
> summary(group1$price_avg)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 161006  780216 1141359 1167111 1455310 3935022

> summary(group1$CPU_score)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    884    3275    4009    4074    4697   10110

> summary(group1$graphic_outer)
   0    1
1505  143

> summary(group1$monitor)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   10.1    13.3    14.0    14.4    15.6    17.3

> summary(group1$weight)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.690   0.980   1.340   1.448   1.860   3.540
```

1. 중간값 110만원 대로 상대적으로 낮은 가격
2. 낮은 CPU 성능, 대부분이 듀얼 코어
3. 대부분 외장 그래픽카드가 없음
4. 최소 10.10인치부터 최대 17.30인치까지 다양한 크기
5. 대부분의 키보드 관련 튜닝 기능이 적용되지 않음.
6. 중간값 1.3kg의 가벼운 무게

3 Group 1 – 군집 분석 결과

1. 중간값 110만원의 상대적으로 낮은 가격
2. 낮은 CPU 성능, 대부분이 듀얼 코어
3. 대부분 외장 그래픽카드가 없음
4. 최소 10.10인치부터 최대 17.30인치까지 다양한 크기
5. 대부분의 키보드 관련 튜닝 기능이 적용되지 않음.
6. 중간값 1.3kg의 가벼운 무게

단순 사무용/넷북/울트라북 등
중저가 제품군!

3 Group 2 – 군집 분석 결과

```
> summary(group2$price_avg)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 399000 1024752 1345713 1616158 1880356 7947050

> summary(group2$memory_vol)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   4.00    8.00    8.00   10.84   16.00   64.00

> summary(group2$CPU_score)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2677    5212    8140    7261    8937   14642

> summary(group2$outerGPU_3D)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      0    1239    2720    4068    5738   15365

> summary(group2$weight)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.090   2.040   2.400   2.454   2.620   5.700
```

1. 중간값 130만원의 높은 가격
2. 최신 고속 DDR4메모리 적용, 높은 용량의 메모리
3. 높은 CPU 성능, 대부분이 쿼드 코어
4. 대부분 외장그래픽카드 탑재, 높은 그래픽카드 성능
5. 중간값 15인치 이상의 대화면 적용
6. 중간값이 2.46kg인 무거운 노트북

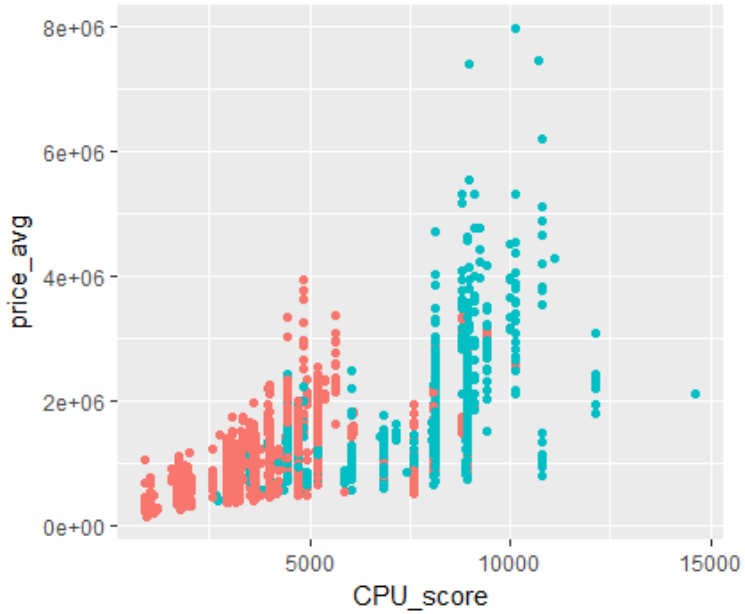
3 Group 2 – 군집 분석 결과

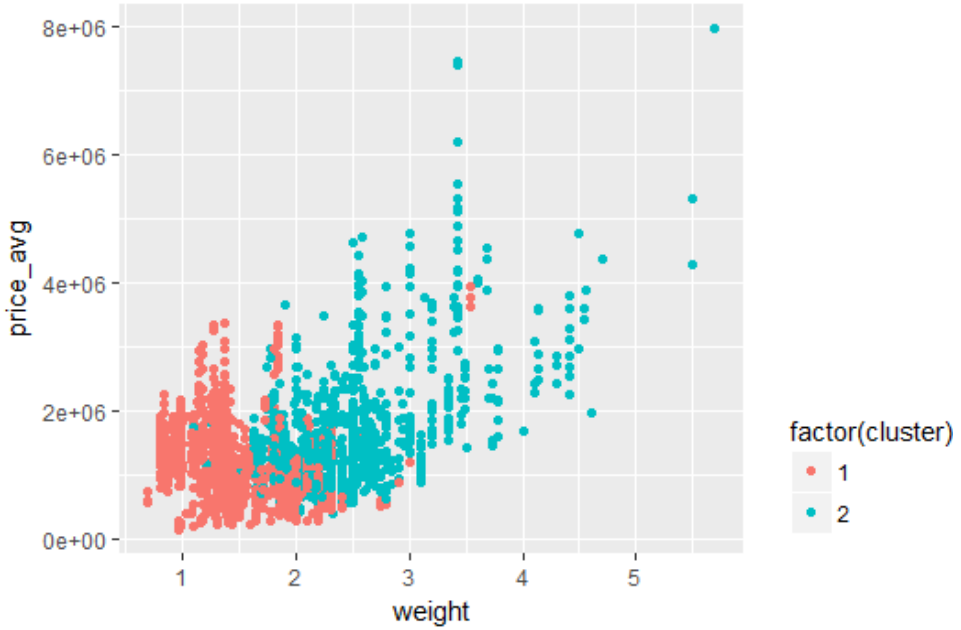
1. 중간값 130만원의 높은 가격
2. 최신 고속 DDR4메모리 적용, 높은 용량의 메모리
3. 높은 CPU 성능, 대부분이 쿼드 코어
4. 대부분 외장그래픽카드 탑재, 높은 그래픽카드 성능
5. 중간값 15인치 이상의 대화면 적용
6. 중간값이 2.46kg인 무거운 노트북

게임용/그래픽 작업용/워크스테이션

고성능 제품군!

4 Clustering - 시각화





➡ 시각화를 통해 군집이 잘 나뉘었음을 알 수 있다!

cafe.naver.com/powersat

연속형 변수 간 상관관계 확인

Group 1

```
> cor(group1$outerGPU_3D,group1$outerGPU_Compute)
[1] 0.9904558
```

Group 2

```
> cor(group2$outerGPU_3D,group2$outerGPU_Compute)
[1] 0.9970793
```

outerGPU_3D	외장 그래픽카드의 3D 성능 점수
outerGPU_Compute	외장 그래픽카드의 연산 성능 점수

높은 상관관계 존재! ➡ outerGPU_3D 제거 후 분석 진행

각 제조회사의 빈도수 확인

Group 1

1	ACER	36
2	AORUS	3
3	APPLE	53
4	ASUS	130
5	DELL	18
6	HP	188
7	LG전자	406
8	MPGIO	1
9	MSI	3
10	디지탈클릭	2
11	레노버	190
12	삼성전자	522
13	정현씨앤씨	3
14	주연테크	7
15	포유디지탈	3
16	한성컴퓨터	40

→

1	ACER	36
2	other	22
3	APPLE	53
4	ASUS	130
5	DELL	18
6	HP	188
7	LG전자	406
8	레노버	190
9	삼성전자	522
10	한성컴퓨터	40

Group 2

1	ACER	24
2	AORUS	5
3	ASUS	138
4	DELL	45
5	GIGABYTE	88
6	HP	309
7	LG전자	64
8	MSI	76
9	TG삼보	2
10	레노버	60
11	삼성전자	95
12	주연테크	10
13	한성컴퓨터	62

→

1	ACER	24
2	other	7
3	ASUS	138
4	DELL	45
5	GIGABYTE	88
6	HP	309
7	LG전자	64
8	MSI	76
9	레노버	60
10	삼성전자	95
11	주연테크	10
12	한성컴퓨터	62

빈도의 **편차**가 **크고**
범주의 **수**가 **매우 많음**

➡ 빈도가 10보다 작은 제조회사를 묶어 'other'로 처리

DATA ANALYSIS

파일 홈 공유 보기 관리

선형 회귀 모형

- 성균관대학교
  - 통계학과
  - 과제


Cluster Analysis


선형 회귀 모형


Tree


Random Forest


Boosting


GAM

Group 1 – Train Set 적합

STEPWISE 실시

```
null<-lm(logprice~1,data=train1)
full<-lm(logprice~battery+adapter+memory+company+release_month+
           CPU_score+core_num+monitor+anti_glare+wideview+touchscreen+
           brightness_auto+rotateLCD+memory_vol+faster_HDD+HDD_vol+
           SSD_vol+DVDrecorder+multiboost+graphic_inner+graphic_outer+graphic_memory+
           outerGPU_Compute+wired_lan+bluetooth+display_output_num+
           webcam+USB_type+Type_C+multireader+number_key+convex_key+antiwater_key+
           light_key+RGBlight_key+fingerscan+weight+world_warranty+dpi,
         data=train1) #delete OS (singularity)
step(null,scope=list(lower=null,upper=full),direction="both",trace=0)

lm.fit1<-lm(formula = logprice ~ memory_vol + company + CPU_score + weight +
              release_month + core_num + touchscreen + monitor + outerGPU_Compute +
              antiwater_key + dpi + display_output_num + memory + SSD_vol +
              number_key + Type_C + fingerscan + wideview + DVDrecorder +
              battery + HDD_vol + anti_glare + multireader + graphic_inner +
              graphic_outer, data = train1) # all variables
```

Check!

- OS 변수 때문에 singularity 발생
- factor의 값이 하나만 존재하는 변수는 제외하고 적합
  (ex. mechanical_key, refresh…)

Group 1 – Train Set 가정 확인

```
> shapiro.test(residuals(lm.fit1))

        Shapiro-Wilk normality test

data:  residuals(lm.fit1)
W = 0.98609, p-value = 7.321e-09

> bptest(lm.fit1)

        studentized Breusch-Pagan test

data:  lm.fit1
BP = 115.18, df = 35, p-value = 1.76e-10

> dwtest(lm.fit1)

        Durbin-Watson test

data:  lm.fit1
DW = 1.9518, p-value = 0.2105
alternative hypothesis: true autocorrelation is greater than 0
```

Check!

1. 정규성 만족 X
2. 등분산성 만족 X
3. 독립성 만족 O (노트북들 간의 독립성)

cafe.naver.com/powersat

Group 1 – Test Set

```
> yhat.lm<-predict(lm.fit1,newdata=test1)
> mean((yhat.lm-test1[,"logprice"])^2)
[1] 0.03077082
> sqrt(mean((yhat.lm-test1[,"logprice"])^2))
[1] 0.1754161
> var(yhat.lm)/var(test1$logprice)
[1] 0.8796989
```

변동(분산)을 얼마나 잘 설명하는가

Check!

1. MSE: 0.03077082
2. RMSE: 0.1754161
3. **설명력**: 0.879689

cafe.naver.com/powersat

Group 2 – Train Set 적합 & 가정 확인

STEPWISE 실시

```
null<-lm(logprice~1,data=train2)
full<-lm(logprice~battery+adapter+memory+company+release_month+CPU_company+
           CPU_score+monitor+anti_glare+wideview+touchscreen+graphic_memory+
           brightness_auto+rotateLCD+memory_vol+faster_HDD+HDD_vol+
           SSD_vol+DVDrecorder+multiboost+graphic_inner+graphic_outer+
           outerGPU_Compute+wired_lan+bluetooth+display_output_num+
           webcam+USB_type+Type_C+multireader+number_key+convex_key+antiwater_key+
           light_key+RGBlight_key+fingerscan+OS+weight+world_warranty+dpi,
         data=train2)
step(null,scope=list(lower=null,upper=full),direction="both",trace=0)

lm.fit2<-lm(formula = logprice ~ battery + memory_vol + company + OS +
              CPU_score + fingerscan + SSD_vol + release_month + CPU_company +
              outerGPU_Compute + antiwater_key + number_key + monitor +
              display_output_num + RGBlight_key + dpi + memory + graphic_inner +
              Type_C + DVDrecorder + faster_HDD + multiboost + wideview +
              brightness_auto + HDD_vol + light_key + multireader, data = train2)
```

```
> shapiro.test(residuals(lm.fit2))

        Shapiro-Wilk normality test

data:  residuals(lm.fit2)
W = 0.97984, p-value = 4.255e-08
```

정규성 X

```
> bptest(lm.fit2)

        studentized Breusch-Pagan test

data:  lm.fit2
BP = 73.163, df = 42, p-value = 0.002044
```

등분산성 X

```
> dwtest(lm.fit2)

        Durbin-Watson test

data:  lm.fit2
DW = 2.0305, p-value = 0.6534
```

독립성 O

Group 2 – Test Set

```
> yhat.lm2<- predict(lm.fit2,newdata=test2)
> mean((yhat.lm2-test2[,"logprice"])^2)
[1] 0.01964755
> sqrt(mean((yhat.lm2-test2[,"logprice"])^2))
[1] 0.1401697
> var(yhat.lm2)/var(test2$logprice)
[1] 0.909529
```

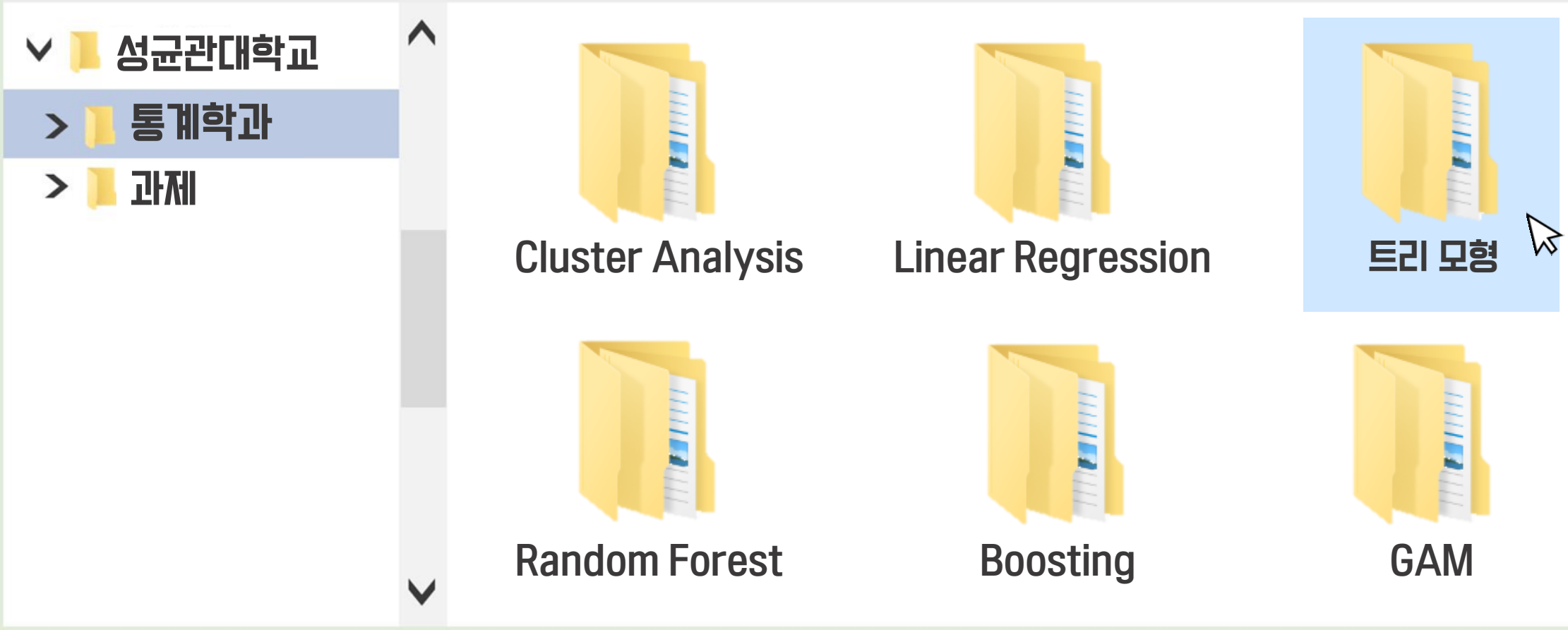
Check!

1. MSE: 0.01964755
2. RMSE: 0.1401697
3. 설명력: 0.909529

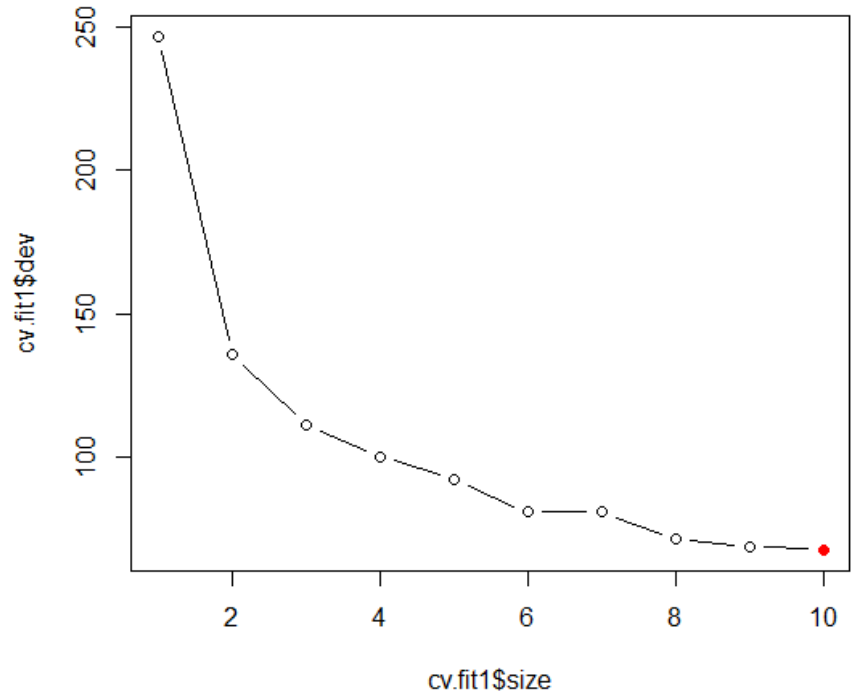
DATA ANALYSIS

파일 홈 공유 보기 관리

트리 모형



Group 1 – Train Set



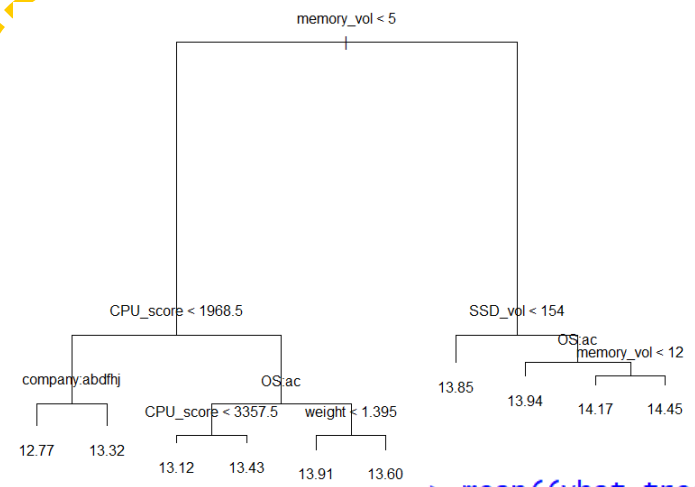
Check!

- Tree의 Node를 몇 개로 했을 때 cross validation 에러가 가장 작을까?

⇒ Deviance가 가장 낮은

Tree의 노드 개수는 10개 (가지치기 X)

Group 1 – Test Set

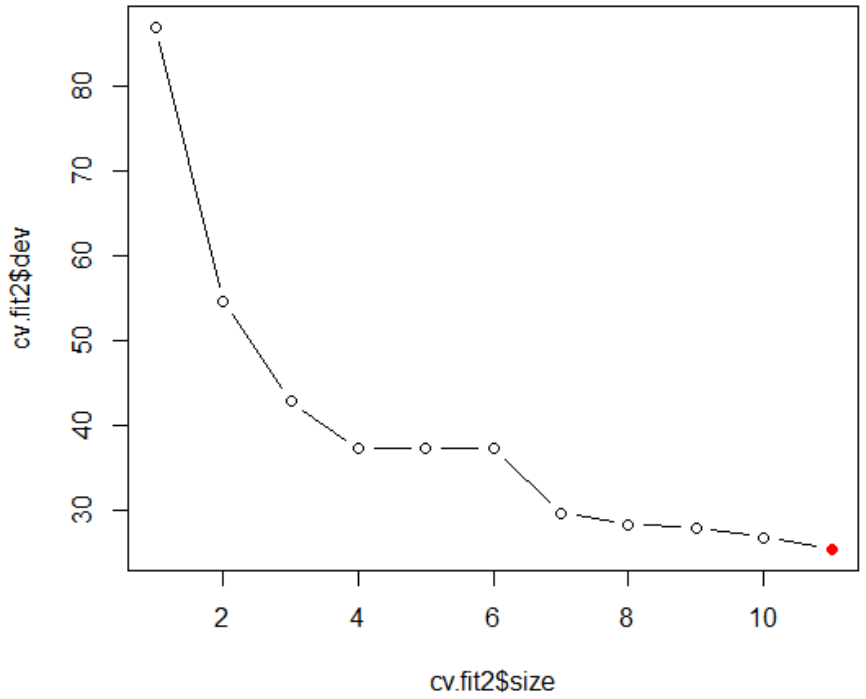



```
> mean((yhat.tree1-test1[,"logprice"])^2)
[1] 0.05853857
> sqrt(mean((yhat.tree1-test1[,"logprice"])^2))
[1] 0.2419475
> var(yhat.tree1)/var(test1$logprice)
[1] 0.706084
```

Check!

1. MSE: 0.05853857
2. RMSE: 0.2419475
3. 설명력: 0.706084

Group 2 – Train Set, Tree



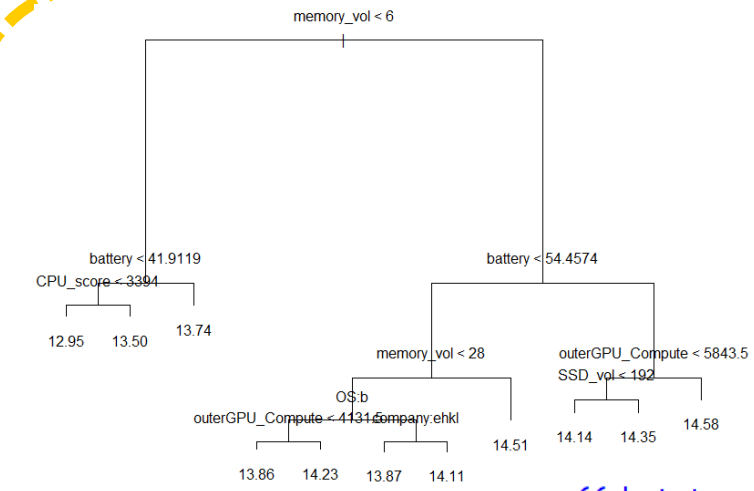
Check!

- Tree의 Node를 몇 개로 했을 때 cross validation 에러가 가장 작을까?

⇒ Deviance가 가장 낮은

Tree의 노드 개수는 **11개** (가지치기 X)

Group 2 – Test Set




```
> mean((yhat.tree2-test2[,"logprice"])^2)
[1] 0.04356419
> sqrt(mean((yhat.tree2-test2[,"logprice"])^2))
[1] 0.2087204
> var(yhat.tree2)/var(test2$logprice)
[1] 0.8152141
```

Check!

1. MSE: 0.043546419
2. RMSE: 0.2087204
3. 설명력: 0.8152141

DATA ANALYSIS

파일 홈 공유 보기 관리

랜덤 포레스트

- 성균관대학교
  - 통계학과
  - 과제


Cluster Analysis


Linear Regression


Tree


랜덤 포레스트


Boosting


GAM

cafe.naver.com/powersat

Group 1

```
set.seed(10)
rf.fit1<-randomForest(train1[,-46],train1$logprice,ntree=500,mtry=15)
yhat.rf1<-predict(rf.fit1,newdata=test1)
```

15개 변수를 선택하여 500개의 나무(숲) 생성

```
> yhat.rf1<-predict(rf.fit1,newdata=test1)
> mean((yhat.rf1-test1[,"logprice"])^2)
[1] 0.01634308
> sqrt(mean((yhat.rf1-test1[,"logprice"])^2))
[1] 0.1278401
> var(yhat.rf1)/var(test1$logprice)
[1] 0.7978367
```

Check!

매우 정확한 예측력!

1. MSE: 0.01634308
2. RMSE: 0.1278401
3. 설명력: 0.7978367

다소 떨어지는 설명력!

Group 2

```
set.seed(10)
rf.fit2<-randomForest(train2[,-46],train2$logprice,ntree=500,mtry=15)
yhat.rf2<-predict(rf.fit2,newdata=test2)
```

15개 변수를 선택하여 500개의 나무(숲) 생성

```
> mean((yhat.rf2-test2[,"logprice"])^2)
[1] 0.01607958
> sqrt(mean((yhat.rf2-test2[,"logprice"])^2))
[1] 0.1268053
> var(yhat.rf2)/var(test2$logprice)
[1] 0.7555339
```

Check!

매우 정확한 예측력!

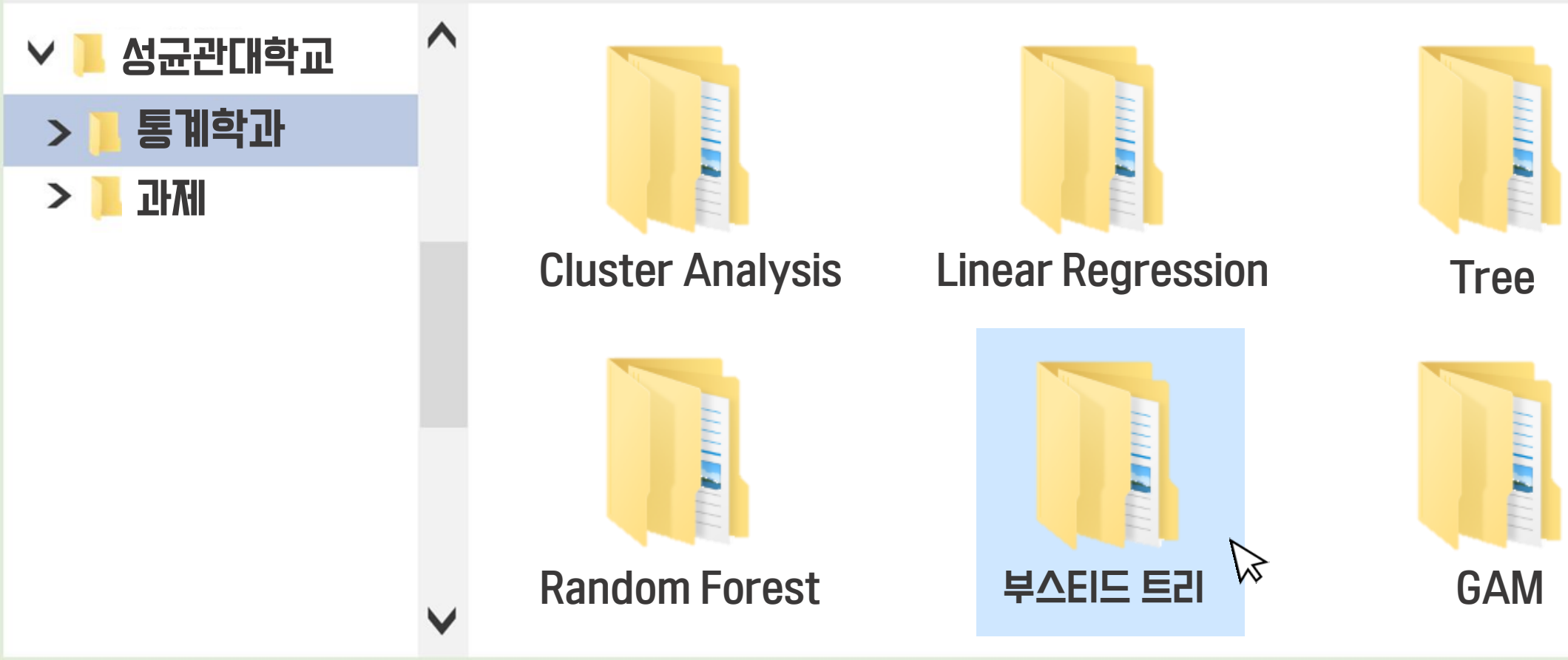
1. MSE: 0.01607958
2. RMSE: 0.1268053
3. **설명력**: 0.7555339

다소 떨어지는 설명력!

DATA ANALYSIS

파일 홈 공유 보기 관리

부스티드 트리



Group 1 – Parameter Tuning I

```
fitControl<-trainControl(method="cv", number=5, returnResamp = "all")
set.seed(1234) # tuning shrinkage & interaction.depth
bt.tuning<-train(logprice~., data=newtrain1, method="gbm",distribution="gaussian",
                 trControl=fitControl, verbose=F,
                 tuneGrid=data.frame(.n.trees=9000, .shrinkage=rep(c(0.03,0.04),2),
                                     .interaction.depth=c(2,3,3,2), .n.minobsinnode=10))
> bt.tuning
Stochastic Gradient Boosting

1123 samples
  40 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 899, 899, 898, 898, 898
Resampling results across tuning parameters:

  shrinkage  interaction.depth  RMSE       Rsquared   MAE
  0.03       2                  0.1341533  0.9170641  0.09363054
  0.03       3                  0.1324395  0.9190405  0.09019787
  0.04       2                  0.1339208  0.9173988  0.09288106
  0.04       3                  0.1313284  0.9205778  0.08918176

Tuning parameter 'n.trees' was held constant at a value of 9000
Tuning
 parameter 'n.minobsinnode' was held constant at a value of 10
RMSE was used to select the optimal model using  the smallest value.
The final values used for the model were n.trees = 9000, interaction.depth = 3,
 shrinkage = 0.04 and n.minobsinnode = 10.
```

shrinkage 0.03 or 0.04

Interaction.depth 2개 or 3개
(각 트리 별 노드 개수)

4개의 조합!

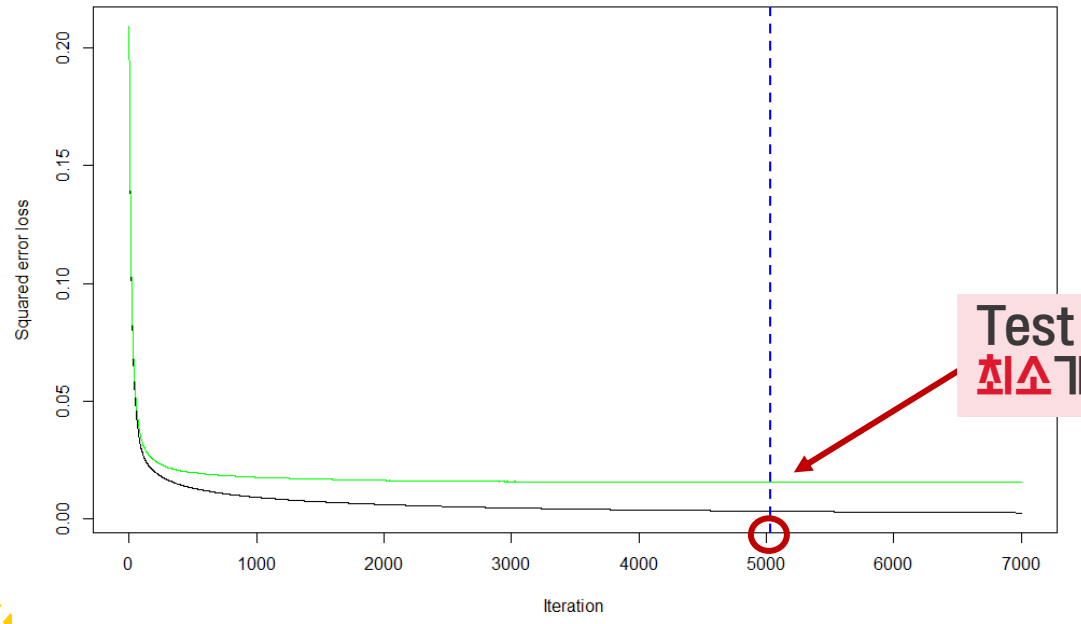
Check!

- 최적의 노드 개수 : 3개
- 수렴속도 : 0.04
- caret 패키지의 파라미터 튜닝 함수를 통해 얻은 값

※ 트리의 개수와 노드 안 최소 데이터 수는 고정

Group 1 – Parameter Tuning II

```
> best.tree<-gbm.perf(boost.cv);best.tree
Using cv method...
[1] 5032
```



Test RMSE가 **최소**가 되는 점

Check!

- 최적의 반복 횟수 (나무 개수) : 5032개
- gbm 함수 내의 cv.folds 옵션 사용

cafe.naver.com/powersat

Group 1

```
boost1<- gbm(logprice~.,data=newtrain1,distribution="gaussian",n.trees=best.tree,
             shrinkage=0.04,interaction.depth=3)
```

앞에서 parameter tuning 과정을 통해 도출한 parameter 사용

```
> mean((yhat.bt-test1[,"logprice"])^2)
[1] 0.011375
> sqrt(mean((yhat.bt-test1[,"logprice"])^2))
[1] 0.1066536
> var(yhat.bt)/var(test1$logprice)
[1] 0.9503649
```

Check!

1. MSE: 0.011375
2. RMSE: 0.1066536
3. 설명력: 0.9503649

예측력과 설명력 모두 뛰어남!

Group 2 – Parameter Tuning I

아까와 같은 방식으로!

```
> bt.tuning
Stochastic Gradient Boosting

684 samples
 42 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 545, 548, 547, 548, 548
Resampling results across tuning parameters:

  shrinkage  interaction.depth  RMSE       Rsquared   MAE
  0.03       2                 0.1142316  0.8970404  0.07925605
  0.03       3                 0.1160836  0.8940164  0.07888228
  0.04       2                 0.1152775  0.8949871  0.07961852
  0.04       3                 0.1166560  0.8926084  0.08022632

Tuning parameter 'n.trees' was held constant at a value of 9000
Tuning
 parameter 'n.minobsinnode' was held constant at a value of 10
RMSE was used to select the optimal model using  the smallest value.
The final values used for the model were n.trees = 9000, interaction.depth = 2,
 shrinkage = 0.03 and n.minobsinnode = 10.
```

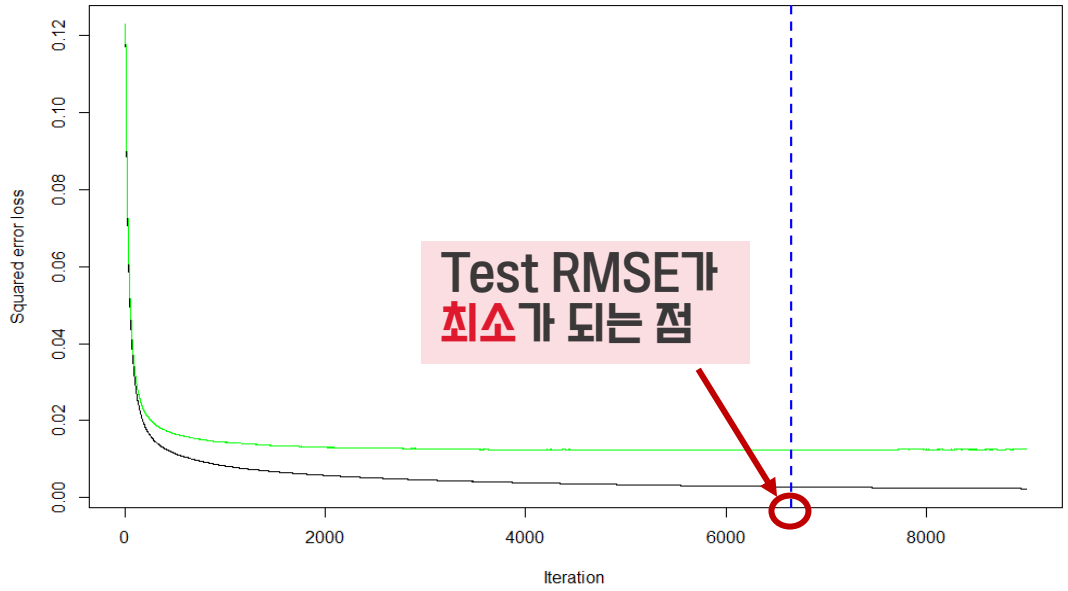
Check!

- 최적의 노드 개수 : 2개
- 수렴속도 : 0.03

※ 트리의 개수와 노드 안 최소 데이터 수는 고정

Group 2 – Parameter Tuning II

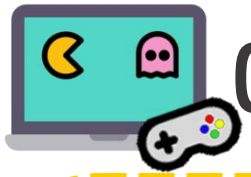
```
> best.tree<-gbm.perf(boost.cv);best.tree
Using cv method...
[1] 6658
```



Check!

- 최적의 반복 횟수 (나무 개수) : 6658개
- gbm 함수 내의 cv.folds 옵션 사용

cafe.naver.com/powersat



Group 2

```
boost2<- gbm(logprice~.,data=newtrain2,distribution="gaussian",n.trees=best.tree,
             shrinkage=0.03,interaction.depth=2)
```

앞에서 parameter tuning 과정을 통해 도출한 parameter 사용

```
> mean((yhat.bt-test2[,"logprice"])^2)
[1] 0.01112946
> sqrt(mean((yhat.bt-test2[,"logprice"])^2))
[1] 0.1054963
> var(yhat.bt)/var(test2$logprice)
[1] 0.9343957
```

Check!

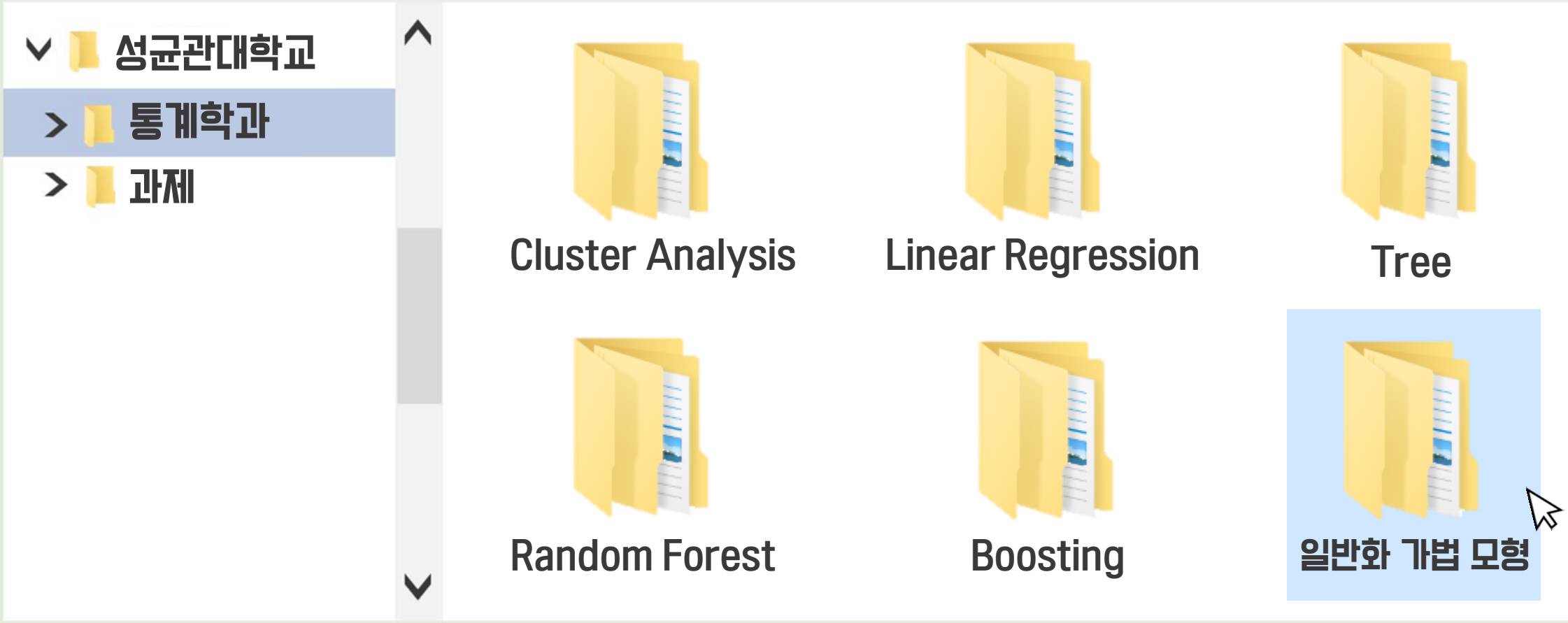
1. MSE: 0.01112946
2. RMSE: 0.1054963
3. 설명력: 0.9343957

예측력과 설명력 모두 뛰어남!

DATA ANALYSIS

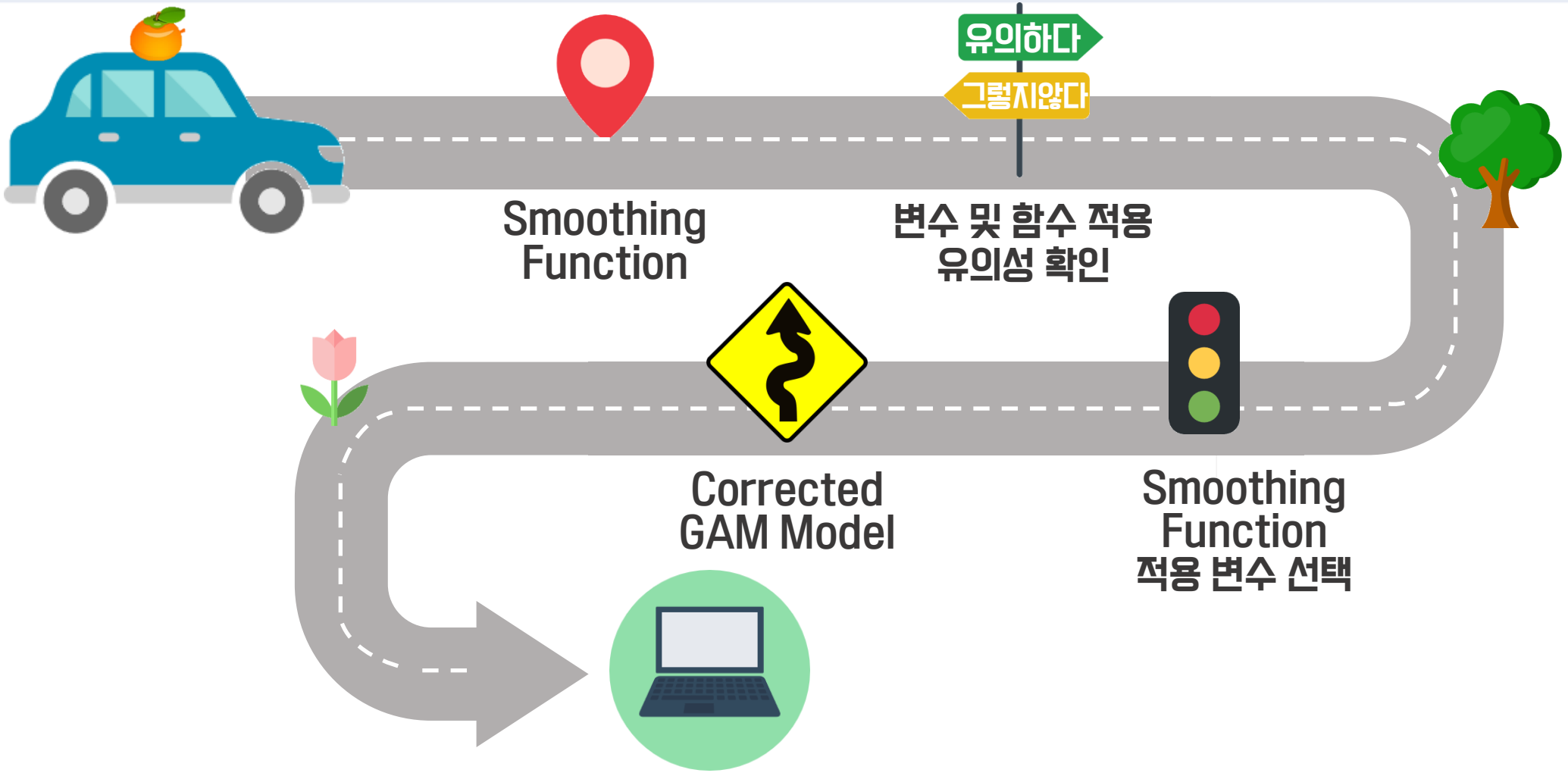
파일 홈 공유 보기 관리

일반화 가법 모형



데이터 소개 **데이터 분석** 분석 활용 한계 및 의의

cafe.naver.com/powersat



cafe.naver.com/powersat

Group 1

```
Call: gam(formula = logprice ~ s(battery) + s(adapter) + memory + company +
    s(release_month) + s(CPU_score) + core_num + s(monitor) +
    anti_glare + wideview + touchscreen + brightness_auto + rotateLCD +
    s(memory_vol) + faster_HDD + HDD_vol + s(SSD_vol) + DVDrecorder +
    multiboost + graphic_inner + graphic_outer + graphic_memory +
    s(outerGPU_Compute) + wired_lan + bluetooth + display_output_num +
    webcam + USB_type + Type_C + multireader + number_key + convex_key +
    antiwater_key + light_key + RGBlight_key + fingerscan + s(weight) +
    world_warranty + s(dpi), data = train1)
Deviance Residuals:
     Min       1Q   Median       3Q      Max
-0.48116 -0.09950 -0.00204  0.09337  0.87725

(Dispersion Parameter for gaussian family taken to be 0.0242)

    Null Deviance: 245.8223 on 1122 degrees of freedom
Residual Deviance: 25.1894 on 1040 degrees of freedom
AIC: -909.4742

Number of Local Scoring Iterations: 14
```

* 연속형 변수에 모두 Smoothing Function 적용

Anova for Parametric Effects

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
s(battery)	1	14.051	14.051	580.1337	< 2.2e-16	***
s(adapter)	1	0.308	0.308	12.7203	0.0003782	***
memory	1	1.659	1.659	68.5122	3.837e-16	***
company	9	51.921	5.769	238.1856	< 2.2e-16	***
s(release_month)	1	3.978	3.978	164.2411	< 2.2e-16	***
s(CPU_score)	1	77.650	77.650	3205.9693	< 2.2e-16	***
core_num	1	7.864	7.864	324.6647	< 2.2e-16	***
s(monitor)	1	2.213	2.213	91.3714	< 2.2e-16	***
anti_glare	1	7.025	7.025	290.0550	< 2.2e-16	***
wideview	1	2.759	2.759	113.9134	< 2.2e-16	***
touchscreen	1	2.952	2.952	121.8875	< 2.2e-16	***
brightness_auto	1	0.680	0.680	28.0791	1.421e-07	***
rotateLCD	1	0.713	0.713	29.4517	7.132e-08	***
s(memory_vol)	1	11.162	11.162	460.8577	< 2.2e-16	***
faster_HDD	1	0.405	0.405	16.7405	4.617e-05	***
HDD_vol	3	0.519	0.173	7.1478	9.406e-05	***
s(SSD_vol)	1	3.094	3.094	127.7611	< 2.2e-16	***
DVDrecorder	1	1.707	1.707	70.4861	< 2.2e-16	***
multiboost	1	0.032	0.032	1.3047	0.2536200	
graphic_inner	1	0.375	0.375	15.5030	8.786e-05	***
graphic_outer	1	0.842	0.842	34.7494	5.063e-09	***
graphic_memory	1	0.440	0.440	18.1638	2.211e-05	***
s(outerGPU_Compute)	1	0.413	0.413	17.0405	3.952e-05	***
wired_lan	1	0.351	0.351	14.4965	0.0001486	***
bluetooth	1	0.015	0.015	0.6395	0.4240676	
display_output_num	3	1.484	0.495	20.4179	7.349e-13	***
webcam	1	0.020	0.020	0.8462	0.3578494	
USB_type	2	0.349	0.174	7.2035	0.0007817	***
Type_C	1	2.099	2.099	86.6826	< 2.2e-16	***
multireader	1	0.038	0.038	1.5585	0.2121688	
number_key	1	0.631	0.631	26.0618	3.930e-07	***
convex_key	1	0.001	0.001	0.0333	0.8553057	
antiwater_key	1	1.435	1.435	59.2414	3.232e-14	***
light_key	1	0.133	0.133	5.4793	0.0194314	*
RGBlight_key	1	0.011	0.011	0.4371	0.5086773	
fingerscan	1	1.149	1.149	47.4514	9.748e-12	***

* 변수 유의성

Anova for Nonparametric Effects

	Npar Df	Npar F	Pr(F)	
(Intercept)				
s(battery)	3	4.4019	0.0043660	**
s(adapter)	3	1.4502	0.2266665	
memory				
company				
s(release_month)	3	6.7245	0.0001706	***
s(CPU_score)	3	12.9587	2.584e-08	***
core_num				
s(monitor)	3	2.1853	0.0881517	.
anti_glare				
wideview				
touchscreen				
brightness_auto				
rotateLCD				
s(memory_vol)	3	3.9278	0.0083985	**
faster_HDD				
HDD_vol				
s(SSD_vol)	3	3.5681	0.0137475	*
DVDrecorder				
multiboost				
graphic_inner				
graphic_outer				
graphic_memory				
s(outerGPU_Compute)	3	2.9658	0.0311562	*
wired_lan				
bluetooth				
display_output_num				
webcam				
USB_type				
Type_C				
multireader				
number_key				
convex_key				
antiwater_key				
light_key				
RGBlight_key				

* 함수 적용 유의성

Group 1 – Corrected GAM

```
Call: gam(formula = logprice ~ s(battery) + adapter + memory + company +
    s(release_month) + s(CPU_score) + core_num + monitor + anti_glare +
    wideview + touchscreen + brightness_auto + rotateLCD + s(memory_vol) +
    faster_HDD + HDD_vol + s(SSD_vol) + DVDrecorder + multiboost +
    graphic_inner + graphic_outer + graphic_memory + s(outerGPU_Compute) +
    wired_lan + display_output_num + USB_type + Type_C + multireader +
    antiwater_key + light_key + fingerscan + s(weight) + s(dpi),
    data = train1)
Deviance Residuals:
      Min        1Q    Median        3Q       Max
-0.462288 -0.100425 -0.001567  0.094104  0.868144

(Dispersion Parameter for gaussian family taken to be 0.0243)

    Null Deviance: 245.8223 on 1122 degrees of freedom
Residual Deviance: 25.5839 on 1052 degrees of freedom
AIC: -916.0225

Number of Local Scoring Iterations: 11
```

* 수정한 GAM 식을 적용하여 분석

Anova for Parametric Effects

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
s(battery)	1	13.733	13.733	564.7067	< 2.2e-16	***
adapter	1	0.155	0.155	6.3567	0.0118410	*
memory	1	1.854	1.854	76.2466	< 2.2e-16	***
company	9	50.794	5.644	232.0694	< 2.2e-16	***
s(release_month)	1	3.889	3.889	159.9258	< 2.2e-16	***
s(CPU_score)	1	75.254	75.254	3094.4306	< 2.2e-16	***
core_num	1	7.276	7.276	299.1923	< 2.2e-16	***
monitor	1	2.369	2.369	97.4095	< 2.2e-16	***
anti_glare	1	7.168	7.168	294.7265	< 2.2e-16	***
wideview	1	2.715	2.715	111.6535	< 2.2e-16	***
touchscreen	1	3.161	3.161	129.9629	< 2.2e-16	***
brightness_auto	1	0.574	0.574	23.6087	1.360e-06	***
rotateLCD	1	0.470	0.470	19.3153	1.220e-05	***
s(memory_vol)	1	11.581	11.581	476.1932	< 2.2e-16	***
faster_HDD	1	0.384	0.384	15.7982	7.528e-05	***
HDD_vol	3	0.451	0.150	6.1885	0.0003616	***
s(SSD_vol)	1	3.292	3.292	135.3474	< 2.2e-16	***
DVDrecorder	1	1.552	1.552	63.8343	3.535e-15	***
multiboost	1	0.018	0.018	0.7342	0.3917361	
graphic_inner	1	0.713	0.713	29.3380	7.533e-08	***
graphic_outer	1	1.152	1.152	47.3549	1.016e-11	***
graphic_memory	1	0.566	0.566	23.2883	1.600e-06	***
s(outerGPU_Compute)	1	0.461	0.461	18.9451	1.477e-05	***
wired_lan	1	0.434	0.434	17.8413	2.609e-05	***
display_output_num	3	1.429	0.476	19.5862	2.325e-12	***
USB_type	2	0.058	0.029	1.1983	0.3021264	
Type_C	1	2.054	2.054	84.4557	< 2.2e-16	***
multireader	1	0.048	0.048	1.9561	0.1622272	
antiwater_key	1	1.457	1.457	59.8981	2.335e-14	***
light_key	1	0.125	0.125	5.1383	0.0236057	*
fingerscan	1	1.332	1.332	54.7520	2.792e-13	***
s(weight)	1	2.035	2.035	83.6839	< 2.2e-16	***
s(dpi)	1	1.852	1.852	76.1495	< 2.2e-16	***
Residuals	1052	25.584	0.024			

* 변수 유의성

Anova for Nonparametric Effects

	Npar Df	Npar F	Pr(F)	
(Intercept)				
s(battery)	3	3.8388	0.0094844	**
adapter				
memory				
company				
s(release_month)	3	6.7349	0.0001679	***
s(CPU_score)	3	12.7313	3.548e-08	***
core_num				
monitor				
anti_glare				
wideview				
touchscreen				
brightness_auto				
rotateLCD				
s(memory_vol)	3	4.3878	0.0044520	**
faster_HDD				
HDD_vol				
s(SSD_vol)	3	3.9333	0.0083297	**
DVDrecorder				
multiboost				
graphic_inner				
graphic_outer				
graphic_memory				
s(outerGPU_Compute)	3	4.3549	0.0046571	**
wired_lan				
display_output_num				
USB_type				
Type_C				
multireader				
antiwater_key				

* 함수 적용 유의성

Group 1

All variables

```
> mean((yhat.gam1-test1[,"logprice"])^2)
[1] 0.02462811
> sqrt(mean((yhat.gam1-test1[,"logprice"])^2))
[1] 0.1569335
> var(yhat.gam1)/var(test1$logprice)
[1] 0.8610968
```

Corrected

```
> mean((yhat.gam11-test1[,"logprice"])^2)
[1] 0.02508191
> sqrt(mean((yhat.gam11-test1[,"logprice"])^2))
[1] 0.1583727
> var(yhat.gam11)/var(test1$logprice)
[1] 0.8724976
```

Check!

- Test error가 다소 증가
- 하지만 설명력도 증가!

Group 2

```
Call: gam(formula = logprice ~ s(battery) + s(adapter) + CPU_company +
    memory + company + s(release_month) + s(CPU_score) + core_num +
    s(monitor) + anti_glare + wideview + touchscreen + brightness_auto +
    rotateLCD + s(memory_vol) + faster_HDD + HDD_vol + refresh +
    s(SSD_vol) + DVDrecorder + multiboost + graphic_inner + graphic_outer +
    graphic_memory + s(outerGPU_Compute) + wired_lan + bluetooth +
    display_output_num + webcam + USB_type + Type_C + multireader +
    number_key + convex_key + antiwater_key + light_key + RGBlight_key +
    fingerscan + OS + s(weight) + world_warranty + s(dpi), data = train2)
Deviance Residuals:
      Min        1Q    Median        3Q       Max
-0.374110 -0.070049 -0.008474  0.061154  0.473124

(Dispersion Parameter for gaussian family taken to be 0.0136)

    Null Deviance: 86.6261 on 683 degrees of freedom
Residual Deviance: 8.0764 on 595.9999 degrees of freedom
AIC: -917.1762

Number of Local Scoring Iterations: 3
```

* 연속형 변수에 모두 Smoothing Function 적용

Anova for Parametric Effects

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
s(battery)	1	23.8834	23.8834	1762.4849	< 2.2e-16	***
s(adapter)	1	1.0599	1.0599	78.2193	< 2.2e-16	***
CPU_company	1	2.3714	2.3714	174.9949	< 2.2e-16	***
memory	1	0.2736	0.2736	20.1893	8.428e-06	***
company	11	6.5624	0.5966	44.0252	< 2.2e-16	***
s(release_month)	1	0.0515	0.0515	3.8036	0.0516099	.
s(CPU_score)	1	5.1874	5.1874	382.8023	< 2.2e-16	***
core_num	2	2.3164	1.1582	85.4689	< 2.2e-16	***
s(monitor)	1	0.7275	0.7275	53.6897	7.671e-13	***
anti_glare	1	0.1908	0.1908	14.0818	0.0001921	***
wideview	1	0.3070	0.3070	22.6584	2.432e-06	***
touchscreen	1	0.0609	0.0609	4.4965	0.0343775	*
brightness_auto	1	0.0045	0.0045	0.3317	0.5648510	
rotateLCD	1	0.0232	0.0232	1.7129	0.1911107	
s(memory_vol)	1	13.2900	13.2900	980.7402	< 2.2e-16	***
faster_HDD	1	0.1911	0.1911	14.0998	0.0001904	***
HDD_vol	3	0.0657	0.0219	1.6160	0.1844894	
refresh	1	0.0758	0.0758	5.5951	0.0183305	*
s(SSD_vol)	1	1.6211	1.6211	119.6303	< 2.2e-16	***
DVDrecorder	1	0.3800	0.3800	28.0399	1.674e-07	***
multiboost	1	0.3846	0.3846	28.3801	1.415e-07	***
graphic_inner	1	0.0511	0.0511	3.7696	0.0526619	.
graphic_outer	1	0.5714	0.5714	42.1678	1.767e-10	***
graphic_memory	1	0.8380	0.8380	61.8392	1.756e-14	***
s(outerGPU_Compute)	1	0.0976	0.0976	7.2020	0.0074846	**
wired_lan	1	0.0885	0.0885	6.5288	0.0108614	*
bluetooth	1	0.0037	0.0037	0.2717	0.6024129	
display_output_num	3	0.3199	0.1066	7.8691	3.720e-05	***
webcam	1	0.0323	0.0323	2.3805	0.1233878	
USB_type	1	0.0050	0.0050	0.3682	0.5442000	
Type_C	1	0.1268	0.1268	9.3607	0.0023165	**
multireader	1	0.0464	0.0464	3.4257	0.0646837	.
number_key	1	0.9040	0.9040	66.7094	1.886e-15	***
convex_key	1	0.0106	0.0106	0.7795	0.3776479	
antiwater_key	1	0.8165	0.8165	60.2527	3.647e-14	***
light_key	1	0.0293	0.0293	2.1618	0.1420024	
RGBlight_key	1	0.1451	0.1451	10.7080	0.0011284	**
fingerscan	1	0.0786	0.0786	5.8017	0.0163127	*

* 변수 유의성

Anova for Nonparametric Effects

	Npar Df	Npar F	Pr(F)	
(Intercept)				
s(battery)	3	12.8535	3.796e-08	***
s(adapter)	3	2.2646	0.07992	.
CPU_company				
memory				
company				
s(release_month)	3	2.8592	0.03634	*
s(CPU_score)	3	2.2030	0.08663	.
core_num				
s(monitor)	2	0.8749	0.41743	
anti_glare				
wideview				
touchscreen				
brightness_auto				
rotateLCD				
s(memory_vol)	3	15.3378	1.265e-09	***
faster_HDD				
HDD_vol				
refresh				
s(SSD_vol)	3	1.2552	0.28893	
DVDrecorder				
multiboost				
graphic_inner				
graphic_outer				
graphic_memory				
s(outerGPU_Compute)	3	18.0621	3.134e-11	***
wired_lan				
bluetooth				
display_output_num				
webcam				
USB_type				
Type_C				
multireader				
number_key				
convex_key				
antiwater_key				
light_key				

* 함수 적용 유의성

Group 2 – Corrected GAM

```
Call: gam(formula = logprice ~ s(battery) + s(adapter) + CPU_company +
    memory + company + CPU_score + core_num + monitor + anti_glare +
    wideview + touchscreen + s(memory_vol) + faster_HDD + refresh +
    SSD_vol + DVDrecorder + multiboost + graphic_outer + graphic_memory +
    s(outerGPU_Compute) + wired_lan + display_output_num + Type_C +
    number_key + antiwater_key + RGBlight_key + fingerscan +
    OS + s(dpi), data = train2)
Deviance Residuals:
     Min       1Q   Median       3Q      Max
-0.34199 -0.07569 -0.01037  0.06810  0.44815

(Dispersion Parameter for gaussian family taken to be 0.0154)

    Null Deviance: 86.6261 on 683 degrees of freedom
Residual Deviance: 9.6263 on 625 degrees of freedom
AIC: -855.0973

Number of Local Scoring Iterations: 4
```

* 수정한 GAM 식을 적용하여 분석

Anova for Parametric Effects

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
s(battery)	1	24.1845	24.1845	1570.2115	< 2.2e-16	***
s(adapter)	1	0.8352	0.8352	54.2292	5.655e-13	***
CPU_company	1	2.5814	2.5814	167.5981	< 2.2e-16	***
memory	1	0.3235	0.3235	21.0068	5.528e-06	***
company	11	6.7087	0.6099	39.5975	< 2.2e-16	***
CPU_score	1	4.9721	4.9721	322.8225	< 2.2e-16	***
core_num	2	2.4812	1.2406	80.5463	< 2.2e-16	***
monitor	1	0.9877	0.9877	64.1291	5.708e-15	***
anti_glare	1	0.1676	0.1676	10.8824	0.0010260	**
wideview	1	0.3119	0.3119	20.2514	8.099e-06	***
touchscreen	1	0.0444	0.0444	2.8802	0.0901751	.
s(memory_vol)	1	13.5014	13.5014	876.5938	< 2.2e-16	***
faster_HDD	1	0.2196	0.2196	14.2607	0.0001743	***
refresh	1	0.0734	0.0734	4.7655	0.0294065	*
SSD_vol	1	1.1341	1.1341	73.6298	< 2.2e-16	***
DVDrecorder	1	0.3239	0.3239	21.0279	5.470e-06	***
multiboost	1	0.3564	0.3564	23.1397	1.890e-06	***
graphic_outer	1	0.4989	0.4989	32.3906	1.940e-08	***
graphic_memory	1	0.8747	0.8747	56.7922	1.708e-13	***
s(outerGPU_Compute)	1	0.0843	0.0843	5.4764	0.0195889	*
wired_lan	1	0.0452	0.0452	2.9366	0.0870886	.
display_output_num	3	0.3631	0.1210	7.8584	3.740e-05	***
Type_C	1	0.1464	0.1464	9.5022	0.0021426	**
number_key	1	0.8393	0.8393	54.4927	4.999e-13	***
antiwater_key	1	0.7157	0.7157	46.4657	2.200e-11	***
RGBlight_key	1	0.2111	0.2111	13.7077	0.0002325	***
fingerscan	1	0.0632	0.0632	4.1043	0.0431996	*
OS	2	2.0873	1.0436	67.7589	< 2.2e-16	***
s(dpi)	1	0.4560	0.4560	29.6049	7.613e-08	***
Residuals	625	9.6263	0.0154			

* 변수 유의성

Anova for Nonparametric Effects

	Npar Df	Npar F	Pr(F)	
(Intercept)				
s(battery)	3	15.7106	7.335e-10	***
s(adapter)	3	0.9292	0.42619	
CPU_company				
memory				
company				
CPU_score				
core_num				
monitor				
anti_glare				
wideview				
touchscreen				
s(memory_vol)	3	16.4737	2.588e-10	***
faster_HDD				
refresh				
SSD_vol				
DVDrecorder				
multiboost				
graphic_outer				
graphic_memory				
s(outerGPU_Compute)	3	16.9529	1.346e-10	***
wired_lan				
display_output_num				
Type_C				
number_key				
antiwater_key				
RGBlight_key				
fingerscan				
OS				
s(dpi)	3	2.1254	0.09581	.

* 함수 적용 유의성

Group 2

All variables

```
> mean((yhat.gam2-test2[,"logprice"])^2)
[1] 0.01623778
> sqrt(mean((yhat.gam2-test2[,"logprice"])^2))
[1] 0.1274276
> var(yhat.gam2)/var(test2$logprice)
[1] 0.9646207
```

Corrected

```
> mean((yhat.gam21-test2[,"logprice"])^
[1] 0.01852431
> sqrt(mean((yhat.gam21-test2[,"logprice"])^2))
[1] 0.1361041
> var(yhat.gam21)/var(test2$logprice)
[1] 0.9715313
```

Check!

- Test error가 다소 증가
- 하지만 설명력도 증가!
- Group2의 경우에는 boosting보다 더 **좋은 설명력**을 보임

분석 요약

		선형회귀	의사결정나무	랜덤 포레스트	Boosting	GAM	GAM corrected
GROUP1	MSE	0.03077082	0.05853857	0.01634308	0.011375	0.02462811	0.02508191
	RMSE	0.1754161	0.2419475	0.1278401	0.1066536	0.1569335	0.1583727
	설명력	0.879689	0.706084	0.7978367	0.9503649	0.8610968	0.8724976
GROUP2	MSE	0.01964755	0.043546419	0.01607958	0.01112946	0.01623778	0.01852431
	RMSE	0.1401697	0.2087204	0.1268053	0.1054963	0.1274276	0.1361041
	설명력	0.909529	0.8152141	0.7555339	0.9343957	0.9646207	0.9715313

04

분석 활용 예시

분석 활용

가성비 좋은 노트북 추천!

가성비가 좋은 노트북 중
개인의 선호에 맞춘 기능으로
필터링하여 노트북 추천!

cafe.naver.com/powersat

step 01

가장 예측력이 좋았던 Boosting을 각 Group에 적합

Train과 Test로 나눴던 데이터를 다시 하나로 통합

```
set.seed(1234)
boost.cv<-gbm(formula = logprice ~ .,distribution = "gaussian",data = newgroup1,
              n.trees = 9000,cv.folds = 5,interaction.depth = 3,
              shrinkage=0.04,n.cores = 2)
best.tree<-gbm.perf(boost.cv);best.tree  # 5425

boost1<-gbm(logprice~.,data=newgroup1,distribution="gaussian",n.trees=best.tree,
            shrinkage=0.04,interaction.depth=3)


set.seed(1234)
boost.cv<-gbm(formula = logprice ~ .,distribution = "gaussian",data = newgroup2,
              n.trees = 9000,cv.folds = 5,interaction.depth = 2,
              shrinkage=0.03,n.cores = 2)
best.tree<-gbm.perf(boost.cv);best.tree # 8949

boost2<- gbm(logprice~.,data=newgroup2,distribution="gaussian",n.trees=best.tree,
            shrinkage=0.03,interaction.depth=2)
```

다른 parameter는 그대로 두고 나무 개수만 다시 조정!

step 02

잔차를 새로운 변수로 지정 : 가성비

예측값 > 실제값 이면 가성비가 좋다! (gsb > 0)

```
raw.group1$gsb<-yhat.final1-raw.group1$price_avg
good1<-filter(raw.group1,gsb>0);good1$cluster<-1

raw.group2$gsb<-yhat.final2-raw.group2$price_avg
good2<-filter(raw.group2,gsb>0);good2$cluster<-2
```

< 가성비 좋은 회사 Top 3 (Group 1) >

APPLE	삼성전자	ACER
57%	54%	53%

< 가성비 좋은 회사 Top 3 (Group 2) >

주연테크	MSI	HP
70%	59%	56%

Code : sort(round(table(good2$company)/table(raw.group2$company),2),decreasing = T)

cafe.naver.com/powersat

step 03

good1과 good2에서 원하는 기능으로 필터링! 예시

```
arrange(filter(good1,weight %in% unique(sort(weight))[1:3]),desc(gsb))$name
```

Group1 (저사양 노트북)에서 무게가 가벼운 노트북을 가성비가 높은 순서로 정렬!

```
> arrange(filter(good1,weight %in% unique(sort(weight))[1:3]),desc(gsb))$name
 [1] "ASUS 트랜스포머3 프로 T303UA-GN042T "          "삼성전자 노트북9 Always NT900X3N-K517S "
 [3] "삼성전자 노트북9 Always NT900X3N-K38D "        "삼성전자 노트북9 Always NT900X3Y-KD3S WIN10 "
 [5] "삼성전자 노트북9 Always NT900X3Y-KD5S WIN10 "  "삼성전자 노트북9 Always NT900X3Y-AD5W WIN10 "
 [7] "삼성전자 노트북9 Always NT900X3N-K79W "        "삼성전자 노트북9 Always NT900X3N-K39S "
 [9] "삼성전자 노트북9 Always NT900X3Y-AD3S WIN10 "  "삼성전자 노트북9 Always NT900X3Y-KD5S WIN8.1 "
[11] "삼성전자 노트북9 Always NT900X3Y-AD5S WIN7 "   "삼성전자 노트북9 Always NT900X3Y-AD2S WIN10 "
[13] "삼성전자 노트북9 Always NT900X3N-K58O "        "삼성전자 노트북9 Always NT900X3Y-AD5W "
[15] "삼성전자 노트북9 Always NT900X3N-K79WS "       "삼성전자 노트북9 Always NT900X3N-K78S "
[17] "ASUS 트랜스포머3 프로 T303UA-GN051R "          "ASUS 트랜스포머3 프로 T303UA-GN045R "
[19] "삼성전자 노트북9 Always NT900X3Y-KD3S "        "ASUS 트랜스포머3 프로 T303UA-GN037R "
[21] "삼성전자 노트북9 Always NT900X3Y-KD5W WIN10 "  "삼성전자 노트북9 Always NT900X3Y-AD3S "
[23] "삼성전자 노트북9 Always NT900X3N-K79S "        "삼성전자 노트북9 Always NT900X3Y-AD2S "
[25] "레노버  YOGA Book LTE W "                       "삼성전자 노트북9 Always NT900X3N-K28S "
[27] "삼성전자 노트북9 Always NT900X3N-K58S "        "삼성전자 노트북9 Always NT900X3Y-AD5S WIN10 "
[29] "레노버  YOGA Book W "
```

cafe.naver.com/powersat

step 03

good1과 good2에서 원하는 기능으로 필터링! 예시

```
arrange(filter(good2,outerGPU_Compute %in% unique(sort(outerGPU_Compute,decreasing=T))[1]),
        desc(gsb))$name
```

Group2 (고사양 노트북)에서 GPU 성능이 가장 좋은 노트북을 가성비가 높은 순서로 정렬!

```
> arrange(filter(good2,outerGPU_Compute %in% unique(sort(outerGPU_Compute,decreasing=T))[1]),desc(gsb
))$name
 [1] "ASUS ROG GL502VS-FY326 "                        "HP 오멘 17-W207TX "
 [3] "GIGABYTE 판타소스 P35X v6 Dual 256 UHD Lite "   "ASUS ROG GL502VS-GZ384 "
 [5] "ASUS ROG GL502VS-Super Edition "                "GIGABYTE 판타소스 P37X V6 Dual 256 UHD Lite "
 [7] "GIGABYTE 판타소스 P35X v6 Dual 256 UHD WIN10 "  "ASUS ROG GL502VS-GZ215T "
 [9] "ASUS ROG GL502VS-GZ384T "                       "GIGABYTE 판타소스 P35X v6 Dual Win10 "
[11] "MSI  GE73VR 7RF 레이더 "                        "MSI  GE63VR 7RF 레이더 "
[13] "ASUS ROG GL502VS-FY007T "                       "GIGABYTE 판타소스 P56XT UHD Dual WIN10 "
[15] "GIGABYTE 판타소스 P57X v7 Dual Lite "           "HP 오멘 17-an019TX "
[17] "GIGABYTE 판타소스 P57X v7 Dual Win10 "          "한성컴퓨터  E57 BossMonster Lv.82 MUXED "
[19] "GIGABYTE 판타소스 P57X v6 Dual Lite "           "MSI  GT62VR 7RE Dominator Pro "
[21] "AORUS  X5 v6 "                                  "HP 오멘 17-W109TX "
[23] "ASUS ROG GL502VS-GZ429T "                       "MSI  GT72VR 7RE Dominator Pro "
```

분석 활용

NEW

새롭게 출시될 노트북의 적정 가격 제시

새롭게 출시된 혹은 출시 될 모델의
성능을 바탕으로 모델에 적합하여
적정 가격 제시!

cafe.naver.com/powersat

step 01

대푯값을 통한 변수 종류 나누기

사용자 맞춤형 노트북 : myown이라는 새로운 observation을 만들 예정!

PART A

Group 1 ≠ Group 2

* 두 그룹 간의 차이가 존재 한다 (연속형 변수)

PART B

Group 1 & Group 2 = 0

* (범주형 변수) 두 그룹 간의 차이가 존재하지 않고, 0으로 동일하다

PART C

Group 1 & Group 2 = 1

* (범주형 변수) 두 그룹 간의 차이가 존재하지 않고, 1로 동일하다

상위 값 사용 + 선택 시 변수의 값을 1로 지정 + Default로 1 사용
(거의 모든 노트북에 존재하는 기능 간주)

cafe.naver.com/powersat

step 01

파트 별 변수 처리 (R 적용)

사용자 맞춤형 노트북 : myown이라는 새로운 observation을 만들 예정!

PART A

PART B

```
> selection
$high_end
 [1] "battery"          "adapter"          "memory"
 [4] "release_month"    "CPU_score"        "core_num"
 [7] "monitor"          "anti_glare"       "memory_vol"
[10] "SSD_vol"          "graphic_inner"    "graphic_outer"
[13] "graphic_memory"   "outerGPU_Compute" "display_output_num"
[16] "number_key"       "light_key"        "weight"
[19] "dpi"

$add
 [1] "touchscreen"     "brightness_auto" "rotateLCD"      "refresh"
 [5] "faster_HDD"      "HDD_vol"         "DVDrecorder"    "multiboost"
 [9] "USB_type"        "antiwater_key"   "RGBlight_key"   "fingerscan"

$company
 [1] "ACER"       "APPLE"      "ASUS"       "DELL"       "GIGABYTE"
 [6] "HP"         "LG전자"     "MSI"        "other"      "레노버"
[11] "삼성전자"   "주연테크"   "한성컴퓨터"

$os
[1] "Linux"   "none"   "Windows"
```

제조회사

운영 체제

cafe.naver.com/powersat

step 01

대푯값을 통한 변수 종류 나누기

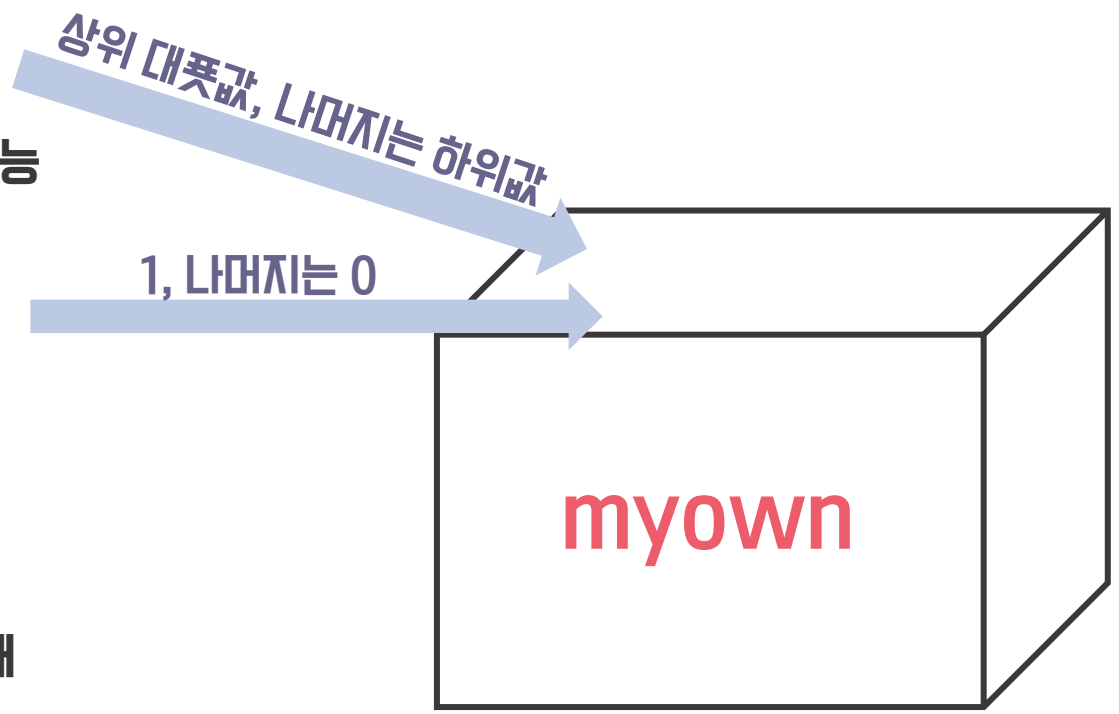
사용자 맞춤형 노트북 : myown이라는 새로운 observation을 만들 예정!

PART A
- 무게
- CPU 성능

PART B
- USB 종류 (USB 3.1)

제조회사
- 삼성전자

운영 체제
- 윈도우 탑재



cafe.naver.com/powersat

step 01

파트 별 변수 처리 (R 적용)

사용자 맞춤형 노트북 : myown이라는 새로운 observation을 만들 예정!

PART A
- 무게
- CPU 성능

PART B
USB 종류 (USB 3.1)

제조회사
삼성전자

운영 체제
윈도우 탑재

myown

```
> pricerange(myown)
Type high-end function: CPU_score weight
Type additional function: USB_type
Which company? 삼성전자
Do you need OS installed?(exlcude Mac) Windows
  CPU_score weight  battery  adapter memory release_month core_num  monitor anti_glare memory_
vol
1  6941.422    1.2 42.29918 46.25112      0      11.03361        2 14.38715          0   6.816
327
   SSD_vol graphic_inner graphic_outer graphic_memory outerGPU_Compute display_output_num numb
er_key
1 213.6999             1             0       0.15006         68.62185                   0
     0
  light_key      dpi USB_type touchscreen brightness_auto rotateLCD refresh faster_HDD HDD_vol
1         0 154.2041  USB 3.1           0              0         0       0          0       0
  DVDrecorder multiboost antiwater_key RGBlight_key fingerscan CPU_company wideview wired_lan
bluetooth
1           0          0             0            0         0        인텔        1         1
        1
  webcam Type_C multireader convex_key world_warranty      OS  company
1      1      1           1          1              0 Windows 삼성전자
```

step 02

Random Forest로 가격 예측 - Bootstrap

Sampling을 여러 번 반복 → 가격을 여러 번 예측 → 예측 가격의 분포가 그려짐

왜 예측력이 좋은 boosting을 두고 Random Forest를 사용했을까?

- 과적합 방지를 위한 parameter tuning을 반복해야 함
- 시간의 압박..

step 03

여러 개의 95% 신뢰구간 중 최소구간으로 가격대 설정

Bonus! 고어의 유사도를 이용하여 각 노트북과 유사도를 계산, 가장 가까운 노트북 추출!

step 02 & step 03 Random Forest로 가격대 예측 (R 활용)

<<예측 가격대>>

```
$range

1328558 1394318
```

<<myown과 가까운 데이터>>

```
$similar_products
  [1] "ASUS 젠북3 UX390UA-GS032T "
  [3] "HP 엔비 13-ad051TU "
  [5] "LG전자 PC그램 13Z950-GA3NK "
  [7] "LG전자 PC그램 13Z950-GP5BML "
  [9] "LG전자 PC그램 13Z950-JOZ5 Pro "
 [11] "LG전자 PC그램 14Z950-P72BK "
 [13] "LG전자 PC그램 14Z960-GA5SE "
 [15] "LG전자 PC그램 14Z960-GP70ML "
 [17] "LG전자 PC그램 14Z960-GR5DL "
 [19] "LG전자 PC그램 14Z960-LR10K "
 [21] "LG전자 PC그램 14Z960-MF5HL "
```

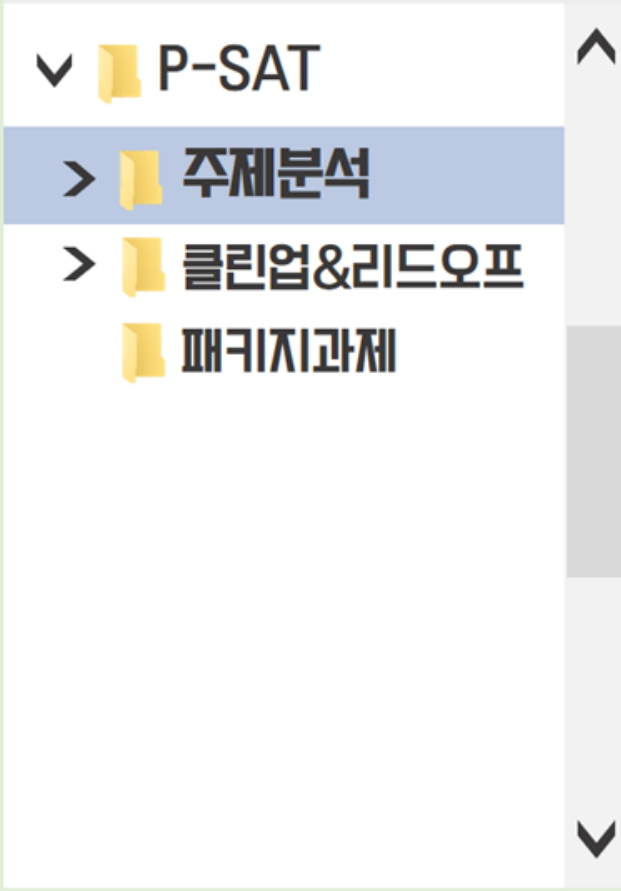
05

한계 및 의의

Wrap Up

파일 홈 공유 보기 관리

한계



* 변하는 시중가를 따지지 못함
* 디자인 고려 하지 않음
* Group1 브랜드 애플의 영향력이 과도함
* 차원의 저주 (변수 개수에 비해서 데이터가 적음)
* 체감 가능한 기능적 결함을 반영하지 못함 (ex. 발열이 심한 LG)
* 모델 자체의 오차는 고려할 수 없음

Wrap Up

파일 홈 공유 보기 관리

의의

- P-SAT
  - 주제분석
  - 클린업&리드오프
  - 패키지과제

* 최소한의 정보로 원하는 노트북을 탐색할 수 있는 모델을 구축
* 추상적인 가성비의 개념을 구체적으로 정의하여 분석


Thank You