면접 준비

데이터 사이언티스트

출처

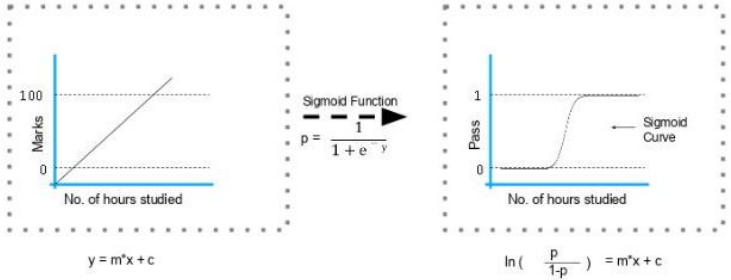
- https://www.simplilearn.com/tutorials/data-science-tutorial/data-science-interview-questions?source=sl_frs_nav_playlist_video_clicked#basic_data_science_interview_questions

1. Supervised vs Unsupervised

Supervised Learning	Unsupervised Learning
- 라벨링이 되어 있는 데이터를 사용 - decision tree - logistic regression - support vector machine	- 라벨링이 되어 있지 않은 데이터를 사용 - k-means clustering - hierarchical clustering - apriori algorithm

2. Logistic regression

- 독립 변수의 선형 조합으로 종속 변수를 예측
  - 종속 변수 : dependent variable, label
  - 독립 변수 : independent variable, feature
- Logistic function ~ Sigmoid function
- Classification problem or 확률 예측
- Linear Regression의 목표는 범위가 정해지지 않은 종속 변수와 독립 변수 사이의 선형 관계를 측정하는 것이지만 Logistic Regression은 Linear Regression을 이용하여 확률을 예측



2-1. Linear regression vs Logistic regression

Linear Regression	Logistic Regression
- Regression problems - Continuous 데이터를 출력 - 종속 변수를 추정 - 직선 형태	- Classification problems 또는 확률값 예측 문제 - Categorical 데이터를 출력 - 종속 변수의 가능성을 계산 - Sigmoid curve

Q. 나이, 성별, 혈중 콜레스테롤 수치라는 3가지 위험 요인을 바탕으로 심장병으로 인한 사망 확률을 예측하고자 할때 적합한 모델은 무엇인가?

A. Logistic Regression

3. Decision Tree

- 목적 : classification problems
- Algorithm
  1. 분기 전 데이터를 입력으로 사용
  2. 분기 전 데이터의 Entropy 계산, 분기 특징 후보들에 대한 Entropy 계산
  3. 분기 특징 후보들의 Information Gain 계산
  4. 가장 높은 Information Gain 값을 가지는 분기 특징을 선택
  5. 사전에 정의한 멈추는 조건이 될때까지 위 과정을 반복
- Entropy : 데이터가 얼마나 균일하게 분류되었는지 알려주는 척도, 즉 작을수록 잘 분류된 상태
- Information Gain : 분기 이전의 Entropy에서 분기 이후의 Entropy를 뺀 수치, 즉 높을수록 잘 분기했다고 판단.
- 단점 : Overfitting 문제 >> **pre-pruning, post-pruning (가지치기)**, Random Forest

3-1. Entropy 계산과정

- Target Value : [0, 0, 0, 1, 1, 1, 1, 1]
- 이때의 Entropy를 계산
- -(5/8 log(5/8) + 3/8 log(3/8))

3-1. Entropy and Information Gain

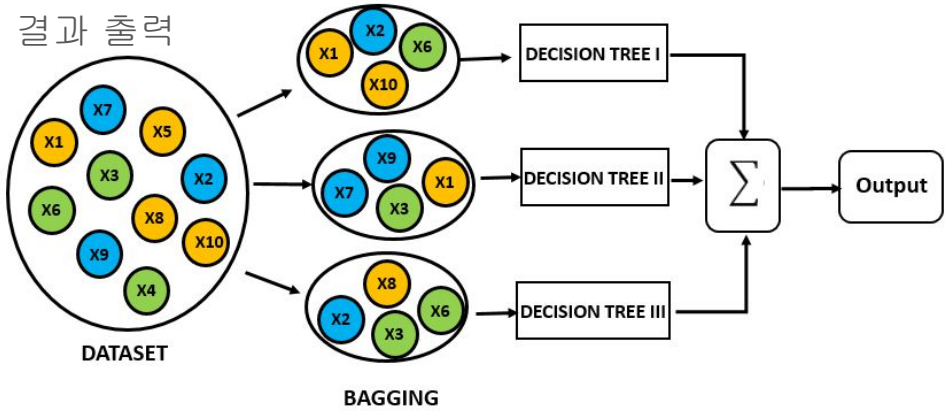
- 공식
-

3-2. Pruning

- 가지치기 역할
- pre-pruning
- post-pruning

4. Random Forest

- 앙상블 머신러닝 모델
- Decision Tree로 생성된 Overfitting Tree에서 일반적인 결과 출력
- Algorithm
  1. 학습 데이터에서 n개 데이터 표본 선택
  2. k개 feature 중 $\sqrt{k}$ 개를 선택
  3. Decision Tree 생성
  4. 1~3 번의 과정을 m 번 반복
  5. 테스트 데이터에서 m개의 결과 중 다수결 결과 출력



https://eunsukimme.github.io/ml/2019/11/26/Random-Forest/

앙상블

- Bagging
- Boosting

5. Overfitting

- 훈련 데이터에 과하게 맞추어져 훈련 데이터의 성능은 좋지만, 테스트 데이터에서는 성능이 저조
- Overfitting 방지
  1. 더 많은 데이터 확보
  2. 모델 복잡도 줄이기
  3. cross validation 방법 사용 : k-fold cross validation
  4. 정규화 사용 (LASSO, Ridge)
  5. 앙상블 학습 방법 사용
- Underfitting 방지
  1. 새로운 특성 추가
  2. 모델 복잡도 증가
  3. 정규화 계수 줄이기

6. Univariate, Bivariate and Multivariate analysis

- Univariate : 일변량 데이터
  - 1개의 feature
  - 평균, 중위수, 최빈값(mode), 산포도, 범위, 최대, 최소 등의 통계 분석 진행

- Bivariate : 이변량 데이터
  - 2개의 다른 feature
  - 원인과 영향을 두 변수 사이의 관계 비교를 통해 분석

- Multivariate : 다변량 데이터
  - 3개 이상의 feature

7. Feature Selection Method

- Filter Method
  - 각 변수들에 대해 통계적인 점수와 순위를 매기고 선택
  - Linear Discrimination Analysis
  - ANOVA
  - Chi-Square
- Wrapper Method
  - 변수의 일부만을 모델링에 사용 후, 평가 작업을 반복하여 변수 선택
  - Forward Selection
  - Backward Selection
  - Recursive Feature Elimination
- Embedded Method
  - 위의 두 방법을 결합하여 어떤 변수가 가장 크게 기여하는 지를 찾아내는 방법
  - LASSO
  - Ridge Regression
  - Elastic Net

7-1. Feature Selection vs Feature Extraction

- https://bioinformaticsandme.tistory.com/188

8. Python Print

- 3의 배수는 “fizz”
- 5의 배수는 “buzz”
- 3과 5의 배수는 “fizzbuzz”

```
for fizzbuzz in range(51):
    if fizzbuzz % 3 == 0 and fizzbuzz % 5 == 0:
        print("fizzbuzz")
        continue
    elif fizzbuzz % 3 == 0:
        print("fizz")
        continue
    elif fizzbuzz % 5 == 0:
        print("buzz")
        continue
    print(fizzbuzz)
```

9. Missing Value

- Missing data 삭제 (확보한 데이터가 충분히 클때)
- 특정 값으로 채우기
- 결측값의 앞 또는 뒷 방향의 값으로 채우기
- mean, mode, medium, trimmed mean
- **Knnimputer** 를 사용하여 결측치 채울 수 있음

- Pandas, scipy 등의 라이브러리를 사용하여 쉽게 채울 수 있음 (fillna)

https://rfriend.tistory.com/262

10. Euclidean Distance in Python

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_i - q_i)^2 + \cdots + (p_n - q_n)^2}.$$

```
import math

plot1 = [1,3]
plot2 = [2,5]

euclidean_distance = math.sqrt((plot1[0]-plot2[0])**2 + (plot1[1]-plot2[1])**2)
```

```
euclidean_distance
```

```
2.23606797749979
```

11. Dimensionality Reduction

- 원래의 차원에서 작은 차원으로 변환
- 장점
  - 데이터 압축하여 저장 공간 감소
  - 계산 시간 감소
- 종류
  - pca
  - auto-encoder
  - Linear Discriminant Analysis

12. Eigenvalues and Eigenvectors

- Eigenvector : 어떤 벡터에 선형변환 결과가 방향은 변하지 않고 크기만 변환되는 벡터를 의미
- Eigenvalue : Eigenvector가 변환되는 크기를 의미

DEFINITION 1. 고윳값, 고유벡터

임의의 $n \times n$ 행렬 A 에 대하여, 0이 아닌 솔루션 벡터 $\vec{x}$ 가 존재한다면 숫자 λ 는 행렬 A 의 고윳값라고 할 수 있다.

$$A\vec{x} = \lambda\vec{x} \tag{2}$$

이 때, 솔루션 벡터 $\vec{x}$ 는 고윳값 λ 에 대응하는 고유벡터이다.

https://angeloyeo.github.io/2019/07/17/eigen_vector.html

12-1. Eigenvalues and Eigenvectors 계산 과정

선형 변환 A에 대해 Eigenvalue와 Eigenvector를 구하면 다음과 같다.

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

$$det(A - \lambda I) = det\left(\begin{bmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{bmatrix}\right) = 0$$

$$\Rightarrow (2-\lambda)^2 - 1$$

$$= (4 - 4\lambda + \lambda^2) - 1$$

$$= \lambda^2 - 4\lambda + 3 = 0$$

그러므로, $\lambda_1 = 1$, $\lambda_2 = 3$이다.

$\lambda_1 = 1$인 경우에 대해,

$$A\vec{x} = \lambda_1 \vec{x}$$

$$\Rightarrow \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 1 \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$2x_1 + x_2 = x_1$$

$$x_1 + 2x_2 = x_2$$

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$\lambda_2 = 3$인 경우의 고유벡터는

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

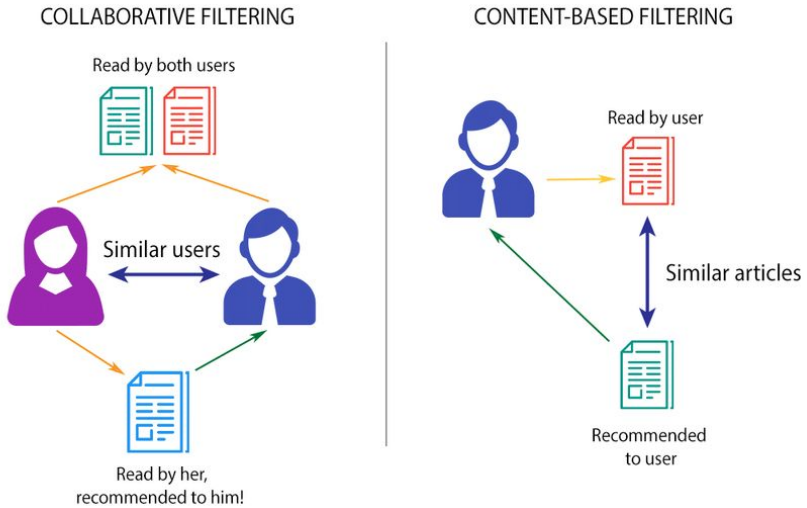
https://angeloyeo.github.io/2019/07/17/eigen_vector.html

13. Model maintain

1. 모니터 : 제대로 작동하는지에 대한 지속적인 모니터링이 필요
2. 평가 : 새로운 알고리즘이 필요한지에 대한 여부를 판단
3. 비교 : 기존 모델과 새 모델을 비교
4. 재작성 : 성능이 더 좋은 모델로 변경

14. Recommender System

- 사용자가 자신의 선호도에 따라 특정 제품을 어떻게 생각할지 예측
- Collaborative Filtering
  - 다른 사용자와의 유사함에 기초
  - 비슷한 사용자가 좋아하는 아이템을 추천
  - ex) 아마존 추천 시스템..
- Content-based Filtering
  - 다른 아이템과의 유사함에 기초
  - 유사한 아이템을 추천
- 그 외
  - Hybrid Recommender System
  - Context-based Recommender System
  - …



15. RMSE and MSE

```
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

# answer : y = 1 * x_0 + 2 * x_1 + 3
X = np.array([[1, 1], [1, 2], [2, 2], [2, 3]])
noise = np.array([0.00201, 0.00032, -0.0001, -0.071902])
y = np.dot(X, np.array([1, 4])) + 3 + noise

reg = LinearRegression().fit(X, y)
reg.score(X, y)

print('coefficients : ',reg.coef_)

print('intercept : ', reg.intercept_)
```

```
coefficients :  [0.99958  3.963254]
intercept :  3.056704
```

```
y_hat = reg.predict(np.array([[3, 5], [4, 5], [6, 7]]))
y_true = np.dot(np.array([[3, 5], [4, 5], [6, 7]]), np.array([1, 4])) + 3
```

```
print('y_hat : ', y_hat)
print('y_true : ', y_true)
```

```
y_hat :  [25.871714 26.871294 36.796962]
y_true :  [26 27 37]
```

```
print('rmse : ', mean_squared_error(y_true, y_hat, squared=False))
print('mse  : ', mean_squared_error(y_true, y_hat))
```

```
rmse :  0.1573181083834126
mse  :  0.024748987225335153
```

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \qquad RMSE = \sqrt{\sum_{i=1}^{n}\frac{(\hat{y}_i - y_i)^2}{n}}$$

15-1. Regression Metrics

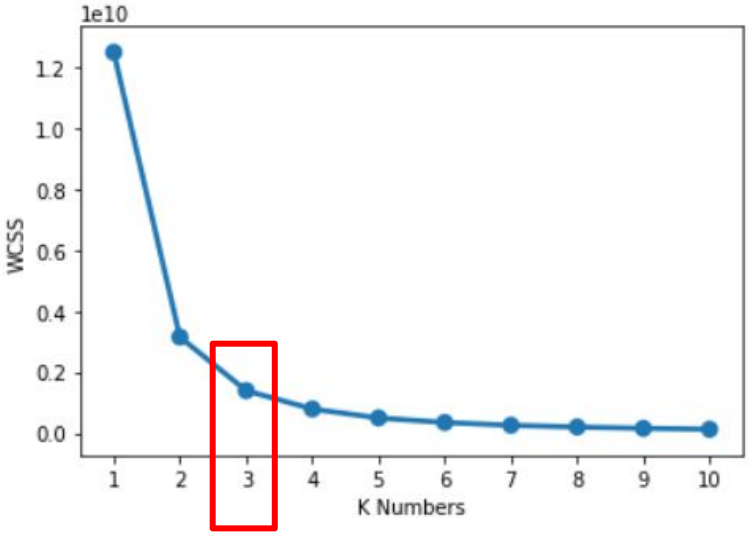
- MSE
- RMSE
- MAE
- R-Squared
- 등등.. 장단점

16. Select k for k-means?

- Elbow Method
  - 군집분석에서 군집수를 결정하는 방법
  - 군집내 총 제곱합 (WSS : Within cluster Sum of Squares) 을 계산하여 적절한 군집 수 설정
- WWS (Within Cluster Sum of Squares)

- **Within Cluster Sums of Squares :** $\mathrm{WSS} = \sum_{i=1}^{N_C} \sum_{x \in C_i} d(\mathbf{x}, \bar{\mathbf{x}}_{\mathbf{C_i}})^2$
- **Between Cluster Sums of Squares:** $\mathrm{BSS} = \sum_{i=1}^{N_C} |C_i| \cdot d(\bar{\mathbf{x}}_{\mathbf{C_i}}, \bar{\mathbf{x}})^2$

C_i = Cluster, N_c = # clusters, $\bar{x}_{c_i}$ = Cluster centroid, $\bar{x}$ = Sample Mean



16-1. WSS, BSS, TSS

- WSS
- BSS
- TSS

17. P-value

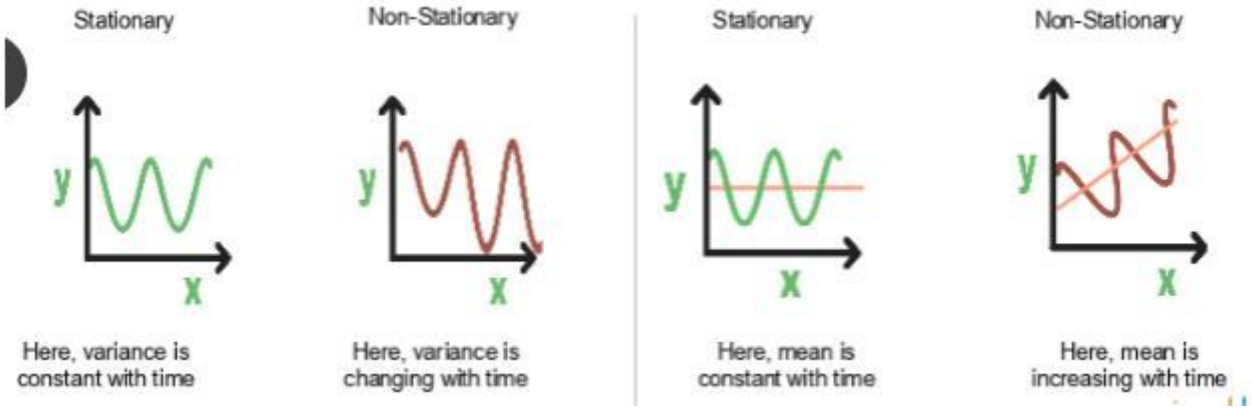
- 검정 통계량에 관한 확률로 크거나 같은 값을 얻을 수 있을 확률
- 귀무가설의 기각 여부를 결정
  - P-value < alpha : 귀무가설을 기각
  - P-value > alpha : 귀무가설을 수락
- 귀무가설 : 새로울게 없다는 가설, 똑같다는 가설
  - ex) 두 확률분포는 차이가 없다.
  - ex) 흡연 여부는 뇌혈관 질환의 발생에 영향을 미치지 않는다.

18. Outlier values treat

- 필요없는 데이터라면 삭제
- 다른 모델을 선택 (linear -> nonlinear)
- 데이터를 정규화
- 특이치에 강한 모델을 사용 (random forest)

19. Time series stationarity

- Stationarity : 시간이 변해도 일정한 분포를 따르는 경우
- 확인 방법
  - 그래프를 그려서 확인
  - 통계량의 변화를 확인
  - 통계적 검정



19-1. Time series stationarity

- 통계적 검정 방법

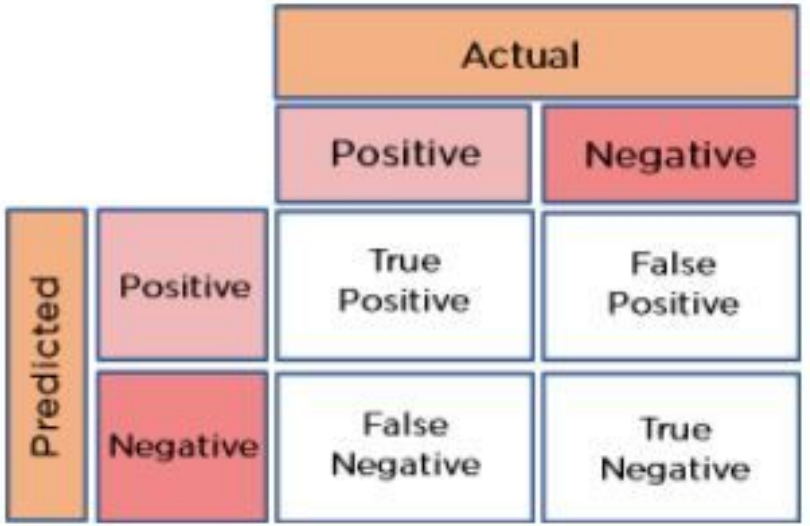
20. Confusion matrix

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

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F1-Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$



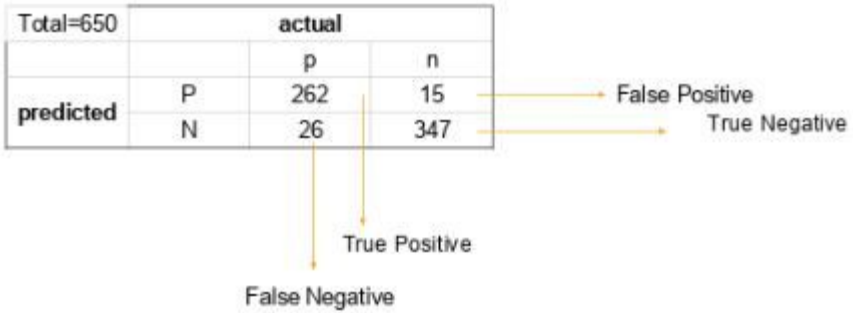
21. Precision and Recall Calculate

- precision = tp / (tp + fp)

  262 / 277 = 0.94

- recall = tp / (tp + fn)

  262 / 288 = 0.9



22. Basic SQL Query

- Order Table
  - OrderId
  - CustomerId
  - OrderNumber
  - TotalAmount
- Customer Table
  - Id
  - FirstName
  - LastName
  - City
  - Country

SQL query (모든 주문 리스트를 고객 정보와 같이 나열)

```
SELECT OrderNumber, TotalAmount, FirstName, LastName, City, Country

FROM Order

JOIN Customer

ON Order.CustomerId = Customer.Id
```

23. Data Imbalance Performance Matrix

- 라벨의 분포가 불균형한 경우
- Accuracy로 본다면 좋은 성능을 나타내지만 실제로 보면 좋지 못한 모델일 수 있음
  - 학습 데이터 : 99% 정상 데이터, 1% 이상 데이터
  - 모두 정상 데이터로 예측 시 Accuracy는 99% 이상일 수 있음
  - Positive를 이상 데이터로 할때
  - Precision은 낮게 나오고 Recall이 높게 나옴
  - fp(정상을 이상치로 예측) 가 높고
  - fn(이상치를 정상으로 예측) 가 낮음
- 위 경우 F1-score 를 이용

24. K-means clustering

- Algorithm
- 장점
- 단점
- 실제 사용 예시

25. Linear Regression

- Algorithm
- 장점
- 단점
- 실제 사용 예시

26. KNN

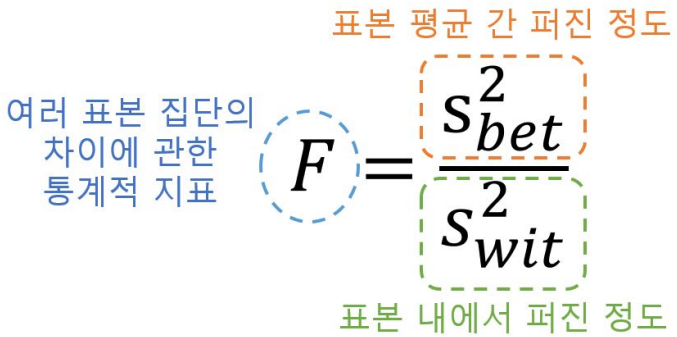
- Algorithm
- 장점
- 단점
- 실제 사용 예시

27. Association Rule

- Algorithm
- 장점
- 단점
- 실제 사용 예시

28. ANOVA

- 3개 이상 다수의 집단을 비교할 때 사용하는 가설검정 방법
- F 분포를 이용
- t-value ~ f-value 같은 의미를 지님



28-1. ANOVA 예제

-

28-1. ANOVA 예제

-