AI기반 데이터 분석 및 AI Agent 개발 과정

『1과목 : 』
AI기반 데이터 분석

2025.09.22-10.02(9일, 62시간)

Prepared by DaeKyeong K

Ph.D.

삼성전자 구미사업부 G.제조팀

HS Global
HARDWARE SOFTWARE

목차

『1-8』
인과 관계 및 예측 분석 이해

회귀분석

...

Syllabus

학습목표

- 회귀분석의 정의와 가정을 이해한다.
- 회귀추정식의 통계적 가설 검증을 이해한다.
- 파이썬 프로그램을 통해 회귀분석을 활용하고 내용을 해설할 수 있다.
- 다중회귀분석에서 변수선택법을 이해하고 활용할 수 있다.

눈높이 체크

- 회귀분석을 들어본 적 있으신가요?
- 단순회귀분석과 다중회귀분석을 이해하시나요?
- 회귀분석을 통계패키지로 구현해 본 적이 있으신가요?

2. 회귀분석

회귀분석의 개요

- 19 세 기 영국의 우생학자 프 랜 시 스 갤 턴 (Francis Galton,1822~1911)은 부모의 키가 큰 자식들의 키가 점점 더 커지지 않고 다시 평균 키로 회귀하는 경향을 발견하였는데, 이를 통계학에서는 '평균으로의 회귀'라고 하며 회귀분석의 '회귀'란 용어는 여기서 파생됨
- 회귀분석의 정의
  - 하나나 그 이상의 독립변수들이 종속변수에 미치는 영향을 추정할 수 있는 통계기법
  - 변수들 사이의 인과관계를 밝히고 모형을 적합하여 관심 있는 변수를 예측하거나 추론하기 위한 분석 방법
  - 독립변수의 개수가 하나 -> 단순 선형 회귀분석으로 분석
  - 독립변수의 개수가 두 개 이상 -> 다중 선형 회귀분석으로 분석할 수 있다

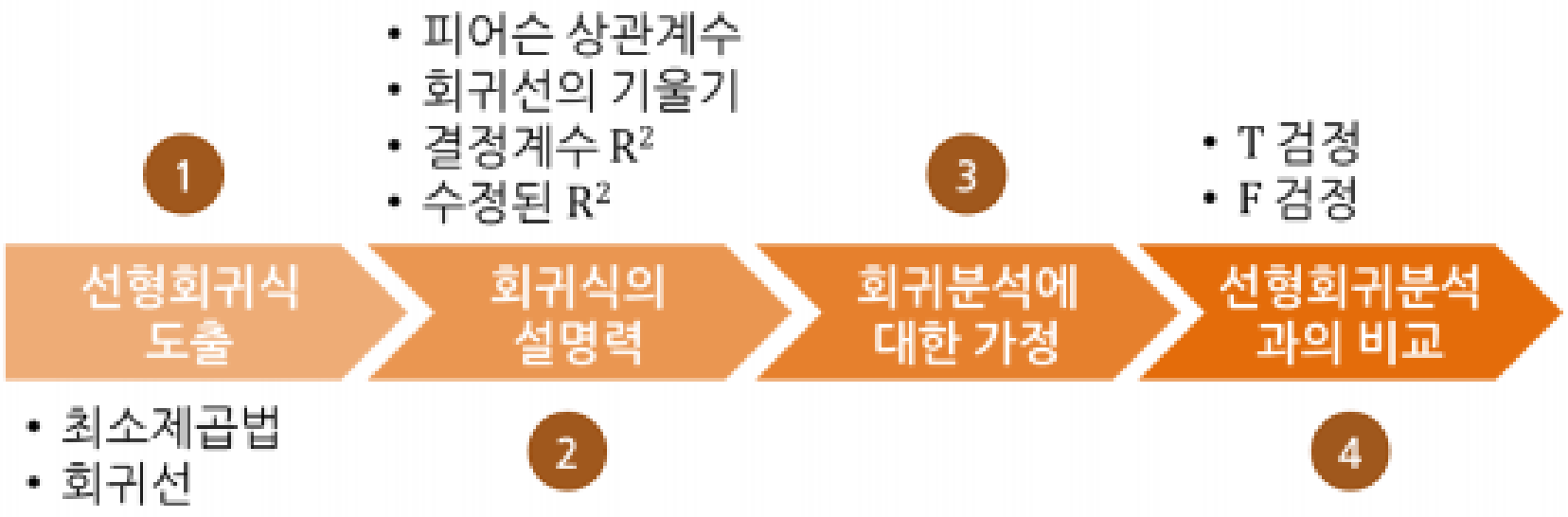
2. 회귀분석

회귀분석의 개요

- 회귀분석의 변수

변수	종류
영향받는 변수(y)	반응변수, 종속변수, 결과변수
영향주는 변수(x)	설명변수, 독립변수, 예측변수

- 회귀분석의 4단계



2. 회귀분석

회귀분석의 개요

- 그래프를 활용한 선형 회귀 분석의 가정 검토
  - 선형성: 선형 회귀모형에서는 설명변수 x와 반응 변수 y가 선형 관계에 있음을 전재되어야 한다.
  - 등분산성
    - 등분산성을 만족: 설명변수 x에 대한 잔차의 산점도를 그렸을 때 설명변수 x 값에 관계없이 잔차들의 변동성이 일정한 형태를 보임
    - 등분산성을 만족 x: 설명변수 x가 커질수록 잔차의 분사가 줄어드는 이분산 형태 / 2차항 설명변수가 필요 / 새로운 설명변수가 필요
  - 정규성: Q-Q Plot을 출력했을 때, 잔차가 대각선 방향의 직선의 형태를 지니고 있으면 잔차는 정규분포를 따른다고 할 수 있다.
- 가정에 대한 검증
  - 단순 선형 회귀분석: 입력 변수와 출력 변수 간의 선형성을 점검하기 위해 산점도를 확인
  - 다중 선형 회귀분석: 선형 회귀분석의 가정인 선형성, 등분산성, 독립성, 정상성 이 모두 만족하는지 확인해야 한다.

2. 회귀분석

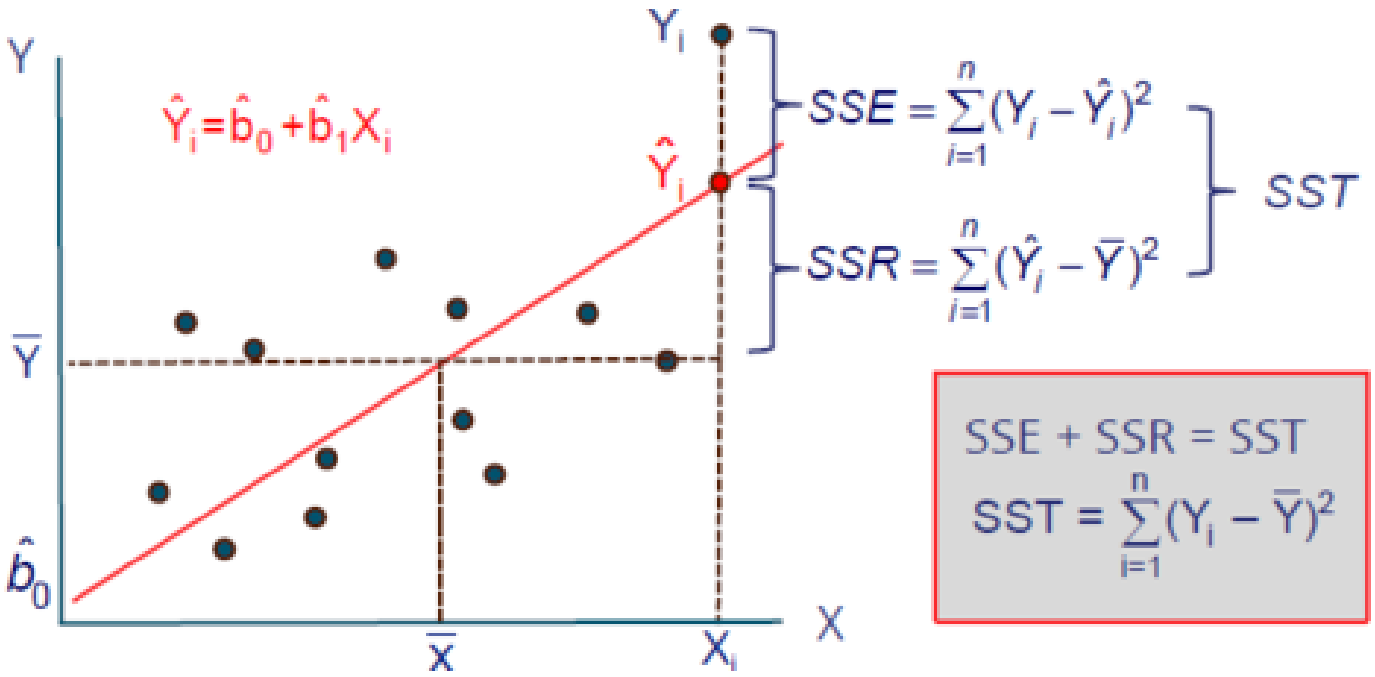
회귀분석의 종류

종류	모형
단순회귀	독립변수가 1개이며 종속변수와의 관계가 직선
다중회귀	독립변수가 k개이며 종속변수와의 관계가 선형(1차함수)
로지스틱회귀	종속변수가 범주형(2진변수)인 경우에 적용되며, 단순 로지스틱 회귀 및 다중, 다항 로지스틱 회귀로 확장할 수 있음
다항회귀	독립변수와 종속변수와의 관계가 1차함수 이상인 관계 (단 k=1이면 2차함수 이상)
곡선회귀	독립변수가 1개이며 종속변수와의 관계가 곡선
비선형회귀	회귀식의 모양이 미지의 모수들의 선형관계로 이뤄져 있지 않는 모형

2. 회귀분석

단순 선형 회귀분석

- 회귀분석의 검정
  - 결정계수(R^2)는 전체 제곱합에서 회귀 제곱 합의 비율(SSR/SST), $0 \le R^2 \le 1$, 1에 가까울수록 설명력 높음
  - 결정계수(R^2)는 전체 데이터의 회귀모형이 설명할 수 있는 설명력을 의미(단순 회귀분석에서 결정계수는 상관계수 r의 제곱을 의미)



2. 회귀분석

단순 선형 회귀분석

- 회귀 직선의 적합도 검토
  - 결정계수(R^2)를 통해 추정된 회귀 식이 얼마나 타당한지 검토(결정계수(R^2)가 1에 가까울수록 회귀모형임)
  - 독립변수가 종속변수 변동의 몇 %를 설명하는지 나타내는지 표
  - 다변량 회귀분석에서는 독립변수의 수가 많아지면 결정계수(R^2)가 높아지므로 독립변수가 유의하든, 유의하지 않든 독립변수의 수가 많아지면 결정계수가 높아지는 단점이 있다.
  - 이러한 결정계수의 단점을 보완하기 위해 수정된 결정계수(adjusted R^2)를 활용한다. 수정된 결정계수는 결정계수보다 작은 값으로 산출된다.

2. 회귀분석

단순 선형 회귀분석

- 오차와 잔차의 차이
  - 오차: 모집단에서 실제값이 회귀선과 비교해 볼 때 나타나는 차이(정확치와 관측치 차이)
  - 잔차: 표본에서 나온 관측값이 회귀선과 비교해볼 때 나타나는 차이
  - 회귀모형에서 오차항은 측정할 수 없으므로 잔차를 오차항의 관찰 값으로 해석하여 오차항에 대한 가정들의 성립 여부를 조사함

2. 회귀분석

다중 선형 회귀분석

- 다중회귀식 $Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \varepsilon$

- 모형의 통계적 유의성 (F 통계량 확인)
  - 유의 수준 5% 하에 F 통계량의 p-값이 0.05보다 작으면 추정된 회귀식은 통계적으로 유의하다고 볼 수 있다.
  - F통계량이 크면 p-value가 0.05보다 작아지고, 이렇게 되면 귀무가설을 기각한다. 즉 모형이 유의하다고 결론

	제곱합	자유도	제곱평균	F-총계량
회차	회차제곱합(SSR)	k	MSR=SSR/k	F=MSR/MSE
오차	오차제곱합(SSE)	n-k+1	MSE=SSE(n-k+1)	
계	전체제곱합(SST)	n-1		

2. 회귀분석

다중 선형 회귀분석

- 회귀계수의 유의성 (t 통계량을 통해 확인)
  - 모든 회귀계수의 유의성이 통계적으로 검증되어야 선택된 변수들의 조합으로 모형을 활용할 수 있다.
- 모형의 설명력
  - 결정계수(R^2)나 수정된 결정계수(R^2)를 확인한다.
- 모형의 적합성
  - 모형이 데이터를 적합하고 있는지 잔차와 종속변수의 산점도를 확인
- 데이터가 전제하는 가정을 만족시키는가?
  - 선형성, 독립성, 등분산성, 빙상관성, 정상성
- 다중공선성
  - 다중회귀분석에서 설명변수들 사이에 선형관계가 존재하면 회귀계수의 정확한 추정이 곤란하다.

2. 회귀분석

다중 선형 회귀분석

- 다중공선성 검사방법

1) 분산 팽창 요인: 4보다 크면 다중공선성이 존재한다고 볼 수 있고, 10보다 크면 심각한 문제가 있는 것으로 해석
2) 상태 지수: 10 이상이면 문제가 있다고 보고, 30보다 크면 심각한 문제가 있다고 해석할 수 있다. 다중 선형 회귀분석에서 다중공선성 문제가 발생하면 문제가 있는 변수를 제거하거나 주성분 회귀, 능형회귀 모형을 적용하여 문제를 해결

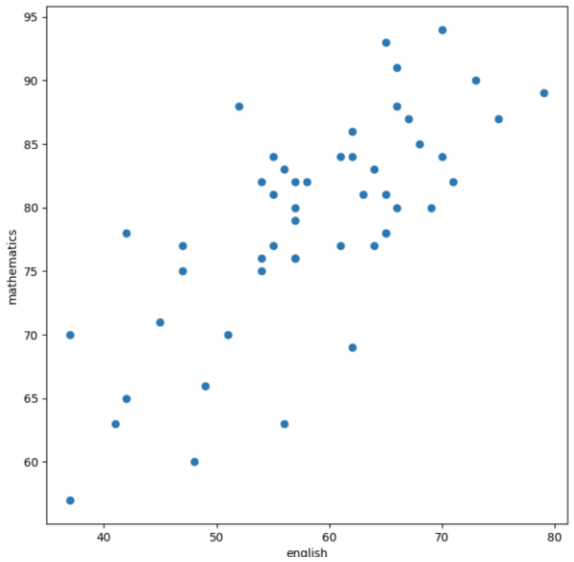
2. 회귀분석

파이썬 실습

● 회귀직선

파일	소스코드
실습환경	py3_10_basic

소스코드	import matplotlib.pyplot as plt
	
	%matplotlib inline
	
	import numpy as np
	import pandas as pd
	
	pd.set_option("display.max_columns", 3)
	
	df = pd.read_csv('datasets/student_scores.csv',
	index_col='student number')
	
	english_scores = np.array(df['english'])
	math_scores = np.array(df['mathematics'])
	
	fig = plt.figure(figsize=(8, 8))
	ax = fig.add_subplot(111)
	# 산점도
	ax.scatter(english_scores, math_scores)
	ax.set_xlabel('english')
	ax.set_ylabel('mathematics')
	
	plt.show()



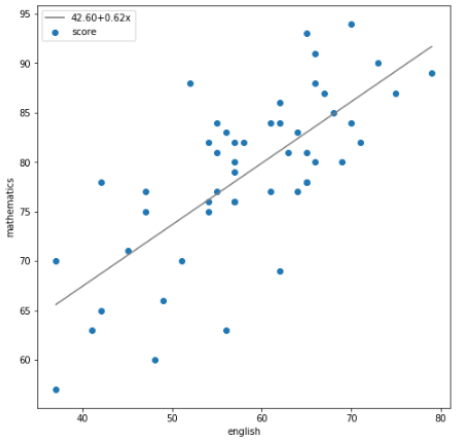
2. 회귀분석

파이썬 실습

● 회귀직선

파일	소스코드
실습환경	py3_10_basic

소스코드	# 계수β_0와β_1를 구한다
	poly_fit = np.polyfit(english_scores, math_scores, 1)
	# β_0+β_1 x를 반환하는 함수를 작성
	poly_1d = np.poly1d(poly_fit)
	# 직선을 그리기 위해 x좌표를 생성
	xs = np.linspace(english_scores.min(), english_scores.max())
	# xs에 대응하는 y좌표를 구한다
	ys = poly_1d(xs)
	
	fig = plt.figure(figsize=(8, 8))
	ax = fig.add_subplot(111)
	ax.set_xlabel('english')
	ax.set_ylabel('mathematics')
	ax.scatter(english_scores, math_scores, label='score')
	ax.plot(xs, ys, color='gray',
	label=f'{poly_fit[1]:.2f}+{poly_fit[0]:.2f}x')
	# 범례의 표시
	ax.legend(loc='upper left')
	
	plt.show()



2. 회귀분석

파이썬 실습

● 회귀직선

파일	소스코드
실습환경	py3_10_basic pip install statsmodels

소스코드	import numpy as np
	
	# 데이터프레임 다루기
	import pandas as pd
	# 기초 통계분석 패키지
	from scipy import stats
	# 그래프 그리기
	import matplotlib.pyplot as plt
	#회귀분석 가능 패키지
	import statsmodels.api as sm
	import statsmodels.formula.api as smf
	
	# 직업만족도
	# survey 변수들: name gender income English jobSatisfaction stress
	df = pd.read_csv("./datasets/survey.csv")
	
	# 가상의 분석자료로 회귀분석 결과 구하기
	model = smf.ols(formula = 'jobSatisfaction ~ English', data = df)
	result = model.fit()
	print(result.summary())

2. 회귀분석

파이썬 실습

- 회귀직선

파일	소스코드

결과값1

OLS Regression Results

==========

Dep. Variable: jobSatisfaction **R-squared:**

Model: OLS Adj. R-squared: 0.054

Method: Least Squares F-statistic: 2.266

Date: Sun, 23 Apr 2023 **Prob (F-statistic):**

Time: 05:42:43 Log-Likelihood: -36.245

No. Observations: 23 AIC: 76.49

Df Residuals: 21 BIC: 78.76

Df Model: 1

Covariance Type: nonrobust

==========

	coef	std err	t	P>|t|	[0.025	0.97
Intercept	5.7052	1.615	3.532	0.002	2.346	
English	-0.0039	0.003	-1.505	0.147		

==========

Omnibus: 0.120 Durbin-Watson:

Prob(Omnibus): 0.942 Jarq

Skew: -0.126 Prob(JB): 0.8

Kurtosis: 2.495 Cond. No. 3.90e+03

==========

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[2] The condition number is large, 3.9e+03. This might indicate that there are strong multicollinearity or other numerical problems.

R-squared: 0.097 로 해당 모델이 관측치 설명 무의미 0.3 ~0.7이 좋음. 너무 낮거나 높으면 무의미

Prob (F-statistic): 0.147에서 0.05보다 크면, 해당 모델이 무의미

Coef English -0.0039은 영어 점수의 계수값. 영어 점수가 낮을 수록 직업 만족도가 유의미함

Prob(Omnibus): 0.942 총괄 검정 유의확률 1에 가까울 수록 오차가 정규분포되어 있어 좋다고 할 수 있음

비고

직업만족도 영어에 의해 설명될 수 있다는 가설에 대해 단일회귀분석 y = -0.0039*English + 5.7052

2. 회귀분석

파이썬 실습

- 다중 회귀직선

파일	소스코드
실습환경	py3_10_basic

소스코드	import numpy as np
	
	# 데이터프레임 다루기
	import pandas as pd
	# 기초 통계분석 패키지
	from scipy import stats
	# 그래프 그리기
	import matplotlib.pyplot as plt
	#회귀분석 가능 패키지
	import statsmodels.api as sm
	import statsmodels.formula.api as smf
	
	# 직업만족도
	# survey 변수들: name gender income English jobSatisfaction stress
	df = pd.read_csv("./datasets/survey.csv")
	
	# 독립표본 t-검정
	# 변수생성
	male = df.income[df.gender == "m"] # 남성
	female = df.income[df.gender == "f"] # 여성
	
	#Levene의 등분산 검정
	l_result = stats.levene(male, female)

2. 회귀분석

파이썬 실습

- 다중 회귀직선

파일	소스코드
실습환경	py3_10_basic pip install statsmodels

소스코드	# 유의 수준 표시하기
	if l_result[1] > .05:
	  print('P value는 %f 로 95 수준에서 유의하지 않음' % l_result[1])
	else :
	  print('P value는 %f 로 95 percent 수준에서 유의함' % l_result[1])
	
	print('남성', round(male.mean(),2), '여성',round(female.mean(),2),'\n등분산검정 결과
	LeveneResult(F) : %.3f \np-value : %.3f' % (l_result))
	
	# 가상의 분석자료로 회귀분석 결과 구하기
	model = smf.ols(formula = 'jobSatisfaction ~ English + stress + income', data = df)
	result = model.fit()
	print(result.summary())

결과값1	
비고	최소제곱법(OLS) smf.ols(formula = ' 종속변수 ~ 독립 변수', data = 데이터 프레임) 직업만족도 영어, 스트레스, 수입에 의해 설명될 수 있다는 가설에 대해 다중회귀분석 결론 : 직업만족도는 영어, 스트레스, 소득 등으로 설명하기 어려운 것으로 판단된다.

2. 회귀분석

파이썬 실습

● 다중 회귀직선

파일	소스코드

결과값1

P value는 0.258716 로 95 수준에서 유의하지 않음

남성 4285.64 여성 4333.11
등분산검정 결과 LeveneResult(F) : 1.348
p-value : 0.259

OLS Regression Results

Dep. Variable:	jobSatisfaction	**R-squared:**	
Model:	OLS	Adj. R-squared:	0.059
Method:	Least Squares	F-statistic:	1.458
Date:	Sun, 23 Apr 2023	**Prob (F-statistic):**	
Time:	05:30:16	Log-Likelihood:	-35.038
No. Observations:	23	AIC:	78.08
Df Residuals:	19	BIC:	82.62
Df Model:	3		
Covariance Type:	nonrobust		

R-squared: 0.187로 단순회귀직선보다 높아짐

Prob (F-statistic): 0.258

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	4.9159	1.712	2.871	0.010	1.333	8.499
English	-0.0064	0.003	-1.931	0.069	-0.013	0.000
stress	0.2141	0.187	1.145	0.266	-0.177	0.606
income	0.0004	0.000	1.125	0.275	-0.000	0.001

y = -0.0064 * English + 4.9159

Omnibus:	0.278	Durbin-Watson:	0.989
Prob(Omnibus):	**0.870**	Jarque-Bera (JB):	
Skew:	-0.036	Prob(JB):	0.796
Kurtosis:	2.313	Cond. No.	3.00e+04

Prob(Omnibus): 0.870

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 3e+04. This might indicate that there are
strong multicollinearity or other numerical problems.

비고

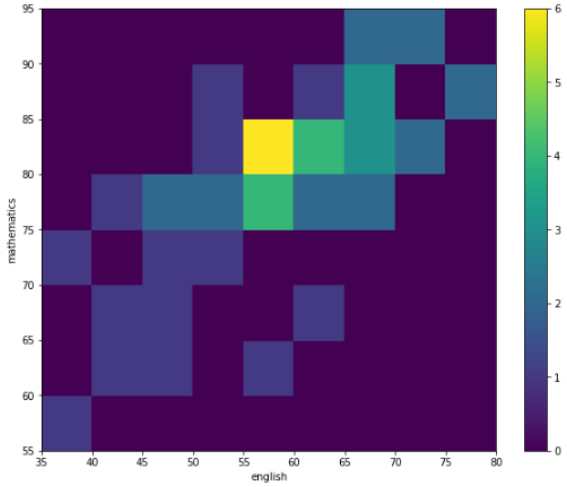
2. 회귀분석

히트맵

- 히스토그램의 2차원 버전으로 색을 이용해 표현하는 그래프

파일	소스코드
실습환경	Tf38_cpu

소스코드	fig = plt.figure(figsize=(10, 8))
	ax = fig.add_subplot(111)
	
	c = ax.hist2d(english_scores, math_scores,
	bins=[9, 8], range=[(35, 80), (55, 95)])
	ax.set_xlabel('english')
	ax.set_ylabel('mathematics')
	ax.set_xticks(c[1])
	ax.set_yticks(c[2])
	# 컬러 바의 표시
	fig.colorbar(c[3], ax=ax)
	plt.show()

결과값1	
비고	

3. 최적 회귀

최적 회귀 방식

- 종속변수에 유의미한 영향을 미칠 것으로 생각되는 독립변수를 찾는 방법
- 보통 모델의 성능을 향상하기 위해 사용

● 최적화 회귀 방정식의 선택

1) 설명 변수 선택

- 필요한 변수만 상황에 따라 타협을 통해 선택
- y에 영향을 미칠 수 있는 모든 설명변수 x들은 y의 값을 예측하는데 참여한다.
- 데이터에 설명변수 x들의 수가 많아지면 관리하는데 많은 노력이 요구 -> 가능한 범위 내에서 적은 수의 설명변수 포함

2) 모형 선택

- 분석 데이터에 가장 잘 맞는 모형을 찾아내는 방법
- 모든 가능한 조합의 회귀 분석: 모든 가능한 독립변수들의 조합에 대한 회귀모형을 생성 & 가장 적합한 회귀모형 선택

3. 최적 회귀

최적 회귀 방식

● 최적화 회귀 방정식의 선택

3) 단계적 변수 선택

종류	내용
전진선택법	절편만 있는 상수모형으로부터 시작해 중요하다고 생각되는 설명변수로부터 차례로 모형에 추가 - 장점: 이해하기 쉽고, 변수의 개수가 많은 경우에도 사용 가능 - 단점: 변수값의 작은 변동에도 그 결과가 크게 달라져 안전성이 부족
후진제거법	독립변수 후보 모두 포함한 모형에서 출발해 가장 적은 영향을 주는 변수부터 하나씩 제거하면서 더이상 제거할 변수가 없을 때의 모형을 선택함 - 장점: 전체 변수들의 정보를 이용할 수 있음 - 단점: 변수의 개수가 많은 경우 사용하기 어려움
단계선택법	전진석택법에 의해 변수를 추가하면서 새롭게 추가된 변수에 기인해 기존 변수의 중요도가 약화되면 해당 변수를 제거하는 등 단계별로 추가 또는 제거되는 변수의 여부를 검토해 더 이상 없을 때 중단한다.

3. 최적 회귀

최적 회귀 방식

- 벌점화된 선택 기준
- 모형의 복잡도에 벌점을 주는 방법. AIC와 BIC방법이 주로 사용
- 방법: AIC, BIC 방법
- 설명
- 모든 후보 모형들에 대해 AIC 또는 BIC를 계산하고 그 값을 최소가 되는 모형을 선택
- 모형 선택의 일치성: 자료가 늘어날 때 참인 모형이 주어진 모형 선택 기준의 최솟값을 갖게 되는 성질
- 이론적으로 AIC에 대해 일치성이 성립되지 않지만 BIC는 주요 분포에서 이러한 성질이 성립
- AIC를 활용하는 방법이 보편화하는 방법 그 밖의 선택 기준으로 RIC, CIC가 있다.

3. 최적 회귀

모델 선택 기준(Model Selection Criteria)

AIC (Akaike Information Criterion)
아카이케 정보 기준
- 공식: AIC = -2 × log(likelihood) + 2k
- k = 모델의 매개변수 개수
- **작을수록 좋은 모델**
- 예측 성능과 모델 복잡성의 균형을 맞춤
- 가장 널리 사용되는 기준

BIC (Bayesian Information Criterion)
베이지안 정보 기준
- 공식: BIC = -2 × log(likelihood) + k × log(n)
- n = 표본 크기
- **작을수록 좋은 모델**
- AIC보다 복잡성에 더 큰 패널티 부여
- 표본이 클수록 AIC보다 더 간단한 모델 선호

3. 최적 회귀

모델 선택 기준(Model Selection Criteria)

RIC (Risk Inflation Criterion)
위험 팽창 기준
- 상대적으로 덜 알려진 기준
- 주로 고차원 데이터에서 사용
- 변수 선택 시 과적합 방지에 초점

CIC (Covariance Information Criterion)
공분산 정보 기준
- 주로 시계열 분석이나 구조방정식 모델에서 사용
- 모델의 공분산 구조를 평가
- 특정 분야에서 제한적으로 사용

3. 최적 회귀

단순회귀모형?

- 설명변수와 반응변수가 각각 1개씩인 모형
- 사례 :
- 기말고사와 쪽지 시험, 수면 시간, 통학 방법 중 어느 변수가 기말고사에 영향을 끼칠까? 어떤 변수 사용이 좋은 모형을 만들까?
- 반응변수는 기말고사 점수 y, 설명변수는 쪽지 시험의 평균 점수 x, 설명변수의 개수는 p

3. 최적 회귀

단순회귀모형?

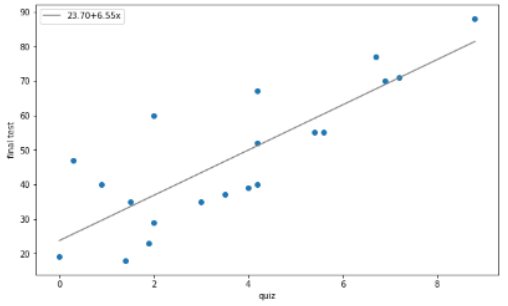
- 설명변수와 반응변수가 각각 1개씩인 모형

파일	소스코드
실습환경	Tf38_cpu pip install statsmodels
소스코드	import numpy as np import pandas as pd import matplotlib.pyplot as plt from scipy import stats import statsmodels.formula.api as smf %precision 3 %matplotlib inline df = pd.read_csv('.././datasets/scores_reg.csv') n = len(df) print(n) df.head()
결과값1	20 quiz final_test sleep_time school_method 0 4.2 67 7.2 bus 1 7.2 71 7.9 bicycle 2 0.0 19 5.3 bus 3 3.0 35 6.8 walk 4 1.5 35 7.5 walk
비고	

3. 최적 회귀

단순회귀모형?

- 단순회귀모형

파일	소스코드
실습환경	Tf38_cpu pip install statsmodels
소스코드	x = np.array(df['quiz']) y = np.array(df['final_test']) p = 1 poly_fit = np.polyfit(x, y, 1) poly_1d = np.poly1d(poly_fit) xs = np.linspace(x.min(), x.max()) ys = poly_1d(xs) fig = plt.figure(figsize=(10, 6)) ax = fig.add_subplot(111) ax.set_xlabel('quiz') ax.set_ylabel('final test') ax.plot(xs, ys, color='gray', label=f'{poly_fit[1]:.2f}+{poly_fit[0]:.2f}x') ax.scatter(x, y) ax.legend() plt.show()
결과값1	
비고	

3. 최적 회귀

회귀계수

- β_0, β_1 는 데이터로부터 추정되어야 하는 모수(회귀계수) 이며, ϵ 은 기댓값 0과 분산 σ2를 가지는 오차항이다.

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i \quad (i = 1, 2, \cdots, n)$$

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**formula = 'final_test ~ quiz'** **result = smf.ols(formula, df).fit()** **result.summary()**
결과값1	OLS Regression Results Dep. Variable: final_test R-squared: 0.676 Model: OLS **Adj. R-squared: 0.658** Method: Least Squares F-statistic: 37.61 Date: Thu, 13 Feb 2020 Prob (F-statistic): 8.59e-06 Time: 11:41:56 Log-Likelihood: -76.325 No. Observations: 20 AIC: 156.7 Df Residuals: 18 BIC: 158.6 Df Model: 1 Covariance Type: nonrobust coef std err t P>|t| [0.025 0.975] Intercept **23.6995** 4.714 5.028 0.000 13.796 33.603 quiz **6.5537** 1.069 6.133 0.000 4.309 8.799 Omnibus: 2.139 Durbin-Watson: 1.478 Prob(Omnibus): 0.343 Jarque-Bera (JB): 1.773 Skew: 0.670 Prob(JB): 0.412 Kurtosis: 2.422 Cond. No. 8.32
비고	

3. 최적 회귀

		회귀계수의 추정값		회귀계수에 관한 t 검정통계량		회귀계수의 95% 신뢰구간	
		coef	std err	t	P>|t|	[0.025	0.975]
절편 β_0	Intercept	23.6995	4.714	5.028	0.000	13.796	33.603
기울기 β_1	quiz	6.5537	1.069	6.133	0.000	4.309	8.799

추정값의 표준오차

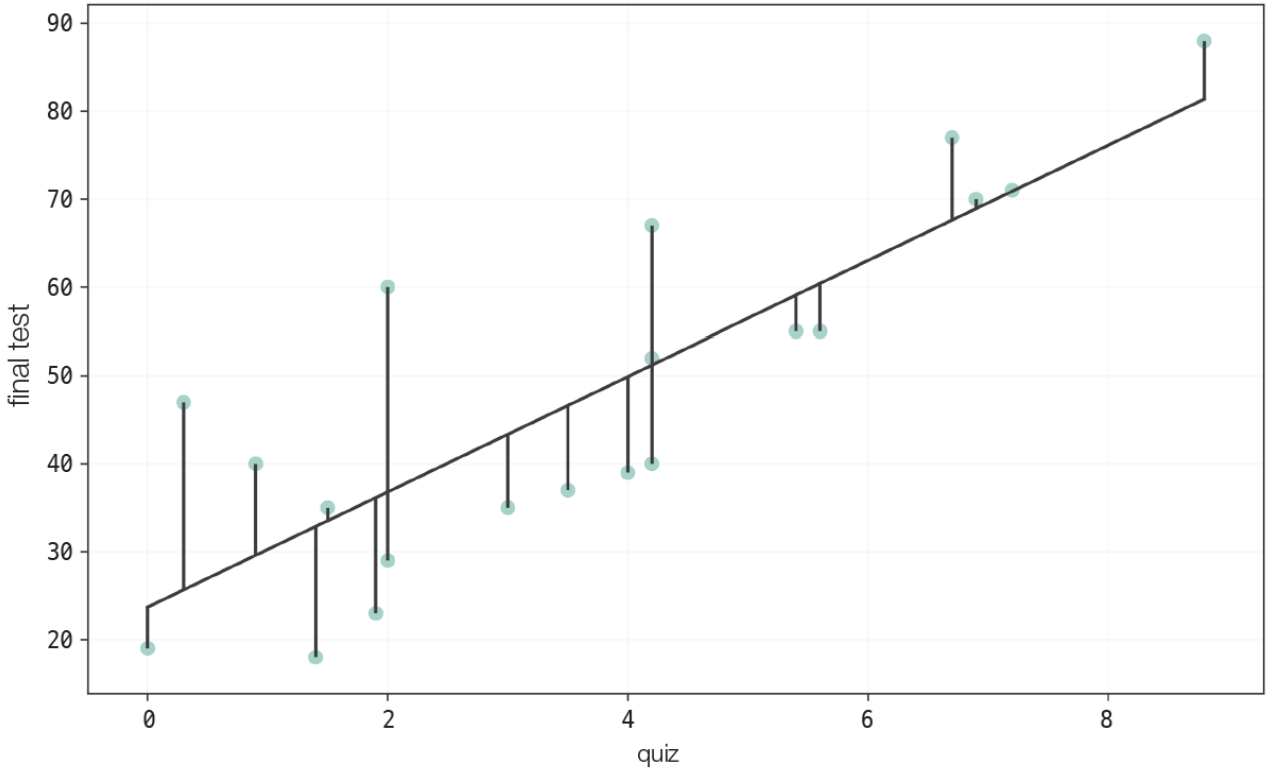
t 검정통계량의 p값

- 점추정 : $\hat{\beta}_0, \hat{\beta}_1$
- 추정값이 생성한 회귀직선 $y = \hat{\beta}_0 + \hat{\beta}_1 x$ 은 예측값 $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$ 과 실제의 데이터 y_i 의 차이가 가장 작은 직선
- 잔차 $\hat{\epsilon}_i$: $y_i - \hat{y}_i$
- 최소제곱법: 잔차제곱합 $\sum_i^n \hat{\epsilon}_i^2$을 최소화하는 $\hat{\beta}_0, \hat{\beta}_1$ 을 구하는 방법

3. 최적 회귀

회귀계수

- 회귀직선과 잔차



3. 최적 회귀

회귀계수

● 회귀직선과 잔차

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**X = np.array([np.ones_like(x), x]).T** **X**
결과값1	array([[1. , 4.2], [1. , 7.2], [1. , 0.], [1. , 3.], [1. , 1.5], [1. , 0.9], [1. , 1.9], [1. , 3.5], [1. , 4.], [1. , 5.4], [1. , 4.2], [1. , 6.9], [1. , 2.], [1. , 8.8], [1. , 0.3], [1. , 6.7], [1. , 4.2], [1. , 5.6], [1. , 1.4], [1. , 2.]])
비고	

절편용 1은 "상수항을 곱할 변수" 역할을 합니다.

- $1 \times \beta_0 = \beta_0$ → 절편이 그대로 나타남
- 모든 관측값에 대해 동일하게 β_0가 더해짐
- **절편용 1이 없으면** 회귀직선이 반드시 원점(0,0)을 지나야 하지만, **절편용 1이 있으면** 회귀직선이 y축의 임의의 점에서 시작할 수 있다.

3. 최적 회귀

회귀계수

- 최소제곱법은 np.linalg.lstsq 실행
- 예측값과 잔차

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**beta0_hat, beta1_hat = np.linalg.lstsq(X, y)[0]** **beta0_hat, beta1_hat** **y_hat = beta0_hat + beta1_hat * x** **eps_hat = y - y_hat** **s_var = np.var(eps_hat, ddof=p+1)** σ^2 **s_var**
결과값1	(23.699495346731226, 6.553732606043085)
결과값2	134.290434734959
비고	**모분산 추정** **np.linalg.lstsq() 함수는 최소자승법(Least Squares)을 사용하여, 오차를 최소화하는 회귀 계수를 찾을 수 있습니다. 이 함수는 선형 회귀 분석에서 회귀 계수를 추정하는 데 자주 사용됩니다.**

ddof=p+1의 의미

ddof (Delta Degrees of Freedom): 자유도 보정값

- **일반 분산**: σ² = Σ(x - x̄)² / n
- **표본 분산**: s² = Σ(x - x̄)² / (n-1) ← ddof=1
- **회귀 잔차 분산**: s² = Σ(eps)² / (n-p-1) ← ddof=p+1

왜 p+1인가?

회귀분석에서는 **p+1개의 모수를 추정**하기 때문입니다:

- p: 독립변수의 개수
- +1: 절편
- 단순선형회귀의 경우: p=1이므로 ddof=2
- 자유도의 의미
- # 예시: n=10개 관측값, 2개 모수(절편+기울기) 추정
- # 자유도 = 10 - 2 = 8
- # 분산 = Σ(잔차²) / 8

3. 최적 회귀

회귀계수

- 구간추정

회귀계수의 **분산-공분산 행렬**은 $\mathrm{Var}(\hat{\beta}) = \sigma^2 (X'X)^{-1}$

```
# (X'X)⁻¹ 행렬의 형태:
[[C0,  C01],    # C0: Var(β₀)/σ²
 [C01, C1 ]]    # C1: Var(β₁)/σ²
                # C01: Cov(β₀,β₁)/σ²
```

t-통계량 계산
t_stat_beta0 = beta0_hat / np.sqrt(s_var * C0) t_stat_beta1 = beta1_hat / np.sqrt(s_var * C1
s_var: 잔차분산 추정값 ($\hat{\sigma}^2$)
C0, C1: 분산 계수들
표준오차 = √(분산) = √($\hat{\sigma}^2$ × 분산계수))

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**C0, C1 = np.diag(np.linalg.pinv(np.dot(X.T, X)))** **np.sqrt(s_var * C0), np.sqrt(s_var * C1)**
결과값1	(4.713837012645705, 1.0685841387335373)
비고	**최소자승법을 사용하는 선형 회귀 분석에서는 특이 행렬이나 역행렬이 존재하지 않을 수 있습니다. 이런 경우에 np.linalg.pinv() 함수를 사용하여 유사 역행렬을 계산하여 회귀 계수를 추정할 수 있습니다.**

$\hat{\beta}_0$과 $\hat{\beta}_1$의 표준오차 각각 $\sqrt{C_0\hat{\sigma}^2}$과 $\sqrt{C_1\hat{\sigma}^2}$

$(XX^T)^{-1}$의 대각성분의 첫 번째가 C_0, 두 번째가 C_1

3. 최적 회귀

회귀계수

- 구간추정

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**rv = stats.t(n-2)** **lcl = beta0_hat - rv.isf(0.025) * np.sqrt(s_var * C0)** **hcl = beta0_hat - rv.isf(0.975) * np.sqrt(s_var * C0)** **print(lcl, hcl)** **rv = stats.t(n-2)** **lcl = beta1_hat - rv.isf(0.025) * np.sqrt(s_var * C1)** **hcl = beta1_hat - rv.isf(0.975) * np.sqrt(s_var * C1)** **print(lcl, hcl)**
결과값1	(13.79609127276026, 33.602899420702194)
결과값2	(4.308720637125893, 8.798744574960278)
비고	회귀계수 β_0, β_1의 신뢰수준 $100(1-\alpha)\%$의 신뢰구간은 $\left[\hat{\beta}_i - t_{\alpha/2}(n-2)\sqrt{\hat{\sigma}^2 C_i},\ \hat{\beta}_i - t_{1-\alpha/2}(n-2)\sqrt{\hat{\sigma}^2 C_i}\right] \quad (i=0,1)$

3. 최적 회귀

회귀계수

- t 검정

  - 귀무가설 : $\beta_1 = 0$
  - 대립가설 : $\beta_1 \neq 0$

검정통계량은 $t = \dfrac{\hat{\beta}_1 - \beta_1}{\sqrt{\hat{\sigma}^2 C_1}}$ 이고, 귀무가설은 $\beta_1 = 0$이므로

$t = \dfrac{\hat{\beta}_1}{\sqrt{\hat{\sigma}^2 C_1}}$ 을 계산

3. 최적 회귀

회귀계수

● t 검정

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**t = beta1_hat / np.sqrt(s_var * C1)** **print(t)** **print((1 - rv.cdf(t)) * 2)**
결과값1	6.133099274532023
결과값2	(4.308720637125893, 8.798744574960278)
비고	**t-통계량 계산** **pythont = beta1_hat / np.sqrt(s_var * C1)** **귀무가설: H_0: $\beta_1 = 0$ (기울기가 0이다, 즉 x와 y는 무관하다)** **대립가설: H_1: $\beta_1 \neq 0$ (기울기가 0이 아니다)** **t-통계량: 추정된 계수를 표준오차로 나눈 값** **t값이 클수록 → 계수가 0과 멀리 떨어져 있음 → 통계적으로 유의미할 가능성 ↑** **p-값 계산** **print((1 - rv.cdf(t)) * 2)** **rv.cdf(t): t-분포의 누적분포함수(CDF) 값** **귀무가설은 기각**

3. 최적 회귀

회귀계수

- t 검정

• 귀무가설 : $\beta_0 = 0$

• 대립가설 : $\beta_0 \neq 0$

검정통계량은 $t = \frac{\hat{\beta}_0 - \beta_0}{\sqrt{\hat{\sigma}^2 C_0}}$ 이고, 귀무가설은 $\beta_0 = 0$ 이므로

$t = \frac{\hat{\beta}_0}{\sqrt{\hat{\sigma}^2 C_0}}$ 을 계산

3. 최적 회귀

회귀계수

- t 검정

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**t = beta0_hat / np.sqrt(s_var * C0)** **print(t)** **print((1 - rv.cdf(t)) * 2)**
결과값1	5.027644206440129
결과값2	(8.745298393186829e-05
비고	귀무가설은 기각

4. 다중회귀모형

다중회귀모형 개념

- 선형 회귀분석은 설명변수와 목적변수(혹은 반응변수) 간의 관계가 선형인 회귀분석을 말하며, 설명변수(혹은 독립변수)가 한 개인 경우 단순선형 회귀분석이라고 하며, 설명변수가 2개 이상이면 다중회귀분석이라고 부른다.
- 독립변수가 한 개인 경우 2차원 그래프로 관계를 확인하는 것이 유용하다. 그러나 독립변수가 2개 이상이 되면, 3차원 공간상에서 2차 평면으로 회귀적합이 표현되며, 독립변수가 3개를 넘어가면 4차원 이상이 되어 시각적으로 파악하기가 어려워진다. 하지만 일반적으로 특정 현상을 설명하는 데 있어서 목표변수 Y의 변화가 하나의 독립변수로 잘 설명되는 경우는 거의 없다고 할 수 있다. 따라서 적절한 독립변수들을 선정하여 이들의 함수로서 목표변수를 설명하고자 하는 다중회귀분석을 일반적으로 시행하게 된다.
- 이러한 다중회귀분석의 모형은 아래와 같이 표현할 수 있다.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p + \epsilon$$

$$Y = X\beta + \epsilon$$

4. 다중회귀모형

다중회귀모형

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**formula = 'final_test ~ quiz + sleep_time'** **result = smf.ols(formula, df).fit()** **result.summary()**
결과값1	OLS Regression Results Dep. Variable: final_test R-squared: 0.756 Model: OLS Adj. R-squared: 0.727 Method: Least Squares F-statistic: 26.35 Date: Wed, 29 Nov 2023 Prob (F-statistic): 6.19e-06 Time: 18:43:22 Log-Likelihood: -73.497 No. Observations: 20 AIC: 153.0 Df Residuals: 17 BIC: 156.0 Df Model: 2 Covariance Type: nonrobust coef std err t P>|t| [0.025 0.975] Intercept -1.8709 11.635 -0.161 0.874 -26.420 22.678 quiz 6.4289 0.956 6.725 0.000 4.412 8.446 sleep_time 4.1917 1.778 2.357 0.031 0.440 7.943 Omnibus: 2.073 Durbin-Watson: 1.508 Prob(Omnibus): 0.355 Jarque-Bera (JB): 1.716 Skew: 0.660 Prob(JB): 0.424 Kurtosis: 2.437 Cond. No. 38.0
비고	

4. 다중회귀모형

회귀계수

- 설명변수인 쪽지 시험과 수면 시간의 데이터는 각각 x1, x2, 반응변수인 기말고사는 y로 설정, 설명변수의 개수 p는 2

파일	소스코드
실습환경	Tf38_cpu pip install statsmodels
소스코드	x1 = df['quiz'] x2 = df['sleep_time'] y = df['final_test'] p = 2 X = np.array([np.ones_like(x1), x1, x2]).T beta0_hat, beta1_hat, beta2_hat = np.linalg.lstsq(X, y)[0] print(beta0_hat, beta1_hat, beta2_hat) y_hat = beta0_hat + beta1_hat * x1 + beta2_hat * x2 eps_hat = y - y_hat s_var = np.sum(eps_hat ** 2) / (n - p - 1) C0, C1, C2 = np.diag(np.linalg.pinv(np.dot(X.T, X))) rv = stats.t(n-p-1) lcl = beta2_hat - rv.isf(0.025) * np.sqrt(s_var * C2) hcl = beta2_hat - rv.isf(0.975) * np.sqrt(s_var * C2) print(lcl, hcl)
결과값1	(-1.8709143470996081, 6.428878343002374, 4.191706546398686)
결과값12	(0.44025333254349563, 7.943159760253876)
비고	

4. 다중회귀모형

가변수

- 질적변수를 변환하여 양적변수와 동일하게 취급할 수 있게 하는 기법
- 0과 1을 취하는 2진 변수
- 변환하고 싶은 질적변수의 카테고리 수에서 하나를 줄인 수만큼 필요
- 통학 방법 '버스' '자전거' '도보'인 카테고리 수는 3이므로

가변수는 2개 $x_{도보}, x_{자전거}$

도보 $(x_{도보}=1, x_{자전거}=0)$, 자전거 $(x_{도보}=0, x_{자전거}=1)$,

버스 $(x_{도보}=0, x_{자전거}=0)$

회귀모형 $Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i도보} + \beta_4 x_{i자전거} + \epsilon_i$

4. 다중회귀모형

가변수

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**formula = 'final_test ~ quiz + sleep_time + school_method'** **result = smf.ols(formula, df).fit()** **result.summary()**
결과값1	Dep. Variable: final_test R-squared: 0.782 Model: OLS Adj. R-squared: 0.724 Method: Least Squares F-statistic: 13.46 Date: Thu, 13 Feb 2020 Prob (F-statistic): 7.47e-05 Time: 11:42:45 Log-Likelihood: -72.368 No. Observations: 20 AIC: 154.7 Df Residuals: 15 BIC: 159.7 Df Model: 4 Covariance Type: nonrobust coef std err t P>|t| [0.025 0.975] Intercept 1.3330 12.434 0.107 0.916 -25.169 27.835 school_method[T.bus] -1.8118 6.324 -0.286 0.778 -15.292 11.668 school_method[T.walk] -7.6555 6.420 -1.192 0.252 -21.339 6.028 quiz 6.0029 1.033 5.809 0.000 3.800 8.206 sleep_time 4.5238 1.809 2.501 0.024 0.668 8.380 Omnibus: 1.764 Durbin-Watson: 1.418 Prob(Omnibus): 0.414 Jarque-Bera (JB): 0.989 Skew: 0.545 Prob(JB): 0.610 Kurtosis: 2.985 Cond. No. 41.8
비고	

5. 모형의 선택

과적합

- 적합이 좋다는 것은 모형이 주변에 있는 데이터에 어느 정도 들어 맞는다. 회귀직선이 데이터에 완전하게 들어맞고 잔차가 작으면 그 모형은 좋은 모형이라고 할 수 있다. 모르는 데이터의 설명변수라도 모형이 반응변수를 정확하게 예측할 수 있다면 그것은 좋은 모형이라고 할 수 있다.
- 적합이 좋다는 것은 설명변수를 증가시켜 가는 것만으로 간단하게 달성된다. 그러나 이렇게 만든 모형은 일반적으로 예측 정확도가 떨어진다. 이것을 과적합(overfitting)이라고 하는데, 복잡한 모형은 표현력이 너무 높아 나머지 주변 데이터에 지나치게 적합되어 일반적인 예측성을 잃게 된다. 따라서 모형을 고를 때 적합도가 좋은 것보다 예측 정확도가 좋은 것을 고르게 된다.

5. 모형의 선택

모형 선택

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**x = np.array(df['quiz'])** **y = np.array(df['final_test'])** **p = 1** **formula = 'final_test ~ quiz'** **result = smf.ols(formula, df).fit()** **result.summary()**
결과값1	Dep. Variable: final_test R-squared: 0.676 Model: OLS Adj. R-squared: 0.658 Method: Least Squares F-statistic: 37.61 Date: Thu, 13 Feb 2020 Prob (F-statistic): 8.59e-06 Time: 11:43:17 Log-Likelihood: -76.325 No. Observations: 20 AIC: 156.7 Df Residuals: 18 BIC: 158.6 Df Model: 1 Covariance Type: nonrobust coef std err t P>|t| [0.025 0.975] Intercept 23.6995 4.714 5.028 0.000 13.796 33.603 quiz 6.5537 1.069 6.133 0.000 4.309 8.799 Omnibus: 2.139 Durbin-Watson: 1.478 Prob(Omnibus): 0.343 Jarque-Bera (JB): 1.773 Skew: 0.670 Prob(JB): 0.412 Kurtosis: 2.422 Cond. No. 8.32
비고	

5. 모형의 선택

모형 선택

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**y_hat = np.array(result.fittedvalues)** **print(y_hat)** **eps_hat = np.array(result.resid)** **print(eps_hat)** **print(np.sum(eps_hat ** 2))**
결과값1	array([51.225, 70.886, 23.699, 43.361, 33.53 , 29.598, 36.152, 46.638, 49.914, 59.09 , 51.225, 68.92 , 36.807, 81.372, 25.666, 67.61 , 51.225, 60.4 , 32.875, 36.807])
결과값2	array([15.775, 0.114, -4.699, -8.361, 1.47 , 10.402, -13.152, -9.638, -10.914, -4.09 , -11.225, 1.08 , -7.807, 6.628, 21.334, 9.39 , 0.775, -5.4 , -14.875, 23.193])
결과값3	2417.227825229262
비고	잔차제곱합

5. 모형의 선택

결정계수

- 모형의 데이터에 대한 적합도를 나타내는 기본적인 지표

$$R^2 = \frac{\text{회귀변동}}{\text{총변동}} = 1 - \frac{\text{잔차변동}}{\text{총변동}}$$

- 총변동 $\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2$ = 회귀변동 + 잔차변동
- 회귀변동 $\sum_{i=1}^{n}(y_i - \bar{y})^2$
- 잔차변동 $\sum_{i=1}^{n}\hat{\epsilon}_i^2$

5. 모형의 선택

결정계수

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**total_var = np.sum((y - np.mean(y))**2)** **exp_var = np.sum((y_hat - np.mean(y))**2)** **unexp_var = np.sum(eps_hat ** 2)** **print(total_var, exp_var + unexp_var)** **print(exp_var / total_var)** **print(np.corrcoef(x, y)[0, 1] ** 2)**
결과값1	(7468.55, 7468.549999999999)
결과값2	0.6763457665505
결과값3	0.6763457665505004
비고	단순회귀의 결정계수는, 설명변수와 반응변수의 상관계수 제곱 r_{xy}^2에 일치

5. 모형의 선택

조정결정계수

- 설명변수를 추가했을 때, 해당 설명변수에 어느 정도 이상의 설명력이 없는 경우 결정계수의 값이 증가하지 않도록 조정하는 결정계수

$\overline{R}^2$ Adj. R-squred

- 자유도를 고려하므로 '자유도조정 결정계수'라고도 부름

$$\overline{R}^2 = 1 - \frac{\text{잔차변동}/n-p-1}{\text{총변동}/n-1}$$

5. 모형의 선택

조정결정계수

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**1 - (unexp_var / (n - p - 1)) / (total_var / (n - 1))**
결과값1	0.6583649758033057
비고	

5. 모형의 선택

F 검정

- **F 검정통계량**

- 귀무가설 : $\beta_1 = \beta_2 = \cdots = \beta_p = 0$
- 대립가설 : 적어도 하나의 β_1 은 0이 아닙니다.

$$F = \frac{\text{회귀변동}/p}{\text{잔차변동}/(n-p-1)}$$

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**f = (exp_var / p) / (unexp_var / (n - p - 1))** **f** **rv = stats.f(p, n-p-1)** **1 - rv.cdf(f)**
결과값1	37.61490671126525
결과값2	8.590875866687497e-06
비고	

5. 모형의 선택

통계모형 평가

- **MLE: 데이터를 가장 잘 설명하는 모수 추정법**
- **AIC는 "예측 정확도"에 초점,**
- **BIC는 "진짜 모형(true model)에 가까운 단순한 모델"을 선택하는 경향**
- **AIC와 BIC 비교**

기준	패널티 항	데이터 크기 영향	특징
AIC	$2k$	표본 수와 무관	비교적 자유롭게 복잡한 모델 선택
BIC	kln(n)	표본 수 커질수록 패널티 ↑	표본 수가 많을 때 단순한 모델 선호

5. 모형의 선택

최대로그우도와 AIC

- **우도(likelihood): 어떤 관측값을 얻을 확률-가능성이 높은지**
- 동전의 확률함수 f(x) 얻은 관측값 x_1, x_2, x_3, x_4, x_5

$$L = \prod_{i=1}^{5} f(x_i) = 0.3^2 0.7^3 \simeq 0.031$$

- 동전의 앞면이 나올 확률 p를 널지 못하는 상황에서 결과 [0, 1, 0, 0, 1]

$$L = \prod_{i=1}^{5} f(x_i) = p^2 (1-p)^3$$

- 우도함수 L(p): 우도 L은 p에 대한 함수로 표현우도는 확률의 곱으로, 곱하면 곱할수록 0에 가까워져 다루기 어려움
- 우도에 로그를 취한 로그우도를 대신 사용

$$\log L = \sum_{i=1} \log f(x_i)$$

5. 모형의 선택

최대로그우도와 AIC

- **AIC (Akaike Information Criterion)**
- 로그우도와 같은 적합도는 의미 없는 설명변수를 늘리면 증가하므로 성능이 나쁜 모형이 선택
- 모형의 복잡도(설명변수의 수)와 데이터에 대한 균형을 잡는 지표 필요. 통계 모형의 "적합도 + 복잡도"를 동시에 고려해 모델을 비교하는 기준
- AIC가 작을수록 좋은 모델 (적합도 높고, 복잡도 낮음)

$$AIC = -2 \times \text{최대로그우도} + 2 \times \text{회귀계수의 수} \qquad AIC = -2\ln(L) + 2k$$

- AIC는 값이 작을수록 모형의 예측 정확도가 좋음

  - 기말고사 ~ 쪽지 시험 : 156.7
  - 기말고사 ~ 쪽지 시험 + 수면 시간 : 153.0
  - 기말고사 ~ 쪽지 시험 + 수면 시간 + 통학 방법 : 154.7

5. 모형의 선택

최대로그우도와 AIC

- **베이지안 정보 기준**
- 회귀계수의 수에 대해 표본 크기 n에 대해서도 페널티 부가
- 통계모형의 적합도와 복잡도를 동시에 고려하여 여러 모델 중 어떤 모델이 더 좋은지 비교할 때 사용하는 기준

$$BIC = -2 \times \text{최대로그우도} + \log n \times \text{회귀계수의 수} \qquad BIC = -2\ln(L) + k\ln(n)$$

- AIC와 달리 BIC는 베이지안 접근에서 유도된 정보 기준

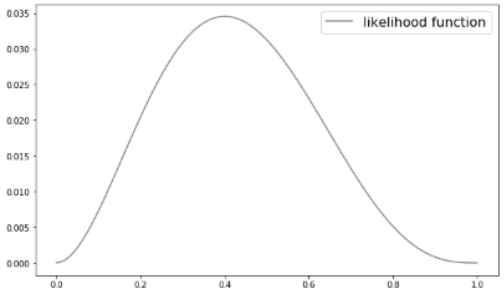
  - 기말고사 ~ 쪽지 시험 : 158.6
  - 기말고사 ~ 쪽지 시험 + 수면 시간 : 156.0
  - 기말고사 ~ 쪽지 시험 + 수면 시간 + 통학 방법 : 159.7

- 값이 작을수록 모형의 예측 정확도가 좋음

5. 모형의 선택

최대로그우도와 AIC

- p를 0에서 1로 변화시킬 때의 우도함수

파일	소스코드
실습환경	Tf38_cpu pip install statsmodels
소스코드	prob = 0.3 coin_result = [0, 1, 0, 0, 1] rv = stats.bernoulli(prob) L = np.prod(rv.pmf(coin_result)) print(L) ps = np.linspace(0, 1, 100) Ls = [np.prod(stats.bernoulli(prob).pmf(coin_result)) for prob in ps] fig = plt.figure(figsize=(10, 6)) ax = fig.add_subplot(111) ax.plot(ps, Ls, label='likelihood function', color='gray') ax.legend(fontsize=16) plt.show()
결과값1	0.030870000000000005
결과값2	
비고	

5. 모형의 선택

최대로그우도와 AIC

- p를 0에서 1로 변화시킬 때의 우도함수

파일	소스코드
실습환경	Tf38_cpu pip install statsmodels
소스코드	prob = 0.4 rv = stats.bernoulli(prob) mll = np.sum(np.log(rv.pmf([0, 1, 0, 0, 1]))) mll rv = stats.norm(y_hat, np.sqrt(unexp_var / n)) mll = np.sum(np.log(rv.pdf(y))) mll aic = -2 * mll + 2 * (p+1) aic bic = -2 * mll + np.log(n) * (p+1) bic
결과값1	-3.365058335046282
결과값2	-76.32521428624038
결과값3	156.65042857248076
결과값4	158.64189311958876
비고	우도함수가 최대로 될 때 로그우도함수도 최대가 되므로, 최우추정은 로그우도함수가 최대가 될 때의 파라미터로 구할 수 있음. 이때의 로그우도의 값이 최대로그우도

$$\sum_{i=1}^{n} \log f(y_i)$$

- 기말고사 ~ 쪽지시험 : −76.325
- 기말고사 ~ 쪽지시험 + 수면 시간 : −73.497
- 기말고사 ~ 쪽지시험 + 수면 시간 + 통학 방법 : −72.368

5. 모형의 선택

모형의 타당성

- 회귀분석에 관해서 세운 '오차항 $\in_i$는 서로 독립이고 $N(0, \sigma^2)$ 을 따른다는 가정을 만족하고 있는지의 여부를 체크하는 것
- 잔차 $\in_i$

5. 모형의 선택

모형의 타당성

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**formula = 'final_test ~ quiz + sleep_time'** **result = smf.ols(formula, df).fit()** **print(result.summary())** **eps_hat = np.array(result.resid)** **print(eps_hat)**
결과값1	OLS Regression Results Dep. Variable: final_test R-squared: 0.756 Model: OLS Adj. R-squared: 0.727 Method: Least Squares F-statistic: 26.35 Date: Thu, 13 Feb 2020 Prob (F-statistic): 6.19e-06 Time: 11:57:32 Log-Likelihood: -73.497 No. Observations: 20 AIC: 153.0 Df Residuals: 17 BIC: 156.0 Df Model: 2 Covariance Type: nonrobust coef std err t P>|t| [0.025 0.975] Intercept -1.8709 11.635 -0.161 0.874 -26.420 22.678 quiz 6.4289 0.956 6.725 0.000 4.412 8.446 sleep_time 4.1917 1.778 2.357 0.031 0.440 7.943 Omnibus: 2.073 Durbin-Watson: 1.508 Prob(Omnibus): 0.355 Jarque-Bera (JB): 1.716 Skew: 0.660 Prob(JB): 0.424 Kurtosis: 2.437 Cond. No. 38.0
결과값2	[11.689 -6.531 -1.345 -10.919 -4.21 4.228 -5.368 -1.235 -4.546 -9.283 -3.574 3.619 -14.682 7.727 18.439 13.581 -1.215 -9.73 -6.316 19.671]
비고	

5. 모형의 선택

더빈-왓슨비

- 다른 오차항이 서로 무상관인지 여부를 체크하는 지표
- 0에 가까우면 양의 상관, 4에 가까우면 음의 상관, 2 전후의 값이면 무상관

$$\frac{\sum_{i=2}^{n}(\hat{\epsilon}_i - \hat{\epsilon}_{i-1})^2}{\sum_{i=1}^{n}\hat{\epsilon}_i^2}$$

파일	소스코드
실습환경	Tf38_cpu pip install statsmodels
소스코드	np.sum(np.diff(eps_hat, 1) ** 2) / np.sum(eps_hat ** 2)
결과값1	1.5082185264423011
결과값2	
비고	0에 가까우면 양의 상관, 4에 가까우면 음의 상관, 2 전후의 값이면 무상관

5. 모형의 선택

다중공선성

- 조건수의 값이 크면 다중공선성과 설명변수에 강한 상관
- 다중공선성이 크면 회귀계수의 분산이 커져 모형의 예측 결과가 나빠짐
- 쪽지 시험의 결과를 2배로 한 중간고사 변수 추가
- 쪽지 시험과 중간고사의 상관관계는 1, 다중공선성이 큼

- 사례

- 조건수가 1.22e+17
- 다중공선성이 생기면 조건수는 매우 큰 값이 됨
- 조건수가 꽤 큰 값이면 다중공선성을 의심
- 설명변수 중에서 한 쪽 변수를 모형에서 제외

5. 모형의 선택

다중공선성

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**df['mid_test'] = df['quiz'] * 2** **df.head()** **formula = 'final_test ~ quiz + mid_test'** **result = smf.ols(formula, df).fit()** **result.summary()**
결과값1	quiz final_test sleep_time school_method mid_test 0 4.2 67 7.2 bus 8.4 1 7.2 71 7.9 bicycle 14.4 2 0.0 19 5.3 bus 0.0 3 3.0 35 6.8 walk 6.0 4 1.5 35 7.5 walk 3.0
결과값2	Dep. Variable: final_test R-squared: 0.676 Model: OLS Adj. R-squared: 0.658 Method: Least Squares F-statistic: 37.61 Date: Thu, 13 Feb 2020 Prob (F-statistic): 8.59e-06 Time: 11:57:40 Log-Likelihood: -76.325 No. Observations: 20 AIC: 156.7 Df Residuals: 18 BIC: 158.6 Df Model: 1 Covariance Type: nonrobust coef std err t P>|t| [0.025 0.975] Intercept 23.6995 4.714 5.028 0.000 13.796 33.603 quiz 1.3107 0.214 6.133 0.000 0.862 1.760 mid_test 2.6215 0.427 6.133 0.000 1.723 3.519 Omnibus: 2.139 Durbin-Watson: 1.478 Prob(Omnibus): 0.343 Jarque-Bera (JB): 1.773 Skew: 0.670 Prob(JB): 0.412 Kurtosis: 2.422 Cond. No. 1.30e+16
비고	

6. 검정

정규성 검정

- 오차항 ϵ_i 가 $N(0, \sigma^2)$ 을 따른다는 가정이 타당한지를 알아보기 위해, 잔차 ϵ_i가 정규분포를 따르고 있는지를 확인하는 정규성 검정을 수행
- **statsmodels의 정규성 검정은 Omnibus와 Jarque-Bera(JB)**

- 귀무가설 : 잔차항은 정규분포를 따릅니다.
- 대립가설 : 잔차항은 정규분포를 따르지 않습니다.

6. 검정

정규성 검정

- 데이터가 정규분포(normal distribution)를 따르는지 알아보는 것은 통계분석을 위해 매우 중요한 과정이다. 정규분포를 따르지 않는다면, 통계분석 결과가 왜곡될 수 있기 때문이다. 그래서 데이터가 정규분포를 따르고 있는지 여부를 확인하는 작업이 필요하다. 그리고 분포 기반의 이상치 판별을 위해서도 정규분포를 기반으로 접근하
- 오차항 ϵ_i 가 $N(0, \sigma^2)$ 을 따른다는 가정이 타당한지를 알아보기 위해, 잔차 ϵ_i가 정규분포를 따르고 있는지를 확인하는 정규성 검정을 수행
- statsmodels의 정규성 검정은 Omnibus와 Jarque-Bera(JB)

• 귀무가설 : 잔차항은 정규분포를 따릅니다.

• 대립가설 : 잔차항은 정규분포를 따르지 않습니다.

6. 검정

다양한 정규성 검정

- 정규성 검정은 대표적으로 Shapiro-Wilk 검정, Kolmogorov-Smirnov 검정이 있으며 해당 검정은 다음과 같은 특징이 있다.
- Shapiro-Wilk 검정

- 표본 데이터의 순서 통계량과 해당하는 정규 분포에서 예상되는 값 사이의 상관관계를 검정
- 이 검정은 소표본($n < 50$)에 매우 효과적
- 표본 크기가 커질수록 검정의 민감도가 떨어질 수 있음

- Kolmogorov-Smirnov 검정

- 표본 데이터의 누적 분포 함수(CDF)와 참조된 정규 분포의 CDF 사이의 최대 차이를 기반으로 정규성을 검정
- 큰 표본을 대상으로 주로 활용됨
- 기본적으로 모수가 알려져 있어야 함
- 표본의 평균과 표준편차를 사용한다면 Liliefors 검정을 사용하는 것을 권장

6. 검정

다양한 정규성 검정

- 그 외에도 다양한 정규성 검정은 다음과 같다.
- **Lilliefors 검정**

- 표본으로부터 추정된 평균과 표준편차를 사용하여 정규성을 검정
- 소표본(n < 50)의 정규성을 검정하기 위해 사용
- 표본 크기가 커질수록 검정의 민감도가 떨어질 수 있음
- Kolmogorov-Smirnov 검정의 확장된 버전

- **Anderson-Darling 검정**

- 이 검정은 표본의 누적 분포와 정규 분포의 누적 분포 사이의 차이를 측정
- 특히 분포의 꼬리 부분에서의 차이에 더 민감하게 반응하는 특징이 있음
- 정규성을 가정한 모델의 적합성을 평가할 때 유용하게 사용

- **Jarque-Bera 검정**

- 데이터의 왜도(skewness)와 첨도(kurtosis)를 이용하여 정규성을 검정
- 주로 대표본의 정규성 검정에 사용

위의 검정들은 데이터의 정규성을 검정하기 위한 다양한 방법들로, 특정 상황이나 데이터의 특성에 따라 적합한 검정법을 선택하는 것이 중요합니다.

- 그 외에 데이터가 정규분포에 가까운지 알아보는 QQ-plot이 있다.

6. 검정

귀무가설과 대립가설

- 정규성 검정의 기본적인 귀무가설과 대립가설은 다음과 같다.
  - 귀무가설(Ho): 데이터의 분포는 정규분포와 다르다고 보기 어렵다.
  - 대립가설(H1): 데이터의 분포는 정규분포와 다르다.
- 물론 각각의 검정 방법에 따라 귀무가설과 대립가설이 조금씩 다르나 대부분의 경우 위와 같은 귀무가설과 대립가설을 사용한다. 다음은 각 검정 방법에 따른 귀무가설과 대립가설이다.

- **Kolmogorov-Smirnov 검정**
  - 귀무가설(Ho): 두 표본 집단간 분포는 같다.
  - 대립가설(H1): 두 표본 집단간 분포는 다르다.

- **Anderson-Darling 검정**
  - 귀무가설(Ho): 표본이 특정 분포를 따른다.
  - 대립가설(H1): 표본이 특정 분포를 따르지 않는다.

6. 검정

귀무가설과 대립가설

- **Jarque-Bera 검정**
  - 귀무가설(Ho): 표본의 왜도와 (초과)첨도는 정규 분포의 것과 같다.
  - 대립가설(H1): 표본의 왜도와 (초과)첨도는 정규 분포의 것과 다르다.

- 좀 더 엄밀하게 말하자면 표본의 왜도와 (초과)첨도가 0인지 아닌지 검정하는 것이다.

6. 검정

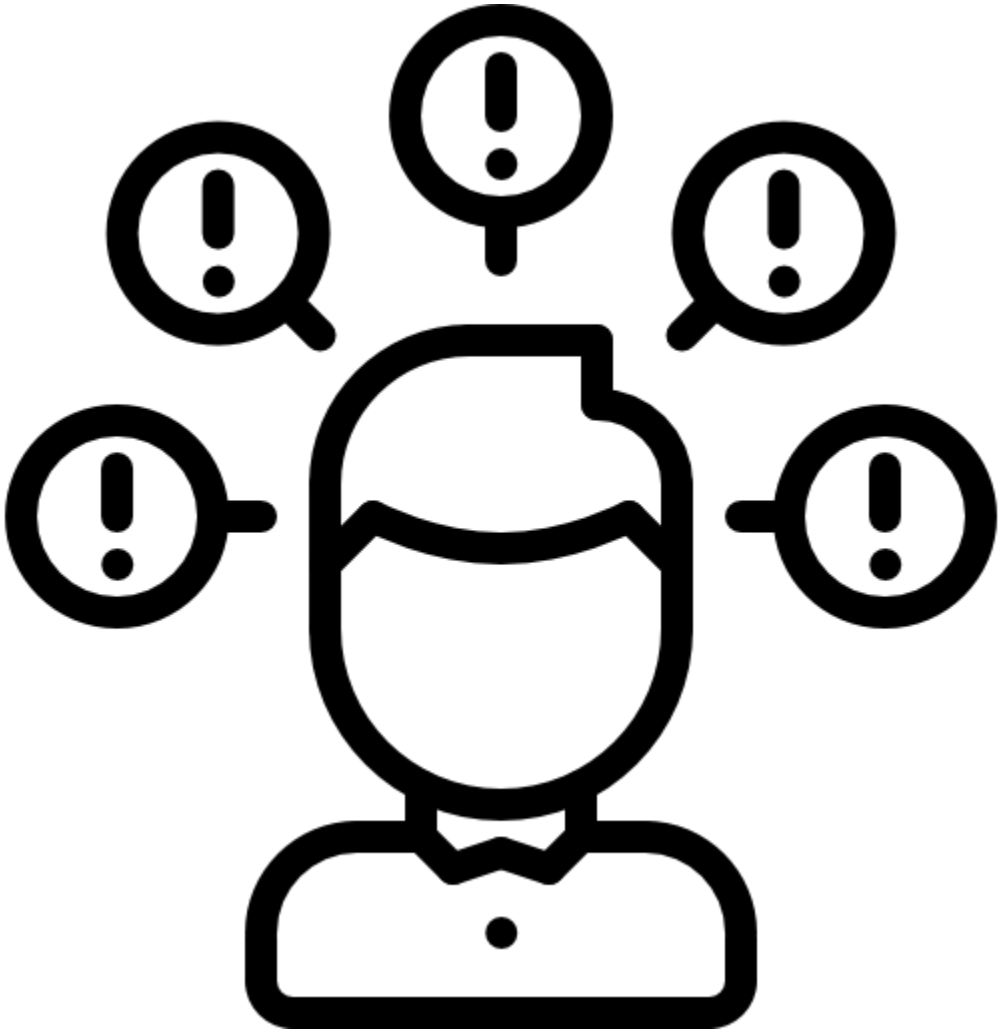
정규성 검정

파일	소스코드
실습환경	**Tf38_cpu** **pip install statsmodels**
소스코드	**print(stats.skew(eps_hat))** **print(stats.kurtosis(eps_hat, fisher=False))**
결과값1	0.660
결과값2	2.437
비고	좌우대칭이면 0, 왼쪽으로 치우치면 0보다 크고, 오른쪽은 0보다 작음 첨도는 분포의 뾰족한 정도를 측정하는 지표 정규분포이면 3, 뾰족하면 3보다 크고, 둥근 정점이면 3보다 작음

7. Others

빅데이터 분석의 해결 과제는?

- 현장의 문제



7. Others

빅데이터 시대, 통계학도 변화한다

- **컴퓨터 활용과 통계학**
  - 통계학은 컴퓨터 덕분에 이론과 활용 면에서 모두 큰 발전을 거듭하여 수학과도 다르고 컴퓨터 과학과도 다른 뚜렷한 지위를 차지하게 되었다.

7. Others

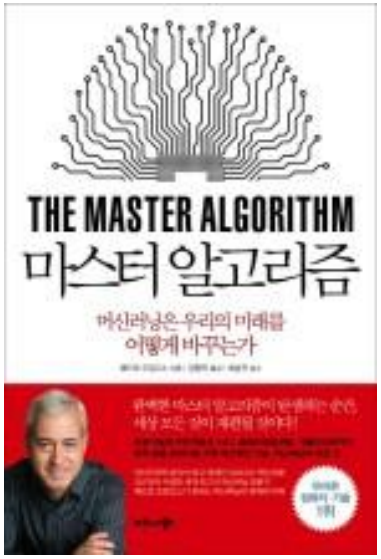
빅데이터의 시대인가, 인공지능의 시대인가

- 이제 공부는 기계가 하는 것?
- 딥러닝을 비롯한 머신러닝이 크게 주목받는 이유는 무엇일까?
- 통계학 가설>추론을 해야 함.
- 머신러닝 지도학습 / 비지도학습
  - 값을 알 수 있음. > 정확하지 않을 수 있어도...

7. Others

빅데이터의 시대인가, 인공지능의 시대인가

- Pedro Domingos - The Master Algorithm



- 기호주의자(Symbolists)는 학습을 연역의 역순으로 보며 철학과 심리학, 논리학에서 아이디어를 얻는다.
- 연결주의자(Connectionists)는 두뇌를 분석하고 모방하며 신경과학과 물리학에서 영감을 얻는다.
- 진화주의자(evolutionaries)는 컴퓨터에서 진화를 모의 시험하며 유전학과 진화생물학에 의존한다.
- 베이즈주의자(Bayesians)는 학습이 확률 추론의 한 형태라고 믿으며 통계학에 뿌리를 둔다.
- 유추주의자(Analogizers)는 유사성(Similarity) 판단을 근거로 추정하면서 배우며 심리학과 수학적 최적화의 영향을 받는다.

7. Others

빅데이터의 시대인가, 인공지능의 시대인가

- 알파고 대 이세돌 혹은 딥마인드 챌린지 매치(Google Deepmind Challenge match)는 2016년 3월 9일부터 15일까지, 하루 한 차례의 대국으로 총 5회에 걸쳐 서울의 포 시즌스 호텔에서 진행된 이세돌과 알파고(영어: AlphaGo) 간의 바둑 대결이다. 최고의 바둑 인공지능 프로그램과 바둑의 최고 중 최고 인간 실력자의 대결로 주목을 받았으며, 최종 결과는 알파고가 4승 1패로 이세돌에게 승리하였다.

- 위키백과

7. Others

빅데이터의 시대인가, 인공지능의 시대인가

- 인간의 뉴런을 카피하다
  - 계산주의
    - 인간의 뇌가 개념과 정보를 기호로 저장한 다음 마치 방정식을 풀듯이 이들을 조작하여 문제를 풀고 사고를 진행
  - 연결주의
    - 인간의 뇌가 뉴런들의 연결에 의해 작동하므로 인공지능 알고리즘 역시 이를 모델로 해야 한다고 보는 생물학적인 관점

7. Others

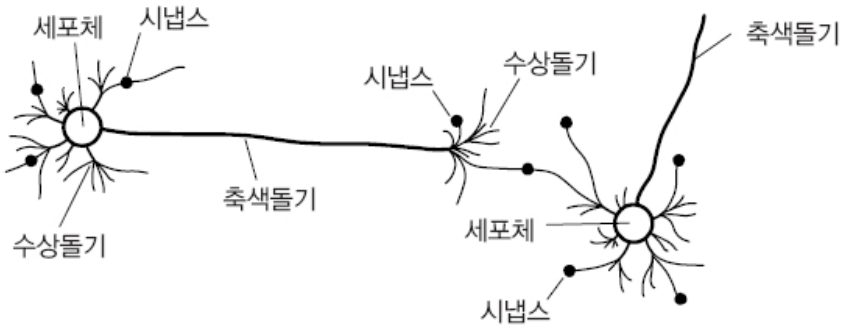
빅데이터의 시대인가, 인공지능의 시대인가

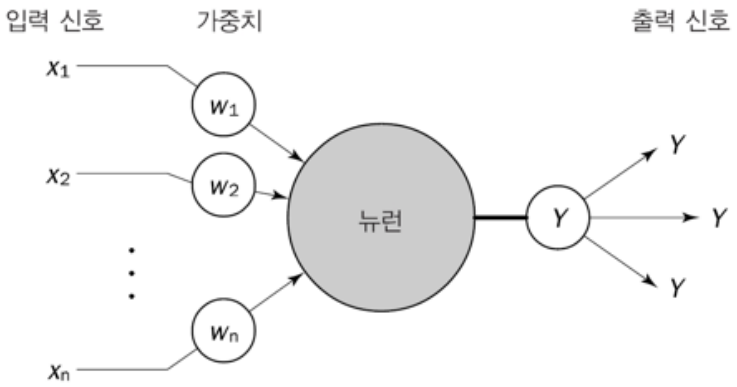
- 인간의 뉴런을 카피하다

  - 인간 뇌를 기반으로 한 추론 모델.
  - 인간 뇌의 추론 모델 - 뉴런(neuron)

생물학적인 신경망	인공 신경망
세포체	뉴런
수상돌기	입력
축색돌기	출력
시냅스	가중치

  - 인간의 뇌 모델링





7. Others

빅데이터의 시대인가, 인공지능의 시대인가

- 머신러닝에게도 선생님이 필요할까?

◦ 지도학습

– 입력 데이터들을 통해 입력변수들의 정보를 받아서 정해진 반응변수에 대한 예측값을 얻는 것

◦ 비지도학습

– 입력 데이터만 있고 미리 정해진 출력변수가 없다

– 데이터 안에 숨어 있는 패턴이나 규칙을 차고자 할 때

8. 실습

실습26 : 회귀 분석-1

문제
[데이터변환축약데이터.xlsx] 업로드
koreanize_matplotlib-0.1.1-py3-none-any.whl, NanumBarunGothic.ttf 업로드한 라이브러리를 설치하고 Matplotlib 한글 사용 환경을 설정 한 다음 나눔체로 한글을 표현해 줘. 해당 데이터로 연봉을 예측하는 모델을 만들어 주세요.

8. 실습

실습27 : 회귀 분석-2

문제
[Advertising_Data.csv] 업로드
koreanize_matplotlib-0.1.1-py3-none-any.whl, NanumBarunGothic.ttf 업로드한 라이브러리를 설치하고 Matplotlib 한글 사용 환경을 설정 한 다음 나눔체로 한글을 표현해 줘. 이 데이터를 탐색해줘.

THANK YOU.

앞으로의 엔지니어는 단순한 '코더'나 '기계 조작자'가 아니라 뇌-기계 인터페이스를 통해 지식과 능력을 즉각 확장하는 존재(뉴로-인터페이스: Neuro Interface)가 될 수 있습니다.

🎯 목표 달성을 위한 여정이 시작됩니다.
✨ 궁금한 점이 있으시면 언제든 문의해주세요!
🚀 함께 코더와 프롬프트 전문가로 성장해 나갑시다!