Statsmodels

최필주

- 널리 사용되는 데이터 분석용 패키지 중 하나

- 주로 활용되는 기능
  - 회귀분석(regression analysis)
  - 시계열분석(time-series analysis)

- Statsmodels의 특징
  - R을 이용한 통계 분석 및 시계열 분석을 파이썬에도 제공
    - R에서 사용하는 formular와 데이터셋 제공

Statsmodels 기본 다루기

Statsmodels 설치하기

- pip 사용

```
In
pip install -U statsmodels
```

- Anaconda 사용

```
In
conda instrall –c conda-forge statsmodels
```

Import하기

```
In
import statsmodels.api as sm
```

- Statsmodels에서 제공되는 데이터셋 가져오기
  - sm.datasets.*item*.load_pandas() 함수 사용
  - DataFrame 접근 형태

```
In
df = sm.datasets.item.load_pandas().data
```

  - sm.dataset?? 확인해보기

예시 – 심장 이식 수술 결과 데이터

- 데이터 정보

In [1]:
```
import statsmodels.api as sm
heart_data = sm.datasets.heart
print(heart_data.SOURCE)
print(heart_data.DESCRLONG)
print(heart_data.NOTE)
```

```
 Miller, R. (1976). Least squares regression with censored dara. Biometrica, 63 (3).
449-464.


This data contains the survival time after receiving a heart transplant, the age of t
he patient and whether or not the survival time was censored.

::

    Number of Observations - 69

    Number of Variables - 3

    Variable name definitions::
        death - Days after surgery until death
        age - age at the time of surgery
        censored - indicates if an observation is censored.  1 is uncensored
```

◉ 예시 – 심장 이식 수술 결과 데이터

- 데이터

In [2]:
```
df = heart_data.load_pandas().data
df.tail()
```

Out [2]:

	survival	censors	age
64	14.0	1.0	40.3
65	167.0	0.0	26.7
66	110.0	0.0	23.7
67	13.0	0.0	28.9
68	1.0	0.0	35.2

R에서 제공되는 데이터셋 가져오기

- sm.datasets.get_rdataset('*item*', package = '*package*') 함수
- 목록
  - http://vincentarelbundock.github.io/Rdatasets/datasets.html
- DataFrame 접근 형태

In
```
data = sm.datasets.get_rdataset('item', package = 'package')
df = data.data
```

예시1 - 타이타닉 데이터

- 데이터 가져오기

In [1]:
```
import statsmodels.api as sm
data = sm.datasets.get_rdataset("Titanic", package="datasets")
df = data.data
df.tail()
```

Out[1]:

	Class	Sex	Age	Survived	Freq
27	Crew	Male	Adult	Yes	192
28	1st	Female	Adult	Yes	140
29	2nd	Female	Adult	Yes	80
30	3rd	Female	Adult	Yes	76
31	Crew	Female	Adult	Yes	20

- 설명을 보고 싶을 경우: __doc__ 참조

In [2]:
```
print(data.__doc__[:1005])
```

예시2 – MASS 패키지의 deaths 데이터 가져오기

- 데이터 가져오기

In [2]:
```
data = sm.datasets.get_rdataset("deaths", "MASS")

df = data.data
df.tail()
```

Out [2]:

	time	value
67	1979.583333	1354
68	1979.666667	1333
69	1979.750000	1492
70	1979.833333	1781
71	1979.916667	1915

- 시계열 데이터
- 1974년부터 1979년까지 영국의 호흡기 질환 사망자수
- 시간표기: 1년 1.0, 1개월 1/12

- 데이터 설명 확인해보기

예시2 – MASS 패키지의 deaths 데이터 가져오기

- 시간 정보 데이터 시각화해서 확인해보기 – 자료형 바꾸기

In [3]:
```
def yearfraction2datetime(yearfraction, startyear=0):
    import datetime
    import dateutil
    year = int(yearfraction) + startyear
    month = int(round(12 * (yearfraction - year)))
    delta = dateutil.relativedelta.relativedelta(months=month)
    date = datetime.datetime(year, 1, 1) + delta
    return date

df["datetime"] = df.time.map(yearfraction2datetime)
df.tail()
```

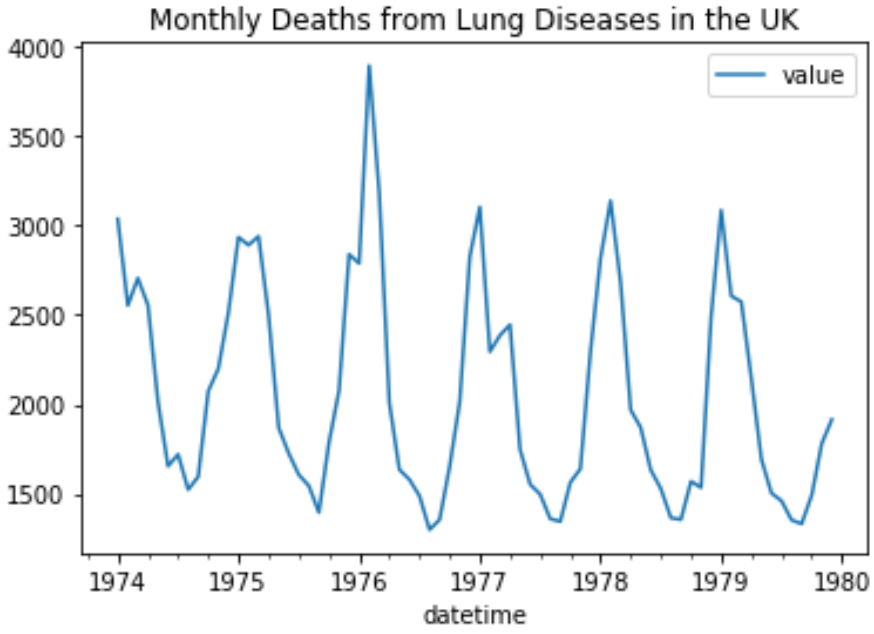
Out [3]:

	time	value	datetime
67	1979.583333	1354	1979-08-01
68	1979.666667	1333	1979-09-01
69	1979.750000	1492	1979-10-01
70	1979.833333	1781	1979-11-01
71	1979.916667	1915	1979-12-01

예시2 – MASS 패키지의 deaths 데이터 가져오기

- 시간 정보 데이터 시각화해서 확인해보기 – 시각화

In [5]:
```
import matplotlib.pyplot as plt
%matplotlib inline
df.plot(x="datetime", y="value")
plt.title(data.title)
plt.show()
```



Statsmodels를 이용한 회귀분석

- Statsmodels을 이용한 선형 회귀분석
  - OLS를 이용한 선형 회귀 분석 수행
    - OLS(Ordinary Least Square): 가장 기본적인 선형 회귀 방법, 잔차제곱합(RSS, Residual Sum of Squares)가 최소값을 갖도록 함

- 실습
  - 실습1: Statesmodels에서 제공되는 유방암 데이터 사용
  - 실습2: R 데이터 셋에서 제공되는 타이타닉 생존자 데이터 사용
  - 실습3: Statesmodels에서 제공되는 심장 이식 수술 데이터 사용

◉ 데이터 가져오기

- 데이터 정보 확인

In [1]:
```
import statsmodels.api as sm
cancer_data = sm.datasets.cancer
print(cancer_data.SOURCE)
print(cancer_data.DESCRLONG)
print(cancer_data.NOTE)
```

```
This is the breast cancer data used in Owen's empirical likelihood.  It is taken from
Rice, J.A. Mathematical Statistics and Data Analysis.
http://www.cengage.com/statistics/discipline_content/dataLibrary.html

The number of breast cancer observances in various counties
::

    Number of observations: 301
    Number of variables: 2
    Variable name definitions:

        cancer - The number of breast cancer observances
        population - The population of the county
```

데이터 가져오기

- X, Y에 데이터 저장하기 – DataFrame 사용

In [2]:
```
df = cancer_data.load_pandas().data
df.tail()
```

Out [2]:

	cancer	population
296	250.0	62931.0
297	267.0	63476.0
298	244.0	66676.0
299	248.0	74005.0
300	360.0	88456.0

In [3]:
```
X = df.population
Y = df.cancer
```

◉ 데이터 가져오기

- X, Y에 데이터 저장하기 – 각각 가져오기

In [4]:
```
data = cancer_data.load()
print(data.exog_name, data.endog_name)
X = cancer_data.load().exog
Y = cancer_data.load().endog
```

```
['population'] cancer
```

- endog (endogenous): 종속변수(y)를 의미
- exog (exogenous): 독립변수(x)를 의미

◉ 데이터 가져오기

- X, Y에 데이터 저장하기 – 각각 가져오기

In [4]:
```
data = cancer_data.load()
print(data.exog_name, data.endog_name)
X = cancer_data.load().exog
Y = cancer_data.load().endog
```

```
['population'] cancer
```

- endog (endogenous): 종속변수(y)를 의미
- exog (exogenous): 독립변수(x)를 의미

◉ 데이터 분석하기

- OLS를 이용한 선형 회귀 분석 - R-style formulas

In [5]:
```
import statsmodels.formula.api as smf

model = smf.ols('cancer ~ population', data = df)
```

- OLS를 이용한 선형 회귀 분석

In [6]:
```
X = sm.add_constant(X, prepend=False)
model = sm.OLS(Y, X)
```

  - X = sm.add_constant(X, prepend = False) 없이 해보기

◉ 분석 결과

In [20]:
```
fit = model.fit()
fit.summary()
```

Out [20]:

OLS Regression Results

Dep. Variable:	cancer	**R-squared:**	0.935
Model:	OLS	**Adj. R-squared:**	0.935
Method:	Least Squares	**F-statistic:**	4315.
Date:	Sun, 04 Aug 2019	**Prob (F-statistic):**	1.04e-179
Time:	16:55:50	**Log-Likelihood:**	-1198.1
No. Observations:	301	**AIC:**	2400.
Df Residuals:	299	**BIC:**	2408.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
population	0.0036	5.45e-05	65.686	0.000	0.003	[illegible]
const	-0.5261	0.969	-0.543	0.588	-2.433	1.381

Omnibus:	67.515	**Durbin-Watson:**	1.822
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	1431.216
Skew:	0.040	**Prob(JB):**	1.64e-311
Kurtosis:	13.682	**Cond. No.**	2.30e+04

각각 확인도 가능

In [9]:
```
fit.pvalues
```

Out [9]:
```
array([1.03663835e-179, 5.87649244e-001])
```

- 분석 결과의 의미 - 적합모델
  - Dep. variable: 종속변수
  - Model: 적합모델
  - Method: 파라미터 적합 방법
  - No. Observations: 사용된 관찰 개수
  - DF Residuals: 잔차의 자유도
    - 관찰 개수에서 파라미터의 수를 뺀 값
  - DF Model: 모델에서 추정하는 파라미터의 개수(상수항은 제외)

Dep. Variable:	cancer
Model:	OLS
Method:	Least Squares
Date:	Sun, 04 Aug 2019
Time:	16:55:50
No. Observations:	301
Df Residuals:	299
Df Model:	1

분석 결과의 의미 – 적합 정도

R-squared:	0.935
Adj. R-squared:	0.935
F-statistic:	4315.
Prob (F-statistic):	1.04e-179
Log-Likelihood:	-1198.1
AIC:	2400.
BIC:	2408.

- 결정계수
  - R-Squared: 분석 결과의 적합도
  - Adj. R-Squared: 파라미터 개수와 관찰 개수에 따라 조정된 값
- F-statistic
  - 간단히 회귀가 단순평균보다 좋은지를 나타냄
  - Prob (F-statistic): 운 좋게 F-statistic에 도달할 확률. 충분히 낮다면 회귀가 단순평균보다 훨씬 더 좋다는 것 확신 가능, 0.05 이하여야 함
- 모델의 평가
  - AIC(Akaike Information Criterion): 관찰 개수와 모델 자체의 복잡성을 기반으로 모델 평가, 낮을 수록 좋음
  - BIC(Bayesian Information Criterion): AIC와 유사

분석 결과의 의미 – 적합 정도

- R-squared의 계산

In [32]:
```
import numpy as np
fitted_values = fit.predict(X)
mean_sum_squared_errors = np.sum((Y-Y.mean())**2)
regr_sum_squared_errors = np.sum((Y-fitted_values)**2)
(mean_sum_squared_errors - regr_sum_squared_errors) / mean_sum_squared_errors
```

Out [32]: 0.9351930401579102

분석 결과의 의미 – 계수 정보

	coef	std err	t	P>|t|	[0.025	0.975]
population	0.0036	5.45e-05	65.686	0.000	0.003	0.004
const	-0.5261	0.969	-0.543	0.588	-2.433	1.381

- coef: 추정계수
- std err: 추정계수의 표준오차
  - 클수록 추정계수의 불확실성 증가
- t: 선형관계(관련성)가 존재하는 정도 (노이즈 대비 시그널)
  - 제곱하면 F-statistic 값이 나옴
  - 크다: 표준 편차가 작다 → 독립-종속 변수간 상관도 높음
  - 작다: 표준 편차가 크다 → 관계 낮음
- P > |t|: t에 대한 P value

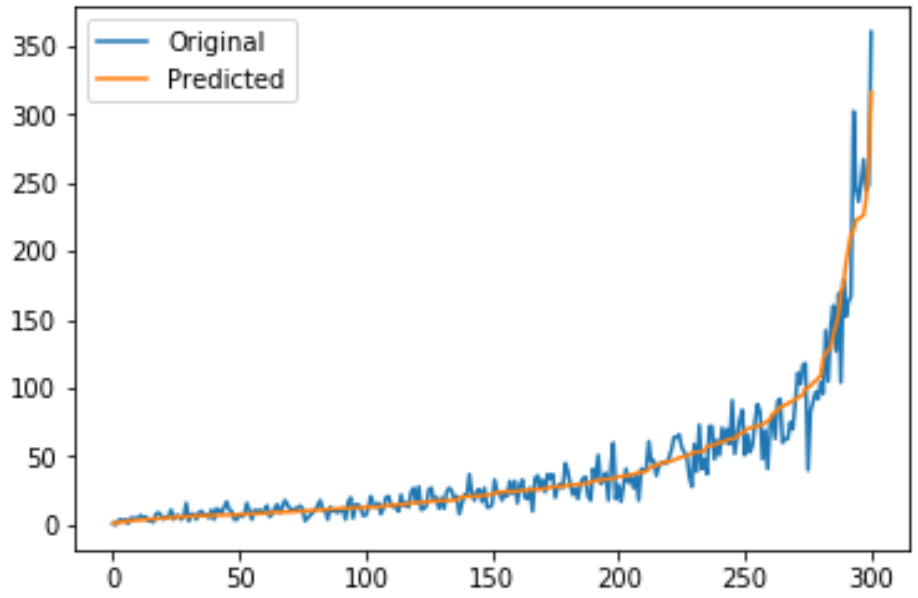
분석 결과의 의미 – 잔차 분석

Omnibus:	67.515	Durbin-Watson:	1.822
Prob(Omnibus):	0.000	Jarque-Bera (JB):	1431.216
Skew:	0.040	Prob(JB):	1.64e-311
Kurtosis:	13.682	Cond. No.	2.30e+04

- Skew(비대칭도)
  - 평균 주위 잔차의 대칭 측정값
  - 0일 수록 대칭
- Kurtosis(첨도)
  - 잔차의 분포 모양
  - 종 모양이면 0, 음수는 평평한 분포, 양수면 뽀족한 분포
- Omnibus: 비대칭도와 첨도를 결합한 통계 테스트
  - Pro(Omnibus): Omnibus를 확률로 변환한 것
- Durbin-Watson
  - 잔차 사이의 상관관계 여부. 시간 기반 데이터 분석과 관련있음
- Jarque-Bera(JB)
  - 비대칭도와 첨도의 또 다른 테스트
- Cond. No
  - 다중공선성에 대한 테스트. 여러 개의 독립변수를 작업할 때 다룸

◉ 분석 결과 시각화

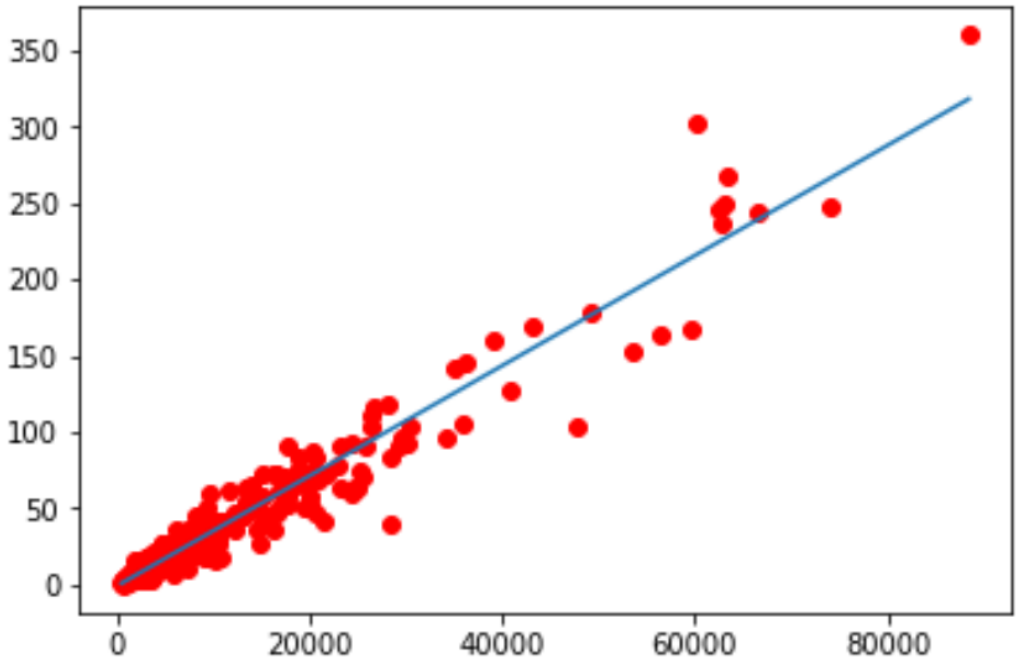
In [44]:
```
import matplotlib.pyplot as plt
%matplotlib inline
plt.plot(Y, label = 'Original')
plt.plot(fitted_values, label = 'Predicted')
plt.legend()
plt.show()
```



◉ 분석 결과 시각화 (p.24의 coef 값 사용)

In [7]:
```
import matplotlib.pyplot as plt
%matplotlib inline
plt.scatter(X.population, Y, label = 'Original', color = 'red')
plt.plot(X.population, X.population * 0.0036 - 0.5261, label = 'Predict')
```

Out[7]: [<matplotlib.lines.Line2D at 0x15767170>]



데이터 가져오기

- 데이터 정보 확인

In [1]:
```
import pandas as pd
import statsmodels.api as sm
titanic_data = sm.datasets.get_rdataset("Titanic", package="datasets")
```

In [2]:
```
print(titanic_data.__doc__[:1005])
```

```
+----+----------+--------------------+
| No | Name     | Levels             |
+----+----------+--------------------+
| 1  | Class    | 1st, 2nd, 3rd, Crew |
+----+----------+--------------------+
| 2  | Sex      | Male, Female       |
+----+----------+--------------------+
| 3  | Age      | Child, Adult       |
+----+----------+--------------------+
| 4  | Survived | No, Yes            |
+----+----------+--------------------+
```

데이터 가져오기

- 데이터 저장하기 – DataFrame 사용

In [3]:
```
df = titanic_data.data
df.tail()
```

Out[3]:

	Class	Sex	Age	Survived	Freq
27	Crew	Male	Adult	Yes	192
28	1st	Female	Adult	Yes	140
29	2nd	Female	Adult	Yes	80
30	3rd	Female	Adult	Yes	76
31	Crew	Female	Adult	Yes	20

◉ 데이터 가져오기

- X, Y에 데이터 저장하기

In [13]:
```
survived_mapping = {"No": 0, "Yes":1}
class_mapping = {"1st": 3, "2nd": 2, "3rd": 1, "Crew":0}
sex_mapping = {"Male": 1, "Female": 0}
age_mapping = {"Adult": 1, "Child": 0}

y = df['Survived'].map(survived_mapping)
X = pd.DataFrame({
    "Class": df['Class'].map(class_mapping),
    "Sex": df['Sex'].map(sex_mapping),
    "Age": df['Age'].map(age_mapping)})
X = sm.add_constant(X)
X.tail()
```

Out[13]:

	const	Class	Sex	Age
27	1.0	0	1	1
28	1.0	3	0	1
29	1.0	2	0	1
30	1.0	1	0	1
31	1.0	0	0	1

◉ 분석 결과

In [15]:
```
fit = sm.OLS(y, X).fit()
fit.summary()
```

Out [15]:

OLS Regression Results

Dep. Variable:	Survived	**R-squared:**	0.000
Model:	OLS	**Adj. R-squared:**	-0.107
Method:	Least Squares	**F-statistic:**	0.000
Date:	Sun, 04 Aug 2019	**Prob (F-statistic):**	1.00
Time:	21:01:40	**Log-Likelihood:**	-23.225
No. Observations:	32	**AIC:**	54.45
Df Residuals:	28	**BIC:**	60.31
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	0.5000	0.207	2.415	0.023	0.076	0.924
Class	-3.261e-16	0.085	-3.86e-15	1.000	-0.173	0.173
Sex	2.429e-16	0.189	1.29e-15	1.000	-0.387	0.387
Age	2.498e-16	0.189	1.32e-15	1.000	-0.387	0.387

Omnibus:	505.225	**Durbin-Watson:**	0.125
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	5.333
Skew:	-0.000	**Prob(JB):**	0.0695
Kurtosis:	1.000	**Cond. No.**	5.84

분석과정

- 데이터 가져오기: statsmodels에서 제공되는 데이터 사용
  - sm.datasets, load_pandas()
- X, Y 설정
  - sm.add_constant
- 모델 적용 및 분석 결과 확인
  - sm.OLS(Y, X).fit(), summary()