Introduction to Big Data

- Developed by Dr. Keungoui KIM
- https://awekim.github.io/portfolio/

Lecture 10. Regression

```
from google.colab import drive
drive.mount('/content/drive')
```

```
import pandas as pd
import seaborn as sns
```

```
import statsmodels.api as sm

from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error
```

Regression with Boston Housing Data

```
import pandas as pd
import numpy as np
import seaborn as sns

from statsmodels.formula.api import ols

#from sklearn.datasets import load_boston # Boston Housing Data
```

```
housing_df = pd.read_csv('/content/drive/MyDrive/[Lecture]/IntBigData/BigData_Python/10_Regression/Housi
housing_df
```

CRIM - per capita crime rate by town

ZN - proportion of residential land zoned for lots over 25,000 sq.ft.

INDUS - proportion of non-retail business acres per town.

CHAS - Charles River dummy variable (1 if tract bounds river; 0 otherwise)

NOX - nitric oxides concentration (parts per 10 million)

RM - average number of rooms per dwelling

AGE - proportion of owner-occupied units built prior to 1940

DIS - weighted distances to five Boston employment centres

RAD - index of accessibility to radial highways

TAX - full-value property-tax rate per $10,000

PTRATIO - pupil-teacher ratio by town

B - 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town

LSTAT - % lower status of the population

MEDV - Median value of owner-occupied homes in $1000's

```
# housing_x_df = pd.DataFrame(housing.data, columns=housing.feature_names)
# housing_x_df = pd.DataFrame(housing['data'],
#                             columns=housing['feature_names'])
```

```
# housing_y_df = pd.DataFrame(housing.target, columns=['MEDV'])
# housing_y_df = pd.DataFrame(housing['target'],
#                             columns=['MEDV'])
```

```
# Combining two tables
# housing_df = pd.concat([housing_x_df, housing_y_df],
#                        axis=1)
housing_df.head()
```

```
# check
import seaborn as sns

sns.pairplot(housing_df[['MEDV','CRIM','LSTAT']])
```

Simple Regression

```
# Regression with statsmodels' sm.OLS
statsOLSModel = ols('MEDV ~ CRIM',
                    data=housing_df)
statsOLSModel
```

```
statsOLSModel_res = statsOLSModel.fit()
statsOLSModel_res
```

```
print(statsOLSModel_res.summary())
```

```
statsOLSModel_res.params
```

```
statsOLSModel_res.predict()[:6,]
```

```
statsOLSModel_res.fittedvalues
```

```
statsOLSModel_res.resid
```

Checking residual

```
statsOLSModel_res.params
```

```
# Checking residual

beta0_hat = statsOLSModel_res.params[0]
beta1_hat = statsOLSModel_res.params[1]
```

```
MEDV_hat = beta0_hat + beta1_hat * housing_df.CRIM
MEDV_hat
```

```
housing_df.MEDV - MEDV_hat
```

```
residual = statsOLSModel_res.resid
residual
```

Checking R-squared

```
housing_df.MEDV
```

```

```

```
housing_df.MEDV.mean()
```

```
# import scipy as sp
# y_mu = sp.mean(housing_df.MEDV)

y_mu = housing_df.MEDV.mean(axis=0)
y_mu
```

```
y = housing_df.MEDV
```

```
y_hat = statsOLSModel_res.predict()
```

```
# 1 - sum(res.resid**2) / sp.sum((y-y_mu)**2)

1 - sum(statsOLSModel_res.resid**2) / sum((y-y_mu)**2)
```

```
statsOLSModel_res.rsquared
```

Visualization

```
sns.lmplot(x='CRIM',y='MEDV', data=housing_df,
           scatter_kws = {'color':'red'},
           line_kws={'color':'black'})
```

Multiple Regression

```
### Multiple regression
statsOLSModel_all = ols('MEDV ~ CRIM+ZN+INDUS+CHAS+NOX+RM+AGE+DIS+RAD+TAX+PTRATIO+B+LSTAT',
                        data=housing_df)
statsOLSModel_all
```

```
statsOLSModel_all_res = statsOLSModel_all.fit()
statsOLSModel_all_res
```

```
print(statsOLSModel_all_res.summary())
```

Multicolliniearity issue

```
housing_df.corr()
```

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

```
statsOLSModel_all.exog
```

```
variance_inflation_factor(statsOLSModel_all.exog,1)
```

```
enumerate(statsOLSModel_all.exog_names)
```

```
all_vif_df = pd.DataFrame({'variable': column,
                           'VIF': variance_inflation_factor(statsOLSModel_all.exog, i)}
                          for i, column in enumerate(statsOLSModel_all.exog_names)
                          if column != 'Intercept')
all_vif_df
```

```
all_vif_df.mean()
```

```
model_vif = ols('MEDV ~ CRIM+ZN+INDUS+CHAS+NOX+RM+AGE+DIS+PTRATIO+B+LSTAT', data=housing_df)

vif_df = pd.DataFrame({'variable': column,
                       'VIF': variance_inflation_factor(model_vif.exog, i)}
                      for i, column in enumerate(model_vif.exog_names)
                      if column != 'Intercept')
vif_df
```

```
vif_df.mean()
```

```
print(model_vif.fit().summary())
```

Normality issue

```
res_all_fix = ols('MEDV ~ CRIM+ZN+INDUS+CHAS+NOX+RM+AGE+DIS+PTRATIO+B+LSTAT',
                  data=housing_df).fit()
```

```
import statsmodels.api as sm
sm.qqplot(res_all_fix.resid, line = 's')
```

```
from scipy.stats import jarque_bera

jarque_bera(res_all.resid)
```

```
from scipy.stats import kstest

kstest(res_all.resid, cdf='norm')
```

```
from scipy.stats import shapiro

shapiro(res_all.resid)
```