# Normality and

# Homoscedasticity  Assumptions

# Influential Observations

# Contents

# Normality and Homoscedasticity

- The errors in Multiple Linear Regression are assumed to follow Normal Distribution.

- If Normality of Errors is not true then statistical tests and associated P values based on F and t distribution are not reliable.

- **Homoscedasticity** describes a situation in which variance of error term is same across all values of the independent variables.

- In the absence of Homoscedasticity ( Or presence of Heteroscedasticity) the standard errors of parameter estimates are incorrect.

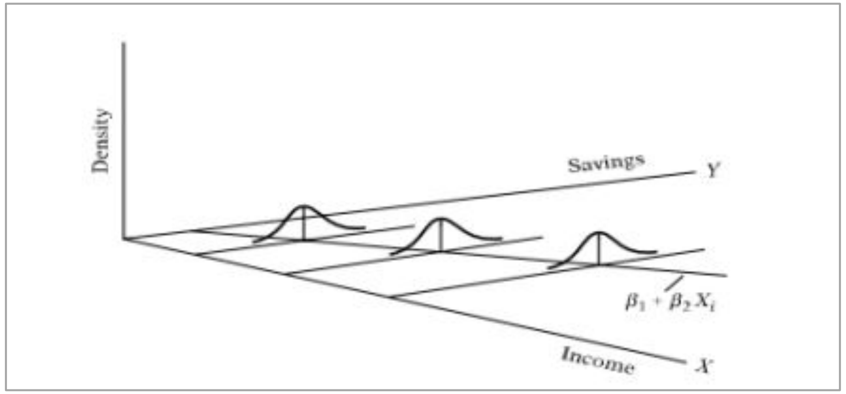
# Assumption of Homoscedasticity

- Variance of error term must be constant across the independent variables (defined by X values )

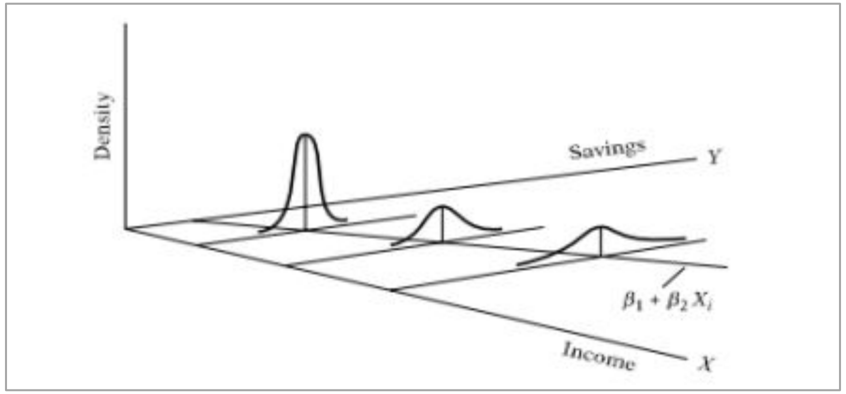
$$V\left({}^{e_i}/_{x_i}\right) = \sigma^2 \;\; \textit{indicates homoscedasticity}$$

$$V\left({}^{e_i}/_{x_i}\right) = \sigma_i^2 \;\; \textit{indicates heteroscedasticity}$$

Homoscedastic Errors

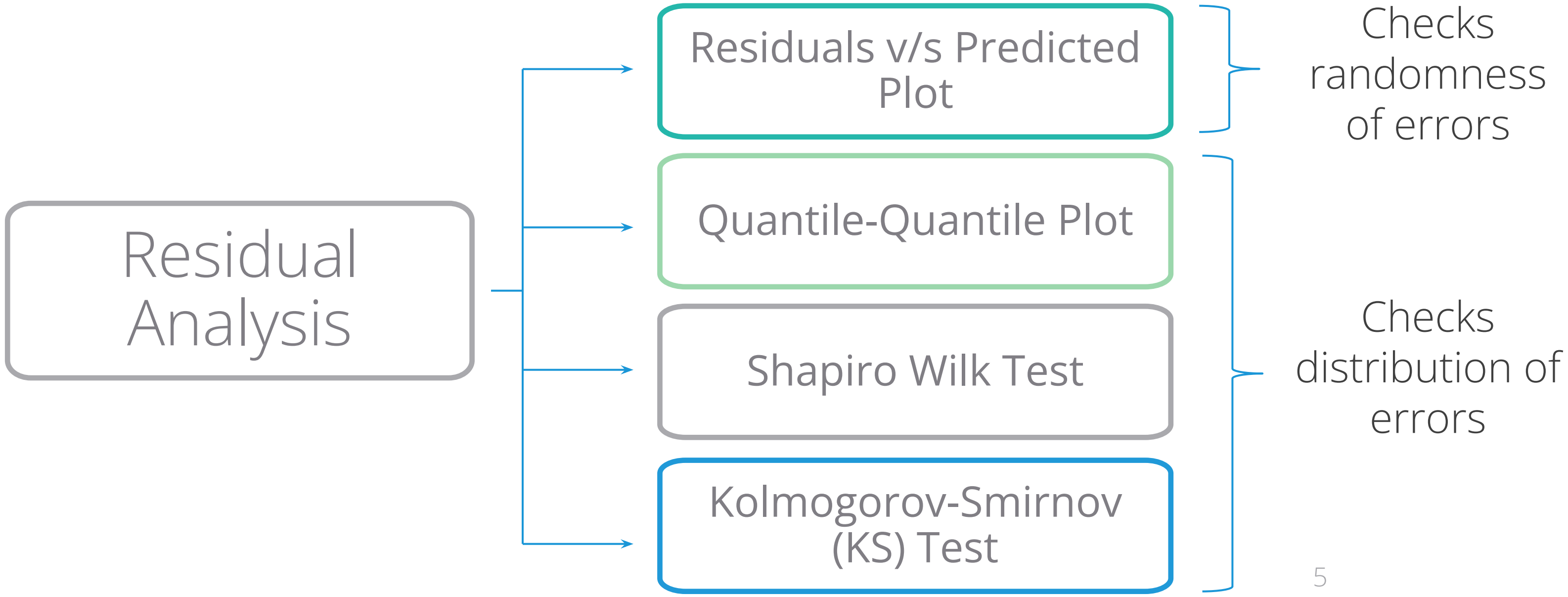



Heteroscedastic Errors



# Residual Analysis

Observed Value – Predicted value = Residual



# Residual Analysis for Performance Index Data

Continuing with the "**Performance Index** " data,

- **Model** job performance index ( **jpi** ) based on aptitude score ( **aptitude** ), test of language ( **tol** ), technical knowledge ( **technical** ) and general information ( **general** )
- Get fitted values and residuals.
- Analyse the distribution of residuals

# Residual v/s Predicted Plot in Python

```
#Importing the Data, Fitting Linear Model and Calculating Fitted
Values and Residuals
```

```
import pandas as pd
perindex= pd.read_csv("Performance Index.csv")

import statsmodels.formula.api as smf
jpimodel = smf.ols('jpi ~ tol + aptitude + technical +general',
data=perindex).fit()

perindex = perindex.assign(pred=pd.Series(jpimodel.fittedvalues))
perindex = perindex.assign(res=pd.Series(jpimodel.resid))
```

- *ols() fits a linear regression.*
- *fittedvalues() and resid() fetch fitted values and residuals*

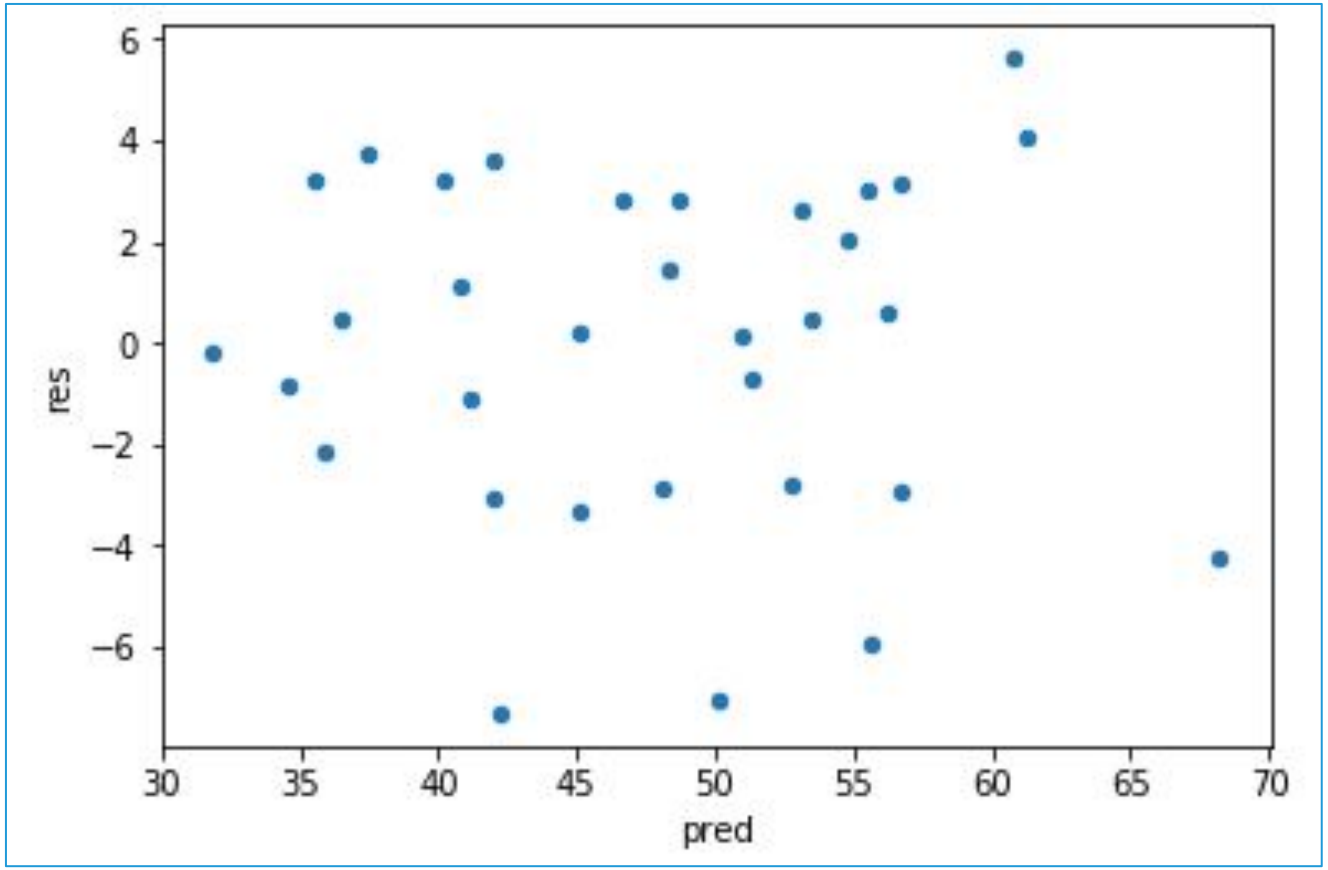
```
#Residuals v/s Predicted Plot
```

```
perindex.plot.scatter(x='pred', y='res')
```

*.plot.scatter() is used to obtain scatter plot of predicted values against residuals.*

# Residual v/s Predicted Plot in Python

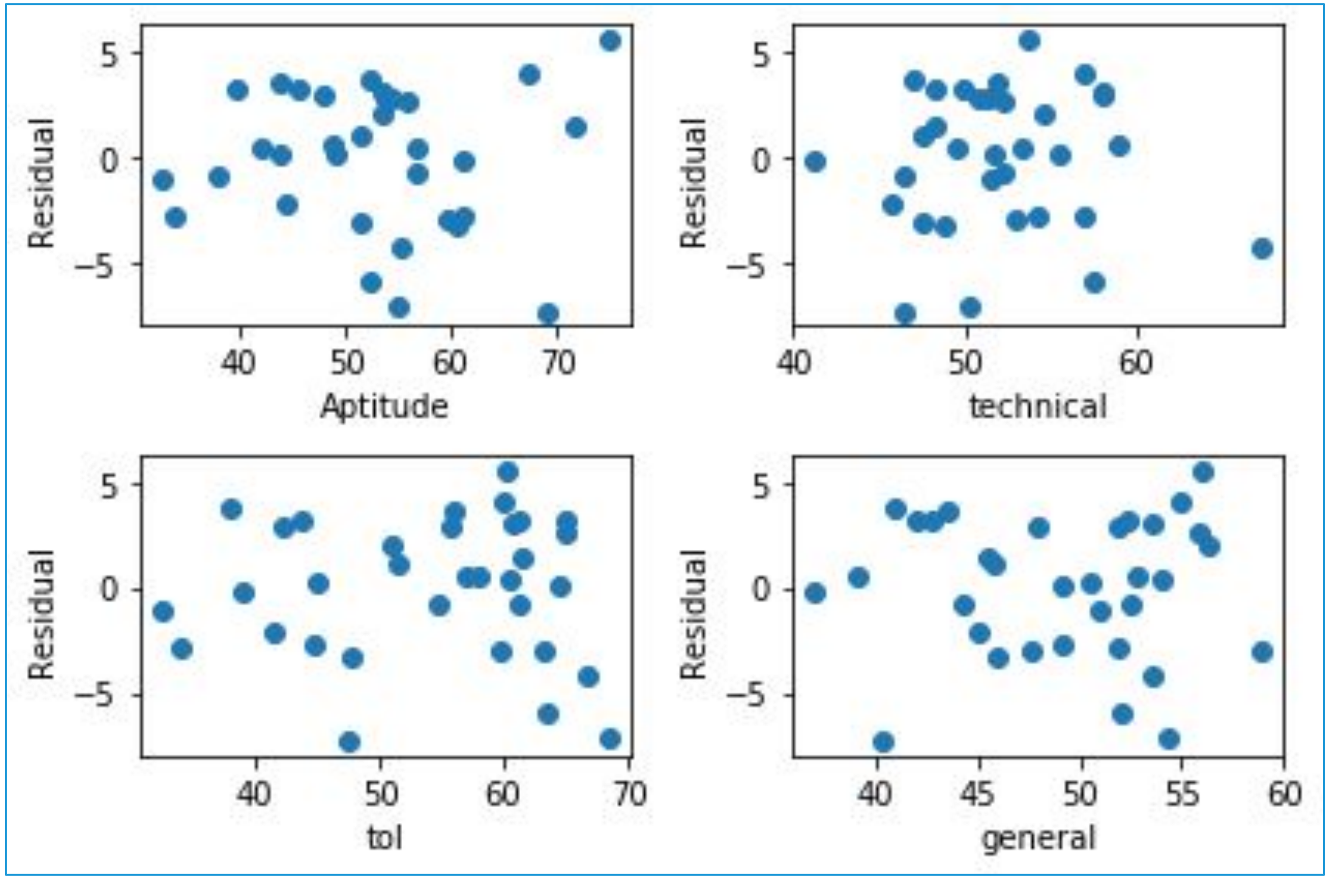
```
# Output
```




*Interpretation:*

- *Residuals in our model are randomly distributed which indicates presence of Homoscedasticity*

# Residual v/s Independent variables Plot in Python




*Interpretation:*

- *Residuals in our model are randomly distributed which indicates presence of Homoscedasticity*

# QQ Plot

- **The Quantile-Quantile (QQ) Plot** is a powerful graphical tool for assessing normality.
- Quantiles are calculated using sample data and plotted against expected quantiles under Normal distribution.

High Correlation between Sample Quantiles and Theoretical Quantiles → Normalit y

- If the data are truly sampled from a Gaussian (Normal) distribution, the **QQ plot will be linear.**

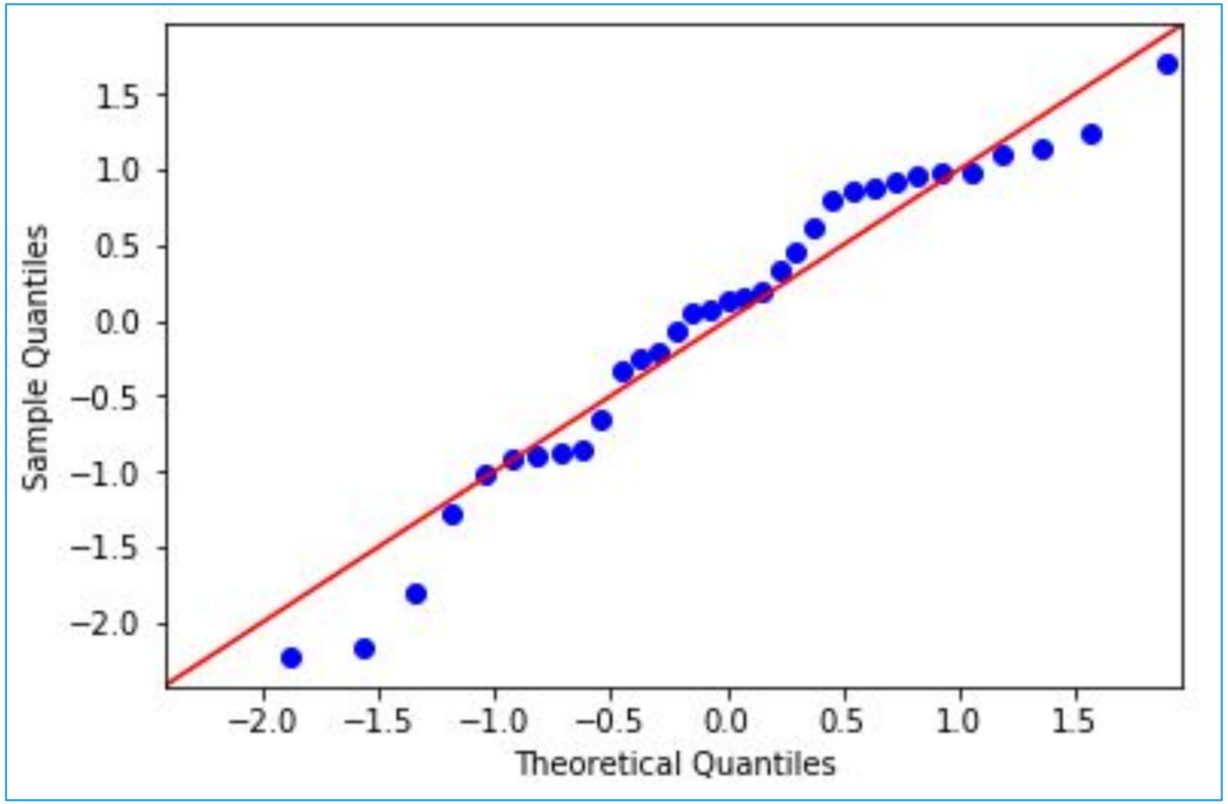
# QQ Plot in Python

```
#QQ Plot
import statsmodels.api as sm
fig = sm.graphics.qqplot(perindex.res, line='45', fit=True)
```

- *qqplot() produces a plot with theoretical quantiles on x axis against the sample quantiles on y axis. Column for which normality is being tested is specified in the first argument.*
- *line= is an argument that adds reference line to the qqplot. Here it adds a 45-degree line*
- *fit=True indicates, parameters are fit using the distribution's fit() method.*

# QQ Plot in Python

```
# Output
```




*Interpretation:*

- *Most of these points are close to the line except few values indicating no serious deviation from Normality.*

# Shapiro Wilk Test

| | |
|---|---|
| Objective | To **correlate**, sample ordered values with expected Normal scores in order **to test normality of the sample** |

Null Hypothesis ($H_0$): Sample is drawn from Normal Population

Alternate Hypothesis ($H_1$): Not $H_0$

| Test Statistic | |
|---|---|
| Decision Criteria | Reject the null hypothesis if p-value < 0.05 |

# Shapiro Wilk Test in Python

```
# Shapiro Wilk Test
import scipy as sp
sp.stats.shapiro(perindex.res)
```

*shapiro() from scipy package, returns correlation coefficient w and p-value.*

```
# Output
(0.9498621821403503, 0.1318102478981018)
```

*Interpretation:*
*p-value>0.05, Do not reject $H_0$. Normality can be assumed.*

# Absence of Normality – Remedial Measure

Mathematical Transformation of the dependent variable is used as a remedial measure in case of serious departure from Normality.

Typically Log Transformation is used. However, there is general transformation called as Box Cox Transformation given as :

- Box Cox transformation

$$Y^* = \frac{Y^\lambda - 1}{\lambda} \qquad \lambda \neq 0$$
$$= \log Y \qquad \lambda = 0$$

Where Y is the response variable

- R can automatically detect the optimum λ using **`boxcox()`** in package MASS

# Influential Observation

- An **influential observation** is an observation whose deletion from the dataset would noticeably change the result of the calculation.

- In particular, in regression analysis an influential point is one whose deletion has a large effect on the parameter estimates.

# Cook's Distance Method

Cook's distance **measures the effect of deleting a given observation.**

Let Di be the Cook's distance for observation i.

$$D_i = \frac{\sum_{j=1}^{n} \left(\hat{Y}_j - \hat{Y}_{j(i)}\right)^2}{p \text{ MSE}}$$

$\hat{Y}_j$ = prediction from the full regression model for observation j

$\hat{Y}_{j(i)}$ = prediction of $j^{th}$ observation from a refitted model after removing $i^{th}$ observation

$MSE$ = mean square error of the regression model

p = number of fitted parameters in the model

Cut off to indicate influential observation,

- Simple operational guideline $D_i > 1$
- Alternative $D_i > 4/n$, where n is the number of observations

* **It is recommended to check model performance by excluding highly influential observation**

# DFBETAs

**DFBETA Statistics**

**DFBETA measures the difference** in each parameter estimate with and without a specific observation. There is a DFBETA for each data point and for each parameter estimate.

**Large values of DFBETAs**

Indicate

**Observations are influential** in estimating a given parameter

Cut off to indicate influential observation,

- general cut off value recommended is **2**
- size adjusted cut off is taken to be **$2/\sqrt{n}$**

# Finding Influential Observations in Python

#Importing the Data

```
import pandas as pd
perindex=pd.read_csv("Performance Index.csv")

import statsmodels.formula.api as smf
jpimodel=smf.ols('jpi ~ aptitude + tol + technical +general',
data=perindex).fit()
```

#Finding Influential Observations

```
influence = jpimodel.get_influence()
influence.summary_frame()
```

- *Influence is an object calling the method get_influence() which in turn allows us to call various measures of influence.*
- *summary_frame() calls a dataframe of 6 influence measures - Cook's Distance, Standardized residuals, dffits, dfbetas among others.*

# Finding Influential Observations in Python

```
# Output
```

| | dfb_Intercept | dfb_aptitude | dfb_tol | dfb_technical | dfb_general | cooks_d | standard_resid | hat_diag | dffits_internal | student_resid | dffits |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.122740655 | -0.148903135 | 0.12930015 | 0.111295503 | -0.231928116 | 0.027053557 | 1.070156154 | 0.105636531 | 0.367787693 | 1.073046027 | 0.368780874 |
| 1 | -0.06974523 | 0.116652494 | 0.133587819 | 0.02982789 | -0.09548941 | 0.010912673 | -0.364255404 | 0.291400351 | -0.233588024 | -0.358542217 | -0.229924297 |
| 2 | 0.007297141 | -0.008657864 | 0.004774056 | 0.014949118 | -0.026376846 | 0.000463024 | -0.206575027 | 0.051460374 | -0.048115671 | -0.203007405 | -0.047284697 |
| 3 | -0.156960588 | 0.094733386 | -0.173852663 | 0.214043541 | -0.081100685 | 0.018150707 | -0.899495717 | 0.100854509 | -0.301253273 | -0.896332474 | -0.30019386 |
| 4 | 0.053857876 | -0.010940236 | 0.00867167 | -0.040889003 | 0.003662146 | 0.001250165 | 0.323269275 | 0.056438877 | 0.079062145 | 0.31803818 | 0.077782773 |
| 5 | 0.244563668 | -0.167648781 | -0.058829828 | 0.006424002 | -0.120351526 | 0.0247544 | 0.95727578 | 0.118994501 | 0.351812453 | 0.9557968 | 0.351268907 |
| 6 | -0.02187633 | -0.017309924 | 0.012271264 | 0.006428915 | 0.013869655 | 0.000302058 | -0.06455514 | 0.266006198 | -0.03886248 | -0.063396607 | -0.038165038 |
| 7 | -0.168200899 | 0.013234244 | 0.04743142 | 0.175275244 | -0.069100347 | 0.01545173 | 0.918453917 | 0.083902352 | 0.277954402 | 0.915804589 | 0.277152627 |
| 8 | -0.00893895 | 0.000748261 | 0.010029123 | -0.012775933 | 0.020594356 | 0.000259695 | 0.13315947 | 0.068233348 | 0.03603436 | 0.130801426 | 0.035396249 |
| 9 | 0.112048775 | -0.224748965 | 0.240141498 | -0.17548137 | 0.078322088 | 0.023475703 | -0.842281862 | 0.141964282 | -0.342605483 | -0.837785996 | -0.340776751 |
| 10 | -0.180550626 | 0.079534465 | 0.074017536 | 0.120829488 | -0.057492908 | 0.01106923 | -0.644822905 | 0.117472155 | -0.235257629 | -0.63795804 | -0.232753046 |
| 11 | -0.340527019 | 0.322925945 | -0.062977416 | 0.148776604 | 0.044546954 | 0.044366369 | 1.221464622 | 0.129438005 | 0.470990281 | 1.232747401 | 0.475340861 |
| 12 | -0.002789222 | 0.152126452 | 0.050719785 | -0.020241173 | -0.060819239 | 0.009463569 | 0.461241654 | 0.181948526 | 0.217526652 | 0.454660862 | 0.214423078 |
| 13 | 0.035353405 | -0.027136734 | 0.033243364 | 0.016690271 | -0.055540496 | 0.001256856 | 0.156009829 | 0.20521207 | 0.079273454 | 0.153265238 | 0.077878842 |
| 14 | -0.060996651 | 1.62E-05 | -0.079422031 | -0.034363374 | 0.147870676 | 0.008619132 | 0.603984002 | 0.105654499 | 0.207594942 | 0.597002274 | 0.205195257 |
| 15 | -0.026013807 | -0.056282646 | 0.145739562 | -0.191264061 | 0.222520013 | 0.020491421 | 0.791281772 | 0.140624982 | 0.320089214 | 0.78585952 | 0.317895805 |
| 16 | 0.005763587 | 0.05225852 | -0.223037193 | -0.045632185 | 0.148853441 | 0.018578284 | 0.85294471 | 0.113226147 | 0.30478094 | 0.848673035 | 0.303254552 |
| 17 | -0.470812698 | 0.716128082 | -0.106108271 | -0.101654484 | 0.324161247 | 0.172591001 | 1.78156995 | 0.213764291 | 0.928953715 | 1.8579384 | 0.968774074 |
| 18 | -0.002557978 | -0.004065261 | 0.008450592 | 0.005075285 | -0.005409883 | 4.13E-05 | 0.044548271 | 0.094144224 | 0.014361446 | 0.043747084 | 0.01410316 |
| 19 | -0.052126913 | -0.182788497 | 0.123094282 | 7.18E-05 | 0.058590983 | 0.01661995 | -0.969160992 | 0.081281261 | -0.288270271 | -0.968072934 | -0.287946635 |
| 20 | -0.097589025 | 0.108404591 | -0.09966008 | 0.077516309 | -0.005991789 | 0.004816919 | -0.272865 | 0.244414631 | -0.155192119 | -0.268305076 | -0.152598659 |

*Interpretation*
*One can use the threshold of 4/n where n is sample size and check cases with Cook's distance greater than the threshold.*

# Finding Influential Observations in Python

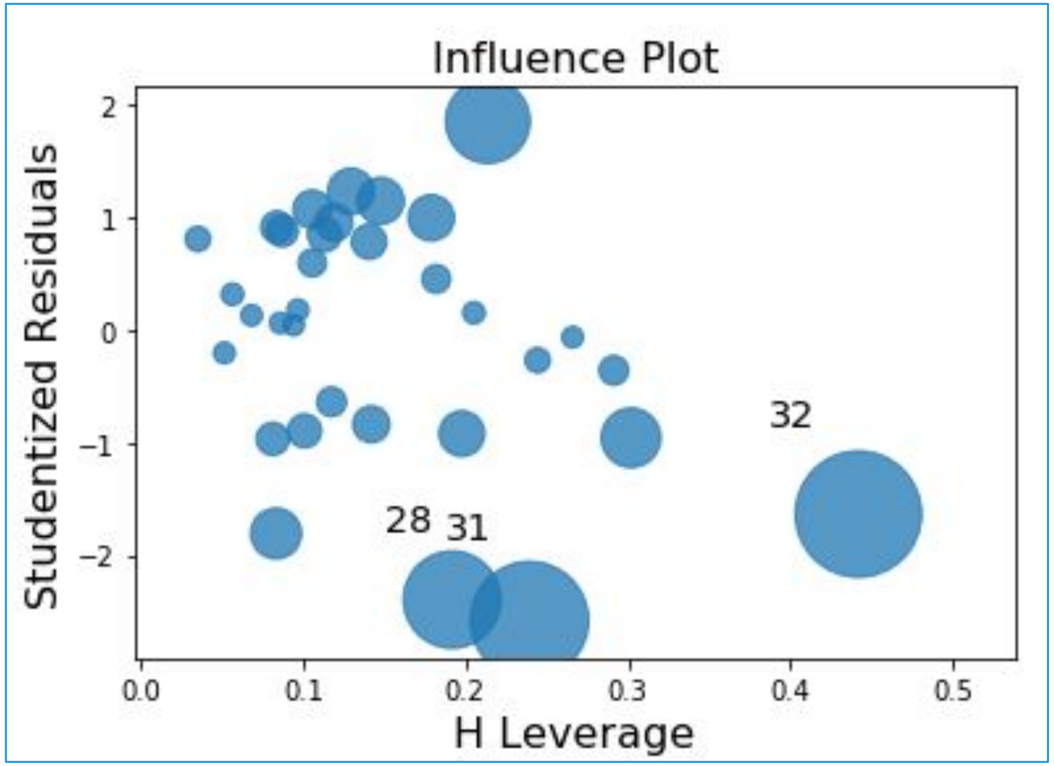
```
#Influence Plot
```

```
from statsmodels.graphics.regressionplots import *
influence_plot(jpimodel, criterion = 'Cooks')
```

- *influence_plot() creates a "bubble" plot of Studentized residuals by hat values, with the areas of the circles representing the observations proportional to criteria specified (in this case Cook's distance).*

# Influence Plot in Python

```
# Output
```




*Interpretation:*
*The data points 28,31 and 32 are detected as influential observations.*

# Quick Recap

| | |
|---|---|
| Normality Assumption | • Error terms should be normally distributed |
| Homoscedasticity | • Errors should have constant variance across X values |
| Residual v/s Predicted Plot | • Ideally, residuals should be randomly distributed |
| Residual v/s Independent variables Plot | • Ideally,residuals should be randomly distributed |
| QQ Plot | • Used to check if errors follow Normal distribution |
| Shapiro Wilk Test | • Test for Normality assessment of errors |

# Quick Recap

| | |
|---|---|
| Box Cox Transformation | • Transforming non normal response to normal |
| Influential Observations in Python | • **`get_influence()`** produces object giving influential observations by different measures<br>• **`Influence_plot()`** creates a “bubble” plot of Studentized residuals by hat values, with the areas of the circles representing the observations proportional to Cook’s distances |