MULTIPLE LINEAR REGRESSION USING PYTHON

DATA SCIENCE
INSTITUTE

Multiple Linear Regression: Recap

- Multiple linear regression is used to explain the relationship between one continuous dependent variable and two or more independent variables.
- The independent variables can be continuous or categorical.
- Multiple Linear Regression is used when we want to predict the value of a variable based on the values of two or more other variables.
- The variable we want to predict is called the dependent variable
- The variables used to predict the value of dependent variable are called independent variables (or explanatory variables/predictors).
- Multiple linear regression requires the model to be linear in the parameters.

- Example: The price house in USD can be dependent variable and area of house, location of house , air quality index in the area, distance from airport etc. can be independent variables.

Statistical Model

$$Y = b_0 + b_1 X_1 + b_2 X_2 + \ldots + b_p X_p + e$$

where,

Y	:	Dependent Variable
$X_1, X_2, \ldots, X_p$	:	Independent Variables
$b_0, b_1, \ldots, b_p$	:	Parameters of Model
e	:	Random Error Component

- Independent variables can either be **Continuous or Categorical**
- Multiple linear regression **requires the model to be linear in the parameters**
- Parameters of the model are estimated by Least Square Method.
- The **least squares (LS)** criterion states that the **sum of the squares of errors** (or residuals) **is minimum**.
- Mathematically, following quantity is minimized to estimate parameters using least square method.
- **Error ss=** $\Sigma (Y_i - \hat{Y}_i)^2$

DATA SCIENCE INSTITUTE

Case Study – Modeling Job Performance Index

Background

- A company conducts different written tests before recruiting employees. The company wishes to see if the scores of these tests have any relation with post-recruitment performance of those employees.

Objective

- To predict employees' job performance index after probationary period, based on scores of tests conducted at the time of recruitment

Available Information

- Sample size is 33
- Independent Variables: Scores of tests conducted before recruitment on the basis of four criteria – Aptitude, Test of Language, Technical Knowledge, General Information
- Dependent Variable: Job Performance Index calculated after an employee finishes probationary period (6 months)

DATA SCIENCE INSTITUTE

Data Snapshot

Performance Index

Dependent Variable

4 Independent Variables

empid	jpi	aptitude	tol	technical	general
1	45.52	43.83	55.92	51.82	43.58

Columns	Description	Type	Measurement	Possible values
empid	Employee ID	integer	-	-
jpi	Job performance Index	numeric	-	positive values
aptitude	Aptitude score	numeric	-	positive values
tol	Test of Language	numeric	-	positive values
technical	Technical Knowledge	numeric	-	positive values
general	General Information	numeric	-	positive values

DATA SCIENCE INSTITUTE

Graphical Representation of Data

- It is always recommended to have a general look at your data and behavior of all the variables before moving to modeling.
- This helps you in making intuitive inferences about the data, which can be statistically validated by your final model.
- The simplest way of doing this is creating a scatter plot matrix, which will give bivariate relationships between variables.

```
#Importing the Data
```

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

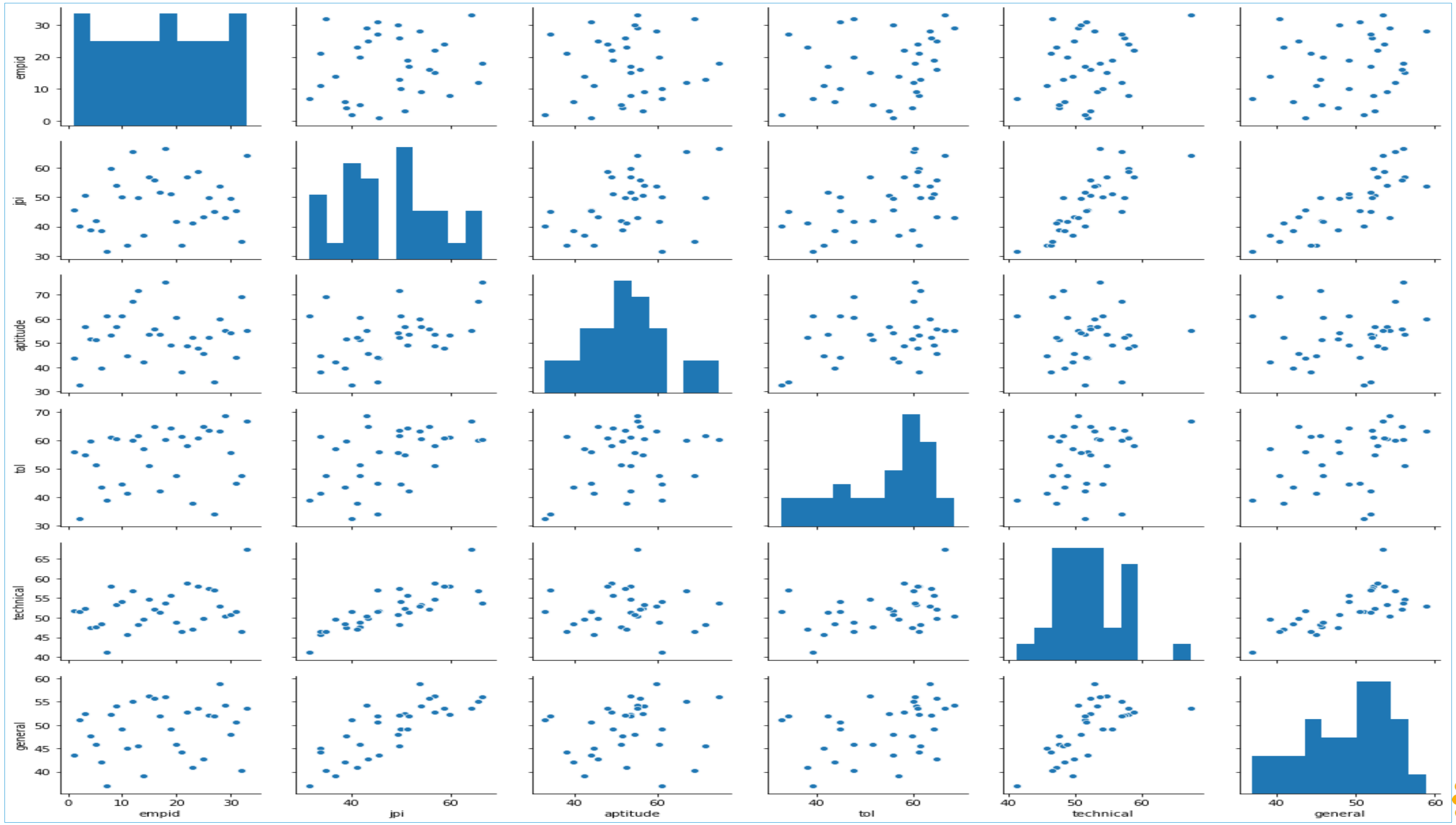
```
#Graphical Representation of the Data
```

```
import seaborn as sns
sns.pairplot(perindex)
```

DATA SCIENCE INSTITUTE

Scatter Plot Matrix

`pairplot()` function in library **`seaborn`** gives scatter plot matrix and distribution of all variables using histogram.



DATA SCIENCE INSTITUTE

Model for the Case Study

Dependent Variable

Independent Variables

$$jpi = b_0 + b_1(\text{aptitude}) + b_2(\text{tol}) + b_3(\text{technical}) + b_4(\text{general}) + e$$

Parameter Estimates

Intercept / Constant

Error Term

Parameter Estimation Using ols() function in Python

#Model Fit

```
import statsmodels.formula.api as smf

jpimodel=smf.ols('jpi ~ tol + aptitude + technical +general',
data=perindex).fit()

jpimodel.params
```

- **ols()** fits a linear regression.
- ~ separates dependent and independent variables
- Left hand side of tilde(~) represents the dependent variable and right-hand side shows independent variables

#Output

```
Intercept    -54.282247
tol            0.033372
aptitude       0.323562
technical      1.095467
general        0.536834
dtype: float64
```

Interpretation :

- **jpimodel.params** gives the model parameters.
- Signs of each parameter represent their relationship with the dependent variable.

DATA SCIENCE INSTITUTE

Interpretation of Partial Regression Coefficients

- For every unit increase in the independent variable (X), expected value of the dependent variable (Y) will change by the corresponding parameter estimate (b), keeping all the other variables constant

Parameters	Coefficients
Intercept	-54.2822
aptitude	0.3236
tol	0.0334
technical	1.0955
general	0.5368

- From the parameter estimates table, we observe that the parameter estimate for Aptitude Test is 0.3236

  We can infer that for one unit increase in aptitude test score, the expected value of job performance index will increase by 0.3236 units

Measure of Goodness of Fit – R Squared

- R^2 is the proportion of variation in the dependent variable which is explained by the independent variables. Note that R^2 always increases if variable is added in the model

$$R^2 = \frac{Explained\,Variation}{Total\,Variation}$$

$$¿\ \frac{\sum_{i=1}^{n} \left(\hat{Y}i - \bar{Y} \right) 2}{\sum_{i=1}^{n} \left(Yi - \bar{Y} \right) 2}$$

$$R_a^{\,2} = 1 - \frac{n - 1}{n - p - 1}\left(1 - R^2\right)$$

- **The adjusted R-squared** is a modified version of R-squared that has been adjusted for the number of predictors in the model

- The adjusted R-squared is a modified version of R-squared that has been adjusted for the number of predictors in the model.
- Normally, greater than 0.7 is considered as a benchmark for accepting goodness of fit of a model.

Understanding Summary Output

```
#Model Summary
jpimodel.summary()
```

summary() generates a detailed description of the model.

```
=====================
Dep. Variable:                    jpi   R-squared:                       0.877
Model:                            OLS   Adj. R-squared:                  0.859
Method:                 Least Squares   F-statistic:                     49.81
Date:                Wed, 23 Oct 2019   Prob (F-statistic):           2.47e-12
Time:                        14:01:20   Log-Likelihood:                -85.916
No. Observations:                  33   AIC:                             181.8
Df Residuals:                      28   BIC:                             189.3
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept    -54.2822      7.395     -7.341      0.000     -69.429     -39.135
tol            0.0334      0.071      0.468      0.643      -0.113       0.179
aptitude       0.3236      0.068      4.774      0.000       0.185       0.462
technical      1.0955      0.181      6.039      0.000       0.724       1.467
general        0.5368      0.158      3.389      0.002       0.212       0.861
==============================================================================
Omnibus:                        2.124   Durbin-Watson:                   1.379
Prob(Omnibus):                  0.346   Jarque-Bera (JB):                1.944
Skew:                          -0.544   Prob(JB):                        0.378
Kurtosis:                       2.518   Cond. No.                     1.25e+03
==============================================================================
```

Interpretation :

- Reject Global Testing null hypothesis that no variables are significant as p-value is<0.05
- Intercept, aptitude, technical, general are significant variables (p-values<0.05)
- tol is not significant (p-value>0.05)

Summary of Findings

Significant variables

- **Aptitude**
- **Technical knowledge**
- **General information**

Out of four dependent variables, **three affect job performance index positively**

R^2 → **0.88**

88% of the variation in job performance index is explained by the model & 12% is unexplained variation

DATA SCIENCE INSTITUTE

Fitted Values and Residuals

```
#Model Fitting after eliminating the insignificant variable
jpimodel_new=smf.ols('jpi ~ aptitude + technical +general', data=perindex).fit()
jpimodel_new.params
```

The insignificant variable **tol** is not included in the new model

```
#Output
Intercept   -54.406443
aptitude      0.333346
technical     1.116627
general       0.543157
dtype: float64
```

Estimated values of the model parameters using the new model

*** To get the fitted values and the residuals values, the model should include only the significant variables**

Fitted Values and Residuals

```
#Adding Fitted Values and Residuals to the Original Dataset
perindex=perindex.assign(pred=pd.Series(jpimodel_new.fittedvalues))
perindex=perindex.assign(res=pd.Series(jpimodel_new.resid))
perindex.head()
```

fittedvalues() and **resid()** fetch fitted values and residuals respectively.

```
#Output
```

	empid	jpi	aptitude	tol	technical	general	pred	res
0	1	45.52	43.83	55.92	51.82	43.58	41.738503	3.781497
1	2	40.10	32.71	32.56	51.49	51.03	41.709731	-1.609731
2	3	50.61	56.64	54.84	52.29	52.47	51.362151	-0.752151
3	4	38.97	51.53	59.69	47.48	47.69	41.691486	-2.721486
4	5	41.87	51.35	51.50	47.59	45.77	40.711451	1.158549

Interpretation :

- **pred** values are calculated based on the values of the model parameters
- **res** is the difference between the actual **jpi** values and the **pred** values.
- Lower the residuals, lesser is the difference between fitted

DATA SCIENCE INSTITUTE

Predictions for New Dataset

- New data set should have all the independent variables used in the model
- Column names of all common variables in the new and old datasets should be identical
- Note that missing values will be taken as 0 (which can be incorrect)

```
#Importing New Dataset
perindex_new=pd.read_csv("Performance Index new.csv")
perindex_new=perindex_new.assign(pred=pd.Series(jpimodel_new.predict(perindex_new)))

perindex_new.head()
```

predict() returns predicted values. Fitted model is the first argument and new dataset object is the second argument. This ensures Python uses parameters from the fitted model for predictions on new data.

	empid	jpi	tol	technical	general	aptitude	pred
0	34	66.35	59.20	57.18	54.98	66.74	61.552576
1	35	56.10	64.92	52.51	55.78	55.45	53.008978
2	36	48.95	63.59	57.76	52.08	51.73	55.621537
3	37	43.25	64.90	50.13	42.75	45.09	39.820600
4	38	41.20	51.50	47.89	45.77	50.85	40.879766

Predictions with Confidence Interval

```
#Predictions with Confidence Interval

result = jpimodel_new.get_prediction(perindex_new)
result.conf_int()
```

- **conf_int()** generates 95% confidence intervals by default.
- Left hand side values in array gives lower confidence interval values, right gives upper.

```
#Output

array([[59.00955719, 64.09559387],
       [50.67791702, 55.34003898],
       [53.65401364, 57.58906082],
       [37.73389546, 41.90730465],
       [39.23363549, 42.52589584],
       [45.41626758, 47.98650295]])
```

Q. Why are confidence intervals needed for predictions?
A. The point estimate is the best guess of the true value of the parameter, while the interval estimate gives a measure of accuracy of that point estimate by providing an interval that contains plausible values.

* **If you wish to specify the level of tolerance/confidence, use alpha= argument in the conf_int() function. For example, to calculate 90% confidence intervals,** alpha = 0.1

DATA SCIENCE INSTITUTE

THANK YOU!!