Ordinary Least Squares with Multiple Linear Regression

Now that you have estimated the parameters for simple linear regression with OLS let's do the same for multiple linear regression.

From L1 content, you have the idea of Multiple Linear Regression. Let's recall a bit. Multiple linear regression generalizes simple linear regression by allowing more than one input variable: $x_1, x_2, \ldots, x_d$. The goal of multiple linear regression is to find a relationship between the input variables and the output variable. This relationship is represented mathematically as follows:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_d x_d$$

β_1 through β_d are the estimated regression coefficients for the independent variables x_1 through x_d. As in simple linear regression, the regression model to get actual output variable is:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_d x_d + \epsilon$$

ϵ is the random error or residual, which reflects the difference between the actual output point and predicted output point.

Multiple linear regression involves more than one input variable, so it is impossible to individually derive a solution for each regression coefficient for each input variable. In a sophisticated regression problem dimension (d) can range to very higher values.

So, how do we estimate all regression coefficients?

From multiple linear regresssion model, we have:

$$y_1 = \beta_0 + \beta_1 x_{11} + \beta_2 x_{12} + \cdots + \beta_d x_{1d} + \epsilon_1$$

We have n set of observations. So we can write:

$$\begin{aligned}
y_1 &= \beta_0 + \beta_1 x_{11} + \beta_2 x_{12} + \cdots + \beta_d x_{1d} + \epsilon_1 \\
y_2 &= \beta_0 + \beta_1 x_{21} + \beta_2 x_{22} + \cdots + \beta_d x_{2d} + \epsilon_2 \\
y_3 &= \beta_0 + \beta_1 x_{31} + \beta_2 x_{32} + \cdots + \beta_d x_{3d} + \epsilon_3 \\
&\vdots \qquad\qquad\qquad\qquad \vdots \\
y_n &= \beta_0 + \beta_1 x_{n1} + \beta_2 x_{n2} + \cdots + \beta_d x_{nd} + \epsilon_n
\end{aligned}$$

x_{nd} is the nth observation for dth feature or input variable. These n set of equations can be written in matrix form as:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1d} \\ 1 & x_{21} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{nd} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_d \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

Using mathematical notations, we can write as:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \epsilon \ldots (1)$$

$\mathbf{y}$ is a $\mathrm{n} \times 1$ column matrix where each element is the observed value of output variable. Similarly, $\mathbf{X}$ is $\mathrm{n} \times (\mathrm{d}+1)$ matrix. An extra dimension is due to the inclusion of $1's$ in the first column. You can interpret the first column, including $1'$s as being multiplied against β_0. There are $(\mathrm{d}+1)$ unknown parameters, d regression coefficients for each of the input variables and 1 extra for the intercept, β_0. So $\boldsymbol{\beta}$ is $(\mathrm{d}+1) \times 1$ column matrix.

By now, we addressed the multiple features and multiple unknown parameters properly in the form of a matrix. Now, we roll back to the principle of OLS to determine the unknown parameters.

From OLS, our objective is to find a column matrix or a column vector, $\boldsymbol{\beta}$, such that *Sum of Squared Errors*, SSE is minimum. SSE is written as:

$$\mathrm{SSE} = \sum_{i=1}^{n} (y_i - \hat{y_i})^2 = \sum_{i=1}^{n} \epsilon_i^2$$

Since

$$\epsilon^T \epsilon = \begin{bmatrix} \epsilon_1 & \epsilon_2 & \dots & = \epsilon_n \end{bmatrix} \cdot \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix} = \epsilon_1^2 + \epsilon_2^2 + \dots \epsilon_n^2 = \sum_{i=1}^{n} \epsilon_i^2$$

We can also write SSE as:

$$\mathrm{SSE} = \sum_{i=1}^{n} \epsilon_i^2 = \epsilon^T \epsilon$$

From equation (1), we know that

$$\epsilon = \mathbf{y} - \mathbf{X}\boldsymbol{\beta}$$

so we can also write SSE as:

$$\mathrm{SSE} = \epsilon^T \epsilon = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})$$

This positive quadratic error function or SSE or objective function is always a convex surface facing upwards as in a simple linear equation. From calculus, the value of parameters at the minimum point is obtained by setting the first derivative of the objective function, with respect to the parameters, equal to 0. So, we will take the partial derivative of the objective function, with respect to $\boldsymbol{\beta}$, and get the value for the column matrix, $\boldsymbol{\beta}$.

$$\frac{\partial\, \mathrm{SSE}}{\partial \boldsymbol{\beta}} = \frac{\partial}{\partial \boldsymbol{\beta}} (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T (\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) = \frac{\partial}{\partial \boldsymbol{\beta}} (\mathbf{y^T y} + \boldsymbol{\beta}^T \mathbf{X}^T \mathbf{X} \boldsymbol{\beta} - 2\boldsymbol{\beta}^T \mathbf{X}^T \mathbf{y})$$

You can take a pen and paper and try expanding the product term to the sums. You have to use basic transpose rules and matrix multiplication rules. That's it! Now, we will set the derivative to 0 as:

$$\frac{\partial}{\partial \boldsymbol{\beta}} (\mathbf{y^T y} + \boldsymbol{\beta}^T \mathbf{X}^T \mathbf{X} \boldsymbol{\beta} - 2\boldsymbol{\beta}^T \mathbf{X}^T \mathbf{y}) = 0$$

As we saw for the column vector ϵ we know, $\boldsymbol{\beta}^T \boldsymbol{\beta} = \boldsymbol{\beta}^2$. After derivation we can write as:

$$2\boldsymbol{\beta} \mathbf{X}^T \mathbf{X} - 2\mathbf{X}^T \mathbf{y} = 0$$

This can be written as:

$$2\boldsymbol{\beta} \mathbf{X}^T \mathbf{X} = 2\mathbf{X}^T \mathbf{y}$$

Cancelling 2 on both sides and isolating β, we get:

$$\beta = \frac{\mathbf{X}^T\mathbf{y}}{\mathbf{X}^T\mathbf{X}}$$

This is the solution to unknown parameters. But, usually, we don't express the formulae for parameter estimates in this way. We express it in the form of the normal equation as:

$$\boldsymbol{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Thus, this normal equation derived is the solution to the unknown parameters in multiple linear regression.

Since the parameters are estimates, we usually put *hats* on them so, the normal equation to determine estimated parameters is:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Implementation on Real World Dataset

For implementation, we will use same Advertising dataset.

A popular introductory statistics book, An Introduction to Statistical Learning (http://faculty.marshall.usc.edu/gareth-james/ISL/index.html), provides this dataset on their website. This dataset can be downloaded from the following address:

- http://faculty.marshall.usc.edu/gareth-james/ISL/Advertising.csv (http://faculty.marshall.usc.edu/gareth-james/ISL/Advertising.csv)

This dataset has got three inputs as advertising mediums, i.e., *TV*, *radio* and *newspaper*. Similarly the output variable is *sales*. This is a sales prediction problem with investment in any of the advertising mediums.

Imports

In [ ]:

```
import numpy as np
import pandas as pd
import matplotlib as mpl
from matplotlib import pyplot as plt
from IPython.display import display, HTML
```

In [ ]:

```
data_path = "https://storage.googleapis.com/codehub-data/1-lv2-2-2-Advertisement.cs

# Read the CSV data from the link
data_df = pd.read_csv(data_path,index_col=0)

# Print out first 5 samples from the DataFrame
data_df.head()
```

Out[2]:

	TV	radio	newspaper	sales
1	230.1	37.8	69.2	22.1
2	44.5	39.3	45.1	10.4
3	17.2	45.9	69.3	9.3
4	151.5	41.3	58.5	18.5
5	180.8	10.8	58.4	12.9

In [ ]:

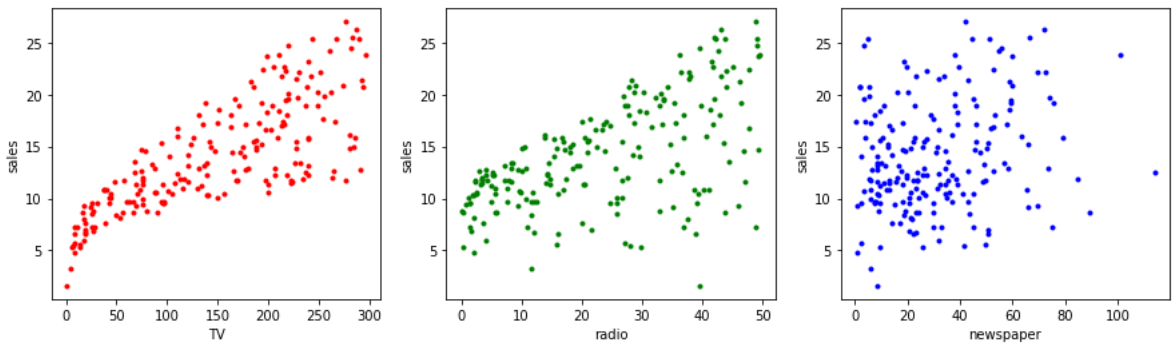
```
fig = plt.figure(figsize=(15,4))
gs = mpl.gridspec.GridSpec(1,3)

# Plot of sales vs TV
ax = fig.add_subplot(gs[0])
ax.scatter(data_df["TV"], data_df["sales"], color="red", marker=".")
ax.set_xlabel("TV")
ax.set_ylabel("sales")

# Plot of sales vs radio
ax = fig.add_subplot(gs[1])
ax.scatter(data_df["radio"], data_df["sales"], color="green", marker=".")
ax.set_xlabel("radio")
ax.set_ylabel("sales")

# Plot of sales vs newspaper
ax = fig.add_subplot(gs[2])
ax.scatter(data_df["newspaper"], data_df["sales"], color="blue", marker=".")
ax.set_xlabel("newspaper")
ax.set_ylabel("sales")

plt.show()
```



The first plot shows a sharp upward trend in the number of units sold as TV advertising increases. A similar trend is also found as radio advertising increases. However, in the last plot, there does not appear to be a

relationship between newspaper advertising and the number of units sold.

Multiple Linear regression using Ordinary Least Squares

Earlier in simple linear regression, we took one variable at a time. But now in multiple linear regression, we will take more input variable. We will see two cases as we saw in L1 content. First, we will see the combined effect of `TV` and `radio` into the output variable `sales` then secondly, we will see the combined effect of all three inputs, `TV`, `radio` and `newspaper` into the output variable `sales`.

In [ ]:

```
# Training Linear Regression using TV and Radio features
X = data_df[["TV", "radio"]]
y = data_df[["sales"]]

# set bias/intercept  term to 1 for each 200 samples
X = np.c_[np.ones((200, 1)), X]
X_transpose = np.transpose(X)

# implementing least square solution of matrix form
betas = np.linalg.inv(X_transpose.dot(X)).dot(X_transpose).dot(y)


message = """<strong>TV and Radio</strong> <br>
$y$ = {:.2f} + {:.2f}$x_{{1}}$ + {:.2f}$x_{{2}}$ <br>
$x_{{1}}$ = TV <br>
$x_{{2}}$ = radio
""".format(*betas[0], *betas[1], *betas[2])
display(HTML( message ))

print("")


# # Training Linear Regression using all features
X = data_df[["TV", "radio", "newspaper"]]
y = data_df[["sales"]]

# set bias/intercept  term to 1 for each 200 samples
X = np.c_[np.ones((200, 1)), X]
X_transpose = np.transpose(X)


# implementing least square solution of matrix form
betas = np.linalg.inv(X_transpose.dot(X)).dot(X_transpose).dot(y)

message = """<strong>TV, Radio, and Newspaper</strong> <br>
$y$ = {:.2f} + {:.2f}$x_{{1}}$ + {:.2f}$x_{{2}}$ + {:.2f}$x_{{3}}$ <br>
$x_{{1}}$ = TV <br>
$x_{{2}}$ = radio <br>
$x_{{3}}$ = newspaper
""".format(*betas[0], *betas[1], *betas[2], *betas[3])
display(HTML( message ))
```

TV and Radio

y = 2.92 + 0.05x_1 + 0.19x_2

x_1 = TV

x_2 = radio

TV, Radio, and Newspaper

y = 2.94 + 0.05x_1 + 0.19x_2 + -0.00x_3

x_1 = TV

x_2 = radio

x_3 = newspaper

TV, radio Vs. sales:

TV and *radio* are the input variables, $x_1 and x_2$ respectively and *sales* is the output variable, y. We obtain a multiple linear regression model of $y = 2.92 + 0.05x_1 + 0.19x_2$. Intercept, β_0 has been estimated as 2.92 and two regression coefficients, β_1 and β_2 have been estimated to 0.05 and 0.19 respectively. The values of the parameters through OLS is same to that through `Scikit-Learn`.

TV, radio, newspaper Vs. sales:

TV , *radio* and *newspaper_are the input variables,* $x_1, x_2 and x_3$ *respectively and _sales* is the output variable, y. We obtain a multiple linear regression model of $y = 2.94 + 0.05x_1 + 0.19x_2 + 0.00x_3$; note that each of the estimated model parameters (i.e., β_0 through β_3) have been rounded to 2 decimal places.. Intercept, β_0 has been estimated as 2.94 and three regression coefficients, β_1, β_2 and β_3 have been estimated to 0.05, 0.19, and 0.00 respectively. The values of the parameters through OLS is same to that through `Scikit-Learn`.

Potential Issues with Ordinary Least Squares

We have now derived OLS for both simple linear regression and multiple linear regression. By both derivations, we are now clear to the key principle of Ordinary Least Squares. OLS tends to find the estimates of parameters such that the *Sum of Squares of Errors, SSE* is minimum.

While deriving the solution, we had made few assumptions. Violation of these assumptions might create serious problems while finding the solution. Some of the assumptions with the potential issues after their violation are:

- Existence of $(\mathbf{X}^T\mathbf{X})^{-1}$

  While calculating β, we assume that $(\mathbf{X}^T\mathbf{X})^{-1}$ exists. When doesn't it exist?

  The inverse doesn't exist when the matrix, $\mathbf{X}^T\mathbf{X}$, is not a full rank matrix. The matrix $\mathbf{X}$ with dimension $n \times (d+1)$ should have at least $(d+1)$ linearly independent rows to make $\mathbf{X}^T\mathbf{X}$ a full rank matrix. If there is perfect colinearity between any two independent input variables, then the matrix doesn't become full rank and the issues arise. There should be correlation between dependent(output) and independent(input) variable but the independent variables should be independent of each other.
- Existence of $n >> d$

  If $n < d+1$, we can not do least squares. Numerous solution are obtained if the number of data points is less than the dimension of the features. So, we need more observations or samples than the number of features.

There is another parameter estimation method, *Gradient Descent*, which addresses the issues of Ordinary Least Squares. We will deep dive into *Gradient Descent* in the upcoming chapter.

Additional Resources

- Lecture Notes
  - http://mezeylab.cb.bscb.cornell.edu/labmembers/documents/supplement%205%20-%20multiple%20regression.pdf (http://mezeylab.cb.bscb.cornell.edu/labmembers/documents/supplement%205%20-%20multiple%20regression.pdf)
    - Checkout the whole document for better understanding.