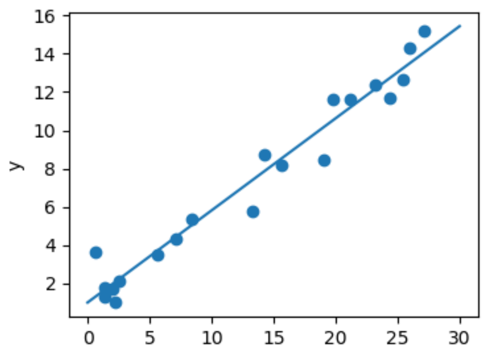
SIMPLE LINEAR REGRESSION WITH JUPYTER NOTEBOOK

DR. ALVIN ANG



COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG

CONTENTS

COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG

INTRODUCTION

- Simple Linear Regression has already been described here (using Excel)
  - https://www.alvinang.sg/s/How-to-Perform-Simple-Linear-Regression-using-Excel-Dr-Alvin-Ang-watermarked.pdf
- In this manuscript, we will make use of a dataset containing 202 car models (and their attributes) and see how Linear Regression can be applied to it (using Jupyter Notebook / Python).
- The dataset is here:
  - https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/DA0101EN/automobileEDA.csv
- Specifically, we will
  - Load and Glance at the Dataset.
  - Visualize / Plot the regression model.
  - Generate a Linear Regression Equation.
  - Use a Residual Plot to visually inspect if Linear Regression fits the model
  - Use R2 and MSE as indicators to determine the accuracy of the Linear Regression fit.

COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG

PART I: LOAD AND GLANCE AT THE DATASET

STEP 1: ENSURE ACCESS TO JUPYTER NOTEBOOK

- There are many ways to access Jupyter Notebook.
- Refer here: https://www.alvinang.sg/s/How-to-Access-CSV-Files-using-PyCharm-Anaconda-or-Skills-Network-Lab-by-Dr-Alvin-Ang.pdf

STEP 2: IMPORT LIBRARIES AND HAVE A GLANCE AT THE DATASET

- Import Libraries Code:
    - import pandas as pd
    - import numpy as np
    - import matplotlib.pyplot as plt
- Load Data and Store in Dataframe df Code:
    - path = 'https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/DA0101EN/automobileEDA.csv'
    - df = pd.read_csv(path)
    - df.head()
- Output:

ymboling	normalized-losses	make	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	length	...	compression-ratio	horsepower	peak-rpm	city-mpg	highway-mpg	price
3	122	alfa-romero	std	two	convertible	rwd	front	88.6	0.811148	...	9.0	111.0	5000.0	21	27	13495.0
3	122	alfa-romero	std	two	convertible	rwd	front	88.6	0.811148	...	9.0	111.0	5000.0	21	27	16500.0
1	122	alfa-romero	std	two	hatchback	rwd	front	94.5	0.822681	...	9.0	154.0	5000.0	19	26	16500.0
2	164	audi	std	four	sedan	fwd	front	99.8	0.848630	...	10.0	102.0	5500.0	24	30	13950.0
2	164	audi	std	four	sedan	4wd	front	99.4	0.848630	...	8.0	115.0	5500.0	18	22	17450.0

vs × 29 columns

- Or you could just click on https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/DA0101EN/automobileEDA.csv to view the dataset.

COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG

PART II: VISUALIZE / PLOT THE REGRESSION MODEL

STEP 1: LOAD THE LR MODULES AND CREATE THE LR OBJECT

- Load the LR Module Code:
    - from sklearn.linear_model import LinearRegression
- Create the LR Object Code:
    - lm = LinearRegression()
    - lm
- Output:
    - LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)

STEP 2: DEFINE OUR X AND Y

- We create
    - X → "highway-mpg"
    - Y → "price"
- Code:
    - X = df[['highway-mpg']]
    - Y = df['price']

STEP 3: FIT / TRAIN THE LINEAR MODEL

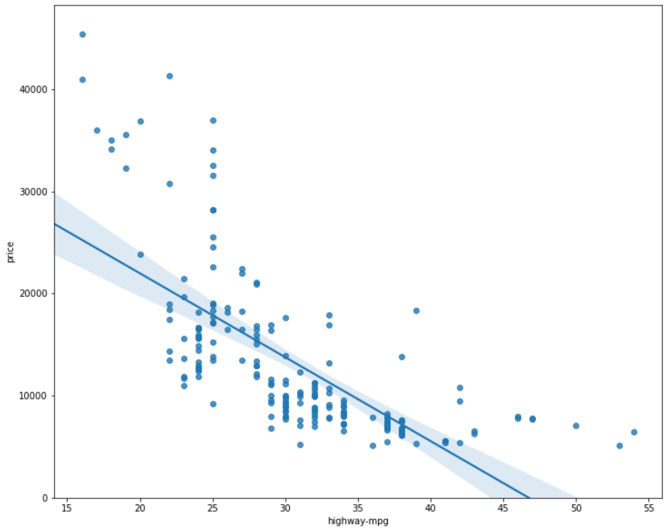
- Code:
    - lm.fit(X,Y)
- Output:
    - LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)

COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG

STEP 4: IMPORT THE VISUALIZATION PACKAGE SEABORN

- Code:
  - import seaborn as sns
  - %matplotlib inline

STEP 5: VISUALIZE PRICE VS HIGHWAY-MPG

- Code:
  - width = 12
  - height = 10
  - plt.figure(figsize=(width, height))
  - sns.regplot(x="highway-mpg", y="price", data=df)
  - plt.ylim(0,)
- Output:
  - (0.0, 48250.106831059355)
  - 

- Comments:

COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG

- Price is negatively correlated to highway-mpg.
- The data points are scattered badly around the regression line.
- A linear model is NOT the best fit.

COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG

PART III: GENERATE A LINEAR REGRESSION EQUATION

STEP 1: FIND THE Y-INTERCEPT

- Y-Intercept refers to the C of the Y = mX + C.
- Code:
    - lm.intercept_
- Output:
    - 38423.3058581574

STEP 2: FIND THE GRADIENT

- Gradient refers to the m of the Y = mX +C
- Code:
    - lm.coef_
- Output:
    - array([-821.73337832])
- This means that the Linear Equation is
    - price = 38423.31 - 821.73 x highway-mpg → Y = C + mX

STEP 3: TEST SOME PREDICTIONS

- Since we already have the LR Equation Y = mX +C, we test it using the first 5 rows of values of the Dataset.
- Code:
    - Yhat=lm.predict(X)
    - Yhat[0:5]
- Output:

COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG

- array([16236.50464347, 16236.50464347, 17058.23802179, 13771.3045085 , 20345.17153508])

- Note that the first 5 rows of the "highway-mpg" are as follows:

highway-mpg	price
27	13495
27	16500
26	16500
30	13950
22	17450

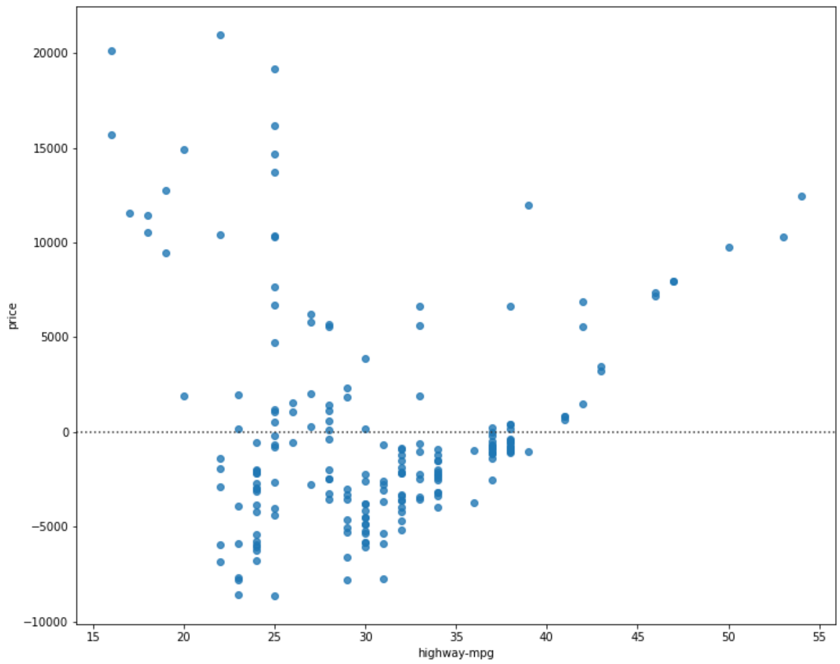
- In other words, the "forecasted" values in the prediction array were using the values
    - 27 / 27 / 26 / 30 / 22
- This differs quite a bit from the real pricings!

COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG

PART IV: USE A RESIDUAL PLOT TO VISUALLY INSPECT IF LINEAR REGRESSION FITS THE MODEL

- Residual plot has been described and defined here:
  - https://www.alvinang.sg/s/Multiple-Regression-MR-by-Dr-Alvin-Ang.pdf
  - A residual plot is a graph that shows the residuals on the vertical y-axis and the independent variable on the horizontal x-axis.
- What is a Residual? The difference between the observed value (y) and the predicted value (Yhat).
- If the points in a Residual Plot are randomly spread out around the x-axis, then a linear model is appropriate for the data.
- Because randomly spread out residuals means that the variance is constant, and thus the linear model is a good fit for this data.
- Code:
  - width = 12
  - height = 10
  - plt.figure(figsize=(width, height))
  - sns.residplot(df['highway-mpg'], df['price'])
  - plt.show()
- Output:

COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG



- Comments:
    - This residual plot shows that the residuals are not randomly spread around the x-axis.
    - Maybe a non-linear model is more appropriate for this data.

COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG

PART V: USE R2 AND MSE AS INDICATORS TO DETERMINE THE ACCURACY OF THE LINEAR REGRESSION FIT

- R2 has been explained here:
  - https://www.alvinang.sg/s/How-to-Perform-Simple-Linear-Regression-using-Excel-Dr-Alvin-Ang-watermarked.pdf
  - R squared, also known as the coefficient of determination, is a measure to indicate how close the data is to the fitted regression line.
- Mean Squared Error (MSE) has been explained here:
  - https://www.alvinang.sg/s/Forecasting-by-Dr-Alvin-Ang-watermarked-hjr9.pdf
  - The Mean Squared Error measures the average of the squares of errors, that is, the difference between actual value (y) and the estimated value (ŷ).

STEP 1: CALCULATE THE R2 FOR "HIGHWAY_MPG" VS "PRICE"

- Code:
  - #highway_mpg_fit
  - lm.fit(X, Y)
  - # Find the R^2
  - print('The R-square is: ', lm.score(X, Y))
- Output:
  - The R-square is: 0.4965911884339176
- Comment:
  - We can say that ~ 49.659% of the variation of the "price" is explained by this simple linear model "highway_mpg".
  - Below 50% means that actually a linear model is not a good fit…which means that the actual data is far from the fitted line…

COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG

STEP 2: CALCULATE THE MSE

FIRSTLY, PREDICT THE OUTPUT "YHAT"

- Code:
  - Yhat=lm.predict(X)
  - print('The output of the first four predicted value is: ', Yhat[0:4])
- Output:
  - The output of the first four predicted value is: [16236.50464347 16236.50464347 17058.23802179 13771.3045085]

SECONDLY, IMPORT THE FUNCTION "MEAN_SQUARED_ERROR"

- Code:
  - from sklearn.metrics import mean_squared_error

THIRDLY, OBTAIN THE MSE

- Code:
  - mse = mean_squared_error(df['price'], Yhat)
  - print('The mean square error of price and predicted value is: ', mse)
- Output:
  - The mean square error of price and predicted value is: 31635042.944639888
- Comment:
  - At this point, we are unable to say if MSE is high or low.
  - MSE is used to measure against another method of fitting i.e. it cannot be used as a standlone measure.
  - That is, currently we are doing Linear Regression (LR) for model fitting and we have this MSE.

COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG

- We can only compare this MSE with another MSE of another model fit… E.g. Multiple Regression (MR)… in which we will showcase this in another article.

COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG

CONCLUSION

- In this article, we used Linear Regression (LR) on a dataset containing 200 car models.
- We used Python / Jupyter Notebook to run the codes.
- Specifically, we wanted to find out whether LR was a good fit or not.
- These we presented in this article:
    - Visualize / Plot the regression model.
    - Generate a Linear Regression Equation.
    - Use a Residual Plot to visually inspect if Linear Regression fits the model
    - Use R2 and MSE as indicators to determine the accuracy of the Linear Regression fit.

COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG

ABOUT DR. ALVIN ANG

Dr. Alvin Ang earned his Ph.D., Masters and Bachelor degrees from NTU, Singapore. He is a scientist, entrepreneur, as well as a personal/business advisor. More about him at www.AlvinAng.sg.

COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG