

DR. ALVIN'S PUBLICATIONS

CROSS VALIDATING A MOTORCARS DATASET

WITH PYTHON
DR. ALVIN ANG



1 | PAGE

COPYRIGHTED BY DR ALVIN ANG
WWW.ALVINANG.SG

CONTENTS

I. Without Cross Validation.....	3
A. Step 1: Read in Dataset	4
1. 1a) Import Libraries.....	4
2. Import CSV	4
B. Step 2: Assigning Columns.....	5
1. Assign 'Price' Column to y_data.....	5
2. Assigning all other Columns to x_data.....	6
C. Step 3: Train Test Split.....	7
D. Step 4: Using Linear Regression	8
1. Fitting Linear Regression Model to Train Dataset.....	8
2. Getting R2 Score	8
3. Do a Prediction.....	9
II. With Cross Validation	10
A. Step 1: Import Cross_Val_Score.....	10
B. Step 2: Get the Cross_Val_Score.....	10
1. Use Cross_Val_Score to first Fit Linear Regression of 'horsepower' and 'price'.....	10
2. Obtain the R2(or Rcross) for Every Fold.....	11
3. Obtain the Mean of all 4 R2	11
C. Step 3: Do a Prediction using Cross_Val-Predict	12
1. Import Cross_Val_Predict	12
III. Conclusion	13
IV. About Dr. Alvin Ang	14

I. WITHOUT CROSS VALIDATION

https://www.alvinang.sg/s/Cross_Validating_a_European_and_Japanese_Car_Dataset_by_Dr_Alvin_Ang.ipynb

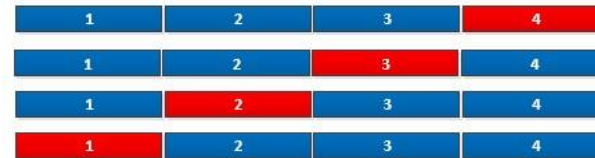
https://www.alvinang.sg/s/cleansed_autocsv.csv

$K = 4$

Without Cross
Validation



With Cross Validation



Blue = Training Data

Red = Testing Data

A. STEP 1: READ IN DATASET

1. 1A) IMPORT LIBRARIES

```
▼ WITHOUT CROSS VALIDATION
▼ Step 1: Read in Dataset
▼ 1a) Import Libraries

[84] import pandas as pd
import numpy as np
```

2. IMPORT CSV

1b) Import CSV

```
[ ] df = pd.read_csv('https://www.alvinang.sg/s/cleansed_autocsv.csv')
```

```
[ ] df=df._get_numeric_data()
df.head()
```

```
#we only want to get the Numeric Data from the dataset
```

	Unnamed: 0	Unnamed: 0.1	symboling	normalized-losses	wheel-base	length	width	height	curb-weight	engine-size	...	stroke	compression-ratio	horsepower	peak-rpm	city-mpg	highway-mpg	price	city-L/100km	diesel	gas
0	0	0	3	122	88.6	0.811148	0.890278	48.8	2548	130	...	2.68	9.0	111.0	5000.0	21	27	13495.0	11.190476	0	1
1	1	1	3	122	88.6	0.811148	0.890278	48.8	2548	130	...	2.68	9.0	111.0	5000.0	21	27	16500.0	11.190476	0	1
2	2	2	1	122	94.5	0.822681	0.909722	52.4	2823	152	...	3.47	9.0	154.0	5000.0	19	26	16500.0	12.368421	0	1
3	3	3	2	164	99.8	0.848630	0.919444	54.3	2337	109	...	3.40	10.0	102.0	5500.0	24	30	13950.0	9.791667	0	1
4	4	4	2	164	99.4	0.848630	0.922222	54.3	2824	136	...	3.40	8.0	115.0	5500.0	18	22	17450.0	13.055556	0	1

5 rows x 21 columns

B. STEP 2: ASSIGNING COLUMNS

1. ASSIGN 'PRICE' COLUMN TO Y_DATA

▼ Step 2: Assigning Columns

▼ 2a) Assign 'price' column to y_data

```
[ ] y_data = df['price']  
    print(y_data[0:3])
```

```
0    13495.0  
1    16500.0  
2    16500.0  
Name: price, dtype: float64
```

2. ASSIGNING ALL OTHER COLUMNS TO X_DATA

2b) Assigning all other columns to x_data

```
x_data=df.drop('price',axis=1)
print(x_data[0:3])
```

```

  Unnamed: 0  Unnamed: 0.1  symboling  normalized-losses  wheel-base  \
0           0           0           3             122           88.6
1           1           1           3             122           88.6
2           2           2           1             122           94.5

   length  width  height  curb-weight  engine-size  bore  stroke  \
0  0.811148  0.890278  48.8         2548         130  3.47  2.68
1  0.811148  0.890278  48.8         2548         130  3.47  2.68
2  0.822681  0.909722  52.4         2823         152  2.68  3.47

   compression-ratio  horsepower  peak-rpm  city-mpg  highway-mpg  \
0                   9.0         111.0    5000.0      21           27
1                   9.0         111.0    5000.0      21           27
2                   9.0         154.0    5000.0      19           26

   city-L/100km  diesel  gas
0    11.190476      0     1
1    11.190476      0     1
2    12.368421      0     1
```

C. STEP 3: TRAIN TEST SPLIT

Step 3: Train Test Split

```
[ ] from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x_data, y_data, test_size=0.25, random_state=1)

#Test size = 25%
#Train size = 75%
#Random State is a seed for random dataset splitting.
```

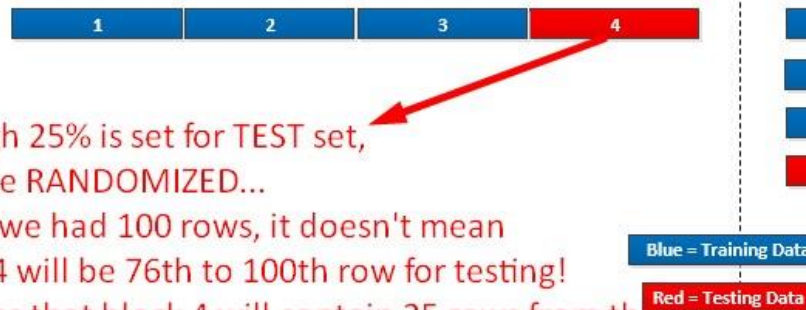
Without Cross Validation

just a note:

even though 25% is set for TEST set,
the rows are RANDOMIZED...

meaning if we had 100 rows, it doesn't mean
that block 4 will be 76th to 100th row for testing!

it just means that block 4 will contain 25 rows from the
dataset..but without Cross Validation means that the dataset
is only SAMPLED ONCE!



D. STEP 4: USING LINEAR REGRESSION

Step 4: Using Linear Regression

```
[ ] from sklearn.linear_model import LinearRegression  
    lre=LinearRegression()
```

1. FITTING LINEAR REGRESSION MODEL TO TRAIN DATASET

4a) Fitting Linear Regression Model to Train Dataset

```
[ ] lre.fit(x_train[['horsepower']], y_train)  
  
    #we fit a linear model to 'price' vs 'horsepower'  
    #(of the TRAIN dataset)  
  
LinearRegression()
```

2. GETTING R2 SCORE

4b) Getting R2 score

```
[ ] lre.score(x_test[['horsepower']], y_test)  
  
    #great fit!  
    #linear regression fits 63% of 'price' to 'horsepower'  
    #on the TEST dataset!  
  
0.6296166860021476
```


3. DO A PREDICTION

4c) Do a Prediction

```
[ ] yhat = lre.predict(x_test[['horsepower']])
    yhat[0:3]

#just for fun, we use the LR model we created
#to predict and preview the first 3 'pricing' rows
#(labelled as yhat)

array([12159.57270173,  7182.0124868 ,  9928.25260538])
```

II. WITH CROSS VALIDATION

A. STEP 1: IMPORT CROSS_VAL_SCORE

WITH CROSS VALIDATION

Step 1: Import Cross_Val_Score from SKLearn

```
from sklearn.model_selection import cross_val_score
```

B. STEP 2: GET THE CROSS_VAL_SCORE

1. USE CROSS_VAL_SCORE TO FIRST FIT LINEAR REGRESSION OF 'HORSEPOWER' AND 'PRICE'

Step 2: Get the Cross_Val_Score

2a) Use Cross_Val_Score to first fit Linear Regression of 'horsepower' and 'price'

```
[ ] Rcross = cross_val_score(lre, x_data[['horsepower']], y_data, cv=4)

#cv = 4 means 4 folds
#fit Linear Regression to 'horsepower' vs 'price'
#(of the ENTIRE dataset...x_data and y_data...
# NOT just the Train/Test split dataset! i.e.not just x_test...y_test...)
```

<https://stackoverflow.com/questions/25006369/what-is-sklearn-cross-validation-cross-val-score>

- scikit learn doesn't specifically say that "cross_val_score" = R2 or R...
- thus for simplicity, we just take it that it's R2

2. OBTAIN THE R2(OR RCROSS) FOR EVERY FOLD

2b) Obtain the R2 (or Rcross) for Every Fold

```
[ ] Rcross
#we see that for every fold, the Rcross is between 50 ~ 70%
array([0.7746232 , 0.51716687, 0.74785353, 0.04839605])
```

3. OBTAIN THE MEAN OF ALL 4 R2

2c) Obtain the Mean of all 4 R2

```
▶ print("Rcross mean = ", Rcross.mean())
#we obtain the Average of the 4 folds
#and its 52%!
```

```
⊙ Rcross mean = 0.522009915042119
```

C. STEP 3: DO A PREDICTION USING CROSS_VAL-PREDICT

1. IMPORT CROSS_VAL_PREDICT

Step 3: Do a Prediction using Cross_Val_Predict

3a) Import Cross_Val_Predict

```
[ ] from sklearn.model_selection import cross_val_predict
```

```
[ ] yhat = cross_val_predict(lre,x_data[['horsepower']], y_data,cv=4)  
    yhat[0:3]
```

```
array([14141.63807508, 14141.63807508, 20814.29423473])
```

Comparing WITHOUT vs WITH Cross Validation

Without Cross Validation

- $R^2 = 63\%$
- We might be fooled that Linear Regression fits the dataset well (63% fitting!)

- $Y_1 = \$12,159$
- $Y_2 = \$7,182$
- $Y_3 = \$9,928$

**Purpose of Cross Validation
= Highlight signs of
OVERFITTING**

With Cross Validation

- $R^2 = 52\%$ (only fits at 52%!)

- $Y_1 = \$14,141$
- $Y_2 = \$14,141$
- $Y_3 = \$20,814$

- The 3 sample predictions show how far off the predictions can get
- The values from “Without Cross Validation” bluffs us that the model works well.
- The values from With Cross Validation shows us the true stuff → Model is not working well with predictions quite far off.



THE END

IV. 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.