2. Multilinear Regression

Bruce E. Shapiro

Getting Started in Machine Learning

Last revised: February 3, 2019

Multilinear Regression

Goal: Find curve $f(x)$ that gives the **best fit** to the points

$$\begin{gathered}(x_0, x_1, \ldots, x_{p-1}, y)_0 \\ (x_0, x_1, \ldots, x_{p-1}, y)_1 \\ \vdots \\ (x_0, x_1, \ldots, x_{p-1}, y)_{n-1}\end{gathered}$$

such that the **objective function**:

$$\mathcal{E} = \sum_{i=0}^{n-1} |f(\mathbf{x}_i) - y_i|^2$$

is minimized, where

$$f(\mathbf{x}) = a + b_0 x_0 + b_1 x_1 + \cdots b_{p-1} x_{p-1}$$

Notation: $\mathbf{x}_i = (x_0, x_1, \ldots, x_{p-1})_i = (x_0^i, x_1^i, \ldots, x_{p-1}^i)$

Derivation of Normal Equations (1)

Let $y_i = f(\mathbf{x}_i) + r_i$ where r_i is the residual error, where

$$|r_i| = |f(x)_i - y_i|$$

Derivation of Normal Equations (1)

Let $y_i = f(\mathbf{x}_i) + r_i$ where r_i is the residual error, where

$$|r_i| = |f(x)_i - y_i|$$

Then:

$$\begin{aligned}
a + b_0 x_0^0 + b_1 x_1^0 + \cdots + b_{p-1} x_{p-1}^0 &= y_0 + r_0 \\
a + b_0 x_0^1 + b_1 x_1^1 + \cdots + b_{p-1} x_{p-1}^1 &= y_1 + r_1 \\
&\vdots \\
a + b_0 x_0^{n-1} + b_1 x_1^{n-1} + \cdots + b_{p-1} x_{p-1}^{n-1} &= y_{n-1} + r_{n-1}
\end{aligned}$$

Derivation of Normal Equations (1)

Let $y_i = f(\mathbf{x}_i) + r_i$ where r_i is the residual error, where

$$|r_i| = |f(x)_i - y_i|$$

Then:

$$\begin{aligned}
a + b_0 x_0^0 + b_1 x_1^0 + \cdots + b_{p-1} x_{p-1}^0 &= y_0 + r_0 \\
a + b_0 x_0^1 + b_1 x_1^1 + \cdots + b_{p-1} x_{p-1}^1 &= y_1 + r_1 \\
&\vdots \\
a + b_0 x_0^{n-1} + b_1 x_1^{n-1} + \cdots + b_{p-1} x_{p-1}^{n-1} &= y_{n-1} + r_{n-1}
\end{aligned}$$

$$\underbrace{\begin{bmatrix} 1 & x_0^0 & x_1^0 & \cdots & x_{p-1}^0 \\ 1 & x_0^1 & x_1^1 & \cdots & x_{p-1}^1 \\ 1 & x_0^2 & x_1^2 & \cdots & x_{p-1}^2 \\ & & & & \\ 1 & x_0^{n-1} & x_1^{n-1} & \cdots & x_{p-1}^{n-1} \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} a \\ b_0 \\ b_1 \\ \vdots \\ b_{p-1} \end{bmatrix}}_{\mathbf{b}} = \underbrace{\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{bmatrix}}_{\mathbf{y}} + \underbrace{\begin{bmatrix} r_0 \\ r_1 \\ \vdots \\ r_{n-1} \end{bmatrix}}_{\mathbf{r}}$$

Derivation of Normal Equations (2)

Matrix Equation: $\mathbf{Ab} = \mathbf{y} + \mathbf{r}$

Derivation of Normal Equations (2)

Matrix Equation: $\mathbf{A}\mathbf{b} = \mathbf{y} + \mathbf{r}$

Let: $\mathbf{A} = \begin{bmatrix} \mathbf{1} & | & \mathbf{a}_0 & | & \mathbf{a}_1 & | & \cdots & | & \mathbf{a}_{n-1} \end{bmatrix}$ where: $\mathbf{a}_j = \begin{bmatrix} x_j^0 \\ x_j^1 \\ \vdots \\ x_j^{n-1} \end{bmatrix}$

Derivation of Normal Equations (2)

Matrix Equation: $\mathbf{Ab} = \mathbf{y} + \mathbf{r}$

Let: $\mathbf{A} = \left[\mathbf{1} \;\middle|\; \mathbf{a}_0 \;\middle|\; \mathbf{a}_1 \;\middle|\; \cdots \;\middle|\; \mathbf{a}_{n-1}\right]$ where: $\mathbf{a}_j = \begin{bmatrix} x_j^0 \\ x_j^1 \\ \vdots \\ x_j^{n-1} \end{bmatrix}$

$$\mathbf{Ab} = \left[\mathbf{1} \;\middle|\; \mathbf{a}_0 \;\middle|\; \mathbf{a}_1 \;\middle|\; \cdots \;\middle|\; \mathbf{a}_{p-1}\right] \begin{bmatrix} a \\ b_0 \\ b_1 \\ \vdots \\ b_{p-1} \end{bmatrix}$$

$$= a\mathbf{1} + b_0\mathbf{a}_0 + b_1\mathbf{a}_1 + \cdots + b_{p-1}\mathbf{a}_{p-1}$$

$$= b_{-1}\mathbf{a}_{-1} + b_0\mathbf{a}_0 + b_1\mathbf{a}_1 + \cdots + b_{p-1}\mathbf{a}_{p-1} = \sum_{j=-1}^{p-1} b_j\mathbf{a}_j$$

where $b_{-1} = a$ and $\mathbf{a}_{-1} = \mathbf{1}$.

Derivation of Normal Equations (3)

Note that $\mathbf{r} = \mathbf{A}\mathbf{b} - \mathbf{y}$ is a vector. The objective function is

$$\begin{aligned}\mathcal{E}(b_0, b_1, \ldots, b_{p-1}) &= \sum_{i=0}^{n-1} |f(\mathbf{x}_i) - y_i|^2 = \sum r_i^2 \\ &= \mathbf{r}^\mathsf{T}\mathbf{r} = (\mathbf{A}\mathbf{b} - \mathbf{y})^\mathsf{T}(\mathbf{A}\mathbf{b} - \mathbf{y})\end{aligned}$$

Derivation of Normal Equations (3)

Note that $\mathbf{r} = \mathbf{A}\mathbf{b} - \mathbf{y}$ is a vector. The objective function is

$$\begin{aligned} \mathcal{E}(b_0, b_1, \ldots, b_{p-1}) &= \sum_{i=0}^{n-1} |f(\mathbf{x}_i) - y_i|^2 = \sum r_i^2 \\ &= \mathbf{r}^\mathsf{T}\mathbf{r} = (\mathbf{A}\mathbf{b} - \mathbf{y})^\mathsf{T}(\mathbf{A}\mathbf{b} - \mathbf{y}) \end{aligned}$$

Differentiate to minimize:

$$\frac{\partial \mathcal{E}}{\partial b_i} = \left[\frac{\partial}{\partial b_i}(\mathbf{A}\mathbf{b} - \mathbf{y})^\mathsf{T}\right](\mathbf{A}\mathbf{b} - \mathbf{y}) + (\mathbf{A}\mathbf{b} - \mathbf{y})^\mathsf{T}\left[\frac{\partial}{\partial b_i}(\mathbf{A}\mathbf{b} - \mathbf{y})\right]$$

Derivation of Normal Equations (3)

Note that $\mathbf{r} = \mathbf{A}\mathbf{b} - \mathbf{y}$ is a vector. The objective function is

$$\begin{aligned}\mathcal{E}(b_0, b_1, \ldots, b_{p-1}) &= \sum_{i=0}^{n-1} |f(\mathbf{x}_i) - y_i|^2 = \sum r_i^2 \\ &= \mathbf{r}^\mathsf{T}\mathbf{r} = (\mathbf{A}\mathbf{b} - \mathbf{y})^\mathsf{T}(\mathbf{A}\mathbf{b} - \mathbf{y})\end{aligned}$$

Differentiate to minimize:

$$\frac{\partial \mathcal{E}}{\partial b_i} = \left[\frac{\partial}{\partial b_i}(\mathbf{A}\mathbf{b} - \mathbf{y})^\mathsf{T}\right](\mathbf{A}\mathbf{b} - \mathbf{y}) + (\mathbf{A}\mathbf{b} - \mathbf{y})^\mathsf{T}\left[\frac{\partial}{\partial b_i}(\mathbf{A}\mathbf{b} - \mathbf{y})\right]$$

But:

$$\frac{\partial(\mathbf{A}\mathbf{b} - \mathbf{y})}{\partial b_i} = \frac{\partial(\mathbf{A}\mathbf{b})}{\partial b_i} = \frac{\partial}{\partial b_i}\sum_{j=-1}^{p-1} b_j \mathbf{a}_j = \mathbf{a}_i$$

Derivation of Normal Equations (4)

$$0 = \frac{\partial \mathcal{E}}{\partial b_i} = (\mathbf{a}_i)^\mathsf{T}(\mathbf{Ab} - \mathbf{y}) + (\mathbf{Ab} - \mathbf{y})^\mathsf{T}\mathbf{a}_i$$

Derivation of Normal Equations (4)

$$
\begin{aligned}
0 = \frac{\partial \mathcal{E}}{\partial b_i} &= (\mathbf{a}_i)^\top (\mathbf{A}\mathbf{b} - \mathbf{y}) + (\mathbf{A}\mathbf{b} - \mathbf{y})^\top \mathbf{a}_i \\
&= \mathbf{a}_i^\top \mathbf{A}\mathbf{b} - \mathbf{a}_i^\top \mathbf{y} + (\mathbf{A}\mathbf{b})^\top \mathbf{a}_i - \mathbf{y}^\top \mathbf{a}_i
\end{aligned}
$$

Derivation of Normal Equations (4)

$$
\begin{aligned}
0 = \frac{\partial \mathcal{E}}{\partial b_i} &= (\mathbf{a}_i)^\top(\mathbf{Ab} - \mathbf{y}) + (\mathbf{Ab} - \mathbf{y})^\top \mathbf{a}_i \\
&= \mathbf{a}_i^\top \mathbf{Ab} - \mathbf{a}_i^\top \mathbf{y} + (\mathbf{Ab})^\top \mathbf{a}_i - \mathbf{y}^\top \mathbf{a}_i \\
&= (\mathbf{a}_i^\top \mathbf{Ab}) + (\mathbf{a}_i^\top \mathbf{Ab})^\top - \mathbf{a}_i^\top \mathbf{y} - (\mathbf{a}_i^\top \mathbf{y})^\top
\end{aligned}
$$

Derivation of Normal Equations (4)

$$
\begin{aligned}
0 = \frac{\partial \mathcal{E}}{\partial b_i} &= (\mathbf{a}_i)^\mathsf{T}(\mathbf{Ab} - \mathbf{y}) + (\mathbf{Ab} - \mathbf{y})^\mathsf{T}\mathbf{a}_i \\
&= \mathbf{a}_i^\mathsf{T}\mathbf{Ab} - \mathbf{a}_i^\mathsf{T}\mathbf{y} + (\mathbf{Ab})^\mathsf{T}\mathbf{a}_i - \mathbf{y}^\mathsf{T}\mathbf{a}_i \\
&= (\mathbf{a}_i^\mathsf{T}\mathbf{Ab}) + (\mathbf{a}_i^\mathsf{T}\mathbf{Ab})^\mathsf{T} - \mathbf{a}_i^\mathsf{T}\mathbf{y} - (\mathbf{a}_i^\mathsf{T}\mathbf{y})^\mathsf{T} \\
&= 2(\mathbf{a}_i^\mathsf{T}\mathbf{Ab}) - 2\mathbf{a}_i^\mathsf{T}\mathbf{y} \quad \text{A scalar is its own transpose}
\end{aligned}
$$

Derivation of Normal Equations (4)

$$
\begin{aligned}
0 = \frac{\partial \mathcal{E}}{\partial b_i} &= (\mathbf{a}_i)^\mathsf{T}(\mathbf{Ab} - \mathbf{y}) + (\mathbf{Ab} - \mathbf{y})^\mathsf{T}\mathbf{a}_i \\
&= \mathbf{a}_i^\mathsf{T}\mathbf{Ab} - \mathbf{a}_i^\mathsf{T}\mathbf{y} + (\mathbf{Ab})^\mathsf{T}\mathbf{a}_i - \mathbf{y}^\mathsf{T}\mathbf{a}_i \\
&= (\mathbf{a}_i^\mathsf{T}\mathbf{Ab}) + (\mathbf{a}_i^\mathsf{T}\mathbf{Ab})^\mathsf{T} - \mathbf{a}_i^\mathsf{T}\mathbf{y} - (\mathbf{a}_i^\mathsf{T}\mathbf{y})^\mathsf{T} \\
&= 2(\mathbf{a}_i^\mathsf{T}\mathbf{Ab}) - 2\mathbf{a}_i^\mathsf{T}\mathbf{y} \quad \text{A scalar is its own transpose} \\
\mathbf{a}_i^\mathsf{T}\mathbf{y} &= \mathbf{a}_i^\mathsf{T}\mathbf{Ab} \quad \text{true for all i}
\end{aligned}
$$

Derivation of Normal Equations (4)

$$
\begin{aligned}
0 = \frac{\partial \mathcal{E}}{\partial b_i} &= (\mathbf{a}_i)^\mathsf{T}(\mathbf{A}\mathbf{b} - \mathbf{y}) + (\mathbf{A}\mathbf{b} - \mathbf{y})^\mathsf{T}\mathbf{a}_i \\
&= \mathbf{a}_i^\mathsf{T}\mathbf{A}\mathbf{b} - \mathbf{a}_i^\mathsf{T}\mathbf{y} + (\mathbf{A}\mathbf{b})^\mathsf{T}\mathbf{a}_i - \mathbf{y}^\mathsf{T}\mathbf{a}_i \\
&= (\mathbf{a}_i^\mathsf{T}\mathbf{A}\mathbf{b}) + (\mathbf{a}_i^\mathsf{T}\mathbf{A}\mathbf{b})^\mathsf{T} - \mathbf{a}_i^\mathsf{T}\mathbf{y} - (\mathbf{a}_i^\mathsf{T}\mathbf{y})^\mathsf{T} \\
&= 2(\mathbf{a}_i^\mathsf{T}\mathbf{A}\mathbf{b}) - 2\mathbf{a}_i^\mathsf{T}\mathbf{y} \quad \text{A scalar is its own transpose} \\
\mathbf{a}_i^\mathsf{T}\mathbf{y} &= \mathbf{a}_i^\mathsf{T}\mathbf{A}\mathbf{b} \quad \text{true for all i} \\
\mathbf{A}^\mathsf{T}\mathbf{y} &= \mathbf{A}^\mathsf{T}\mathbf{A}\mathbf{b} \quad \textbf{Normal Equations}
\end{aligned}
$$

Derivation of Normal Equations (4)

$$
\begin{aligned}
0 = \frac{\partial \mathcal{E}}{\partial b_i} &= (\mathbf{a}_i)^\mathsf{T}(\mathbf{Ab} - \mathbf{y}) + (\mathbf{Ab} - \mathbf{y})^\mathsf{T}\mathbf{a}_i \\
&= \mathbf{a}_i^\mathsf{T}\mathbf{Ab} - \mathbf{a}_i^\mathsf{T}\mathbf{y} + (\mathbf{Ab})^\mathsf{T}\mathbf{a}_i - \mathbf{y}^\mathsf{T}\mathbf{a}_i \\
&= (\mathbf{a}_i^\mathsf{T}\mathbf{Ab}) + (\mathbf{a}_i^\mathsf{T}\mathbf{Ab})^\mathsf{T} - \mathbf{a}_i^\mathsf{T}\mathbf{y} - (\mathbf{a}_i^\mathsf{T}\mathbf{y})^\mathsf{T} \\
&= 2(\mathbf{a}_i^\mathsf{T}\mathbf{Ab}) - 2\mathbf{a}_i^\mathsf{T}\mathbf{y} \quad \text{A scalar is its own transpose} \\
\mathbf{a}_i^\mathsf{T}\mathbf{y} &= \mathbf{a}_i^\mathsf{T}\mathbf{Ab} \quad \text{true for all i} \\
\mathbf{A}^\mathsf{T}\mathbf{y} &= \mathbf{A}^\mathsf{T}\mathbf{Ab} \quad \textbf{Normal Equations} \\
\mathbf{b} &= (\mathbf{A}^\mathsf{T}\mathbf{A})^{-1}\mathbf{A}^\mathsf{T}\mathbf{y} \quad \text{Solution of Normal Equations}
\end{aligned}
$$

Normal Equations: remarks

- The components of $\mathbf{x}$ are called **features**

Normal Equations: remarks

- The components of **x** are called **features**
- The matrix **F** with input vectors as rows is called a **feature matrix**. If there are 3 features, e.g., **x** = (x, y, z)

$$\mathbf{F} = \begin{bmatrix} x_0 & y_0 & z_0 \\ x_1 & y_1 & z_1 \\ \vdots & & \\ x_{n-1} & y_{n-1} & z_{n-1} \end{bmatrix}$$

Thus $\mathbf{A} = \left[\mathbf{1} \mid \mathbf{F}\right]$

Normal Equations: remarks

- The components of **x** are called **features**
- The matrix **F** with input vectors as rows is called a **feature matrix**. If there are 3 features, e.g., $\mathbf{x} = (x, y, z)$

$$\mathbf{F} = \begin{bmatrix} x_0 & y_0 & z_0 \\ x_1 & y_1 & z_1 \\ \vdots & & \\ x_{n-1} & y_{n-1} & z_{n-1} \end{bmatrix}$$

Thus $\mathbf{A} = \left[\mathbf{1} \mid \mathbf{F}\right]$

- Normal Matrix $\mathbf{N} = \mathbf{A}^\mathsf{T}\mathbf{A}$ for 3 features, e.g., $\mathbf{x} = (x, y, z)$

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_0 & x_1 & \cdots & x_{n-1} \\ y_0 & y_1 & \cdots & y_{n-1} \\ z_0 & z_1 & \cdots & z_{n-1} \end{bmatrix} \begin{bmatrix} 1 & x_0 & y_0 & z_0 \\ 1 & x_1 & y_1 & z_1 \\ \vdots & & & \\ 1 & x_{n-1} & y_{n-1} & z_{n-1} \end{bmatrix} = \begin{bmatrix} n & \sum x_i & \sum y_i & \sum z_i \\ \sum x_i & \sum x_i^2 & \sum x_i y_i & \sum x_i z_i \\ \sum y_i & \sum x_i y_i & \sum y_i^2 & \sum y_i z_i \\ \sum z_i & \sum x_i z_i & \sum y_i z_i & \sum z_i^2 \end{bmatrix}$$

Multilinear Regression Using sklearn (1)

```
import pandas as pd
data = pd.read_csv("https://archive.ics.uci.edu/ml/
   machine-learning-databases/cpu-performance/
   machine.data",header=None)
data.columns=["vendor","Model","MYCT","MMIN",
   "MMAX","CACH","CHMIN","CHMAX","PRP","ERP"]
print(data[:5])
```

vendor	Model	MYCT	MMIN	MMAX	CACH	CHMIN	CHMAX	PRP
0	adviser	32/60	125	256	6000	256	16	128
1	amdahl	470v/7	29	8000	32000	32	8	32
2	amdahl	470v/7a	29	8000	32000	32	8	32
3	amdahl	470v/7b	29	8000	32000	32	8	32
4	amdahl	470v/7c	29	8000	16000	32	8	16

Multilinear Regression Using sklearn (2)

Extract five features for X data.

```
import numpy as np
X=np.array(data[["MYCT","MMIN","MMAX","CACH","CHMIN",
    "CHMAX"]])
Y=np.array(data["PRP"]).reshape(-1,1)
```

Multilinear Regression Using sklearn (2)

Extract five features for X data.

```
import numpy as np
X=np.array(data[["MYCT","MMIN","MMAX","CACH","CHMIN",
    "CHMAX"]])
Y=np.array(data["PRP"]).reshape(-1,1)
```

Create train/test split:

```
from sklearn.model_selection import train_test_split
XTRAIN, XTEST, YTRAIN, YTEST = train_test_split(X,Y)
```

Multilinear Regression Using sklearn (2)

Extract five features for X data.

```
import numpy as np
X=np.array(data[["MYCT","MMIN","MMAX","CACH","CHMIN",
   "CHMAX"]])
Y=np.array(data["PRP"]).reshape(-1,1)
```

Create train/test split:

```
from sklearn.model_selection import train_test_split
XTRAIN, XTEST, YTRAIN, YTEST = train_test_split(X,Y)
```

Perform Linear regression on training data

```
from sklearn.linear_model import LinearRegression
LR=LinearRegression()
reg=LR.fit(XTRAIN,YTRAIN)
```

Multilinear Regression Using sklearn (3)

Extract Coefficients

```
print("The intercept is ", reg.intercept_)
```

```
The intercept is  [-48.68341707]
```

```
print("The coefficients are\n", reg.coef_)
```

```
The coefficients are
 [[ 0.06104161  0.01466434  0.00527632  1.02699926
 -1.77763521  1.94911414]]
```

Multilinear Regression Using sklearn (3)

Extract Coefficients

```
print("The intercept is ", reg.intercept_)
```

```
The intercept is  [-48.68341707]
```

```
print("The coefficients are\n", reg.coef_)
```

```
The coefficients are
 [[ 0.06104161  0.01466434  0.00527632  1.02699926
 -1.77763521  1.94911414]]
```

What this means:

$$\begin{aligned}\mathtt{PRP} \approx & -48.68 + 0.061 \times \mathtt{MYCT} + 0.015 \times \mathtt{MMIN} \\ & + 0.0053 \times \mathtt{MMAX} + 1.03 \times \mathtt{CACH} - 1.78 \times \mathtt{CHMIN} \\ & + 1.95 \times \mathtt{CHMAX}\end{aligned}$$

Multilinear Regression Using sklearn (4)

Evaluate Fit

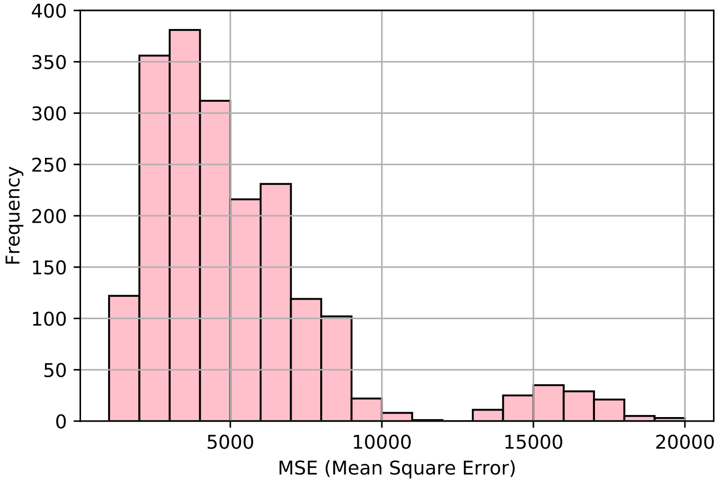
```
from sklearn.metrics import mean_squared_error, r2_score
YP=reg.predict(XTEST)
R2=r2_score(YTEST,YP)
MSE = mean_squared_error(YTEST,YP)
print("R^2=",round(R2,3)," MSE=",round(MSE,3))
```

```
R^2= 0.829  MSE= 6267.859
```

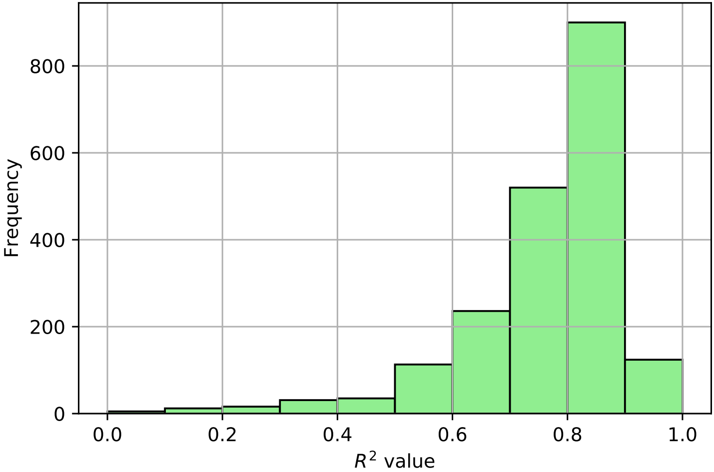
How important is the randomization?

Repeat the train/test split many times

```
r2s=[]
MSES=[]
n=2000
for j in range(n):
    XTRAIN, XTEST, YTRAIN, YTEST = train_test_split(X,Y)
    LR=LinearRegression()
    reg=LR.fit(XTRAIN,YTRAIN)
    YP=reg.predict(XTEST)
    R2=r2_score(YTEST,YP)
    MSE = mean_squared_error(YTEST,YP)
    r2s.append(R2)
    MSES.append(MSE)
```



400
350
300
250
200
150
100
50
0
Frequency
5000
10000
15000
20000
MSE (Mean Square Error)



Frequency
800
600
400
200
0
0.0
0.2
0.4
0.6
0.8
1.0
R^2 value

Remarks

- Its not unusual to compute $\mu \pm 2\sigma$ to describe the center and variability of the measures of error
- But this is based on the observation that in a normal distribution 95% of the data falls withing 1.96 (i.e., about 2) standard deviations of the mean.
- Clearly the noise in the MSE and R^2 is not normally distributed
- Its always a good idea to look at the data distribution, not just a couple of numbers that summarize it

Citations

1. Dua, D. and Karra Taniskidou, E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.
2. Kibler,D. & Aha,D. (1988). Instance-Based Prediction of Real-Valued Attributes. In Proceedings of the CSCSI (Canadian AI) Conference. (data set: [https://archive.ics.uci.edu/ml/datasets/Computer+Hardware])