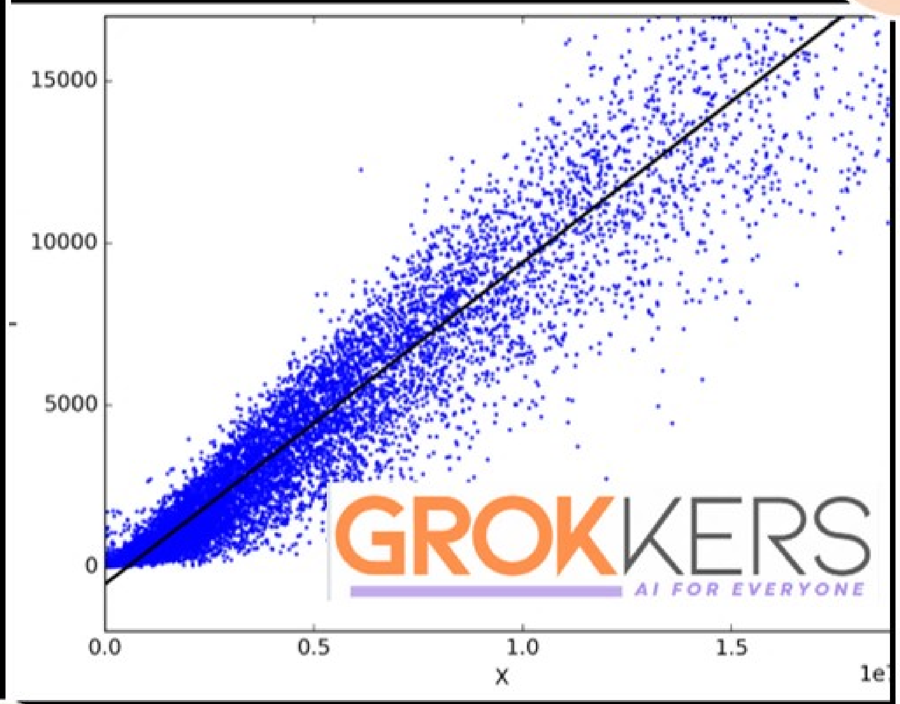
Supervised modeling Linear Regression

From concepts to Implementation with detailed explanation of the algorithm



Agenda

- basic intuition (code)
- statistical way (code)
- sklearn implementation on advertising dataset
- Model evaluations (learning curve and cross validations)
- Test of assumptions (adv dataset)
- MSE plot
- Save/load model
- effect of OHE
- effect of multi-collinearity
- non linear data

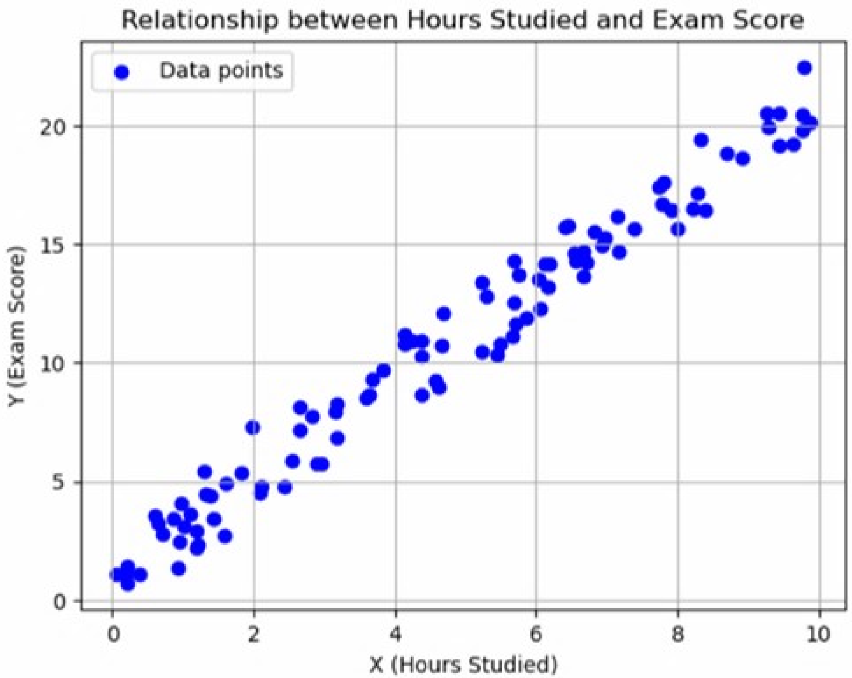
Basic intuition (linear regression)

- Suppose we want to analyze the relationship between the number of hours a student studies and their exam scores.
- We collect data from several students where we have the number of hours they studied (*X*) and their corresponding exam scores (*Y*).

Hours Studied (X)	Exam Score (Y)
2	60
3	70
4	75
5	80
6	85

Plotting the Data

- plot the data points on a scatter plot,
- with the number of hours studied on the x-axis and the exam score on the y-axis.



Observations from the plot

Direction of Relationship:	**Strength of Relationship**:	**Outliers**:	**Patterns**:
• determine whether there is a <u>positive</u> or <u>negative</u> relationship between the <u>number of hours</u> studied and the <u>exam score</u>.	• If the data points form a tight <u>cluster</u> around a line or curve, it suggests a <u>strong relationship</u>, indicating that the exam scores are closely related to the number of hours studied.	• Examining the scatter plot helps us identify any outliers or unusual data points that do not follow the general trend.	• scatter plots can reveal other patterns such as <u>quadratic</u>, <u>exponential</u>, or <u>logarithmic</u> relationships between variables.

Data

- $X = [2,3,4,5,6]$
- Y = [60,70,75,80,85]

- Step 1: Calculate the Mean of *X* and *y*

$$\bar{X} = \frac{\sum X}{n}$$
$$\bar{y} = \frac{\sum y}{n}$$

$$\bar{X} = \frac{2+3+4+5+6}{5} = \frac{20}{5} = 4$$
$$\bar{y} = \frac{60+70+75+80+85}{5} = \frac{370}{5} = 74$$

Step 2: Calculate the Slope m

$$m = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(y_i - \bar{y})}{\sum_{i=1}^{n}(X_i - \bar{X})^2}$$

where *Xi* and *yi* are individual data points.

$$\begin{aligned}
m &= \frac{(2-4)(60-74)+(3-4)(70-74)+(4-4)(75-74)+(5-4)(80-74)+(6-}{(2-4)^2+(3-4)^2+(4-4)^2+(5-4)^2+(6-4)^2} \\
&= \frac{(-2)(-14)+(-1)(-4)+(0)(1)+(1)(6)+(2)(11)}{(-2)^2+(-1)^2+(0)^2+(1)^2+(2)^2} \\
&= \frac{28+4+0+6+22}{4+1+0+1+4} \\
&= \frac{60}{10} \\
&= 6
\end{aligned}$$

Step 3: Calculate the Intercept b

$$b = \bar{y} - m\bar{X}$$

$$\begin{aligned} b &= 74 - 6(4) \\ &= 74 - 24 \\ &= 50 \end{aligned}$$

So, the slope m is 6, and the intercept b is 50.

the equation of the line of best fit is
y=6X+50

use of the line of best fit

the equation

- line of best fit, $y=6X+50$

Prediction:

- use the equation to predict y (exam score) for any given value of X (number of hours studied).
- For example, if a student studies for 7 hours, the exam score: $y=6\times7+50=92$.

Understanding Relationships:

- For every additional hour a student studies ($\Delta X=1$), their predicted exam score increases by 6 points ($\Delta y=6$).
- intercept term (50) indicates that even if a student doesn't study at all ($X=0$), their predicted exam score would still be 50.

Demo using python/sklearn

(Linear regression using python)

How do we handle 2 or more predictors

- **Model Specification**:
  - Specify the multiple linear regression model by defining the equation that relates the dependent variable to ALL the predictor variables.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \epsilon$$

where:

- y is the dependent variable.
- $x_1, x_2, \ldots, x_n$ are the predictor variables.
- $\beta_0, \beta_1, \ldots, \beta_n$ are the coefficients (intercept and slopes).
- ϵ is the error term.

How do compute β0, β1 ... βn

- The equations can be formulated as

$$y_i = \beta_0 + \beta_1 X_1 + e_i$$

- If we actually let i = 1, ..., n, we see that we obtain n equations:

$$\begin{aligned} y_1 &= \beta_0 + \beta_1 X_1 + e_1 \\ y_2 &= \beta_0 + \beta_1 X_1 + e_2 \\ y_3 &= \beta_0 + \beta_1 X_1 + e_3 \\ &\cdots \\ &\cdots \\ y_n &= \beta_0 + \beta_1 X_1 + e_n \end{aligned}$$

pattern

- By taking advantage of the pattern, we can instead formulate the above simple linear regression function in matrix notation:

$$
\begin{aligned}
y_1 &= \beta_0 + \beta_1 X_1 + e_1 \\
y_2 &= \beta_0 + \beta_1 X_1 + e_2 \\
y_3 &= \beta_0 + \beta_1 X_1 + e_3 \\
&\cdots \\
&\cdots \\
y_n &= \beta_0 + \beta_1 X_1 + e_n
\end{aligned}
$$

$$
\begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} 1 & X_1 \\ 1 & X_2 \\ \dots & \\ \dots & \\ 1 & X_n \end{bmatrix} \cdot \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \vdots \\ \epsilon_n \end{bmatrix}
$$

More generally...

$$
\begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ \vdots \\ Y_n \end{bmatrix}_{n \times 1} = \begin{bmatrix} 1 & X_{11} & X_{21} & \dots & X_{k1} \\ 1 & X_{12} & X_{22} & \dots & X_{k2} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ 1 & X_{1n} & X_{22} & \dots & X_{kn} \end{bmatrix}_{n \times k} \cdot \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \vdots \\ \beta_k \end{bmatrix}_{k \times 1} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \vdots \\ \epsilon_n \end{bmatrix}_{n \times 1}
$$

$$y = X\beta + \epsilon$$

Criteria for Estimates (beta coeffs)

- find the estimator β that minimizes the sum of squared residuals $\sum e_i^2$
- vector of residuals e is given by: $e = y - X\beta$
- sum of squared residuals (RSS) is $e^{\top}e$

$$\begin{bmatrix} e_1 & e_2 & \dots & \dots & e_n \end{bmatrix}_{1 \times n} \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ \vdots \\ e_n \end{bmatrix}_{n \times 1} = \begin{bmatrix} e_1 * e_1 + e_2 * e_2 + \dots \dots e_n * e_n \end{bmatrix}_{1 \times 1}$$

Goal is

- Minimize the error
- How
    - Taking derivative
- Why?

$$e^\top e = (y - X\beta)^\top (y - X\beta) = y^\top y - \beta^\top X^\top y - y^\top X\beta + \beta^\top X^\top X\beta = y^\top y - 2\beta^\top X^\top y + \beta^\top X^\top X\beta$$

$$\frac{d}{d\beta} e^\top e = 0$$

$$-2X^\top y + 2X^\top X\beta = 0$$

$$(X^\top X)\beta = X^\top y$$

$$\beta = (X^\top X)^{-1}(X^\top y)$$

Objective of Minimization

01

goal of linear regression is to find the line (or plane in higher dimensions) that <u>best fits the data</u>.

02

We define "best fit" as the line that <u>minimizes the differences</u> between the <u>observed</u> values of the dependent variable and the values <u>predicted</u> by the model.

03

we want to <u>minimize</u> the sum of squared differences between observed and predicted values, which is SSR/ MSE

Derivative and Optimization

In calculus, finding the minimum or maximum of a function often involves taking the derivative of the function with respect to the variable of interest and setting it equal to zero.

This is because at the minimum or maximum point of a function, the derivative (slope) is zero.

In the case of SSR, we want to find the coefficients (**B**) that minimize SSR, so we take the derivative of SSR with respect to **B** and set it equal to zero.

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

Sum of Squares:

- Each element $(\mathbf{X}^T\mathbf{X})_{ij}$ in the matrix represents the sum of the products of the i-th and j-th columns of $\mathbf{X}$.
- captures the "squared" aspect, as it measures the squared magnitudes of the relationships between different predictor variables.
- For example, if $\mathbf{X}$ contains variables for both height and weight, then $(\mathbf{X}^T\mathbf{X})_{ij}$ would capture how height and weight interact, providing information about their joint influence on the dependent variable.

Covariance Matrix:

- $\mathbf{X}^T\mathbf{X}$ can be seen as a covariance matrix,
- captures the variability and relationships among the predictor variables.

Data (X) and X^TX

Some dummy data of 10 rows and 4 columns

```
# Create a dummy X matrix with 10 rows and 4 columns
X = np.random.rand(10, 4)
X
```

```
array([[0.16229901, 0.17844609, 0.23413847, 0.01414331],
       [0.32572355, 0.90879085, 0.47121831, 0.11474841],
       [0.40889962, 0.68183987, 0.07671828, 0.97481349],
       [0.6930081 , 0.81073777, 0.57046056, 0.68612582],
       [0.93838685, 0.48562115, 0.39653582, 0.10438776],
       [0.32200603, 0.73740054, 0.74076723, 0.7500362 ],
       [0.58710261, 0.57405294, 0.15848482, 0.67487777],
       [0.69577519, 0.68485129, 0.58051587, 0.79589448],
       [0.25811869, 0.0207403 , 0.70045795, 0.51519803],
       [0.27158632, 0.45453005, 0.64718974, 0.46576465]])
```

```
# Calculate X^T * X
XTX = np.dot(X.T, X)
XTX
```

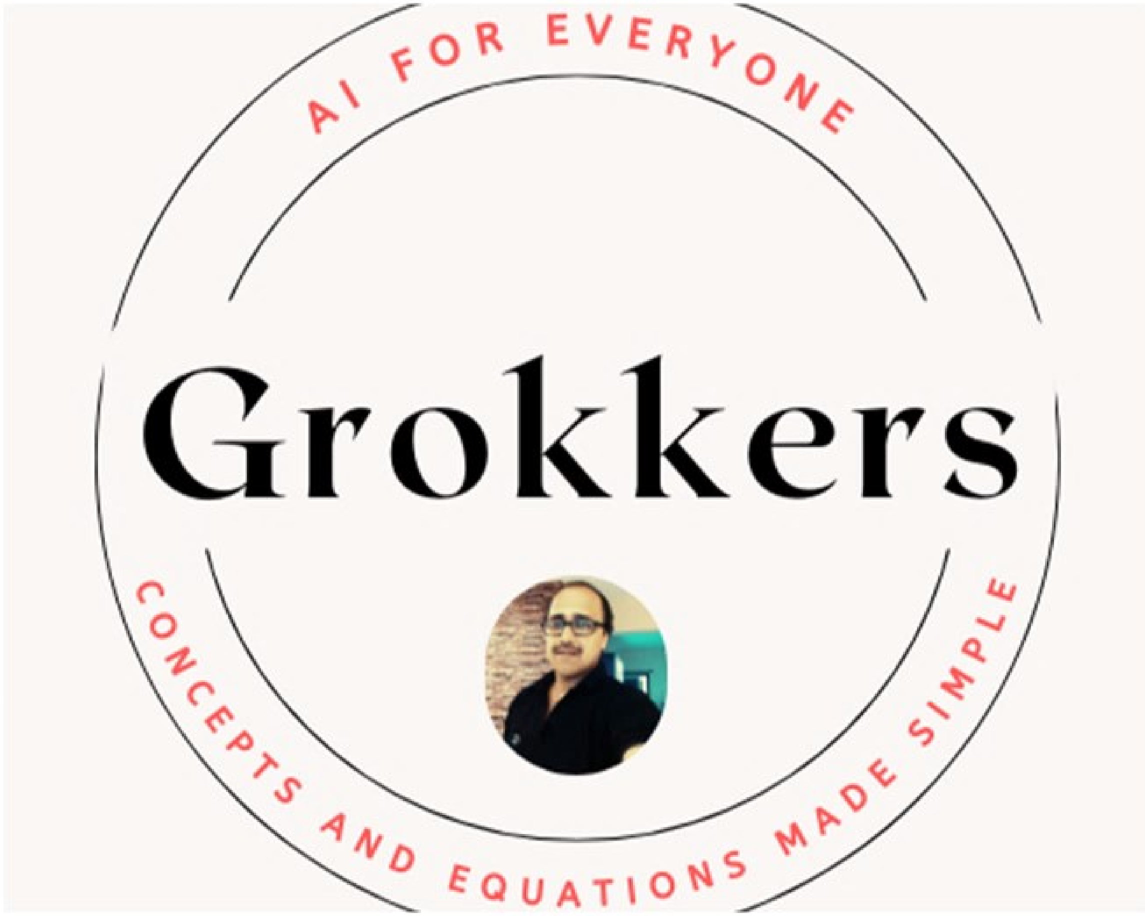
```
array([[3.3498798 , 2.28396234, 2.43730245, 2.88764236],
       [2.28396234, 2.7351605 , 1.97321068, 2.44146574],
       [2.43730245, 1.97321068, 2.50345816, 2.3802519 ],
       [2.88764236, 2.44146574, 2.3802519 , 3.41390663]])
```

Demo using python/sklearn

(Linear regression using matrix formula)

PY

Thanks


AI FOR EVERYONE
Grokkers
CONCEPTS AND EQUATIONS MADE SIMPLE