Linear Regression - 1

How does the data look for Linear Regression?

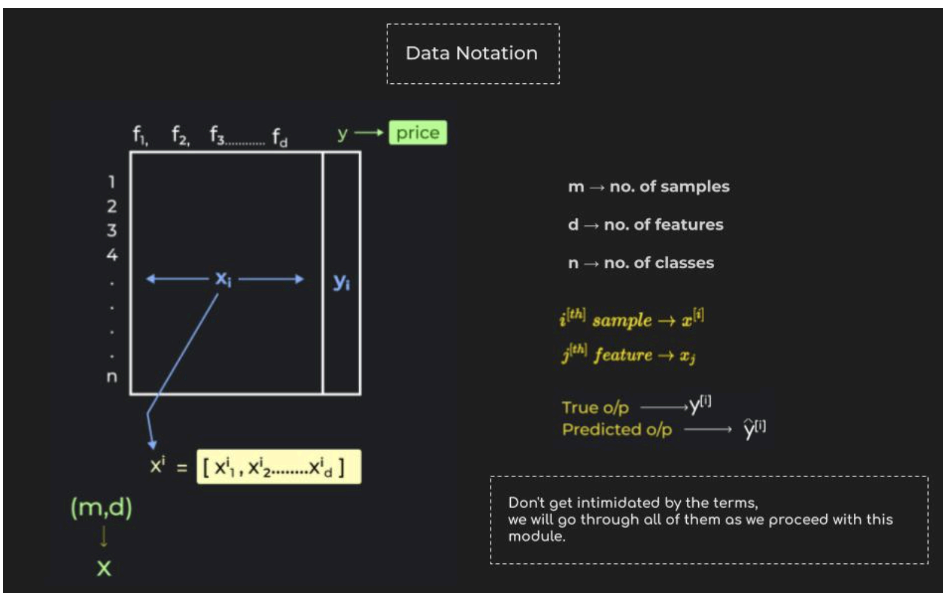
Regression Problem → Data: n samples with $[f_1, f_2, \ldots\ldots, f_d] \epsilon R^{(n \times d)}$ where

- sample X_i consists the features
- That has a target label $y_i \epsilon R$.

How does the training data look for the Regression problem?

Regression → **Supervised** task, the **target** y_i is **numerical**

1. Input sample = $X_i \epsilon R^{nXd}$, $x_i = [x_{i1}, \ldots\ldots, x_{id}]^T$
2. Output sample = $y_i \epsilon R$
3. Training example = (X_i, y_i)
4. Training Dataset = $\{(X_i, y_i),\ i = 1, 2, \ldots n\}$,



What will be the simplest model for predicting a value?

Ans: Mean model → the mean of the entire data as its prediction.

What is the goal of the ML model?

Ans: To find $f: X \rightarrow y$ such that $f(x_i) \approx y_i$

How do we define function f?

Algebraic Intuition → find $\hat{y} = f(x_i)$,

we can say for Linear Regression:

$$f(x_{i1}, x_{i2}, \ldots\ldots, x_{id}) = w_1 x_{i1} + w_2 x_{i2} + w_3 x_{i3} + \ldots. + w_d x_{id} + w_0$$

$$\hat{y}_i = f(x_i) = \sum_{j=1}^{D} w_j x_{ij} + w_0$$

Now, $w^T = [w_1, w_2 \ldots.. w_d]$ and $x_i = [x_{i1}, x_{i2} \ldots.. x_{id}]$, then:

$$\hat{y}_i = f(x_i) = w^T x_i + w_0$$

How does the ML model find the function f?

Ans: By updating the weights of the model on the training dataset

Is $f(x_i)$ in Linear Regression analogous to $y = mx + c$?

Ans: Yes, it is.

Linear Regression: finding the best D Dimensional hyperplane that fits the D-dimensional data such that $\hat{y}_q \approx y_q$

How to find the best-fit line of the Linear Regression model?

Ans: By optimizing the weights vector $W^T = [w_1, w_2, \ldots\ldots, w_d]$ w.r.t the loss function.

How do you say Linear Regression is optimized?

Ans: when we see the loss function is not decreasing anymore i.e. local minima

Sklearn-code

```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(X1_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

What loss function to use for linear regression optimization?

Mean Square Error→finds the mean of the square difference between $\hat{y}$, y .

$$min_{w,w_0} \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

After training the model, how do we measure model performance?

Ans: R-squared **metric**. → measures the performance of Linear Regression over a mean model. It is Defined as:

$$R^2 = 1 - \frac{SS_{res}}{SS_{total}} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2}$$, where $\bar{y}_i$ is the mean model

SS_{res} - Squared sum of error of regression line
SS_{total}-(total sum of squares) Squared sum of error of mean line

Range: Practically, R^2 ranges from (0, 1]

Sklearn code

LinearRegression's default `.score()` function uses R2 score to evaluate the data.

```
model.score(X_train, y_train)
```

```
0.9459004943250285
```

```
[ ] model.score(X_test, y_test)
```

```
0.945987722055055
```

What will be the best value of R^2 ?

Ans: 1, when $\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = 0$.

What will be the minimum value of R^2 ?

Ans: - ∞ , when $\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 >> \sum_{i=1}^{n}(y_i - \bar{y}_i)^2$