. 4th grade student at INSAT university

. Instructor at Data OverFlow

. Member of the engineering team at Data Colab

. Member of the core team at DSC INSAT

achraf.fares@insat.u-carthage.tn

/faresachref

Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

summary

01
Recapitulation

Small recap about supervised Machine learning

02
Supervised ML Metrics

How can we evaluate our models ?

03
Regression Models

.Linear Regression model

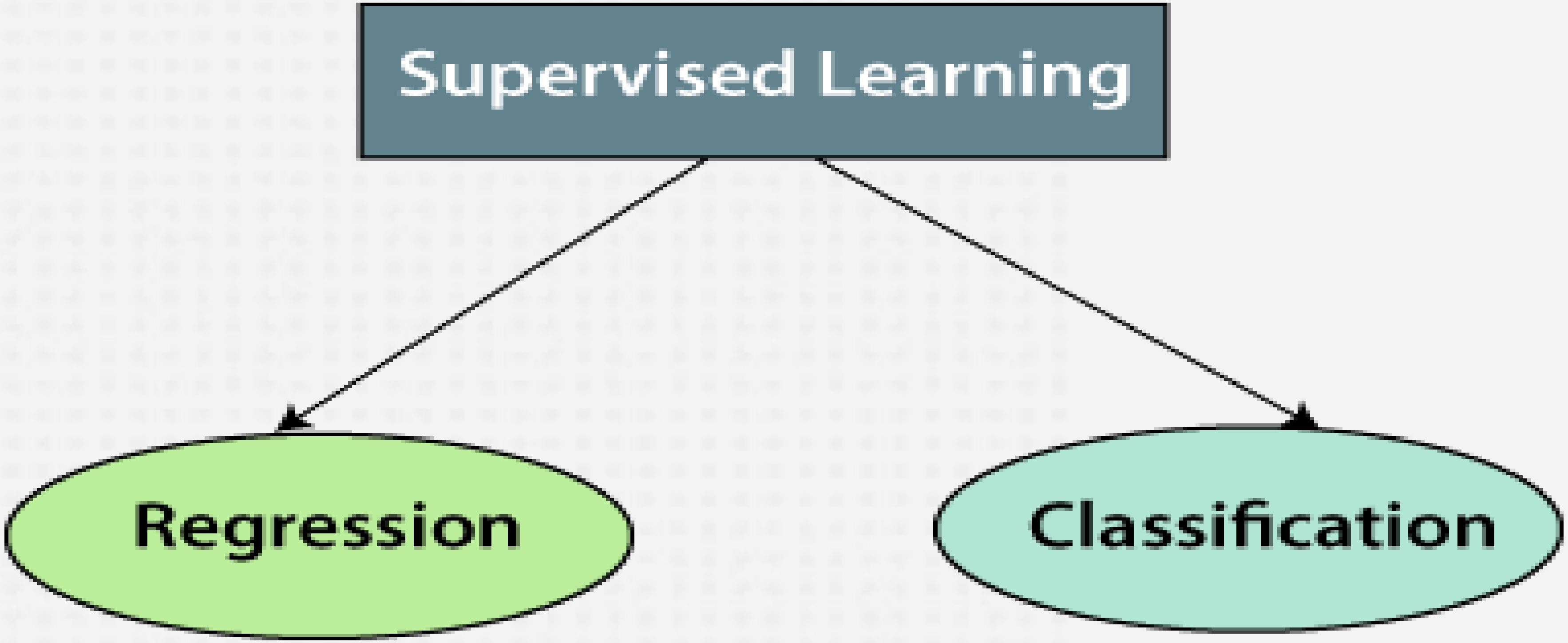
.Polynomial Regression model

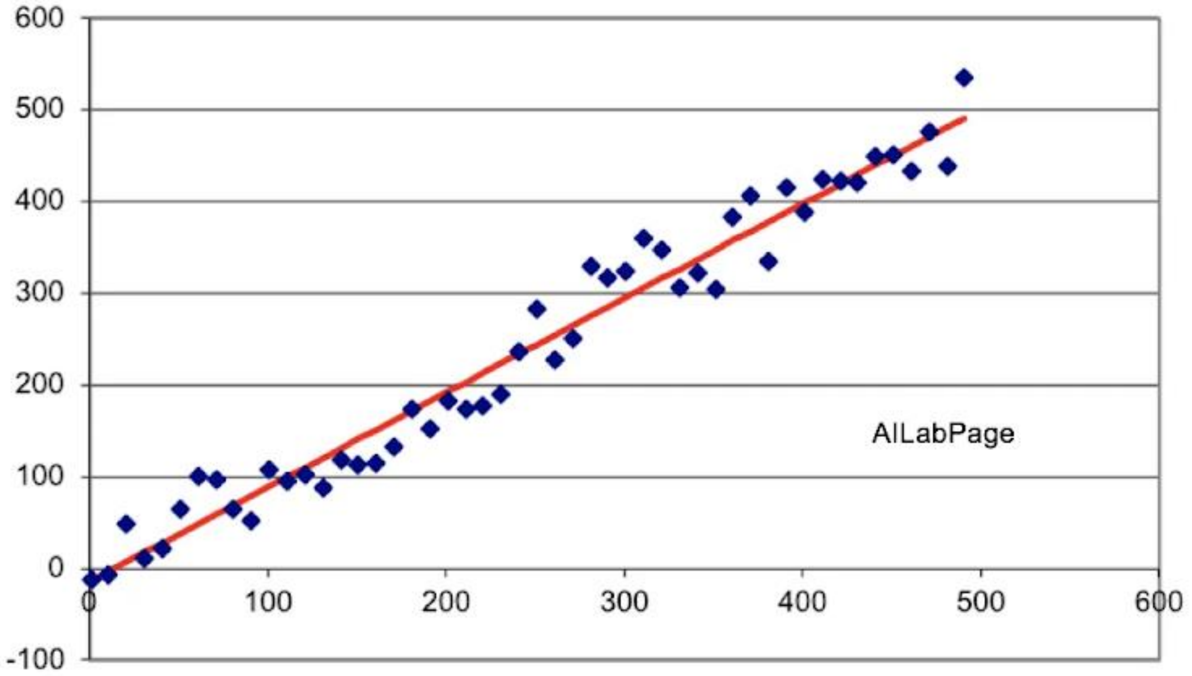
04
Classification Models

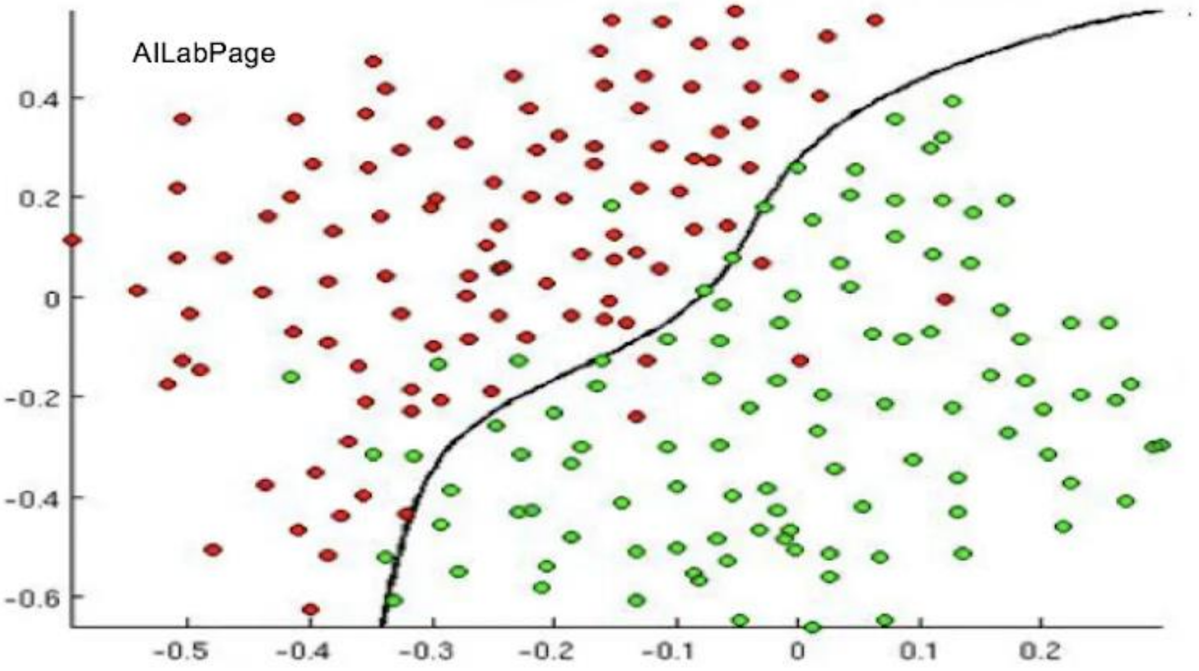
.Logistic Regression model
.DecisionTreeClassifer model

Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

Recapitulation :







Regression

The system attempts to predict a value for an input based on past data.

Example – 1. Temperature for tomorrow

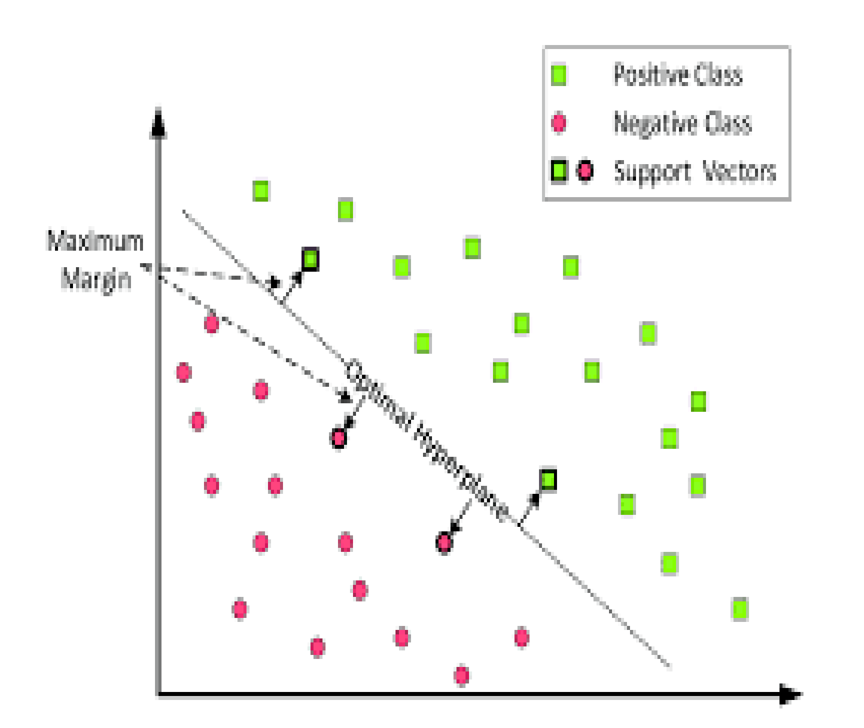
Classification

In classification, predictions are made by classifying them into different categories.

Example – 1. Type of cancer 2. Cancer Y/N

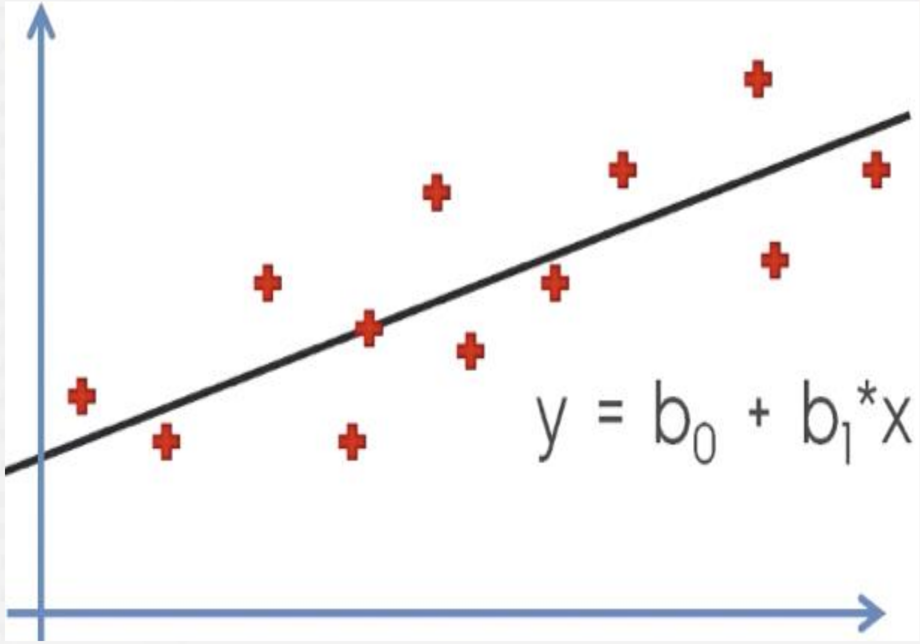
Classification :

•In Classification, the output variable must be a discrete value.
•The task of the classification algorithm is to map the input value(x) with the discrete output variable(y).
•In Classification, we try to find the decision boundary, which can divide the dataset into different classes.



Regression :

•In Regression, the output variable must be of continuous nature or real value
•The task of the regression algorithm is to map the input value (x) with the continuous output variable(y)
•In Regression, we try to find the best fit line, which can predict the output more accurately.



Machine Learning Cycle :

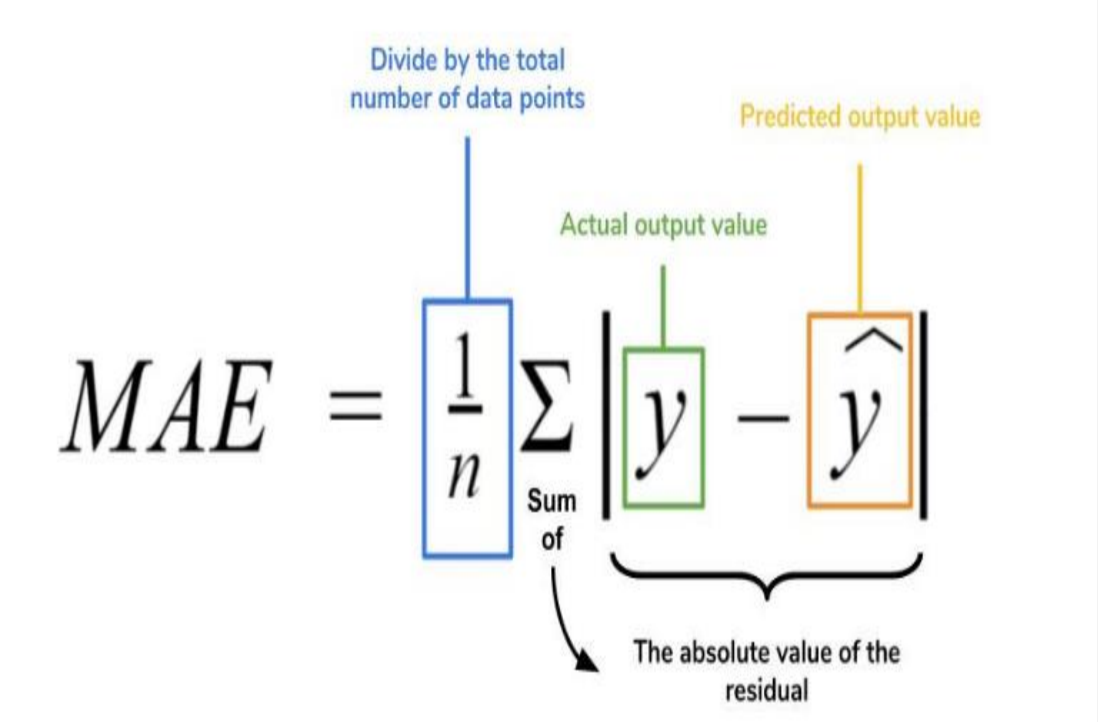
Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

Regression Metrics

Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

Regression Metrics :

1. Mean absolute error (MAE) :



Pros :

. MAE is a linear score which means all the individual differences are weighted equally

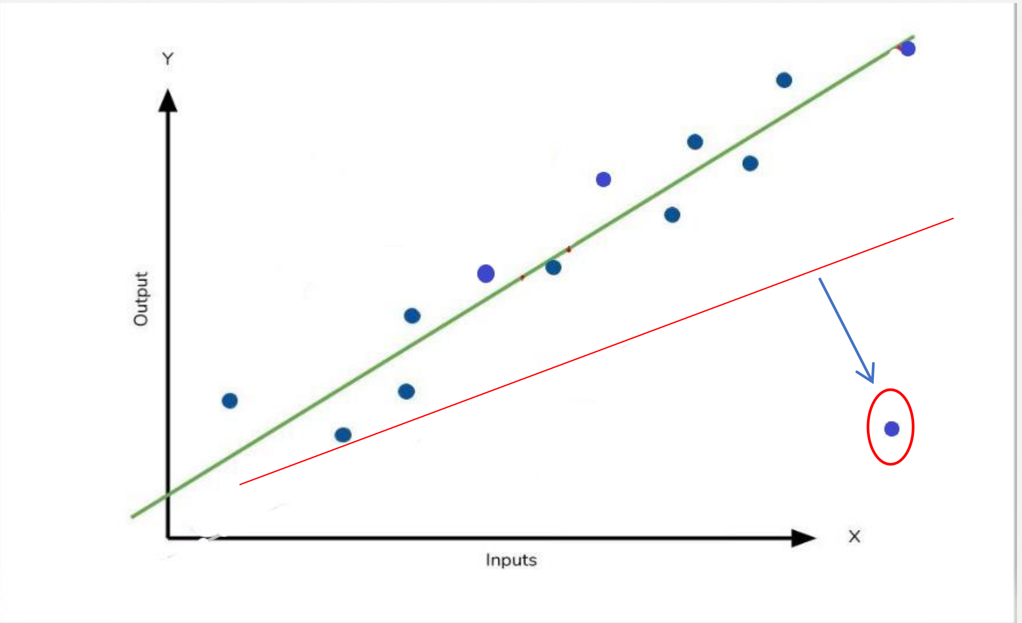
. The MAE is robust to outliers and does not penalize the errors as extremely

Cons :

. It is not suitable for applications where you want to pay more attention to the outliers

2.Mean Sequared error (MSE) :

$$MSE = \frac{1}{n}\Sigma\underbrace{\left(y - \hat{y}\right)^{2}}_{\text{The square of the difference between actual and predicted}}$$



Pros :

. Outliers will produce these exponentially larger differences, and it is our job to judge how we should approach them.

.

Cons :

As it squares the differences, it penalizes even a small error which leads to over-estimation of how bad the model

. If we had many outliers and we try to minimize the MSE in order to outperform our model we can end with decreasing our model performance.

Root Mean Squared Error(RMSE):

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}}$$

Because the MSE is squared, its units do not match that of the original output. Researchers will often use RMSE to convert the error metric back into similar units, making interpretation easier.

R-squared error (r2_score):

$$R^2 = 1 - \frac{RSS}{TSS}$$

R^2 = coefficient of determination

RSS = sum of squares of residuals (error)

TSS = total sum of squares

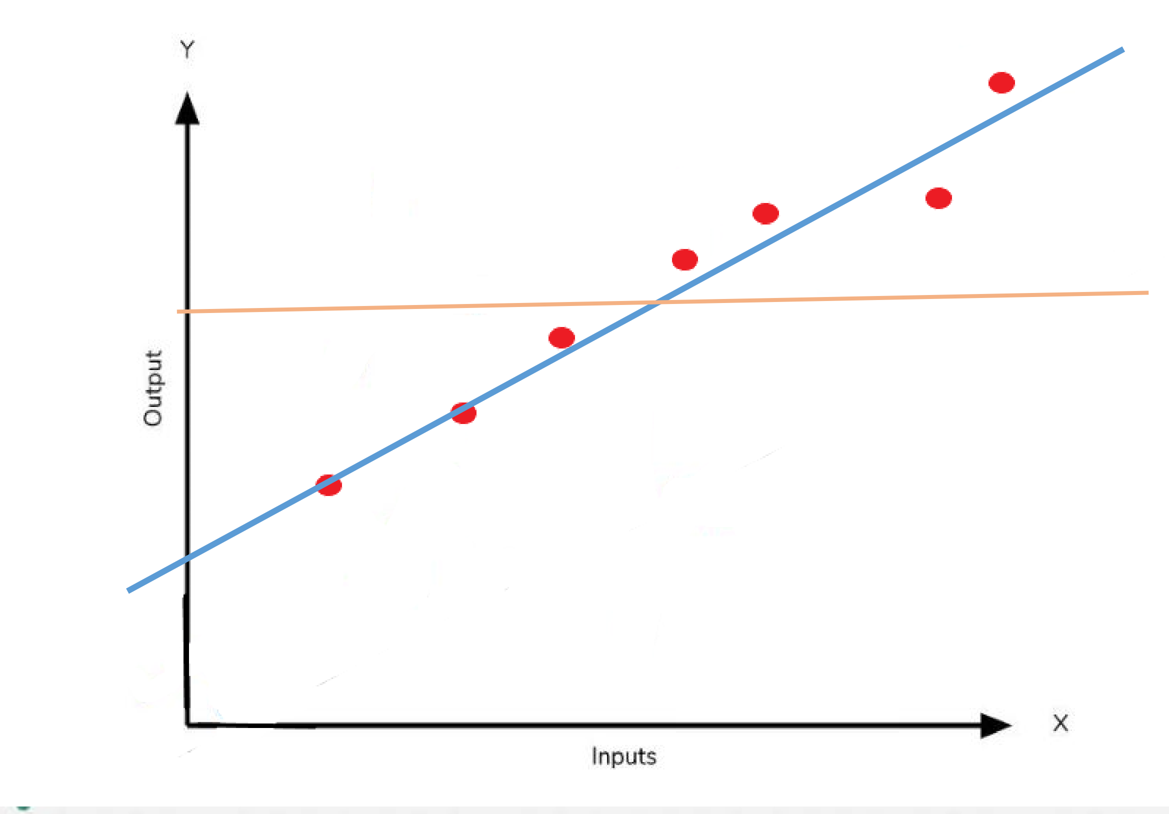
$$R^2 = 1 - \frac{\text{MSE(model)}}{\text{MSE(baseline)}}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

The Squared error (SE)

The Squared error if we choose the mean of the output data as an estimator

The metric helps us to compare our current model with a constant baseline and tells us how much our model is better.



Baseline = Ɏ : average of output

SE(Ɏ) = 50

Model = $\hat{Y}$

SE($\hat{Y}$) = 2

R^2 = 0.96

Classification Metrics

Data
Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

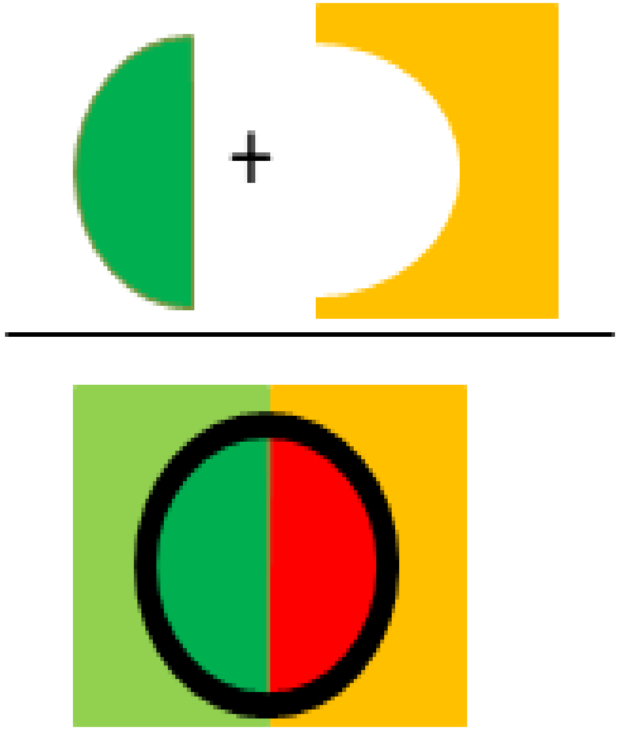
Confusion Matrix :

	Predicted 0	Predicted 1
Actual 0	TN	FP
Actual 1	FN	TP

- **True Positives** : We predicted YES and the actual output was also YES.
- **True Negatives** : We predicted NO and the actual output was NO.
- **False Positives** : We predicted YES and the actual output was NO.
- **False Negatives** : We predicted NO and the actual output was YES.

Accuracy Score :

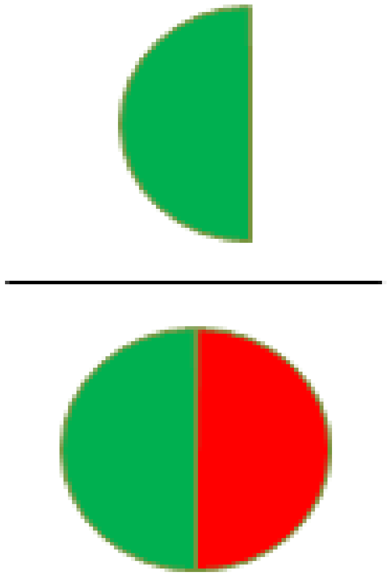
$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} =$$



Fraction predicted correctly

Precision :

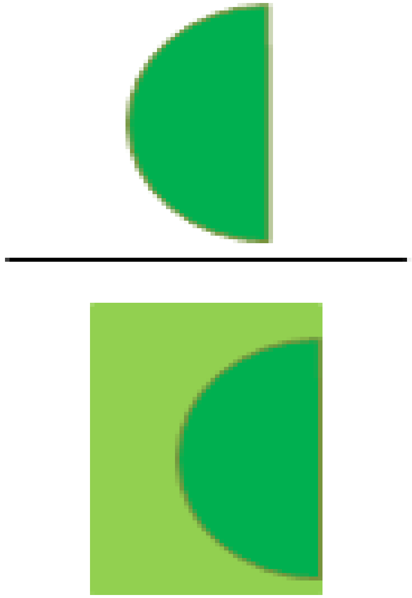
$$\text{Precision} = \frac{TP}{TP + FP} =$$



Fraction of predicted positives that are actually positive

Recall :

$$\text{Recall (Sensitivity)} = \frac{TP}{\underbrace{TP + FN}_{\text{The total of the positive data}}} =$$



Fraction of positives predicted correctly

Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

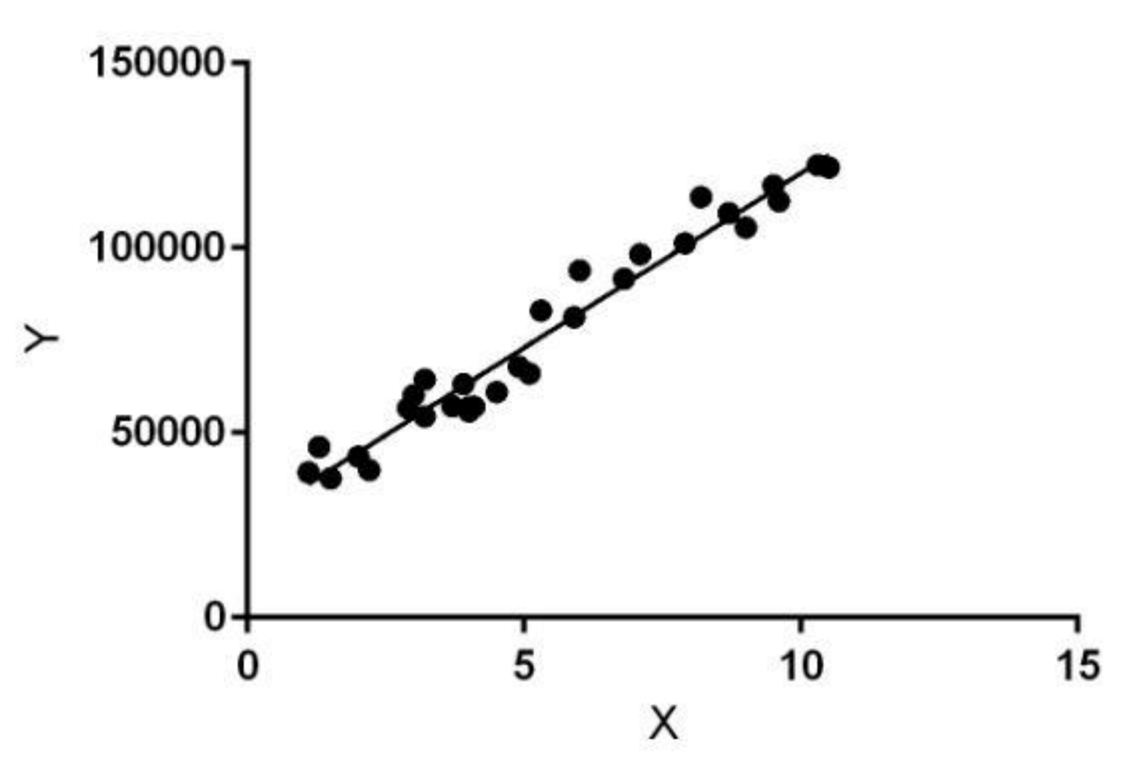
Regression Models

Data
Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

Linear Regression :

Definition :

- Linear regression is used to predict the value of an outcome variable *Y* based on one or more input predictor variables *X*. The aim is to establish a linear relationship (a mathematical formula) between the predictor variable(s) and the response variable, so that, we can use this formula to estimate the value of the response *Y*, when only the predictors (*Xs*) values are known



Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

Linear Regression formula :

Simple Linear Regression

$$y = b_0 + b_1 * x_1$$

Constant

Coefficient

Input feature

Multiple Linear Regression

$$y = b_0 + b_1 * x_1 + b_2 * x_2 + \ldots + b_n * x_n$$

Constant

Coefficient N°1

Coefficient N°n

Input feature N°1

Input feature N°n

How Linear Regression Model Train :

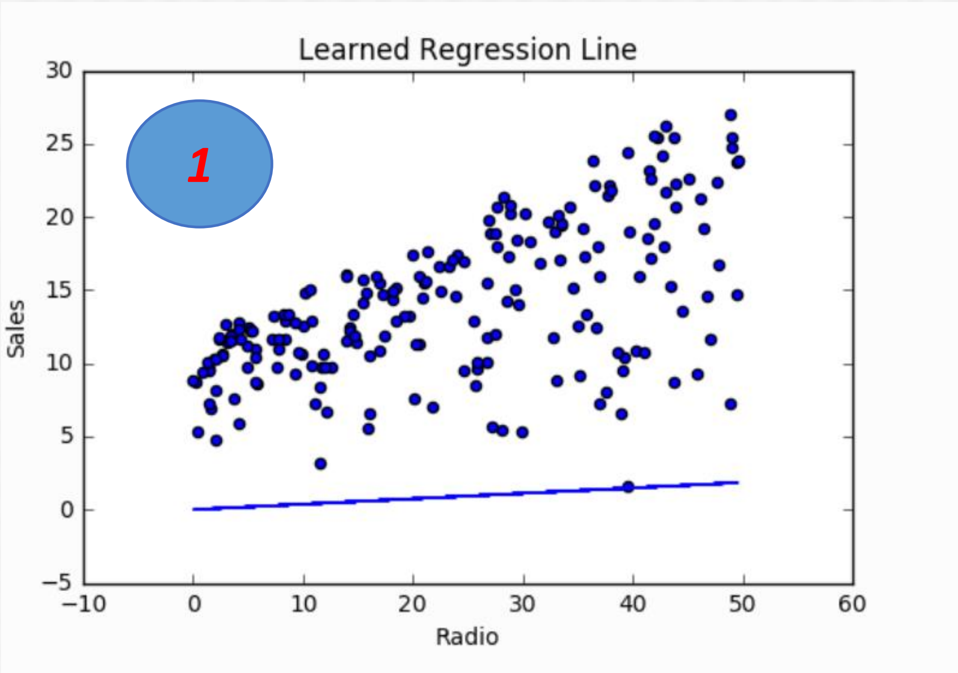
While training the model we are given :

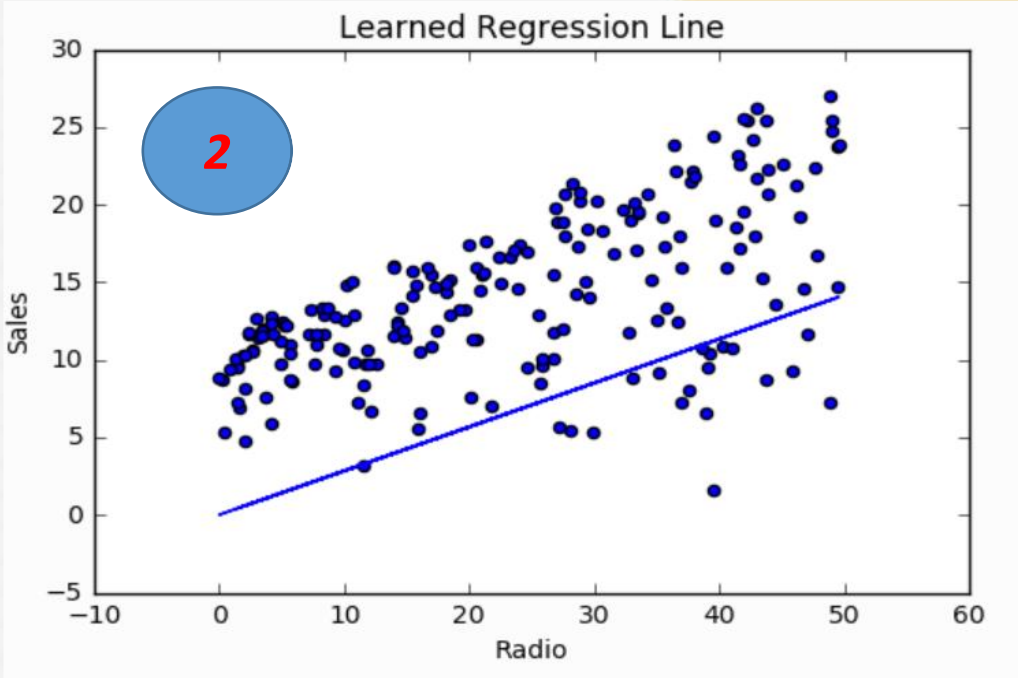
X0,X1,X2,……,Xn: input training features

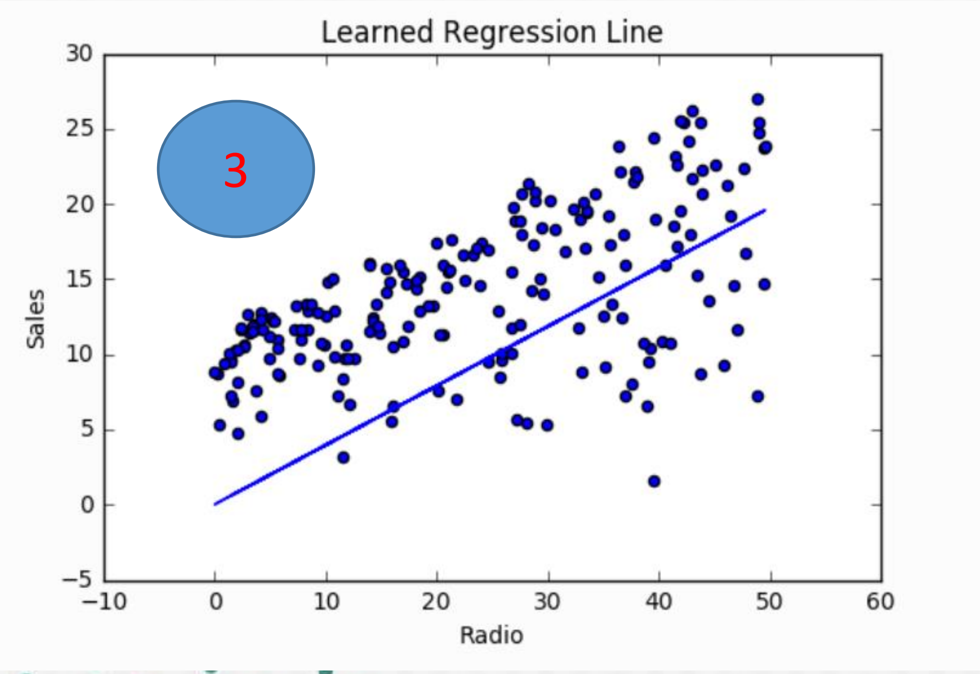
y: labels to data (supervised learning)

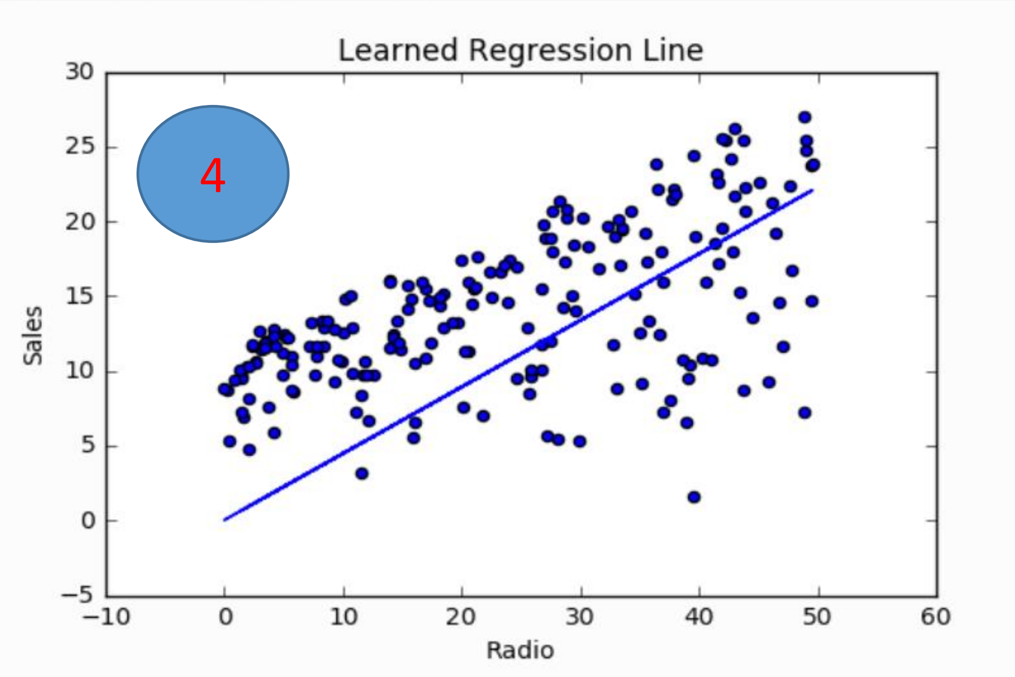
When training the model – it fits the best line to predict the value of y for a given value of x. The model gets the best regression fit line by finding the best b0 , b1 ,b2,……,bn values.

Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB


Learned Regression Line
1
Sales
Radio
30
25
20
15
10
5
0
−5
−10
0
10
20
30
40
50
60


Learned Regression Line
2
Sales
Radio
30
25
20
15
10
5
0
−5
−10
0
10
20
30
40
50
60


Learned Regression Line
3
Sales
Radio
30
25
20
15
10
5
0
−5
−10
0
10
20
30
40
50
60


Learned Regression Line
4
Sales
Radio
30
25
20
15
10
5
0
−5
−10
0
10
20
30
40
50
60

How to update b_0, b_1 ,….., b_n values to get the best fit line ?

Cost function :

By achieving the best-fit regression line, the model aims to predict y value such that the error difference between predicted value and true value is minimum. So, it is very important to update the b_1, b_2 ,……, b_n values, to reach the best value that minimize the error between predicted y value (pred) and true y value (y).

So we had to minimize the Cost function J :

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(\hat{y^{(i)}} - y^{(i)} \right)^2 = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

The Gradient Discent:

Let's suppose we will implement a Simple linear Regression Model :

$$Y = \boldsymbol{\theta}_0 + \boldsymbol{\theta}_1 X$$

Cost Function

$$J(\Theta_0, \Theta_1) = \frac{1}{2m} \sum_{i=1}^{m} [\underbrace{h_\Theta(x_i)}_{\text{Predicted Value}} - \underbrace{y_i}_{\text{True Value}}]^2$$

Gradient Descent

$$\boxed{\Theta_j = \Theta_j - \underbrace{\alpha}_{\text{Learning Rate}} \frac{\partial}{\partial \Theta_j} J(\Theta_0, \Theta_1)}$$

Now,

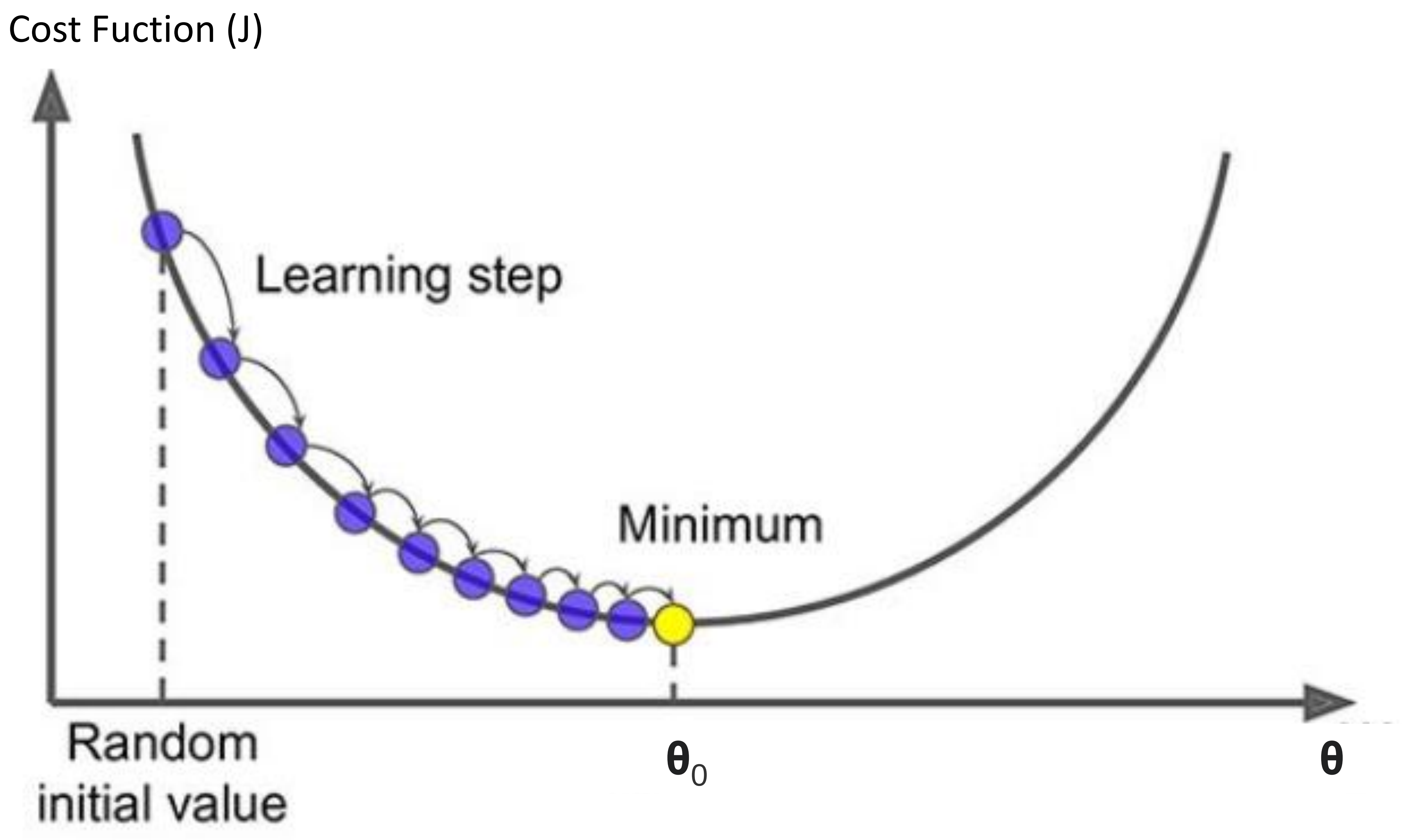
$$\frac{\partial}{\partial \Theta} J_\Theta = \frac{\partial}{\partial \Theta} \frac{1}{2m} \sum_{i=1}^{m} [h_\Theta(x_i) - y]^2$$

$$= \frac{1}{m} \sum_{i=1}^{m} (h_\Theta(x_i) - y) \frac{\partial}{\partial \Theta_j} (\Theta x_i - y)$$

$$= \frac{1}{m} (h_\Theta(x_i) - y) x_i$$

Therefore,

$$\boxed{\Theta_j := \Theta_j - \frac{\alpha}{m} \sum_{i=1}^{m} [(h_\Theta(x_i) - y) x_i]}$$

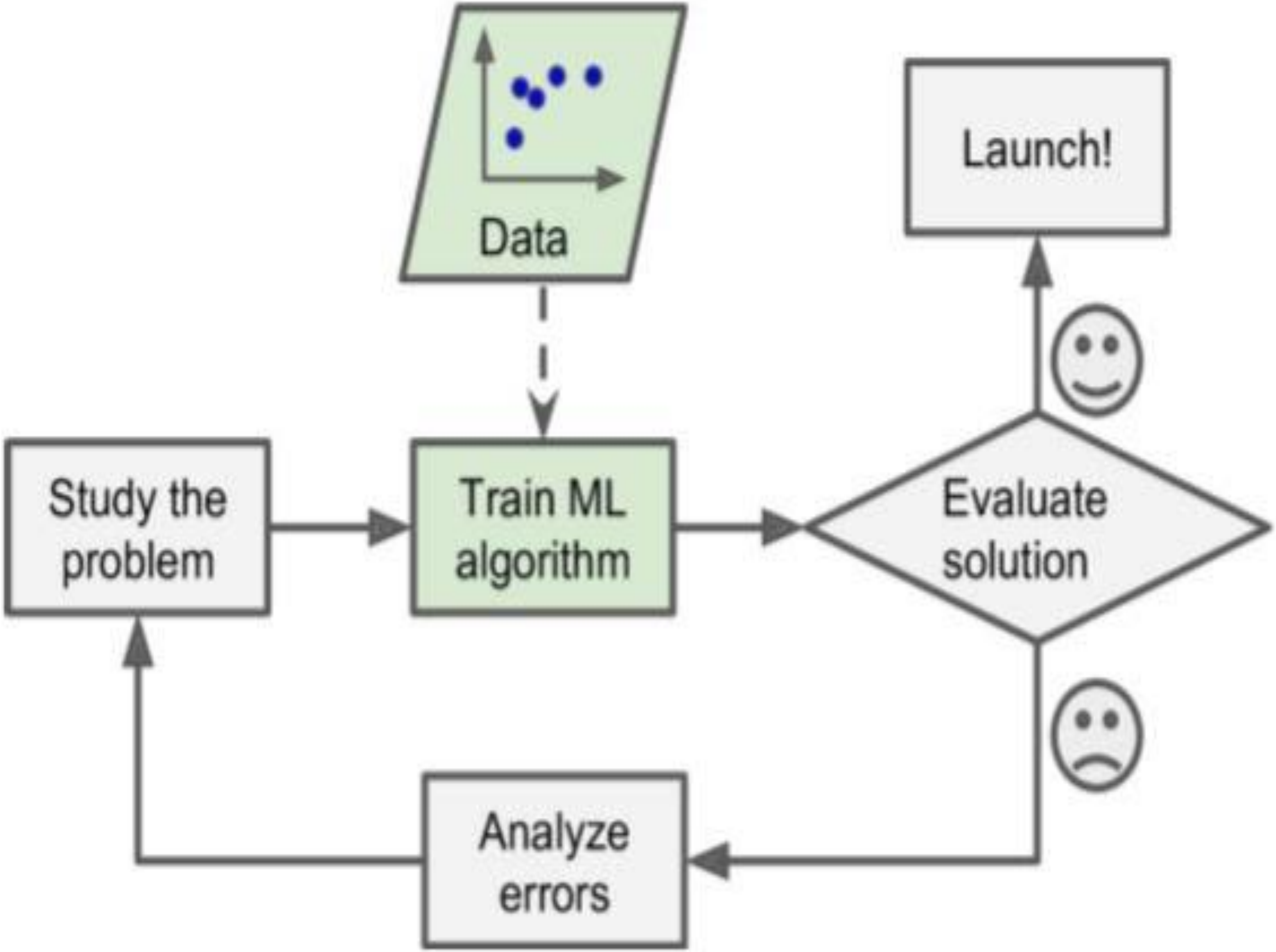

Cost Fuction (J)
Learning step
Minimum
Random
initial value
θ_0
θ

So I will implement all of this from scratch

Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

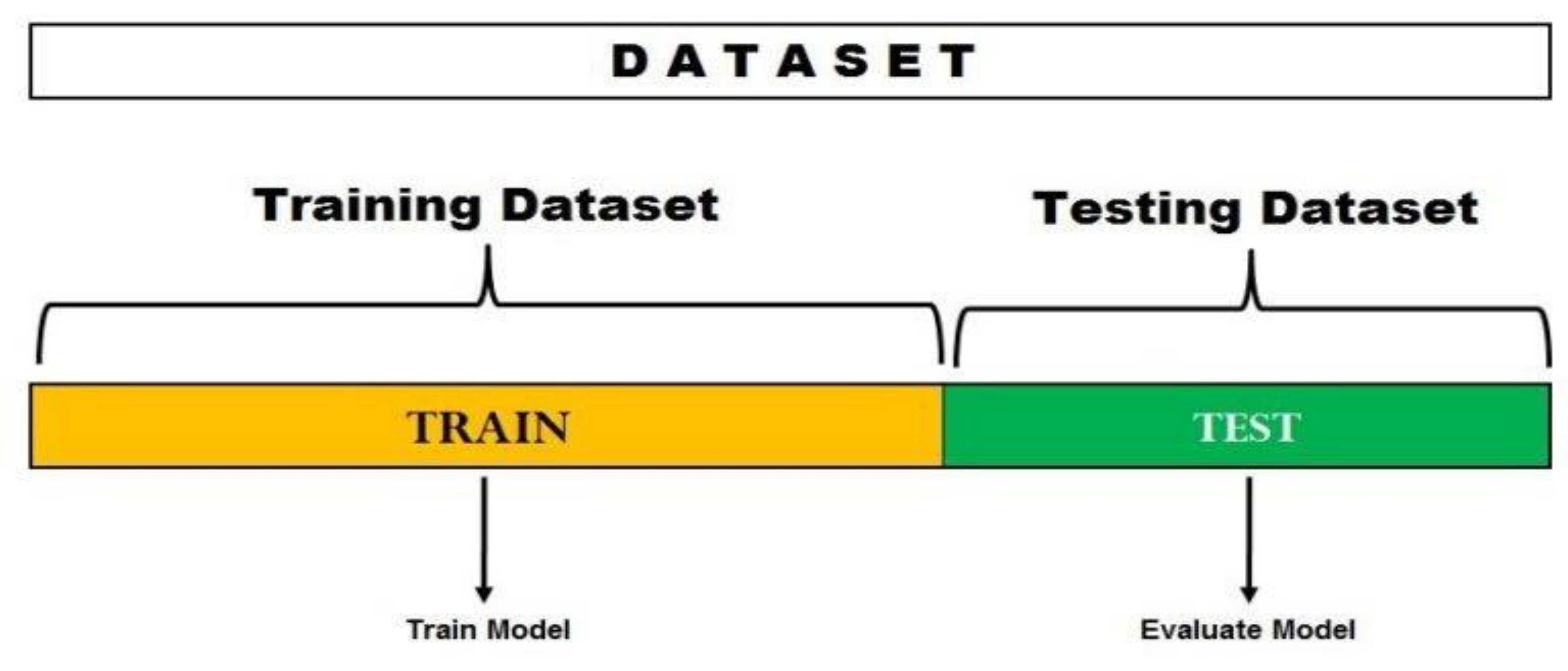
scikit learn

- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license



Data
Launch!
Study the problem
Train ML algorithm
Evaluate solution
Analyze errors

Data Spliting :



Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

Train and test spliting using scikit_learn:

```
Entrée [19]: from sklearn.model_selection import train_test_split
             #split our data to a train and test splits using train_test_split
             X_train,X_test,y_train,y_test = train_test_split(input_data,output,test_size=0.2,random_state=42)
```

The input data for the training data

The input data for the test data

The output data for the test data

the size of test data

Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

Let’s train our first model :

Entrée [78]:
```
from sklearn.linear_model import LinearRegression
#first:intantiate our model
model = LinearRegression()
#second: Train our model
model.fit(X_train,y_train)
```

Out[78]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

Let’s get our first prediction :

Entrée [80]:
```
#make a prediction on the test data
prediction=model.predict(X_test)
#show our first prediction
print("this is my  first prediction : \n",prediction )
```

```
this is my  first prediction :
 [ 99811.49231373 373499.28094536 211518.82835492 ... 493515.62255379
 341942.42964631 237793.92743721]
```

Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

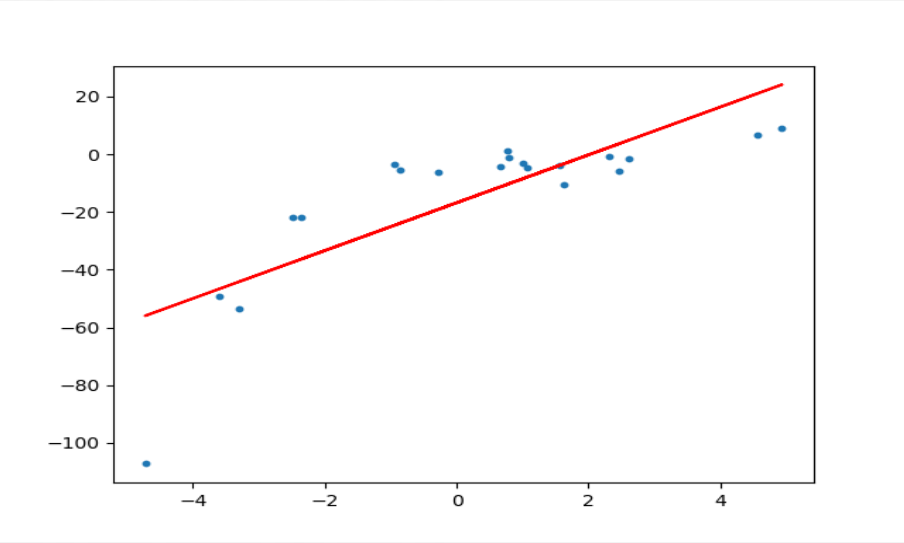
And then Let's evaluate the model :

```
Entrée [81]: #evaluate our model using our metrics
             #MAE
             MAE = mean_absolute_error(y_test,prediction)
             #MSE
             MSE = mean_squared_error(y_test,prediction)
             #R_squared
             r2 = r2_score(y_test,prediction)

             print("MAE :{:.2f}".format(MAE))
             print("MSE :{:.2f}".format(MSE))
             print("R2_score :{:.2f}".format(r2))

             MAE :125433.55
             MSE :39970613249.75
             R2_score :0.69
```

The polynomial Rgression :

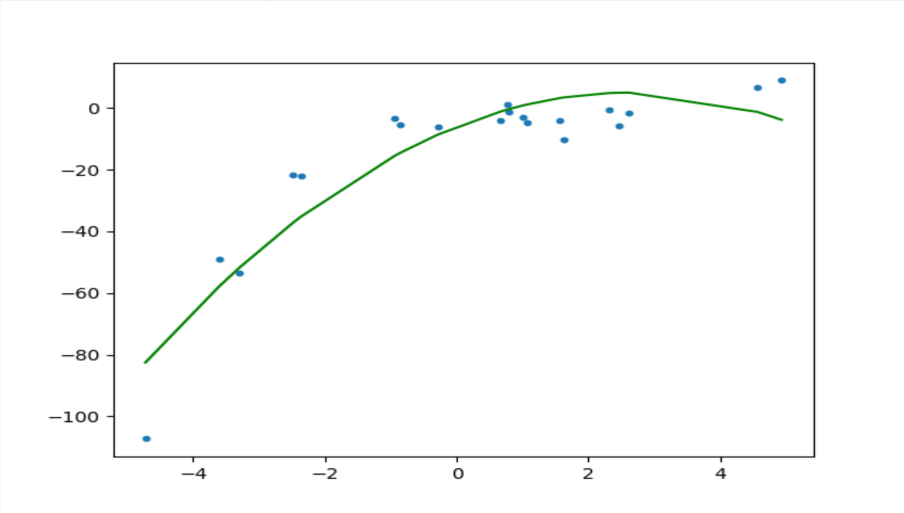
To generate a higher order equation we can add powers of the original features as new features. The linear model,

$$Y=\mathbf{b}_0 + \mathbf{b}_1X$$



can be transformed to:

$$Y=\mathbf{b}_0 + \mathbf{b}_1X + \mathbf{b}_2X^2$$



. This will be considered to be linear model as the coefficients/weights associated with the features are still linear,X^2 is only a feature.

. To convert the original features into thier higher order terms we will use The PolynomialFeatures provided by scikit-learn. Next we will train the model using Linear Regression

```
Entrée [88]: from sklearn.preprocessing import PolynomialFeatures
```

```
Entrée [177]: #Instantiate our PolynomialFeatures Generator
              polynomial_features= PolynomialFeatures(degree=3)
              #Transform our input Data
              x_poly = polynomial_features.fit_transform(X_train)
```

```
Entrée [170]: model.fit(x_poly,y_train)

  Out[170]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

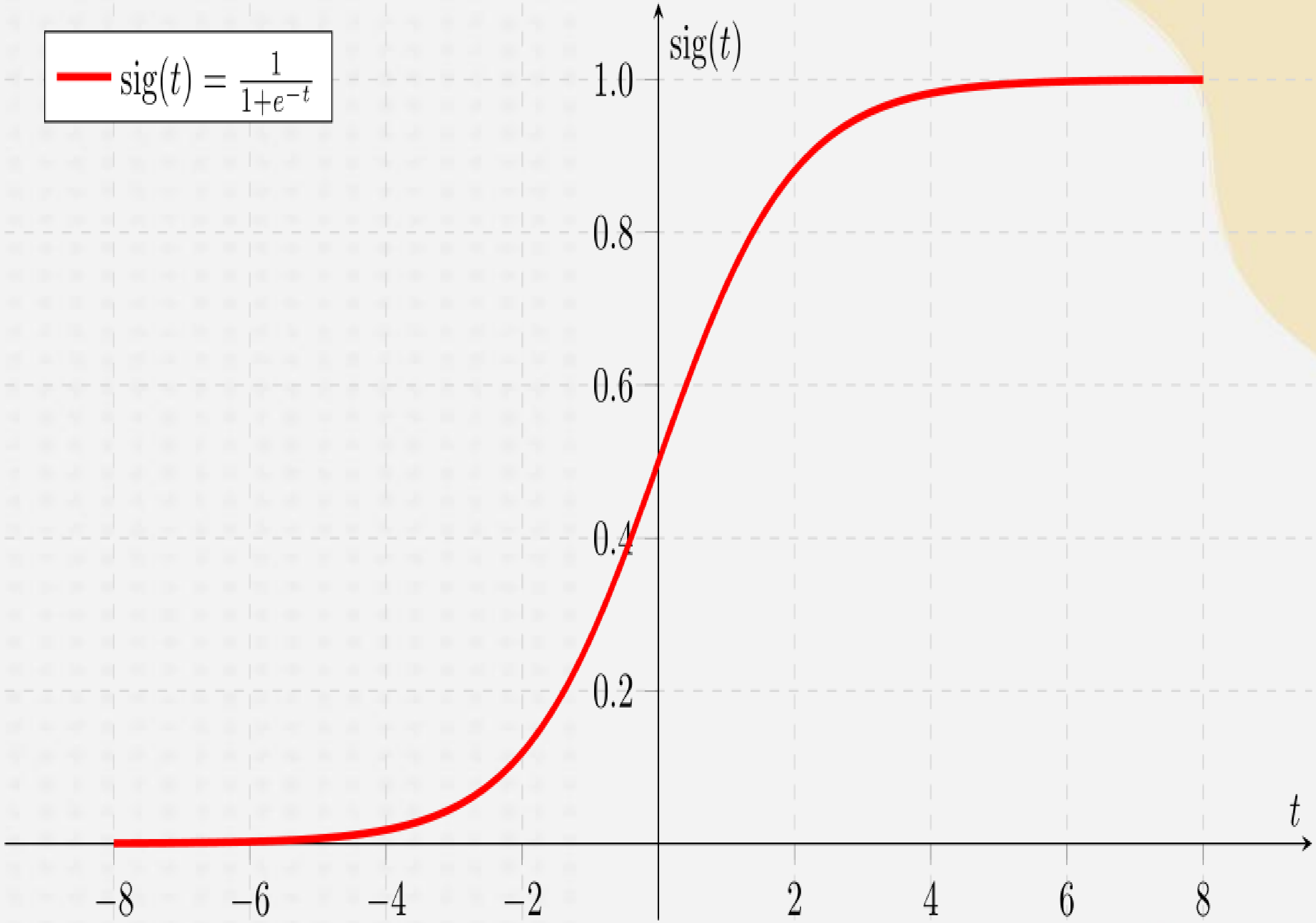

Let’s code

Classification Models

Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

The logistic Regression :

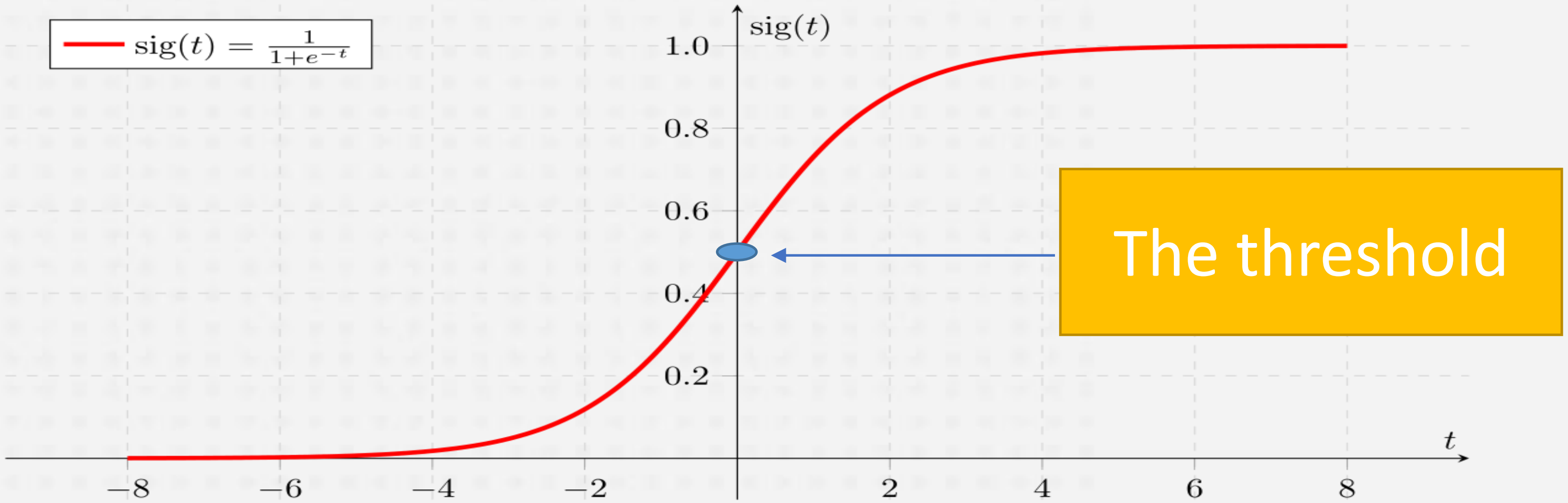
$$f(x) = \frac{1}{1 + e^{-(x)}}$$



Note:

- Contrary to popular belief, logistic regression IS a regression model. The model builds a regression model to predict the probability that a given data entry belongs to the category numbered as “1”. Just like Linear regression assumes that the data follows a linear function, Logistic regression models the data using the sigmoid function.

- Logistic regression becomes a classification technique only when a decision threshold is brought into the picture. The setting of the threshold value is a very important aspect of Logistic regression and is dependent on the classification problem itself.



Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

How it works :

$$f(x) = \frac{1}{1+e^{-h(x)}}$$

$$h(x) = \theta_0 + \theta_1 X_1 + \theta_2 X_2 + \theta_3 X_3 + \theta_4 X_4$$

Feature N°1

And just like the linear Regression when we train our model we will try to optimize our weights using the gradient discent to get the best possible performnace.

Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

Logistic Regression with sciket-learn:

```
Entrée [19]: from sklearn.model_selection import train_test_split
             train_features,test_features,train_labels,test_labels = train_test_split(features,labels,test_size = 0.2,stratify=labels)
```

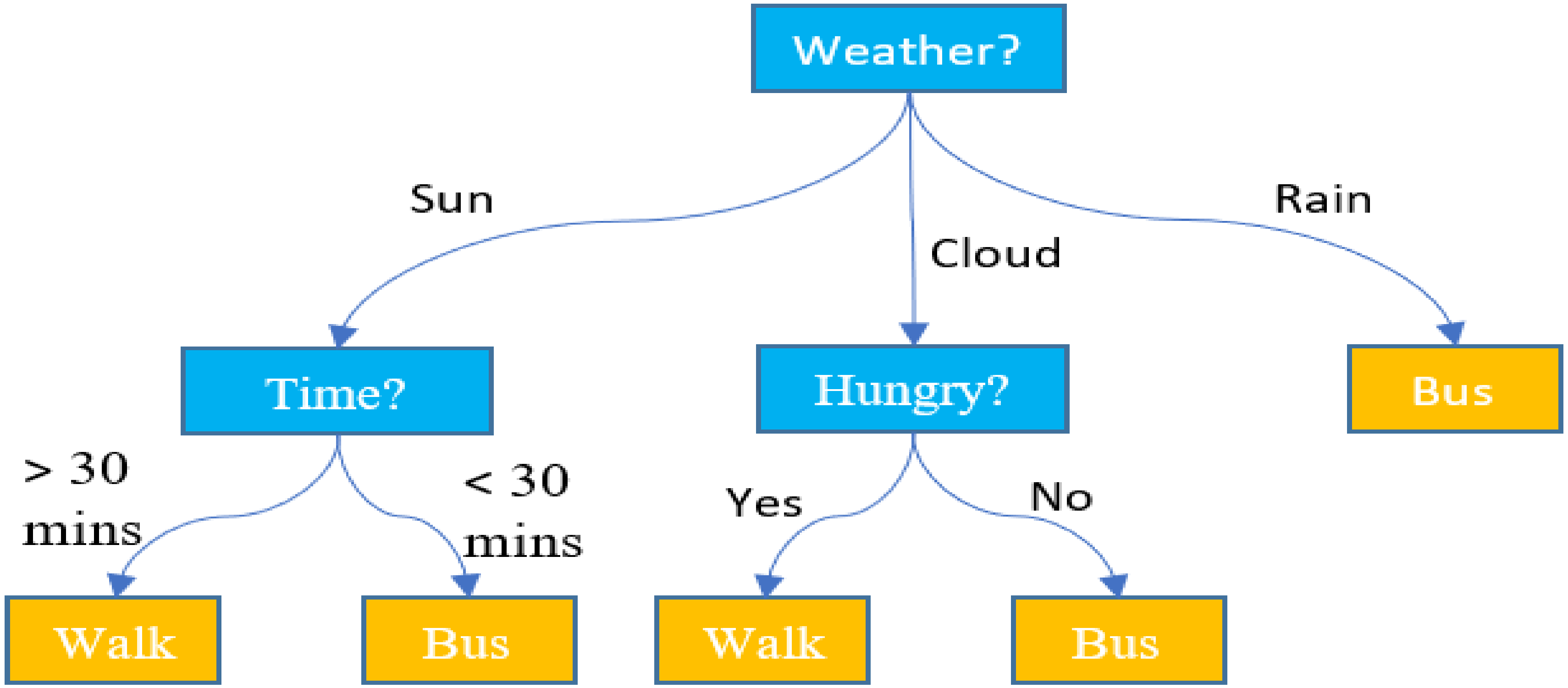
```
Entrée [20]: #Import our model from sklearn
             from sklearn.linear_model import LogisticRegression
```

```
Entrée [28]: #Instantiate our model
             clf = LogisticRegression()
             #final let's train our first model
             clf = clf.fit(train_features, train_labels)
```

Let’s evaluate our model :

```
Entrée [23]: from sklearn.metrics import accuracy_score
             accuracy = accuracy_score(test_labels, predicted_labels)
             accuracy

   Out[23]: 0.7006519377037306
```

Decision tree Model:



How to Build decision trees :

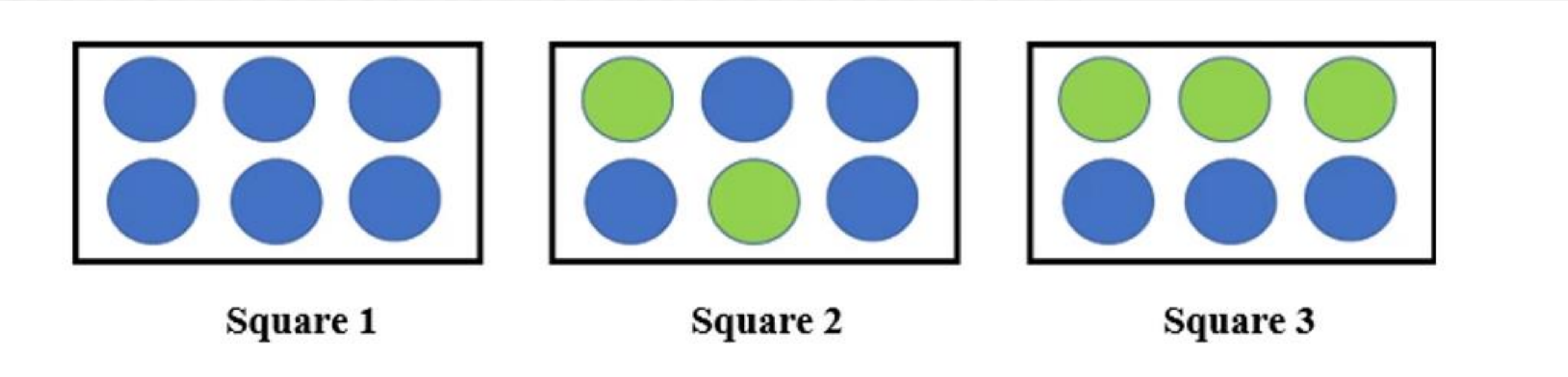
Geni impurity:

$$I_G(n) = 1 - \sum_{i=1}^{J} (p_i)^2$$

$$= \quad 1 - prob(\text{'class 1'})^2 - prob(\text{'class 0'})^2 - \ldots\ldots - prob(\text{'class n'})^2$$

?

Our goal is to get the most pure feature and the most *homogeneous* data



square 1 is a pure node

square 2 is more impure

square 3 is most impure

Square 1 in the best to be our root node in our tree model

Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

Let’s take an exemple :

Days	Meal Type	Spicy	Cuisine	Packed	Price	Liked/Dislike
1	Breakfast	Low	Gujarati	Hot	25	No
2	Breakfast	Low	Gujarati	cold	30	No
3	Lunch	Low	Gujarati	Hot	46	Yes
4	Dinner	normal	Gujarati	Hot	45	Yes
5	Dinner	High	South Indian	Hot	52	Yes
6	Dinner	High	South Indian	cold	23	No
7	Lunch	High	South Indian	cold	43	Yes
8	Breakfast	normal	Gujarati	Hot	35	No
9	Breakfast	High	South Indian	Hot	38	Yes
10	Dinner	normal	South Indian	Hot	46	Yes
11	Breakfast	normal	South Indian	cold	48	Yes
12	Lunch	normal	Gujarati	cold	52	Yes
13	Lunch	Low	South Indian	Hot	44	Yes
14	Dinner	normal	Gujarati	cold	30	No

Meal Type

Meal Type is a nominal data that has 3 values Breakfast, Lunch and Dinner.
Let's classify the instances on basis of liked/dislike.

Meal Type	# Yes	# No	# Total
Breakfast	2	3	5
Lunch	4	0	4
Dinner	3	2	5

Gini index (Meal Type = Breakfast) = $1-(2/5)^2-(3/5)^2 = 1- 0.16 - 0.36 = 0.48$
Gini index (Meal Type = Lunch) = $1- (4/4)^2-(0/4)^2 = 1- 1- 0 = 0$
Gini index (Meal Type = Dinner) = $1- (3/5)^2 -(2/5)^2 = 1- 0.36- 0.16 = 0.48$

Now, we will calculate the weighted sum of Gini index for Meal Type features,

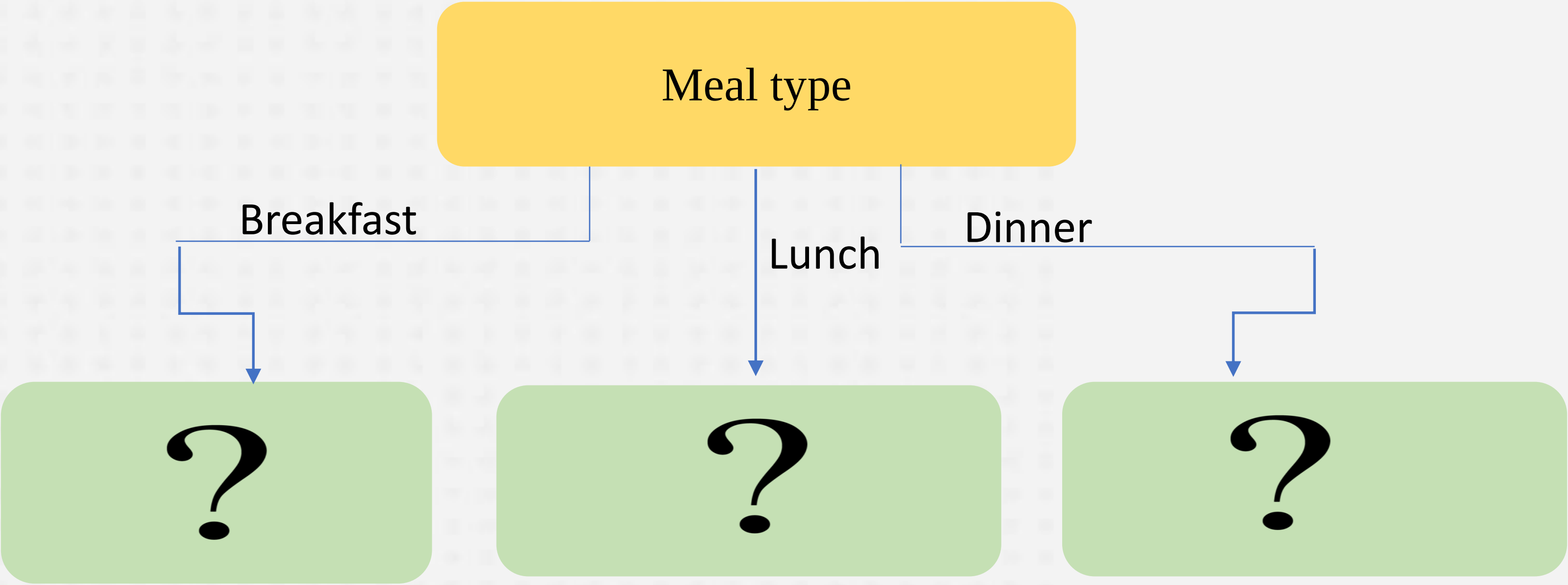
Gini (Meal Type) = $(5/14) * 0.48 + (4/14) * 0 + (5/14) * 0.48 = 0.342$

Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB

I and we continue doing the same process :

Features	Gini Index
Meal type	0.342
Spicy	0.439
Cuisine	0.367
Packed	0.428

So, we can conclude that the lowest Gini index is of “Meal Type” and a lower Gini index means the highest purity and more homogeneity. So, our root node is “Meal type”. So, our tree looks like



DecisionTreeClassifier with scikit-learn :

Entrée [59]:
```
#Instiate our DecisionTreeClassifer
tree_clf = DecisionTreeClassifier()
#Let's train our model
tree_clf.fit(train_features,train_labels)
#let's get our first prediction
pred=tree_clf.predict(test_features)
#let's evaluate our model
accuracy_score(pred, test_labels)
```

Out[59]: 0.9918507787033684

Data Overflow
ACM INSAT Student Chapter
IEEE Computer Society INSAT SB