Lecture 6: *Regression*

AI/ML Foundation Course
with Python

Sample

A part of the whole population of research interest

Dependent Variable

The factor being influenced by others

Independent Variables

The factor that influences other variables.

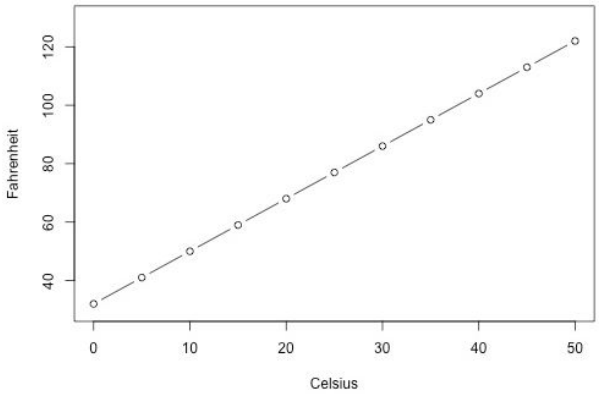
Simple Linear Regression

x → ***Predictor***/Independent/ Explanatory Variable

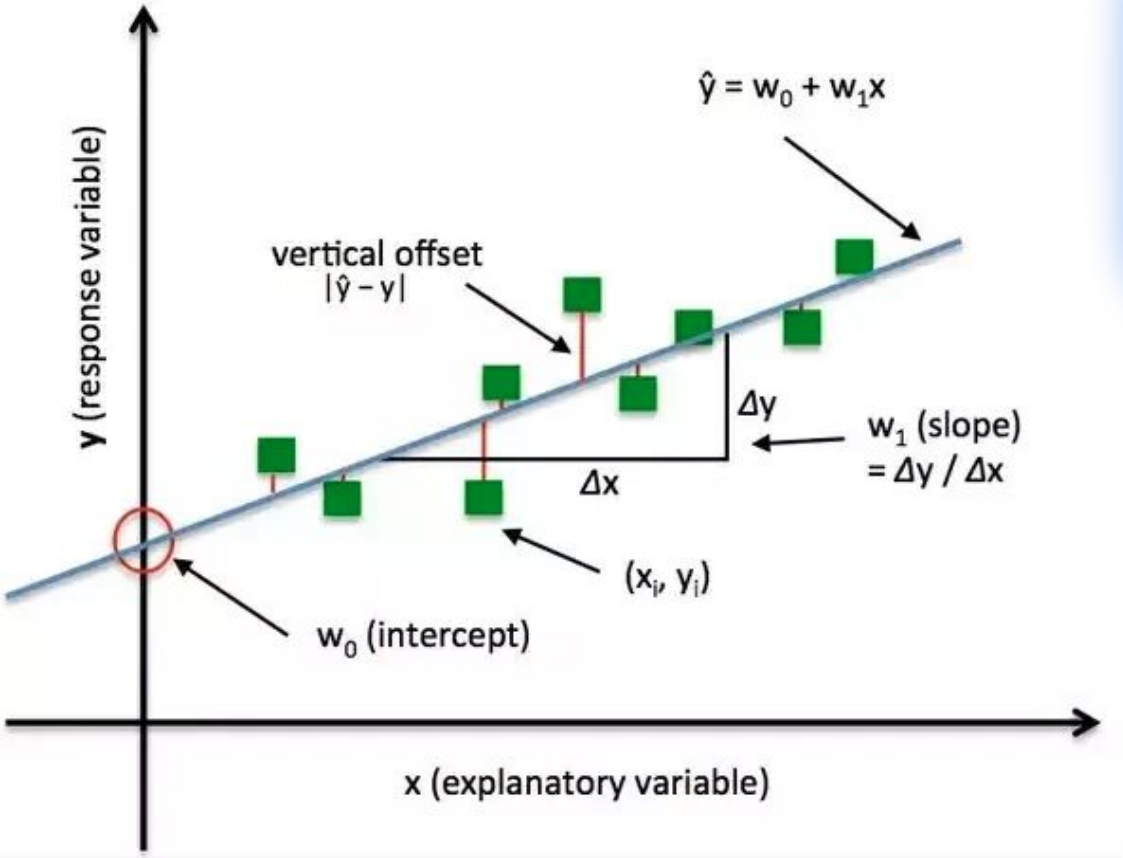
y → ***Response***/Dependent/Outcome Variable

Linear Regression → Relationship Between Two Variables.

Simple Linear Regression → Concerns the study of only one predictor variable.



Simple Linear Regression



Find the best fitting line that minimizes the squared error

1. Deterministic

The observed data points (x,y) fall into a *straight line*

- **Circumference =** $\pi \times \text{diameter}$
  Circumference of Circle
- **Fahrenheit =** $(9/5) * C + 32$
  Temperature Conversion
- **I =** V/R
  Ohm's Law

2. Statistical

The observed data points (x,y) do not fall into a straight line. Therefore, Relationship between x, y is statistical in nature.

- **Height and Weight**
  As height increases, weight tends to increase but not deterministically.
- **Lung function and Years of smoking**
  Lung function tends to decrease with years of smoking but not exactly.
- **More Examples?**
  x and y follow some sort of linear function

Height (x) and Corresponding Weight (y)

ht	wt
63	127
64	121
66	142
69	157
69	162
71	156
71	169
72	165
73	181
75	208

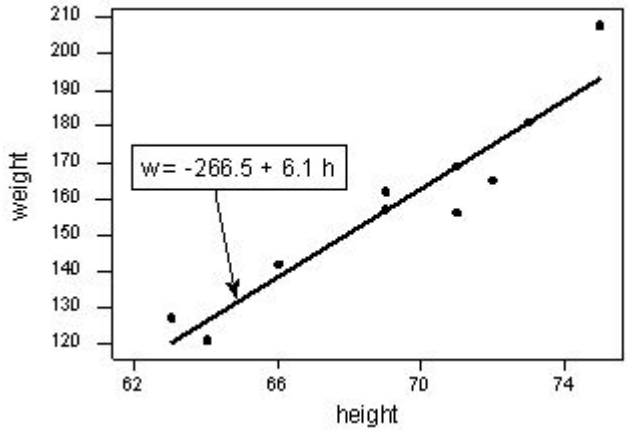
Linear relation between x and y:

$$\hat{y}_i = b_0 + b_1 x_i$$

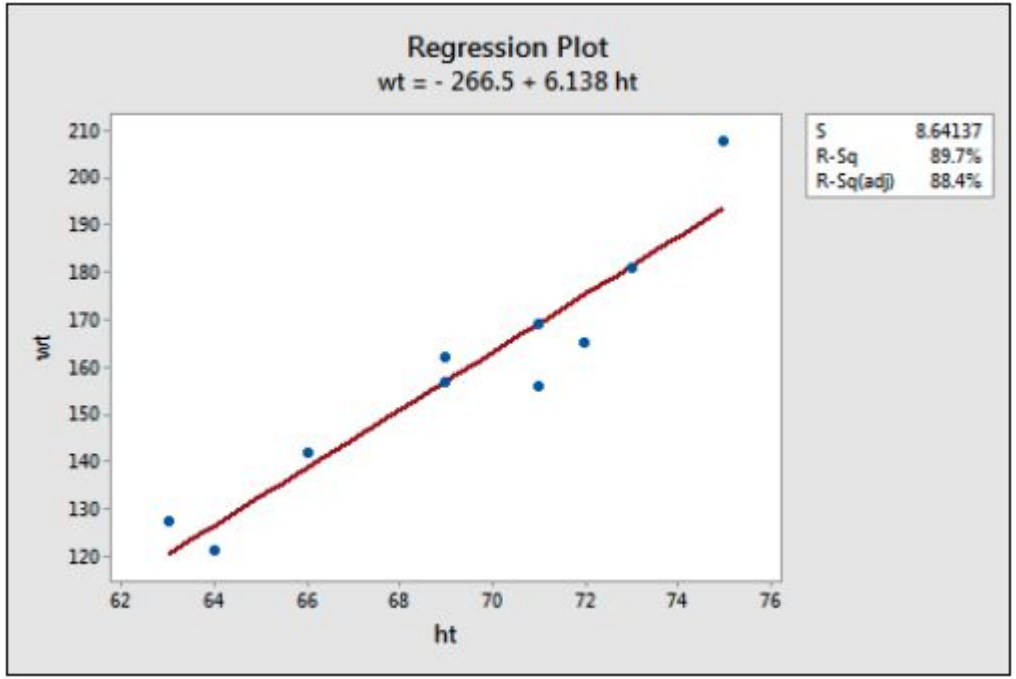
yi → denotes the observed response for experimental unit *i*

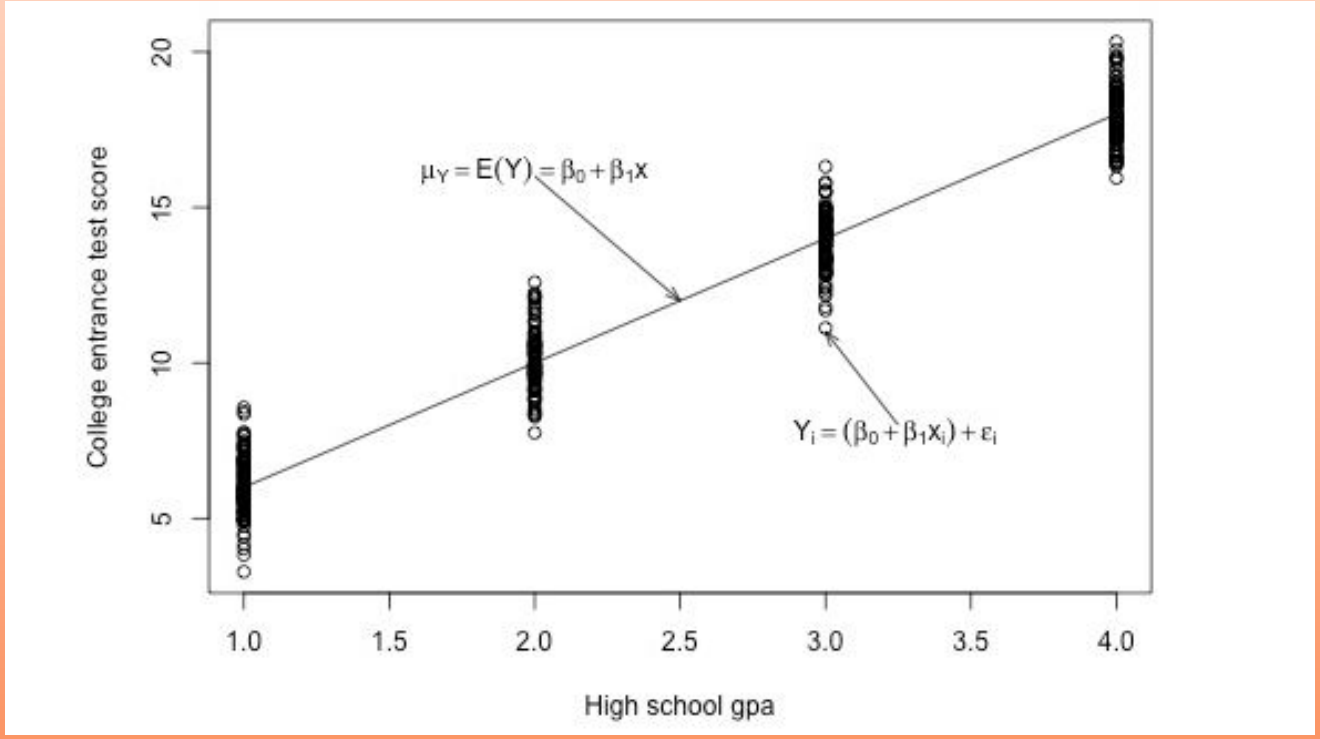
xi→ denotes the predictor value for experimental unit *i*

y^i → is the predicted response (or fitted value) for experimental unit *i*



$w = -266.53 + 6.1376\,h$ (the solid line)					
i	x_i	y_i	$\hat{y}_i$	$(y_i - \hat{y}_i)$	$(y_i - \hat{y}_i)^2$
1	63	127	120.139	6.8612	47.076
2	64	121	126.276	-5.2764	27.840
3	66	142	138.552	3.4484	11.891
4	69	157	156.964	0.0356	0.001
5	69	162	156.964	5.0356	25.357
6	71	156	169.240	-13.2396	175.287
7	71	169	169.240	-0.2396	0.057
8	72	165	175.377	-10.3772	107.686
9	73	181	181.515	-0.5148	0.265
10	75	208	193.790	14.2100	201.924
					597.4



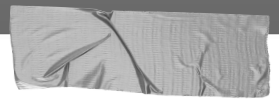


Note

Simple Linear Regression Model

Potential Relationship between the Predictor "High School GPA" & the Response "College Entrance Test Score".

1) The mean of the response $E(Y_i)$ at each value of the predictor x_i is a Linear function of the x_i
2) The errors ε_i are Independent.
3) The errors ε_i at each value of the predictor x_i are Normally distributed.
4) The errors ε_i at each value of the predictor x_i have Equal variances (σ^2).



Note

Four Conditions that comprise a Simple Linear Regression Model

$$\mu_Y = E(Y) = \beta_0 + \beta_1 x.$$

Multiple Linear Regression

- A regression model with two or more predictors (independent variables)
- Multiple Regression models are defined using matrices due to large number of predictor variables.
- Think of distribution of *error(s)* at a fixed value *for all the predictors*.

PIQ	Brain	Height	Weight
124	81.69	64.5	118
150	103.84	73.3	143
128	96.54	68.8	172
134	95.15	65.0	147
110	92.88	69.0	146
131	99.13	64.5	138
98	85.43	66.0	175
84	90.49	66.3	134
147	95.55	68.8	172
124	83.39	64.5	118
128	107.95	70.0	151
124	92.41	69.0	155
147	85.65	70.5	155
90	87.89	66.0	146
96	86.54	68.0	135
120	85.22	68.5	127
102	94.51	73.5	178
84	80.80	66.3	136
86	88.91	70.0	180
84	90.59	76.5	186
134	79.06	62.0	122
128	95.50	68.0	132
102	83.18	63.0	114
131	93.55	72.0	171
84	79.86	68.0	140
110	106.25	77.0	187
72	79.35	63.0	106

Is a person's brain size predictive of his/her intelligence?

- Predictor (x_1) : Brain Counts based on MRI Scans
- Predictor (x_2) : Height (in inches)
- Predictor (x_3) : Weight (in pounds)
- **Response (y)** : Performance IQ (PIQ) Scores as a measure of individual's intelligence.

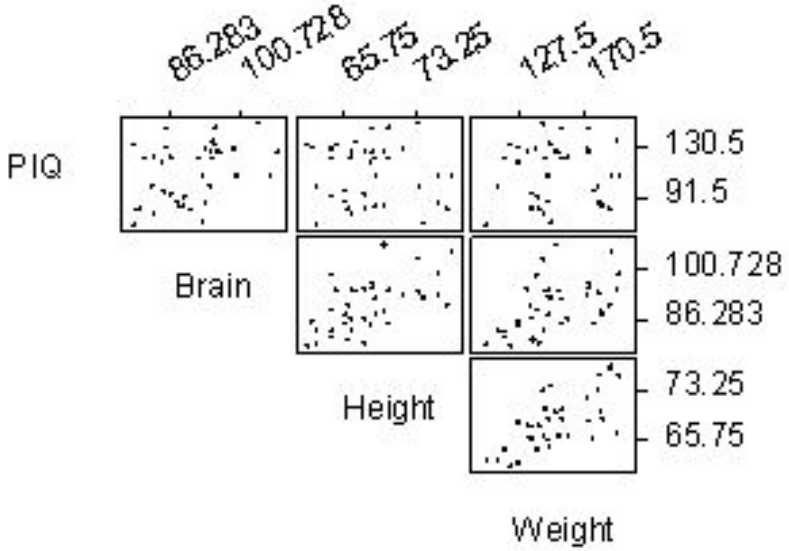
PIQ	Brain	Height	Weight
124	81.69	64.5	118
150	103.84	73.3	143
128	96.54	68.8	172
134	95.15	65.0	147
110	92.88	69.0	146
131	99.13	64.5	138
98	85.43	66.0	175
84	90.49	66.3	134
147	95.55	68.8	172
124	83.39	64.5	118
128	107.95	70.0	151
124	92.41	69.0	155
147	85.65	70.5	155
90	87.89	66.0	146
96	86.54	68.0	135
120	85.22	68.5	127
102	94.51	73.5	178
84	80.80	66.3	136
86	88.91	70.0	180
84	90.59	76.5	186
134	79.06	62.0	122
128	95.50	68.0	132
102	83.18	63.0	114
131	93.55	72.0	171
84	79.86	68.0	140
110	106.25	77.0	187
72	79.35	63.0	106

How?

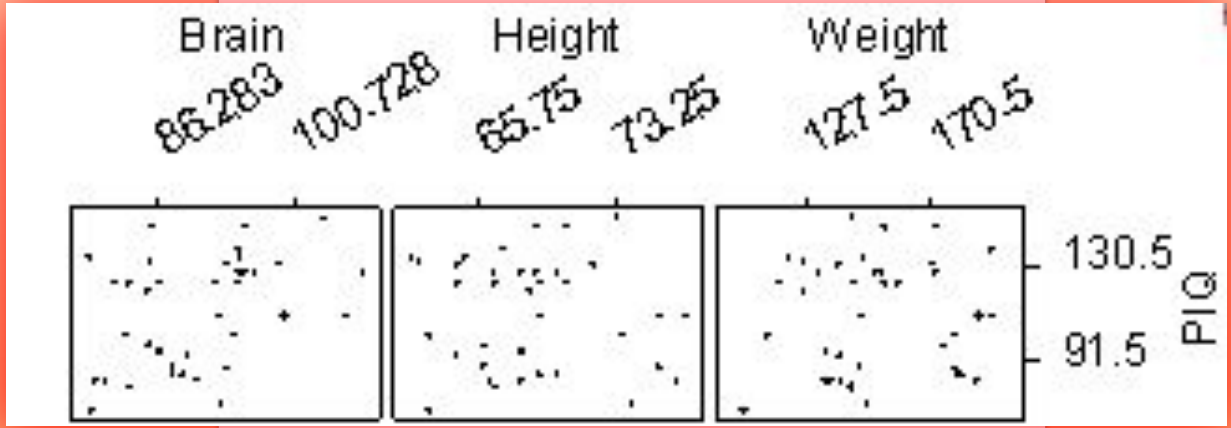
Scatter Plot Matrix

- ➔ One Scatter Plot for Each Pair of Variables
- ➔ Consider the relationship between the response and each of the predictor variable.
- ➔ We also consider how the predictors are related among each other.

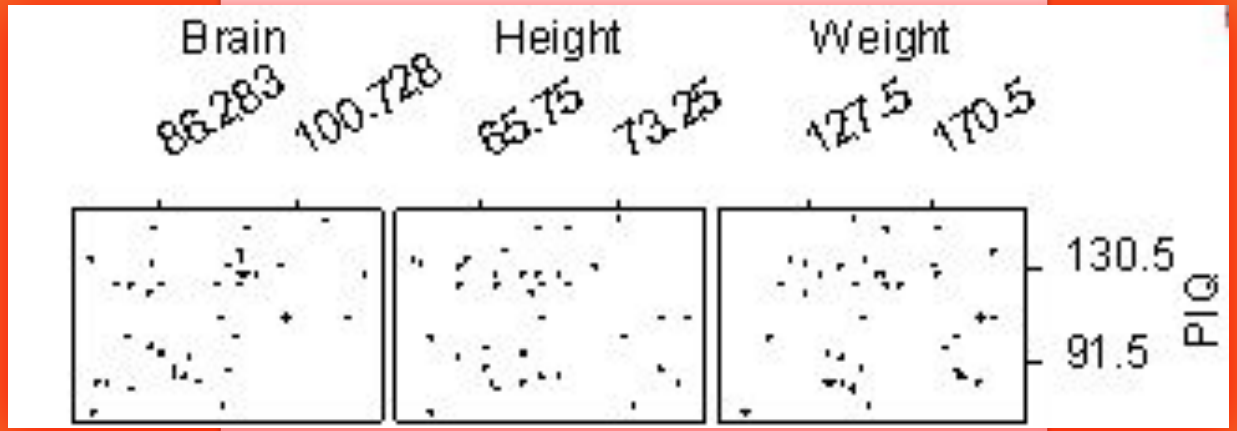
Scatter Plots



Brain Count Vs PIQ



PIQ = 111.4 + 2.060 Brain -
2 .73 Height +
0.001 Weight



Multiple Regression Model with three quantitative predictors

$$y_i = (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3}) + \epsilon_i$$

- y_i is the intelligence (PIQ) of student i
- x_{i1} is the brain size (MRI) of student i
- x_{i2} is the height (Height) of student i
- x_{i3} is the weight (Weight) of student i
- **independent** error terms $\boldsymbol{\varepsilon_i}$ must follow a **normal** distribution with mean 0 and **equal variance** σ^2.

Notation and Assumptions for Multiple Linear Regression Model

- ➔ The estimates of the β coefficients are the values that minimize the sum of squared errors for the sample.
- ➔ b0 is the sample estimate of β0, b1 is the sample estimate of β1 and so on.
- ➔ β1 coefficient represents the change in the mean response, E(*y*), per unit increase in x1 when x2, x3, ..., xp−1 are held constant.
- ➔ Residual (error) term is calculated as e=observed(y)−predicted(y), i.e, the difference between an actual and a predicted value of *y*.

$$y_i = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \ldots + \beta_{p-1} x_{i,p-1} + \epsilon_i.$$

What is a Matrix

An ***r* × *c* matrix** is a rectangular array of symbols or numbers arranged in *r* rows and *c* columns.

$$A = \begin{bmatrix} 1 & 2 \\ 6 & 3 \end{bmatrix}$$

2x2 Square matrix

$$B = \begin{bmatrix} 1 & 80 & 3.4 \\ 1 & 92 & 3.1 \\ 1 & 65 & 2.5 \\ 1 & 71 & 2.8 \\ 1 & 40 & 1.9 \end{bmatrix}$$

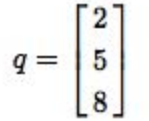
3x5 Square matrix

$$X = \begin{bmatrix} 1 & x_{11} & x_{12} \\ 1 & x_{21} & x_{22} \\ 1 & x_{31} & x_{32} \\ 1 & x_{41} & x_{42} \\ 1 & x_{51} & x_{52} \\ 1 & x_{61} & x_{62} \end{bmatrix}$$

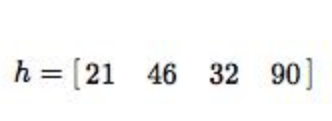
3x6 Square matrix

What is a Matrix

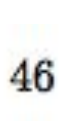
An ***r* × *c* matrix** is a rectangular array of symbols or numbers arranged in *r* rows and *c* columns.



3x1 Column Vector



1x4 Row Vector



1x1 Matrix : Scalar

Matrix Multiplication

- The number of columns of the first matrix must equal the number of rows of the second matrix.
- The number of rows of the resulting matrix equals the number of rows of the first matrix
- The number of columns of the resulting matrix equals the number of columns of the second matrix.

$$C = AB = \begin{bmatrix} 1 & 9 & 7 \\ 8 & 1 & 2 \end{bmatrix} \begin{bmatrix} 3 & 2 & 1 & 5 \\ 5 & 4 & 7 & 3 \\ 6 & 9 & 6 & 8 \end{bmatrix} = \begin{bmatrix} 90 & 101 & 106 & 88 \\ 41 & 38 & 27 & 59 \end{bmatrix}$$

Matrix Addition

- **The number of rows and columns of the first matrix must equal the number of rows and columns of the second matrix.**

$$C = A + B = \begin{bmatrix} 2 & 4 & -1 \\ 1 & 8 & 7 \\ 3 & 5 & 6 \end{bmatrix} + \begin{bmatrix} 7 & 5 & 2 \\ 9 & -3 & 1 \\ 2 & 1 & 8 \end{bmatrix} = \begin{bmatrix} 9 & 9 & 1 \\ 10 & 5 & 8 \\ 5 & 6 & 14 \end{bmatrix}$$

Matrix Transpose

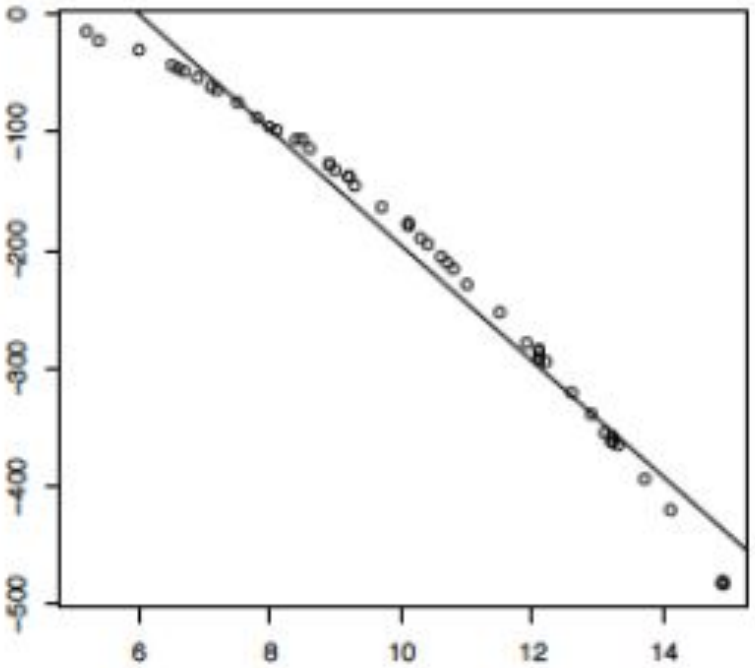
- **The transpose of a 2x3 matrix is a 3x2 matrix with same values.**

$$A = \begin{bmatrix} 1 & 5 \\ 4 & 8 \\ 7 & 9 \end{bmatrix} \qquad A' = A^T = \begin{bmatrix} 1 & 4 & 7 \\ 5 & 8 & 9 \end{bmatrix}$$

Polynomial Regression

Polynomial Regression allows for a nonlinear relationship between y and x.

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \ldots + \beta_h X^h + \epsilon,$$



$$y_i = 5 + 12x_i - 3x_i^2 + \epsilon_i,$$

i	Temperature	Yield
1	50	3.3
2	50	2.8
3	50	2.9
4	70	2.3
5	70	2.6
6	70	2.1
7	80	2.5
8	80	2.9
9	80	2.4
10	90	3.0
11	90	3.1
12	90	2.8
13	100	3.3
14	100	3.5
15	100	3.0

Example : Yield Dataset

Yield Data set contains measurements of yield from an experiment done at five different temperature levels -

50 70 80 90 100

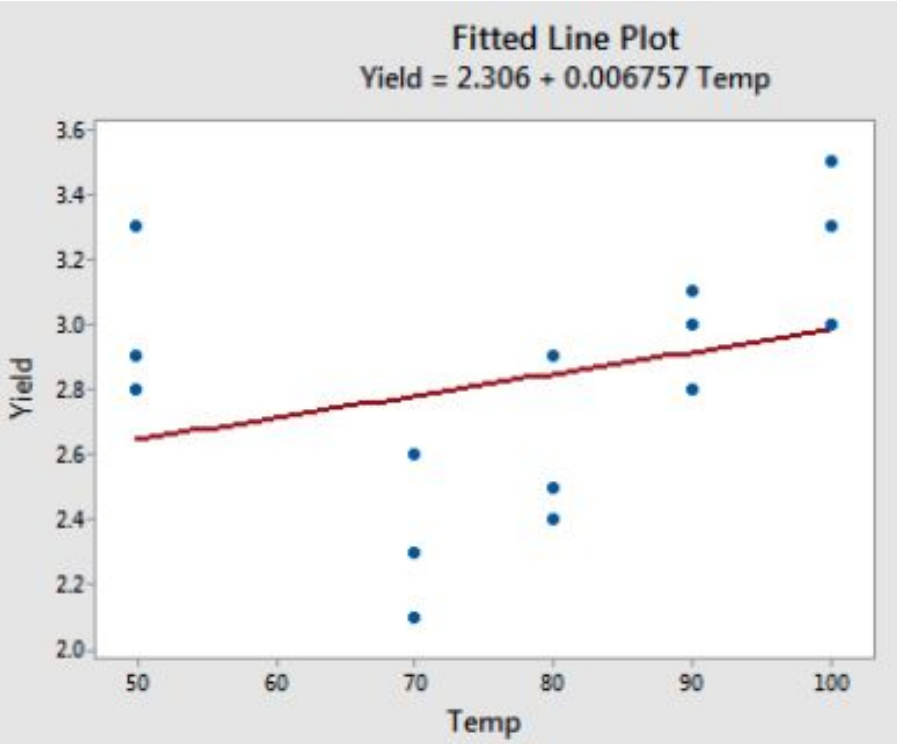
The variables are --

***y* = yield**

And,

***x* = temperature in degree Fahrenheit.**

Example : Yield Dataset : Linear Fit



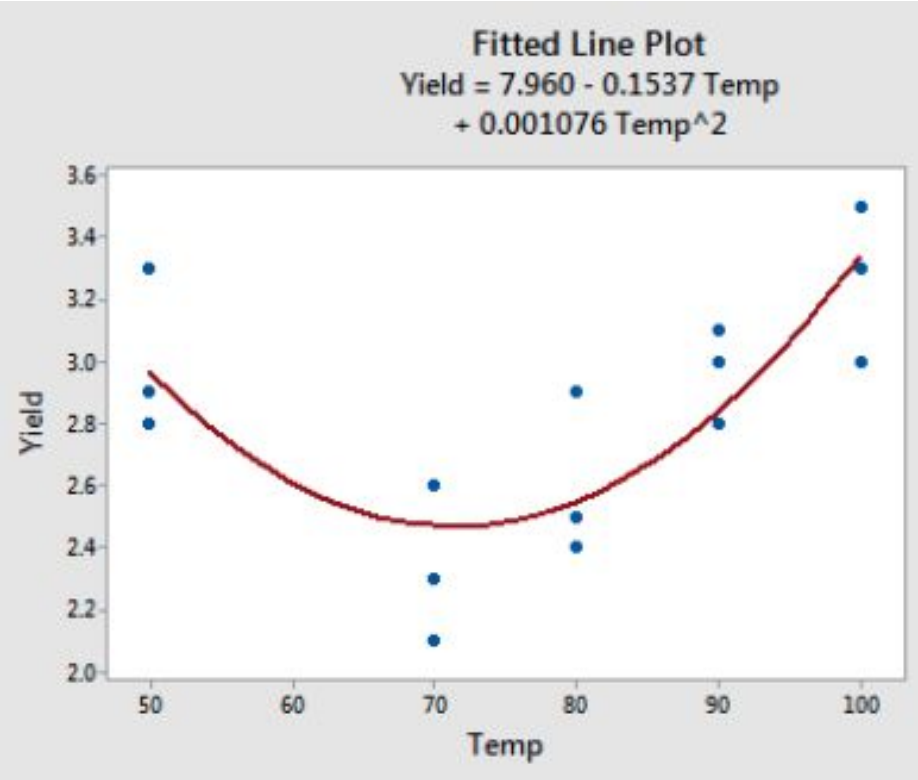
Yield Data set contains measurements of yield from an experiment done at five different temperature levels -

***Y- axis* = yield**

And,

***x* - axis = temperature in degree Fahrenheit.**

Example : Yield Dataset : Quadratic Fit



Yield Data set contains measurements of yield from an experiment done at five different temperature levels -

***Y- axis* = yield**

And,

***x* - axis = temperature in degree Fahrenheit.**

Cost Function

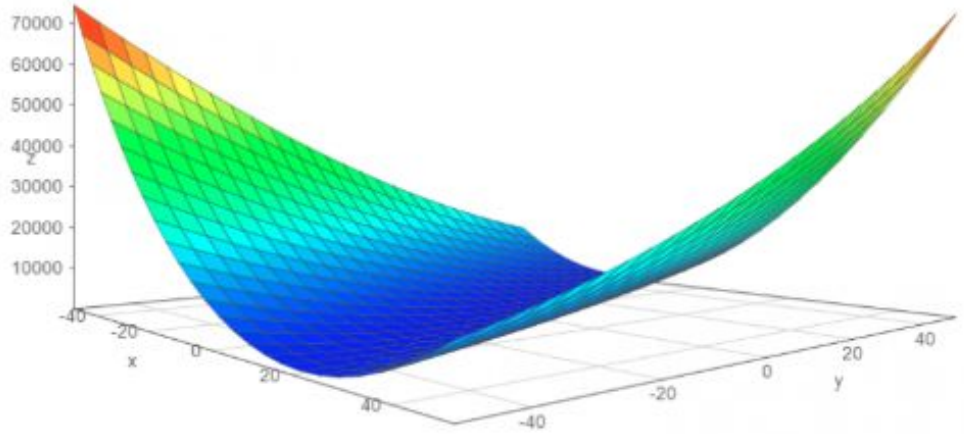
Cost Function

- ➔ **Objective**
  Minimize Cost Function
- ➔ **Brute Force?**
- ➔ **Optimization algorithms?**

This cost function is also called the mean squared error function.

In a 3D Plot with x-axis, y-axis and z-axis --

- The point where the height of the plot is least represents the point where the cost function is minimized.



Gradient Descent

Gradient Descent

➔ An Optimization Algorithm that Minimizes a Function for the Regression Model

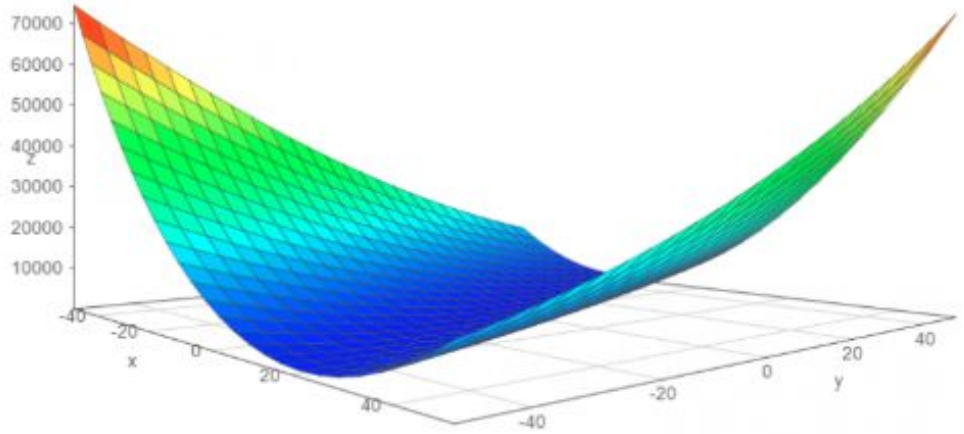
How?

➔ Slope of the tangent line at the local minima, will be equal to zero.

$$Y = mX + B$$

$$Y\_approx = m\_current * X + b\_current$$

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$$



Gradient Descent

- Training a machine learning algorithm or a neural network really is just the process of minimizing the cost function with time.
- Minimizing the derivatives as close as possible to zero determines how close you are to reaching your local maxima/minima.
- Say, after 1000 iterations, it returns m_optimum, b_optimum and cost.

$$\frac{\partial}{\partial \mathrm{m}} = \frac{2}{N}\sum_{i=1}^{N} - x_i(y_i - (mx_i + b))$$

$$\frac{\partial}{\partial \mathrm{b}} = \frac{2}{N}\sum_{i=1}^{N} -(y_i - (mx_i + b))$$

Calculating Gradient of the Slope and the Intercept

Stochastic Gradient Descent

- Gradient descent takes into account entire training set in computing cost & gradient
- One has to run thru all the samples in a single iteration to do one update for the parameters.
- MB-GD computes gradient using few mini-batches (~256) of sample training examples. 1<k<n instead of using all training set.
- MB Convergres in even fewer iterations than GD

- Batch Optimization Methods are Too Slow, Time consuming, Memory Restrictive and impractical in case of very large training sets.
- An alternative approach to Gradient Descent in calculating the parameters of the model is SGD.
- SGD uses a single training example to calculate the parameters updating one example at a time towards the local optima.

Gradient Descent VS SGD

- Correctness
- Cost function iterates over all training samples before every single update.
- Deterministic
- Batch Size : n
- Smaller number of Iterations
- High computational cost of each iteration

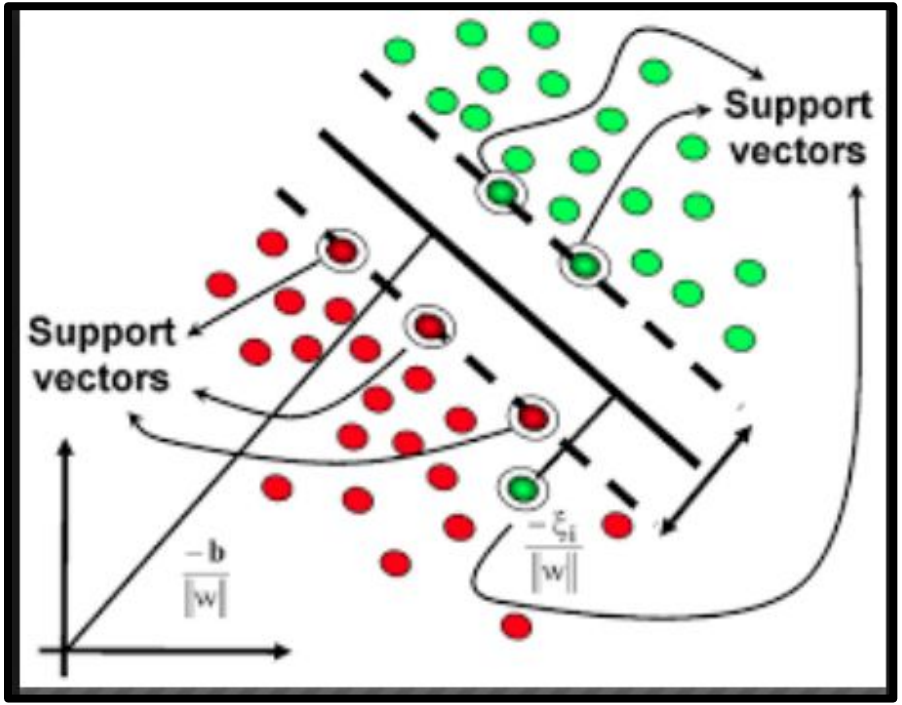
$$J(a,b) = \frac{1}{n}\sum_{i=1}^{n}(y_{i,actual} - y_{i,predicted})^2$$

- Speed
- Cost function only accounts for one training sample chosen at random.
- Stochastic
- Batch Size = 1
- Smaller Learning Rate to avoid skipping Global minima
- Lower computational cost

$$J(a,b) = (y_{i,actual} - y_{i,predicted})^2$$

Support Vector Regression (SVR)

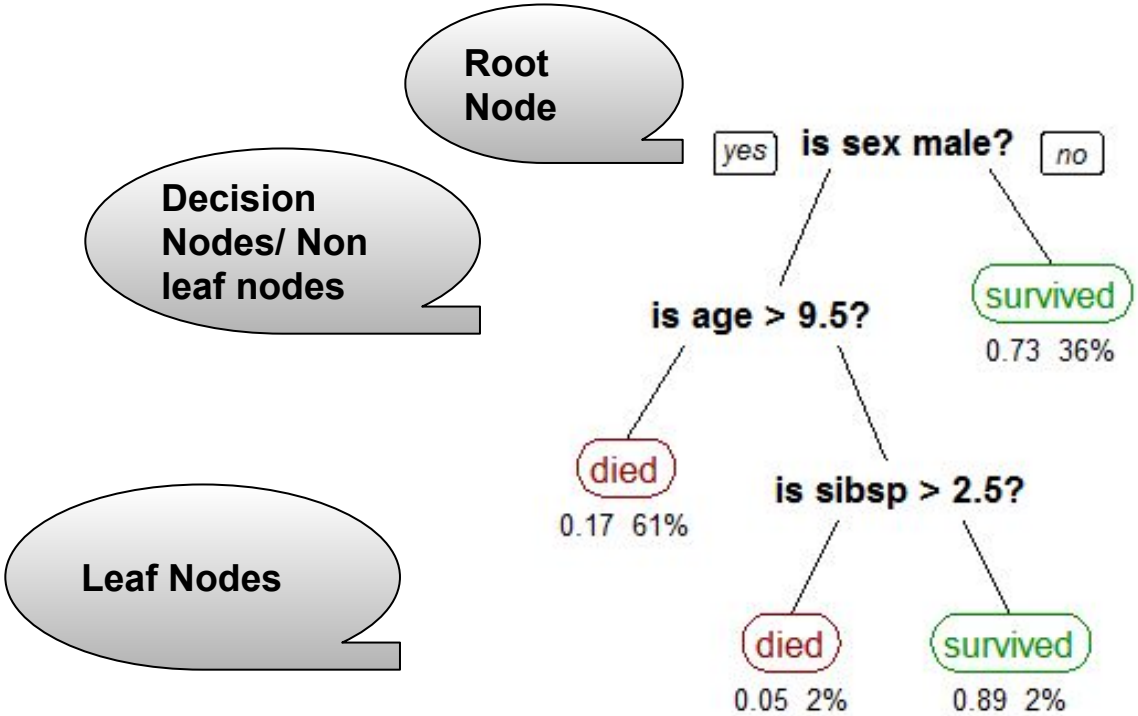
- Training a machine learning algorithm or a neural network really is just the process of minimizing the cost function with time.
- Support Vector.
- Maximum Margin



Decision Tree Regression

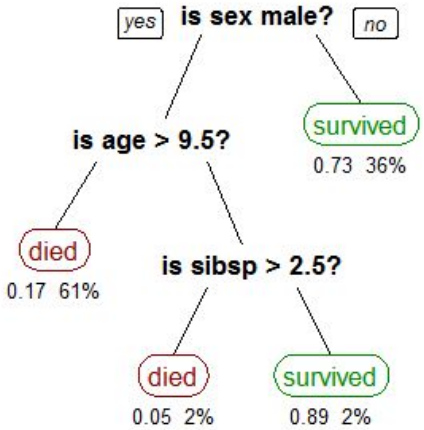
- ➔ Flow-chart-like structure
- ➔ Builds model in the form of a tree structure
- ➔ Each internal (non-leaf) node denotes a test on an attribute
- ➔ Each branch represents the outcome of a test
- ➔ Each leaf (or terminal) node holds a class label.
- ➔ The topmost node in a tree is the root node.
- ➔ Decision trees capture nonlinear interaction between the features much more effectively than linear models.

Decision Tree Regression



Decision Tree Regression

How To Grow A Decision Tree for Regression?



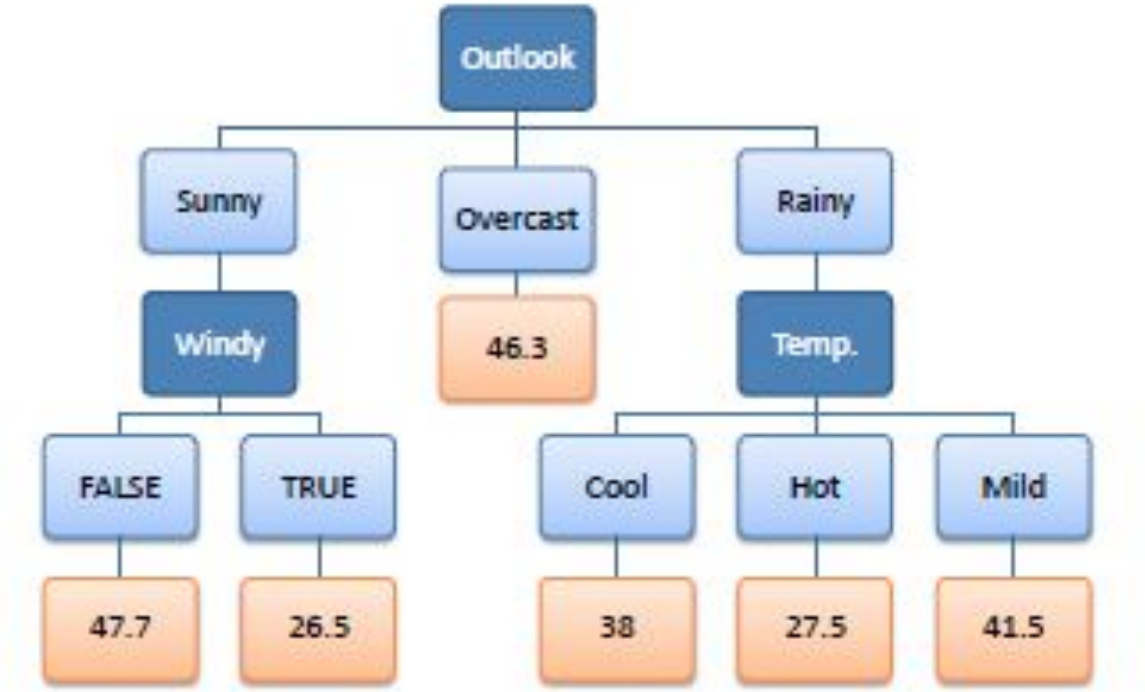
- ➔ **Goal is to find out a f(x) that minimizes the error.**
- ➔ **Tree is built from Top to Bottom.**
- ➔ **At each step, we have to find a better split (branch) in the decision tree.**

Decision Tree Regression: *Example*

Predictors | Target

Outlook	Temp	Humidity	Windy	Hours Played
Rainy	Hot	High	False	25
Rainy	Hot	High	True	30
Overcast	Hot	High	False	48
Sunny	Mild	High	False	46
Sunny	Cool	Normal	False	52
Sunny	Cool	Normal	True	23
Overcast	Cool	Normal	True	43
Rainy	Mild	High	False	35
Rainy	Cool	Normal	False	38
Sunny	Mild	Normal	False	46
Rainy	Mild	Normal	True	48
Overcast	Mild	High	True	52
Overcast	Hot	Normal	False	44
Sunny	Mild	High	True	30

Decision Tree Regression: *Example*



Decision Tree Regression: *Algorithm*

- ➔ **Top-down**
- ➔ **Greedy search through the space of possible branches**
- ➔ **No backtracking.**

- ➔ **Decision tree is built top down from the root node.**
- ➔ **Involves partitioning the data into subsets containing similar values**
- ➔ **If numerical sample is completely homogeneous, standard deviation is zero.**

Decision Tree Regression: *Algorithm*

Standard Deviation for Response Variable (one attribute)

Hours Played
25
30
46
45
52
23
43
35
38
46
48
52
44
30

$$S = \sqrt{\frac{\sum (x - \mu)^2}{n}}$$

Standard Deviation

S = 9.32

Decision Tree Regression: *Algorithm*

Standard Deviation for Response and Predictor (two attributes)

$$S(T,X) = \sum_{c \in X} P(c)S(c)$$

		Hours Played (StDev)	Count
Outlook	Overcast	3.49	4
	Rainy	7.78	5
	Sunny	10.87	5
			14

S(Hours, Outlook) = **P**(Sunny)***S**(Sunny) + **P**(Overcast)***S**(Overcast) + **P**(Rainy)***S**(Rainy)

= (4/14)*3.49 + (5/14)*7.78 + (5/14)*10.87

= 7.66

Decision Tree Regression: *Algorithm*

Standard Deviation Reduction

- **SDR is based on the decrease in std dev after a data is split on an attribute.**
- **Find that attribute which returns the highest standard deviation reduction (i.e the most similar branches).**
- **Assign it as root of the decision tree.**

Decision Tree Regression: *Algorithm*

Standard Deviation Reduction : Steps (1-5)

1. **The standard deviation of the target is calculated.**
2. **The dataset is then split on different attributes.**
3. **The standard deviation for each branch is calculated.**
4. **The resulting standard deviation is subtracted from the standard deviation before the split.**
5. **The result is the standard deviation reduction.**

Decision Tree Regression: *Algorithm*

6. **The standard deviation of the target is calculated.**
7. **The attribute with the largest standard deviation reduction is chosen for the decision node.**
8. **Dataset is divided based on the values of the selected Attribute.**
9. **A branch set with standard deviation more than 0 needs further splitting.**
10. **The process is run recursively on the non-leaf branches, until all data is processed.**

Decision Tree Regression: *Algorithm*

Standard deviation (Hours Played) = 9.32

		Hours Played (StDev)
Outlook	Overcast	3.49
	Rainy	7.78
	Sunny	10.87
SDR=1.66		

		Hours Played (StDev)
Temp.	Cool	10.51
	Hot	8.95
	Mild	7.65
SDR=0.17		

$$SDR(T,X) = S(T) - S(T,X)$$

SDR(Hours , Outlook) = **S**(Hours) – **S**(Hours, Outlook)

= 9.32 – 7.66 = 1.66

		Hours Played (StDev)
Humidity	High	9.36
	Normal	8.37
SDR=0.28		

		Hours Played (StDev)
Windy	False	7.87
	True	10.59
SDR=0.29		

Decision Tree Regression: *Algorithm*

Standard deviation (Hours Played) = 9.32

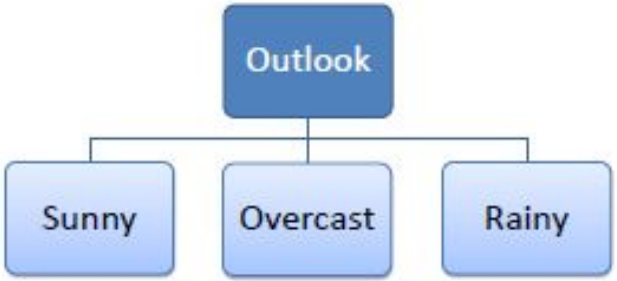
★		Hours Played (StDev)
Outlook	Overcast	3.49
	Rainy	7.78
	Sunny	10.87
SDR=1.66		

$$SDR(T,X) = S(T) - S(T,X)$$

SDR(Hours , Outlook) = **S**(Hours) – **S**(Hours, Outlook)

= 9.32 – 7.66 = 1.66

Decision Tree Regression: *Algorithm*



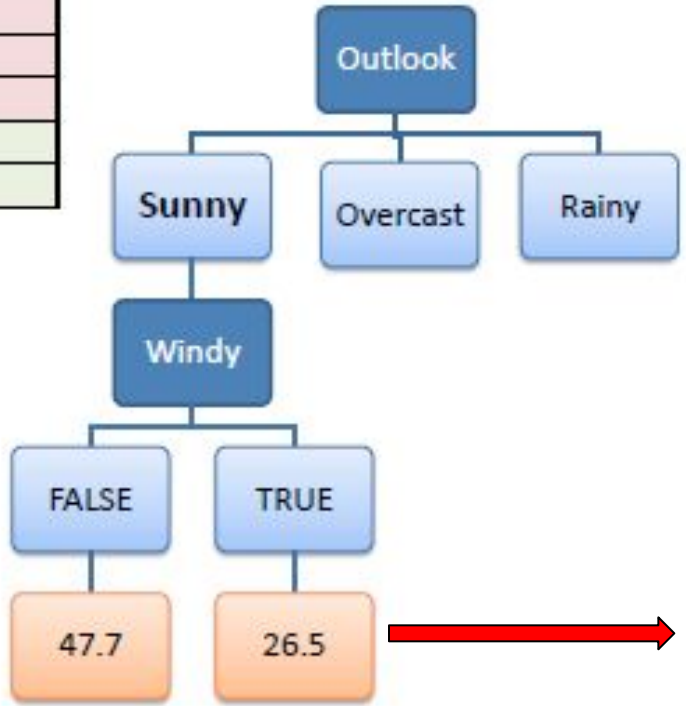
Outlook	Temp	Humidity	Windy	Hours Played
Sunny	Mild	High	FALSE	45
Sunny	Cool	Normal	FALSE	52
Sunny	Cool	Normal	TRUE	23
Sunny	Mild	Normal	FALSE	46
Sunny	Mild	High	TRUE	30
Rainy	Hot	High	FALSE	25
Rainy	Hot	High	TRUE	30
Rainy	Mild	High	FALSE	35
Rainy	Cool	Normal	FALSE	38
Rainy	Mild	Normal	TRUE	48
Overcast	Hot	High	FALSE	46
Overcast	Cool	Normal	TRUE	43
Overcast	Mild	High	TRUE	52
Overcast	Hot	Normal	FALSE	44

Decision Tree Regression: *Algorithm*

Temp	Humidity	Windy	Hours Played
Mild	High	FALSE	45
Cool	Normal	FALSE	52
Mild	Normal	FALSE	46
Cool	Normal	TRUE	23
Mild	High	TRUE	30

★		Hours Played (StDev)
Windy	False	3.09
	True	3.50
SDR= 7.62		

SDR = $10.87-((3/5)*3.09 + (2/5)*3.5)$

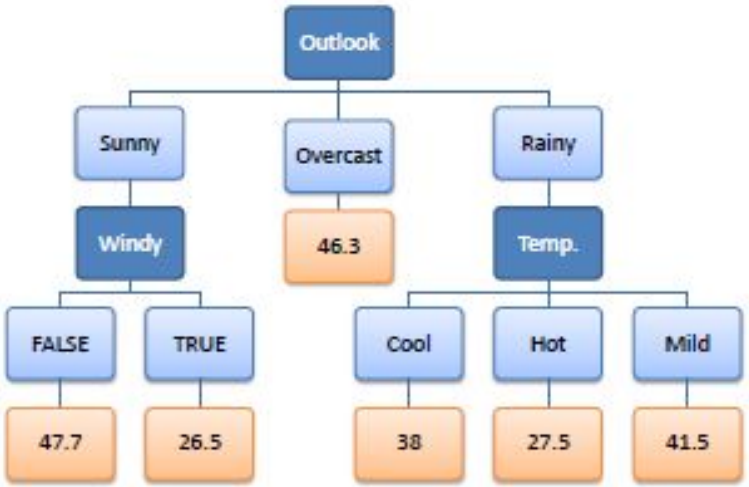


When the number of instances is more than one at a leaf node we calculate the *average* as the final value for the target.

Decision Tree Regression: *Algorithm*

Predictors | Target

Outlook	Temp	Humidity	Windy	Hours Played
Rainy	Hot	High	False	26
Rainy	Hot	High	True	30
Overcast	Hot	High	False	48
Sunny	Mild	High	False	46
Sunny	Cool	Normal	False	62
Sunny	Cool	Normal	True	23
Overcast	Cool	Normal	True	43
Rainy	Mild	High	False	36
Rainy	Cool	Normal	False	38
Sunny	Mild	Normal	False	48
Rainy	Mild	Normal	True	48
Overcast	Mild	High	True	52
Overcast	Hot	Normal	False	44
Sunny	Mild	High	True	30

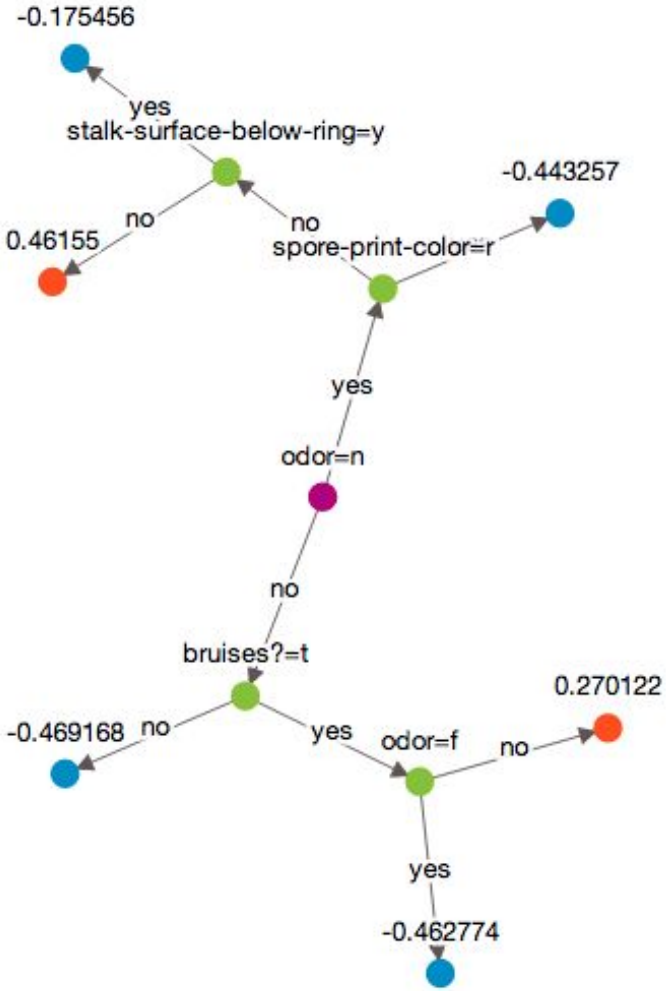


The process is run recursively on the non-leaf branches, until all data is processed.

Random Forest Regression

- ➔ Most Effective model for Predictive Analytics
- ➔ The random forest model is a sum of multiple base models.
- ➔ This is called model ensembling.
- ➔ Predictive Performance is increased by combining predictions from multiple base models.
- ➔ Each base classifier is a decision tree.
- ➔ All base models are creating independently using a different subsample of the data.
- ➔ Random forest works best with tabular data with numerical features.

Random Forest Regression *(Example)*



Tuning Parameters

- **Num_trees :** Controls the number of trees in the final model. More trees, more the training & prediction time. Num_trees~Higher accuracy.
- **Max_depth :** Restricts the depth of each individual tree to prevent overfitting.
- **Step_size :** Smaller value will take more iterations to reach the same level of training error of a larger step size.
- **min_child _weight:** The minimum observations required at a leaf node. Larger value produces simpler trees.
- **Min_loss_reduction :** This restricts the reduction of loss function for a node split. Larger value produces simpler trees.
- **Row_subsample :** Using only a fraction of rows at each iteration to produce more robust model.
- **Column_subsample:** Using only a subset of the columns to use at each iteration.

Evaluating Model Performance using Regression Metrics

In a Regression task, model learns to predict numeric scores.

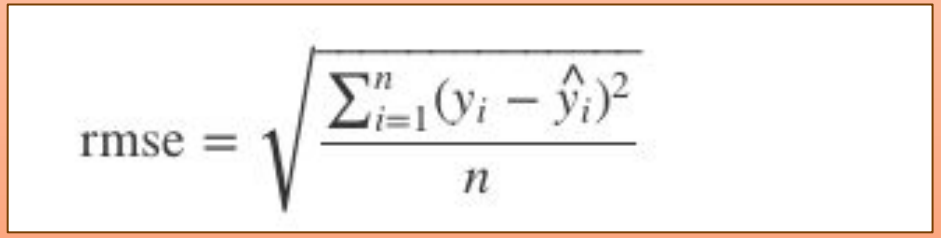
E.g.,
Predicting the price of a stock for future based on past price history.

Or,
Or predicting a user's rating on an item based on past trends.

Two Methods of Measuring Performance

- ★ Root Mean Squared Error
- ★ Max-Error

Root Mean Squared Error (RMSE)



Defined as the square root of the average squared difference between the actual score and the predicted score:

yi → denotes the true score for the i-th data point

And,

Ŷi → denotes the predicted value.

R Squared

$$\text{R-Square} = \frac{\sum(Y\_actual - Y\_predicted)^2}{\sum(Y\_actual - Y\_mean)^2}$$

R2 measures the proportion of variation in the data that is accounted for in the model.

It evaluates how well the model fits the data.

Also known as the coefficient of determination

Max Error

- ➔ Worst case error
- ➔ Rarely used
- ➔ Finds out the maximum error possible between specified Quantile Ranges.

.

Summary

1. Multiple Linear regression models the relation between a response variable Y and multiple predictor variables X1.....Xp.

2. Factor variables need to be converted into numeric variables for use in a regression.

3. The most common method to encode a factor variable with P distinct values is to represent them using P-1 dummy values.

Hands-on Lab

- ➔ Using numpy, Sklearn and matplotlib
- ➔ Plotting regression models using scatterplot
- ➔ Creating correlation plots


Thank
You