Machine Learning

Lesson 5—Regression

Learning Objectives

- ✓ Explain Regression and its types
- ✓ Describe Linear Regression: Equations and Algorithms

Regression

Topic 1—Regression and Its Types

What Is Regression?

> In statistical modeling, regression analysis is a set of statistical processes for estimating the relationships among variables.

wikipedia

Types of Regression

- Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Decision Tree Regression
- Random Forest Regression

Types of Regression

LINEAR REGRESSION

- Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Decision Tree Regression
- Random Forest Regression

Linear regression is a linear approach for modeling the relationship between a scalar dependent variable y and an independent variable x.

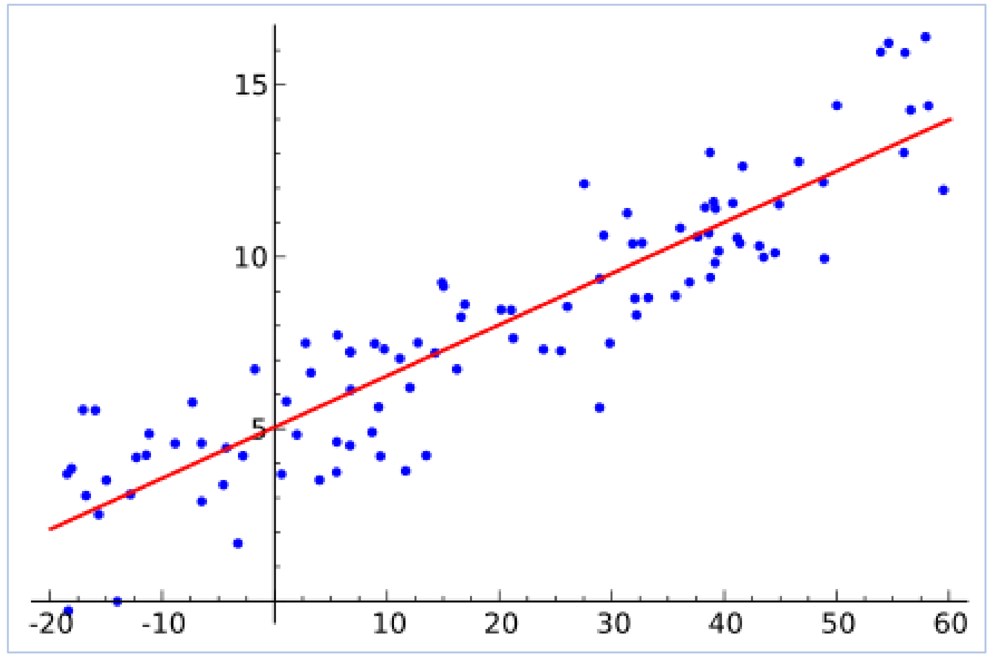
$$\hat{y} = \boldsymbol{w}^\top \boldsymbol{x}$$

where x, y, w are vectors of real numbers and w is a vector of weight parameters.

The equation is also written as:

***y* = *wx* + b**

where b is the bias or the value of output for zero input



Types of Regression

MULTIPLE LINEAR REGRESSION

Linear Regression

Multiple Linear Regression

Polynomial Regression

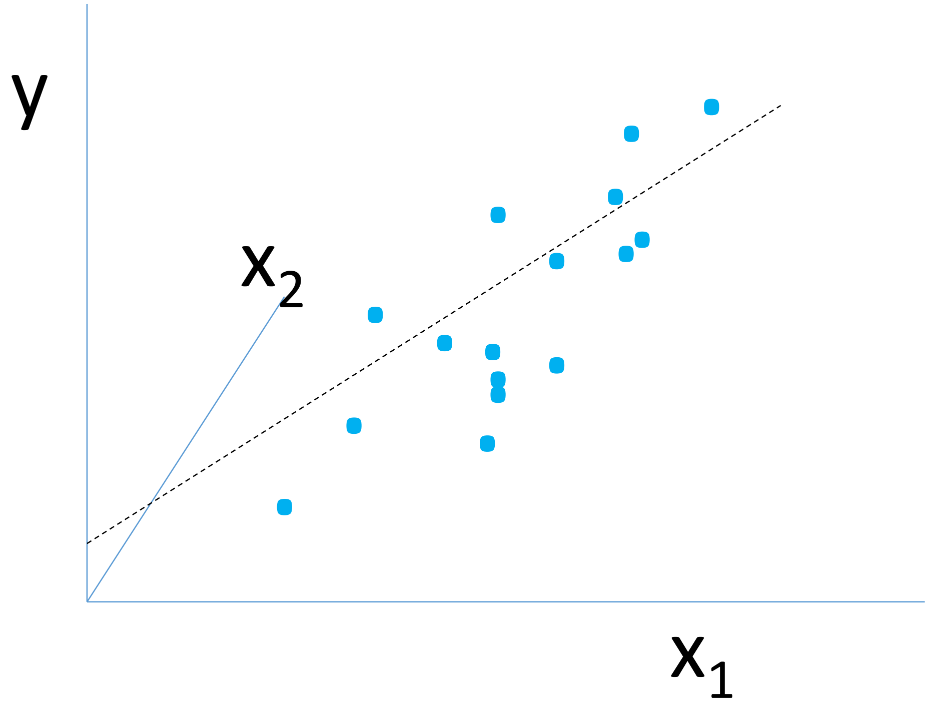
Decision Tree Regression

Random Forest Regression

It is a statistical technique used to predict the outcome of a response variable through several explanatory variables and model the relationships between them.

It represents line fitment between multiple inputs and one output, typically:

$y = w_1x_1 + w_2x_2 + b$



The graph shows dependent variable y plotted against two independent variables x_1 and x_2. It is shown in 3D. More independent variables (if involved) will increase the dimensions further.

Types of Regression

POLYNOMIAL REGRESSION

Linear Regression

Multiple Linear Regression

Polynomial Regression

Decision Tree Regression

Random Forest Regression

Polynomial regression is applied when data is not formed in a straight line.

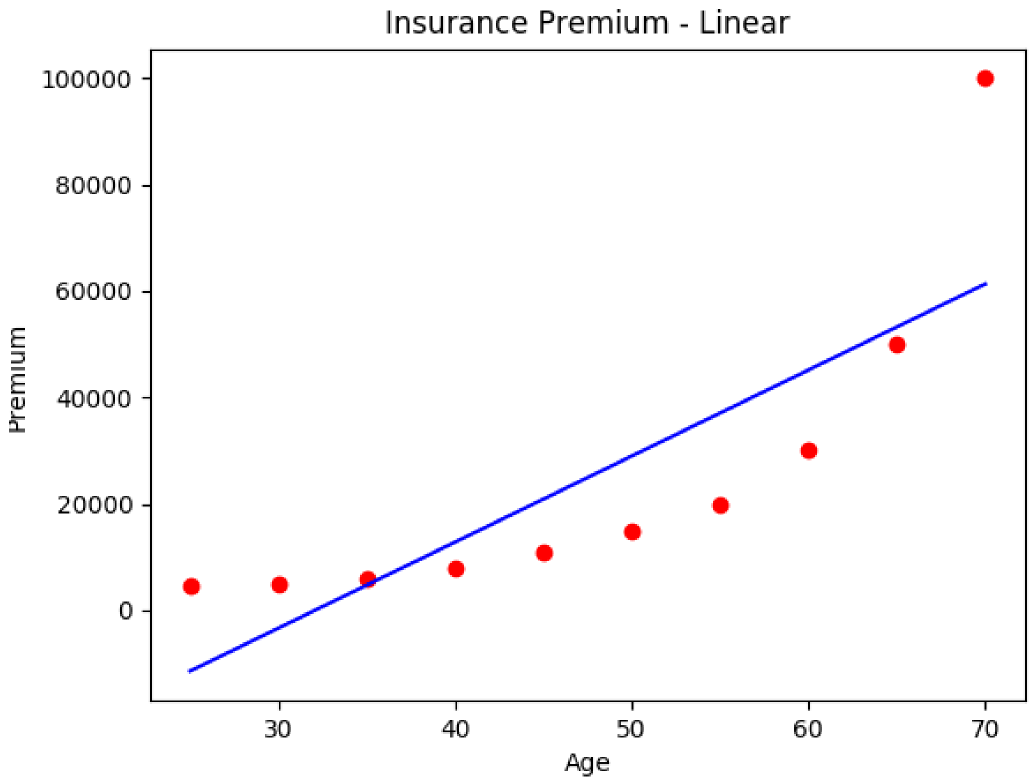
It is used to fit a linear model to non-linear data by creating new features from powers of non-linear features.

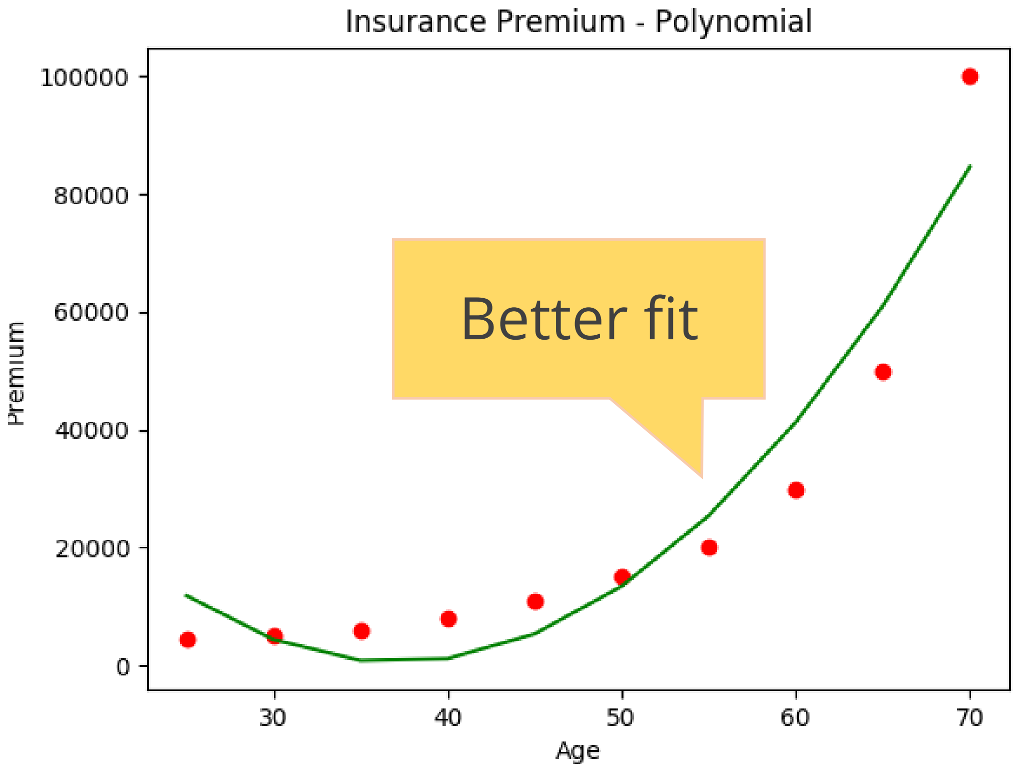
Example: Quadratic features

$$x_2' = x_2^2$$

$$y = w_1x_1 + w_2x_2^2 + 6$$

$$= w_1x_1 + w_2x_2' + 6$$


Insurance Premium - Linear


Insurance Premium - Polynomial

Better fit

Demo

Polynomial Regression

Predict yearly insurance claims based on people's age. You can perform this in your lab environment using the dataset in the LMS.

Types of Regression

WHAT IS A DECISION TREE?

- Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Decision Tree Regression
- Random Forest Regression

> A decision tree is a graphical representation of all the possible solutions to a decision based on a few conditions.

Types of Regression

DECISION TREE REGRESSION ALGORITHM

- Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Decision Tree Regression
- Random Forest Regression

- Consider data with two independent variables, X_1 and X_2.
- The algorithm splits data into two parts. Split boundaries are decided based on reduction in leaf impurity.
- The algorithm keeps on splitting subsets of data till it finds that further split will not give any further value.
- Calculate average of dependent variables (y) of each leaf. That value represents the regression prediction of that leaf.
- This tree splits leaves based on x_1 being lower than 0.1973. At second level, it splits based on x_1 value again.
- At each node, the MSE (mean square error or the average distance of data samples from their mean) of all data samples in that node is calculated. The mean value for that node is provided as “value” attribute.

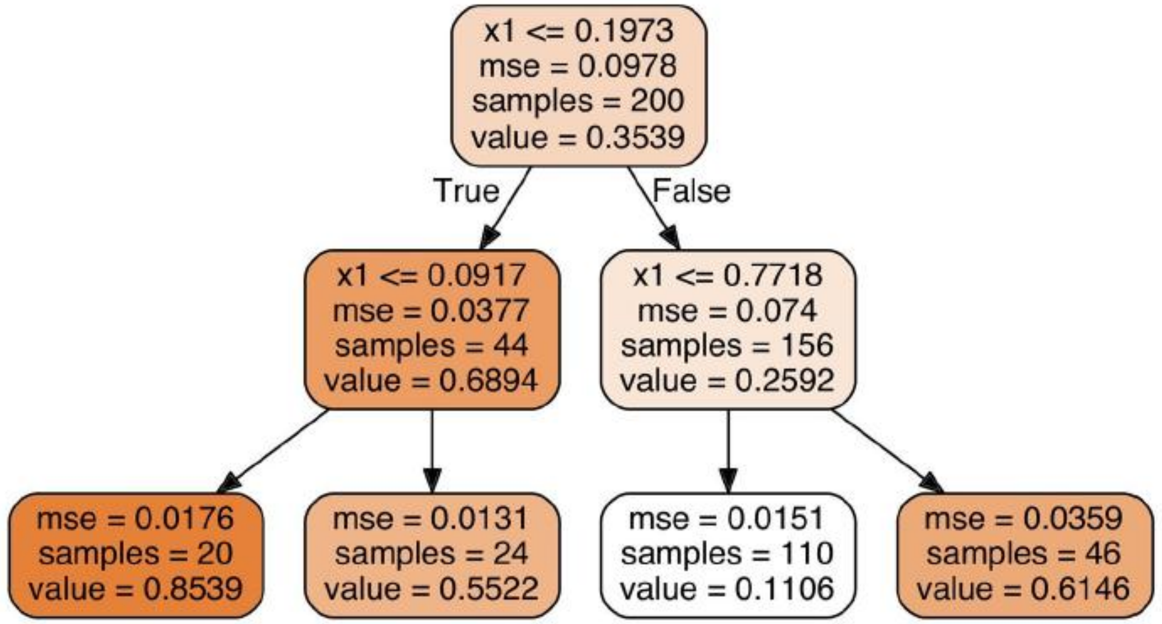


Image Credit : “Hands-on Machine Learning with Scikit-Learn and TensorFlow ” by Aurelien Geron

Types of Regression

DECISION TREE REGRESSION

Linear Regression

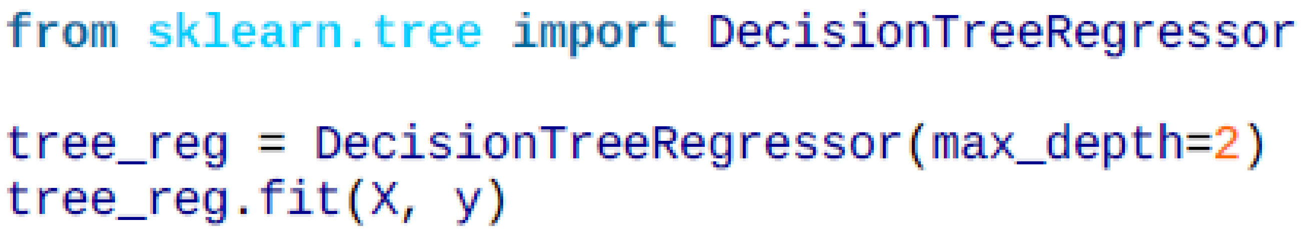
Multiple Linear Regression

Polynomial Regression

Decision Tree Regression

Random Forest Regression

Decision Trees can perform regression tasks. The following is a decision tree on a noisy quadratic dataset:



```python
from sklearn.tree import DecisionTreeRegressor

tree_reg = DecisionTreeRegressor(max_depth=2)
tree_reg.fit(X, y)
```

Types of Regression

HOW DECISION TREES PERFORM REGRESSION

- Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Decision Tree Regression
- Random Forest Regression

- The dataset looks similar to classification DT. The main difference is that instead of predicting class, each node predicts value.
- This value represents the average target value of all the instances in this node.
- This prediction has an associated MSE or Mean Squared Error over the node instances.
- This mean value of the node is the predicted value for a new data instance that ends up in that node.

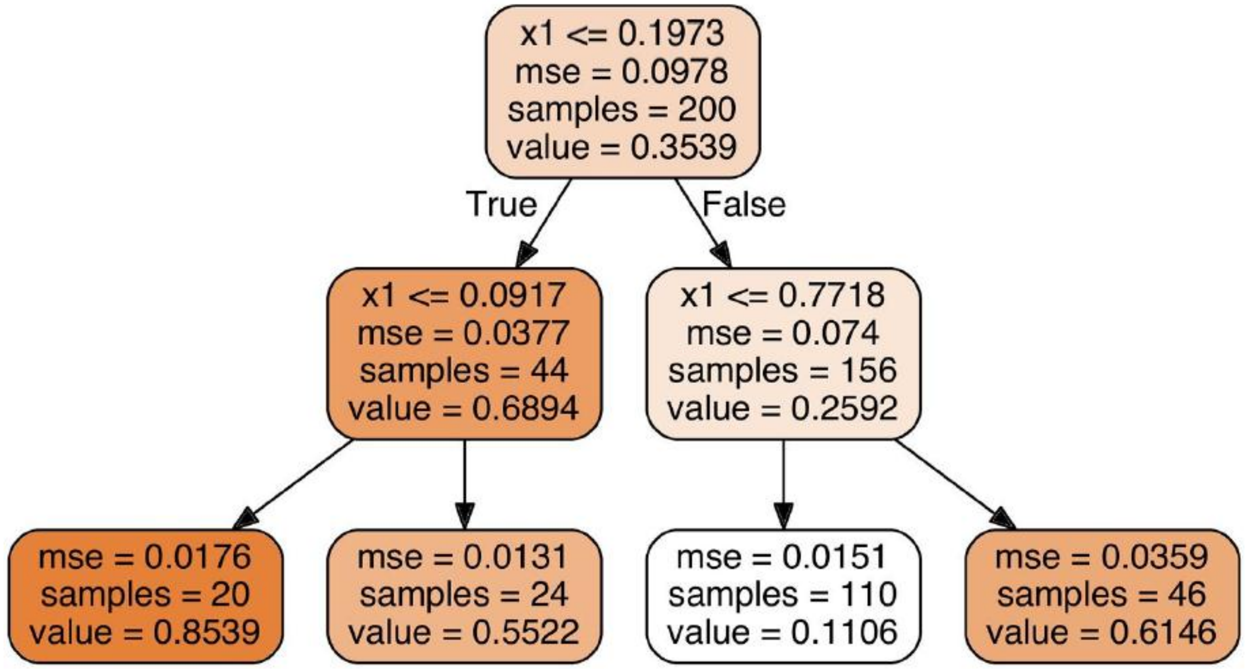


Image Credit : "Hands-on Machine Learning with Scikit-Learn and TensorFlow " by Aurelien Geron

Types of Regression

DECISION TREE REGRESSION PLOT

- Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Decision Tree Regression
- Random Forest Regression

The regression plot is shown below. Notice that predicted value for each region is the average of the values of instances in that region.

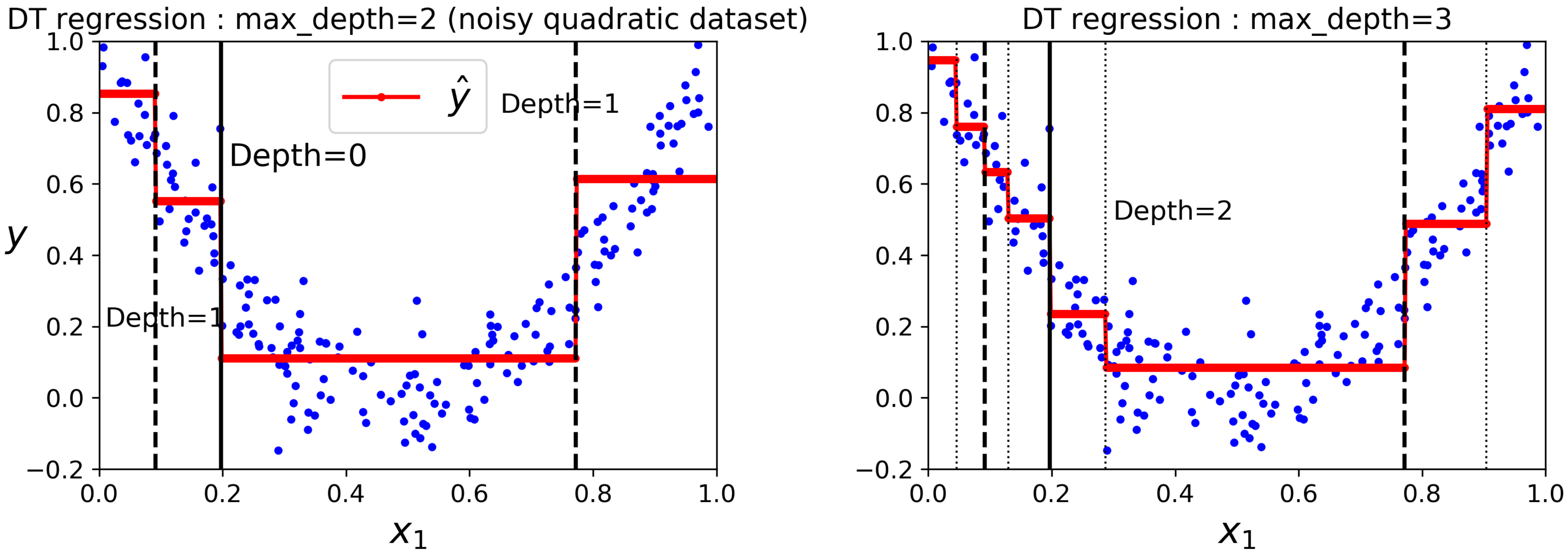


Image Credit : "Hands-on Machine Learning with Scikit-Learn and TensorFlow " by Aurelien Geron

Types of Regression

DECISION TREE: REGULARIZATION

- Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Decision Tree Regression
- Random Forest Regression

- Decision Trees are non-parametric models, which means that the number of parameters is not determined prior to training. Such models will normally overfit data.
- In contrast, a parametric model (such as a linear model) has a predetermined number of parameters, thereby reducing its degrees of freedom. This in turn prevents overfitting.
- To prevent overfitting, one must restrict the degrees of freedom of a Decision Tree. This is called regularization.

Types of Regression

DECISION TREE: FEATURES OF REGULARIZATION

- Regularization is any modification made to the learning algorithm that reduces its generalization error but not its training error.
- In addition to varying the set of functions or the set of features possible for training an algorithm to achieve optimal capacity, one can resort to other ways to achieve regularization.

Types of Regression

DECISION TREE: WAYS TO REGULARIZE DECISION TREES

- Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Decision Tree Regression
- Random Forest Regression

- max_depth – limit the maximum depth of the tree
- min_samples_split – the minimum number of samples a node must have before it can be split
- min_samples_leaf – the minimum number of samples a leaf node must have
- min_weight_fraction_leaf – same as min_samples_leaf but expressed as a fraction of total instances
- max_leaf_nodes – maximum number of leaf nodes
- max_features – maximum number of features that are evaluated for splitting at each node

Types of Regression

DECISION TREE: REGRESSION (CART ALGORITHM)

Linear Regression

Multiple Linear Regression

Polynomial Regression

Decision Tree Regression

Random Forest Regression

To achieve regression task, the CART algorithm follows the logic as in classification; however, instead of trying to minimize the leaf impurity, it tries to minimize the MSE or the mean square error, which represents the difference between observed and target output – (y-y’)^2”

$$J(k, t_k) = \frac{m_{\text{left}}}{m}\text{MSE}_{\text{left}} + \frac{m_{\text{right}}}{m}\text{MSE}_{\text{right}} \quad \text{where} \begin{cases} \text{MSE}_{\text{node}} = \sum_{i \in \text{node}} \left(\hat{y}_{\text{node}} - y^{(i)}\right)^2 \\ \hat{y}_{\text{node}} = \frac{1}{m_{\text{node}}} \sum_{i \in \text{node}} y^{(i)} \end{cases}$$

J(k, t$_k$) represents the total loss function that one wishes to minimize.

It is the sum of weighted (by number of samples) MSE for the left and right node after the split.

Types of Regression

DECISION TREE: REGRESSION (CART ALGORITHM) (CONTD.)

Linear Regression

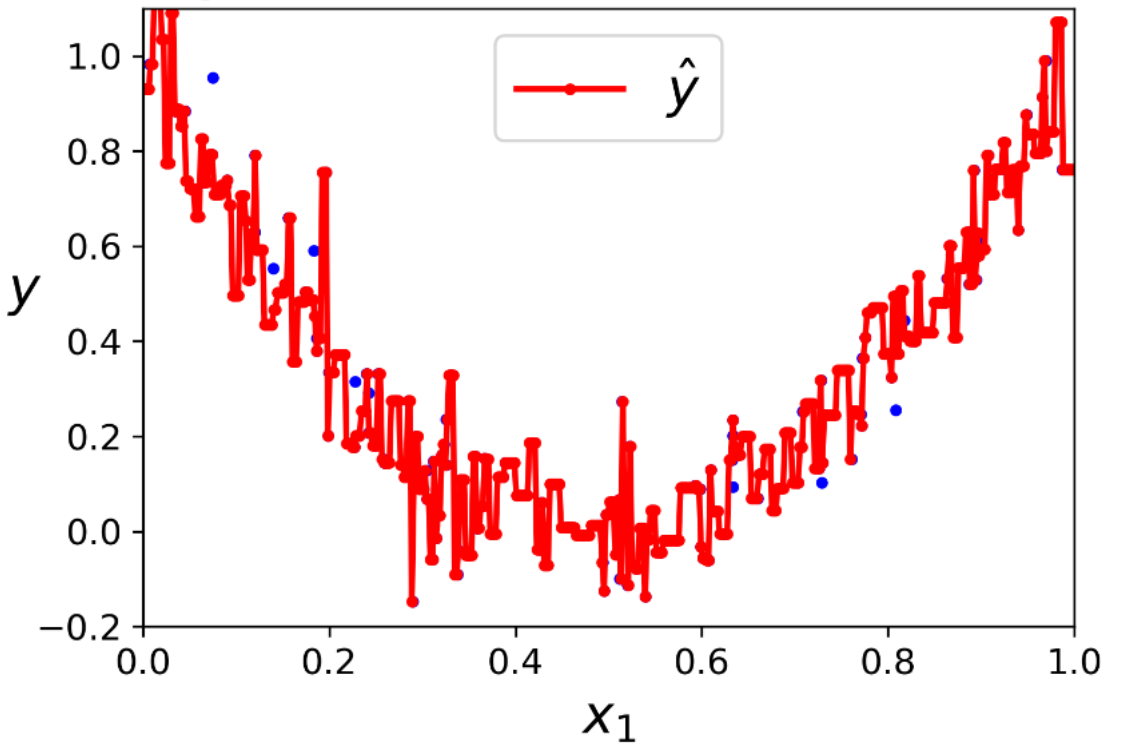
Multiple Linear Regression

Polynomial Regression

Decision Tree Regression

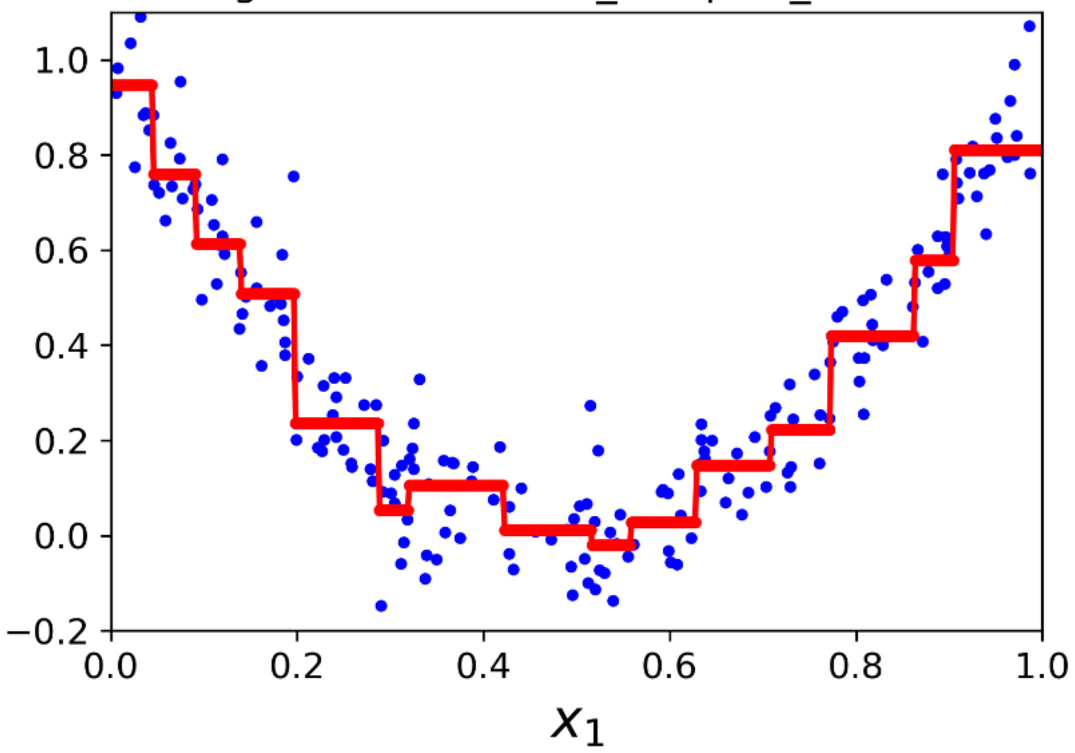
Random Forest Regression

DT regression with no restrictions - leads to overfit



The graph represents regression line if the decision tree is allowed to split continuously (and go deeper and deeper) without any stoppage – this is an overfitted situation.

Regularization : min_samples_leaf=10



The graph represents regularization where a decision tree is not allowed to split any further if the number of samples in a node falls below 10. It prevents overfitting and is more practical to use.

Demo

Decision Tree Regression

1. Predict yearly insurance claims based on people’s age using decision trees.

2. Generate random quadratic data and perform Decision Tree regression.

You can perform this in your lab environment using the dataset in the LMS.

Types of Regression

WHAT IS RANDOM FOREST?

- Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Decision Tree Regression
- Random Forest Regression

Ensemble Learning uses the same algorithm multiple times or a group of different algorithms together to improve the prediction of a model.

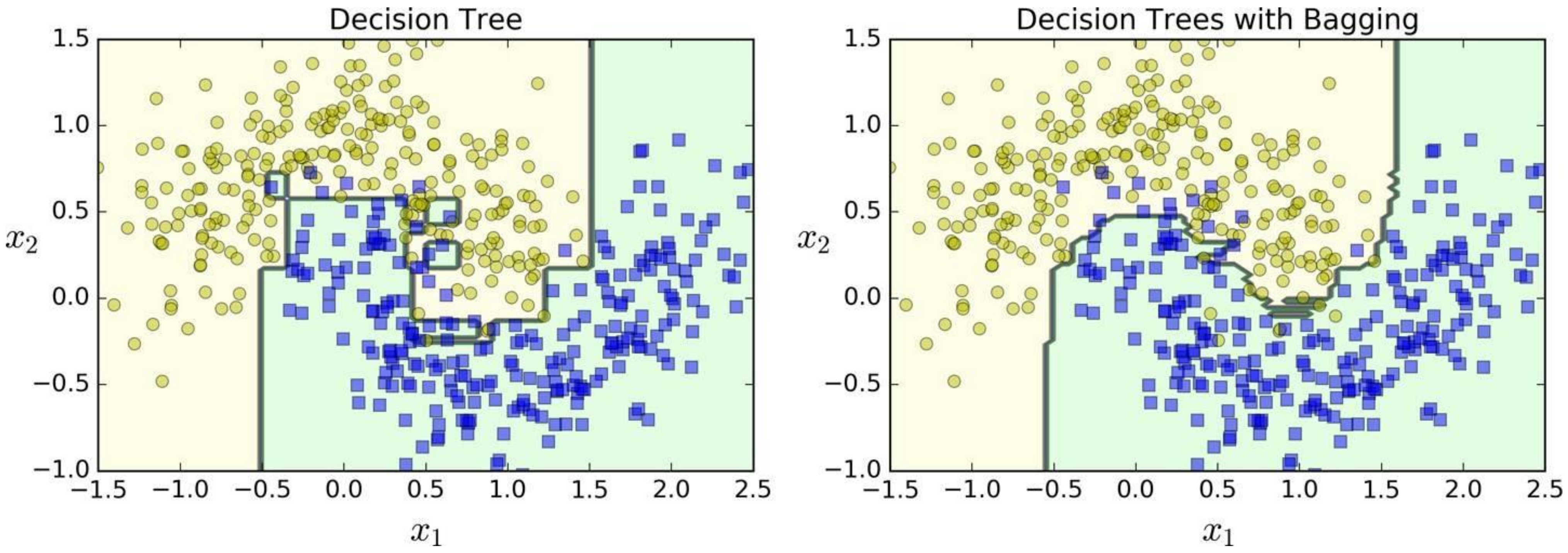
Random Forests use an ensemble of decision trees to perform regression tasks.

Types of Regression

RANDOM FOREST REGRESSION ALGORITHM

- Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Decision Tree Regression
- Random Forest Regression

1. Pick any random K datapoints from the dataset
2. Build a decision tree from these K points
3. Choose the number of trees you want (N) and repeat steps 1 and 2
4. For a new data point, average the value of y predicted by all the N trees. This is the predicted value.



A single decision tree vs. a bagging ensemble of 500 trees

Image Credit : "Hands-on Machine Learning with Scikit-Learn and TensorFlow " by Aurelien Geron

Demo

Random Forest

Predict yearly insurance claims based on people's age using Random Forest. You can perform this in your lab environment using the dataset in the LMS.

Regression

Topic 2—Linear Regression: Equations and Algorithms

Mean Square Error (MSE)

Mean-squared error (MSE) is used to measure the performance of a model.

$\hat{y}^{(\text{test})}$ = predictions made by the model on the test data

$y^{(\text{test})}$ = expected output

$$\text{MSE}_{\text{test}} = \frac{1}{m} \sum_i (\hat{y}^{(\text{test})} - y^{(\text{test})})_i^2$$

- The objective is to design an algorithm that decreases the MSE by adjusting the weights *w* during the training session.
- The above function is also called the LOSS FUNCTION or the COST FUNCTION. The value needs to be minimized.

Solving Linear Regression

Find parameters θ that minimize the least squares (OLS) equation, also called Loss Function:

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

This decreases the difference between observed output [h(x)] and desired output [y]

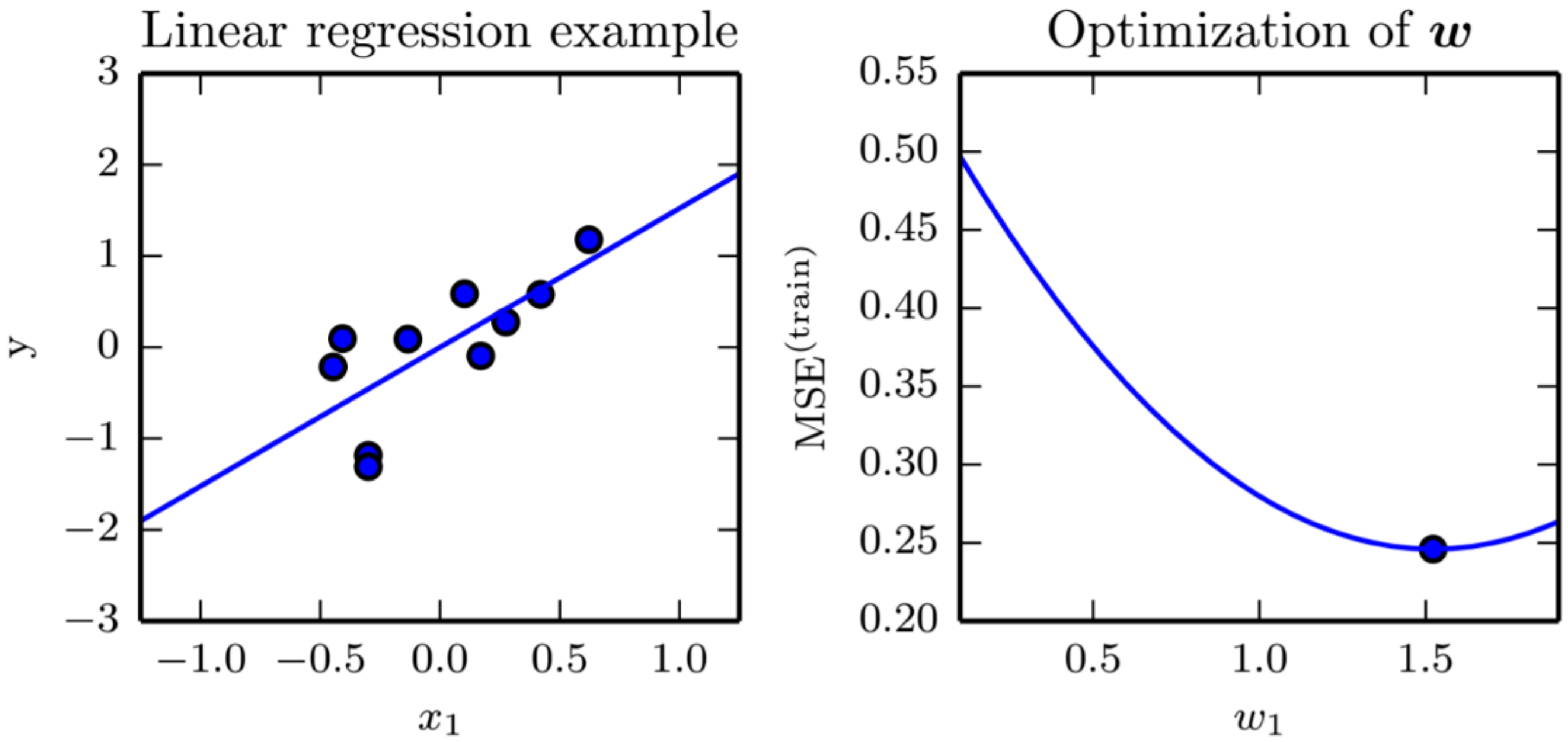


Image Credit : "Hands-on Machine Learning with Scikit-Learn and TensorFlow " by Aurelien Geron

Learning the Parameters

There are two ways to learn the parameters:

1. **Normal Equation:** Set the derivative (slope) of the Loss function to zero (this represents minimum error point).

   $\nabla_\theta J(\theta) = 0$ **and solve for θ**

   $\theta = (X^T.X)^{-1}.X^T.y$ **where X, y are input/output vectors**

2. **LMS algorithm:** The minimization of the MSE loss function in this case is called LMS (least mean squared) rule or Widrow-Hoff learning rule. This typically uses the Gradient Descent algorithm.

Learning the Parameters

DERIVING NORMAL EQUATION

Normal Equation

LMS Algorithm

To minimize MSE$_{\text{train}}$, solve the areas where the gradient (or slope $\partial E/\partial w$) with respect to weight *w* is 0.

$$\nabla_{\boldsymbol{w}} \text{MSE}_{\text{train}} = 0$$

$$\Rightarrow \nabla_{\boldsymbol{w}} \frac{1}{m} || \hat{\boldsymbol{y}}^{(\text{train})} - \boldsymbol{y}^{(\text{train})} ||_2^2 = 0$$

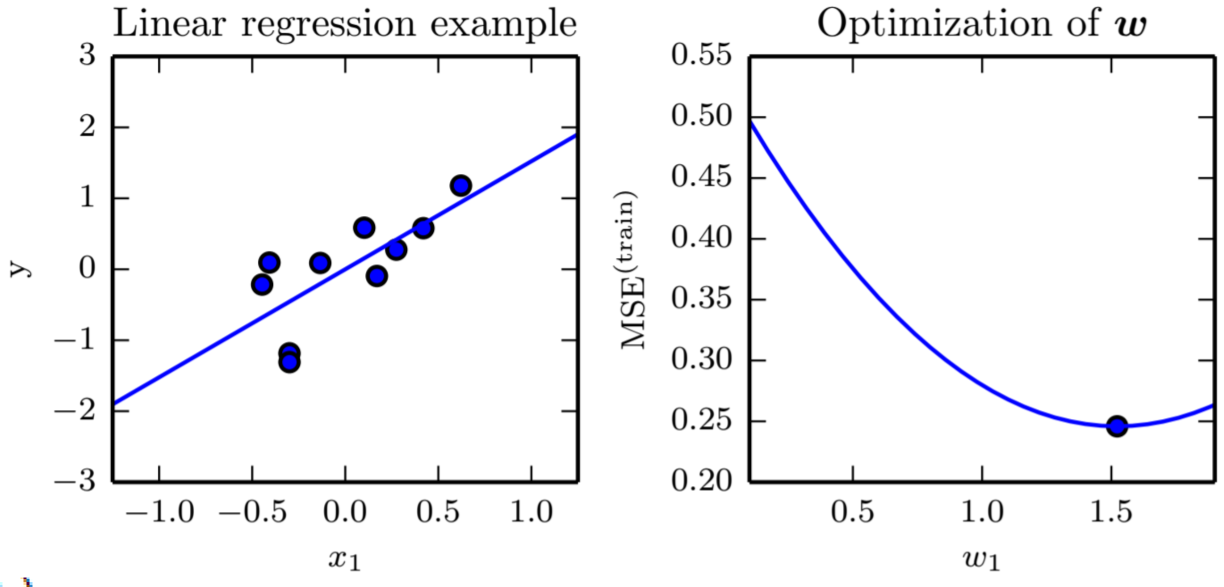
$$\Rightarrow \frac{1}{m} \nabla_{\boldsymbol{w}} || \boldsymbol{X}^{(\text{train})} \boldsymbol{w} - \boldsymbol{y}^{(\text{train})} ||_2^2 = 0$$

$$\Rightarrow \nabla_{\boldsymbol{w}} \left(\boldsymbol{X}^{(\text{train})} \boldsymbol{w} - \boldsymbol{y}^{(\text{train})} \right)^{\top} \left(\boldsymbol{X}^{(\text{train})} \boldsymbol{w} - \boldsymbol{y}^{(\text{train})} \right) = 0$$

$$\Rightarrow \nabla_{\boldsymbol{w}} \left(\boldsymbol{w}^{\top} \boldsymbol{X}^{(\text{train})\top} \boldsymbol{X}^{(\text{train})} \boldsymbol{w} - 2 \boldsymbol{w}^{\top} \boldsymbol{X}^{(\text{train})\top} \boldsymbol{y}^{(\text{train})} + \boldsymbol{y}^{(\text{train})\top} \boldsymbol{y}^{(\text{train})} \right) = 0$$

$$\Rightarrow 2 \boldsymbol{X}^{(\text{train})\top} \boldsymbol{X}^{(\text{train})} \boldsymbol{w} - 2 \boldsymbol{X}^{(\text{train})\top} \boldsymbol{y}^{(\text{train})} = 0$$

$$\Rightarrow \boldsymbol{w} = \left(\boldsymbol{X}^{(\text{train})\top} \boldsymbol{X}^{(\text{train})} \right)^{-1} \boldsymbol{X}^{(\text{train})\top} \boldsymbol{y}^{(\text{train})}$$



Learning the Parameters

DERIVING NORMAL EQUATION (CONTD.)

Normal Equation

LMS Algorithm

$$\boldsymbol{w} = \left(\boldsymbol{X}^{(\text{train})\top}\boldsymbol{X}^{(\text{train})}\right)^{-1}\boldsymbol{X}^{(\text{train})\top}\boldsymbol{y}^{(\text{train})}$$

This can be simplified as:

$w = (X^T.X)^{-1}.X^T.y$

This is called the Normal Equation.

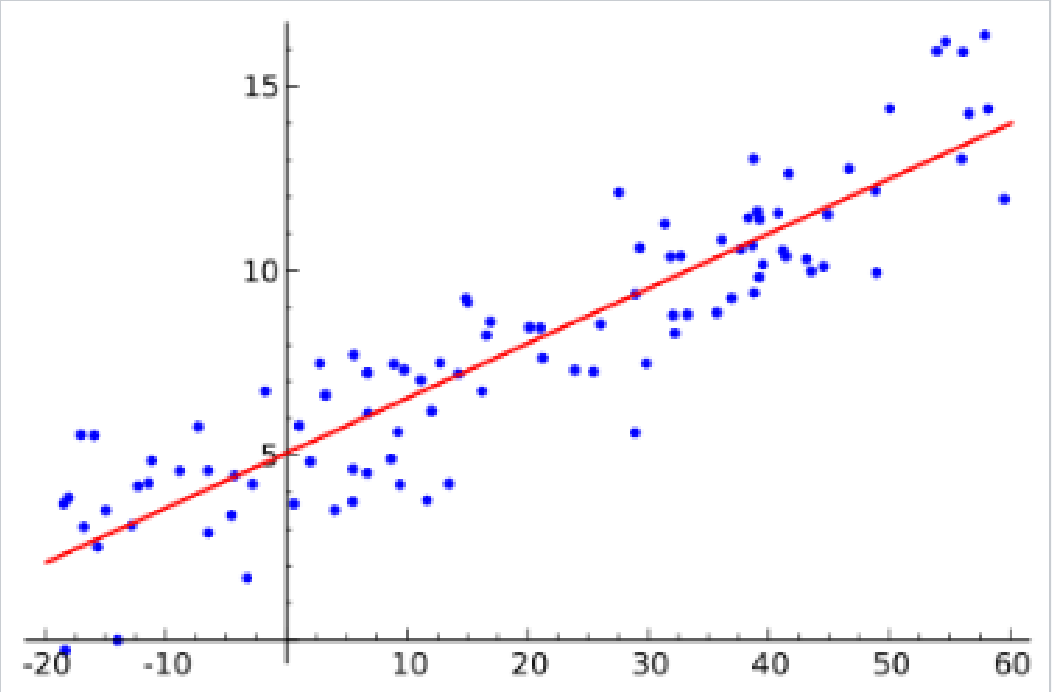


Illustration of linear regression on a data set.

Learning the Parameters

LMS ALGORITHM

Normal Equation

LMS Algorithm

In case of Linear Regression, the hypotheses are represented as:

$$y = h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

Where θ_i's are parameters (or weights).
θ_i's can also be represented as $\theta_0 * x_0$ where $x_0 = 1$, so:

$$h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x$$

The cost function (also called Ordinary Least Squares or OLS) defined is essentially MSE – the ½ is just to cancel out the 2 after derivative is taken and is less significant.

$$J(\theta) = \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

It is advisable to start with random θ. Then repeatedly adjust θ to make $J(\theta)$ smaller.

Learning the Parameters

LMS ALGORITHM: GRADIENT DESCENT

Gradient descent is an algorithm used to minimize the loss function.

Normal Equation

LMS Algorithm

The *J(θ)* in *dJ(θ)/dθ* represents the cost function or error function that you wish to minimize, example: OLS or (y-y')^2.

Minimizing this would mean that y' approaches y. In other words, observed output approaches the expected output.

Other examples of loss or cost function include cross-entropy, that is, y*log(y'), which also tracks the difference between y and y'.

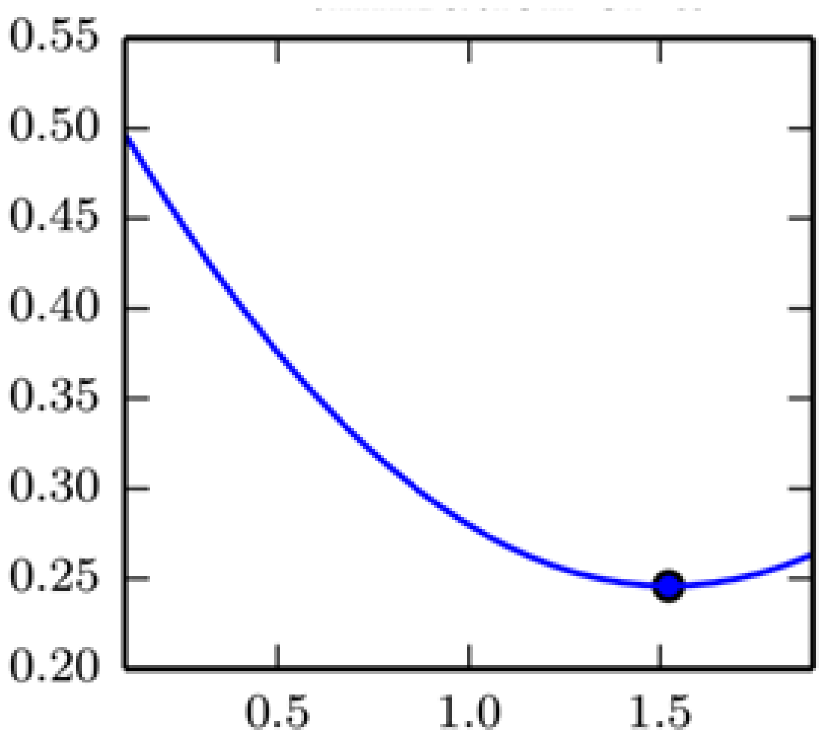


Image Credit : "Hands-on Machine Learning with Scikit-Learn and TensorFlow " by Aurelien Geron

Learning the Parameters

LMS ALGORITHM: GRADIENT DESCENT

Normal Equation

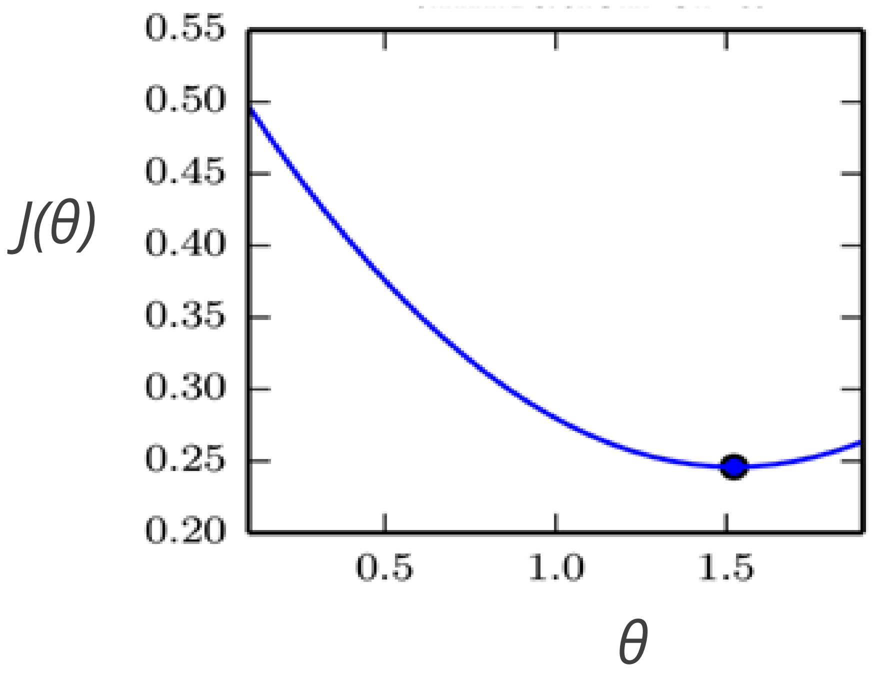
LMS Algorithm

1. The slope of *J(θ)* vs *θ* graph is *dJ(θ)/dθ.*
2. Adjust *θ repeatedly. α* is the learning rate.
3. This algorithm repeatedly takes a step toward the path of steepest descent.

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

4. Calculate derivative term for one training sample (x, y) to begin with.

$$\begin{aligned} \frac{\partial}{\partial \theta_j} J(\theta) &= \frac{\partial}{\partial \theta_j} \frac{1}{2} \left(h_\theta(x) - y\right)^2 \\ &= 2 \cdot \frac{1}{2} \left(h_\theta(x) - y\right) \cdot \frac{\partial}{\partial \theta_j} \left(h_\theta(x) - y\right) \\ &= \left(h_\theta(x) - y\right) \cdot \frac{\partial}{\partial \theta_j} \left(\sum_{i=0}^{n} \theta_i x_i - y \right) \\ &= \left(h_\theta(x) - y\right) x_j \end{aligned}$$



5. Update rule for one training sample $\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)}) \right) x_j^{(i)}$

Image Credit : "Hands-on Machine Learning with Scikit-Learn and TensorFlow " by Aurelien Geron

Learning the Parameters

LMS ALGORITHM: BATCH GRADIENT DESCENT

Normal Equation

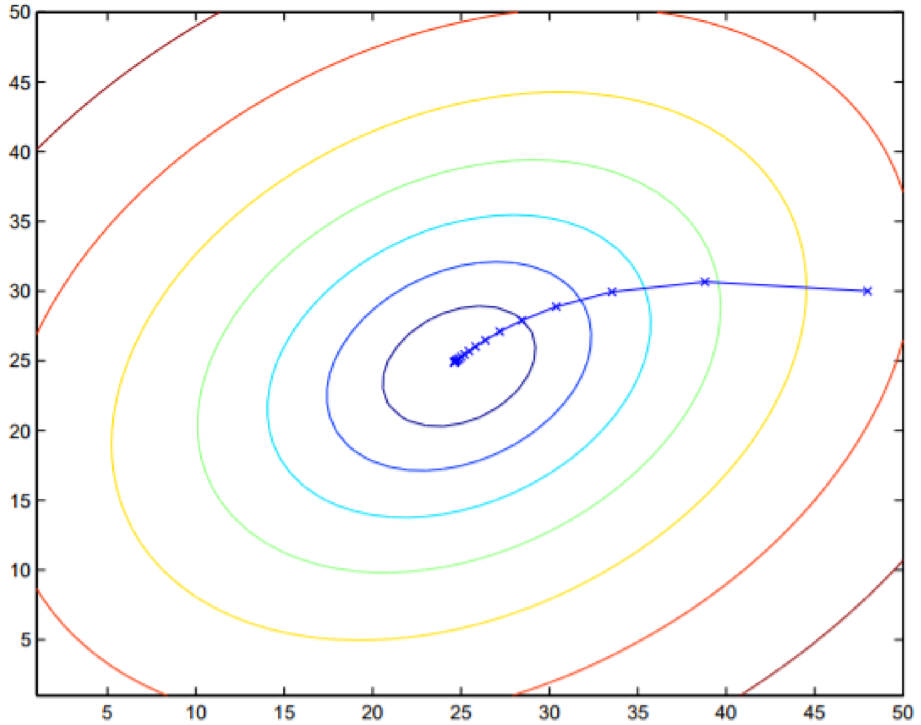
LMS Algorithm

Extend the rule for more than one training sample:

Repeat until convergence {

$$\theta_j := \theta_j + \alpha \sum_{i=1}^{m} \left(y^{(i)} - h_\theta(x^{(i)})\right) x_j^{(i)} \qquad \text{(for every } j\text{)}$$

}



The algorithm moves from outward to inward to reach the minimum error point of the loss function bowl.

- This method considers every training sample on every step and is called batch gradient descent.
- J is a convex quadratic function whose contours are shown in the figure.
- Gradient descent will converge to global minimum, of which there is only one in this case.

Image Credit : "Hands-on Machine Learning with Scikit-Learn and TensorFlow " by Aurelien Geron

Learning the Parameters

LMS ALGORITHM: STOCHASTIC GRADIENT DESCENT

In this type of gradient descent (also called incremental gradient descent), one updates the parameters after each training sample is processed.

Normal Equation

LMS Algorithm

```
Loop {
    for i=1 to m, {
        $\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)})\right) x_j^{(i)}$        (for every $j$)
    }
}
```

- Unlike the batch gradient descent, the progress is made right away after each training sample is processed and applies to large data.
- Stochastic gradient descent offers the faster process to reach the minimum; It may or may not converge to the global minimum, but is mostly close.

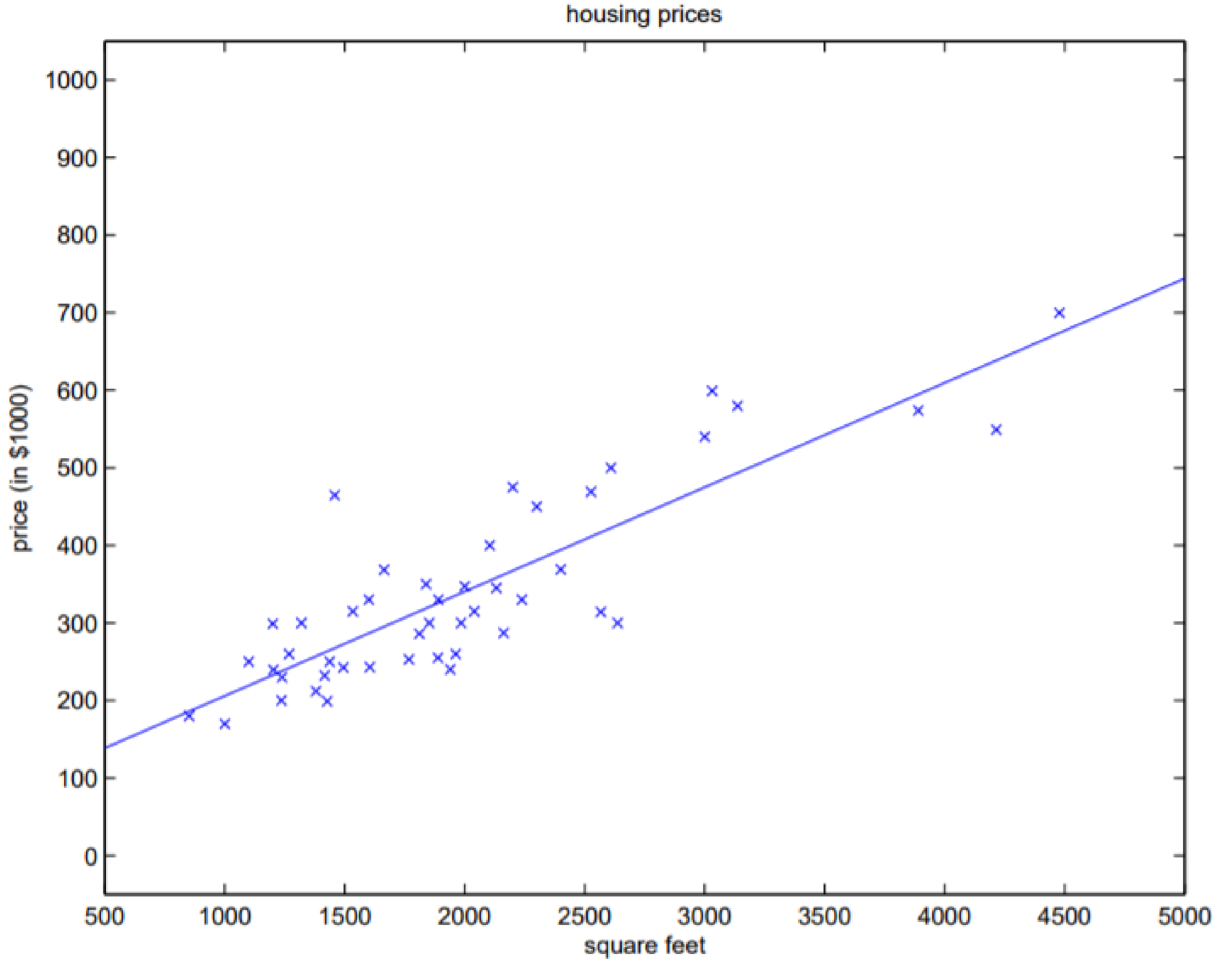


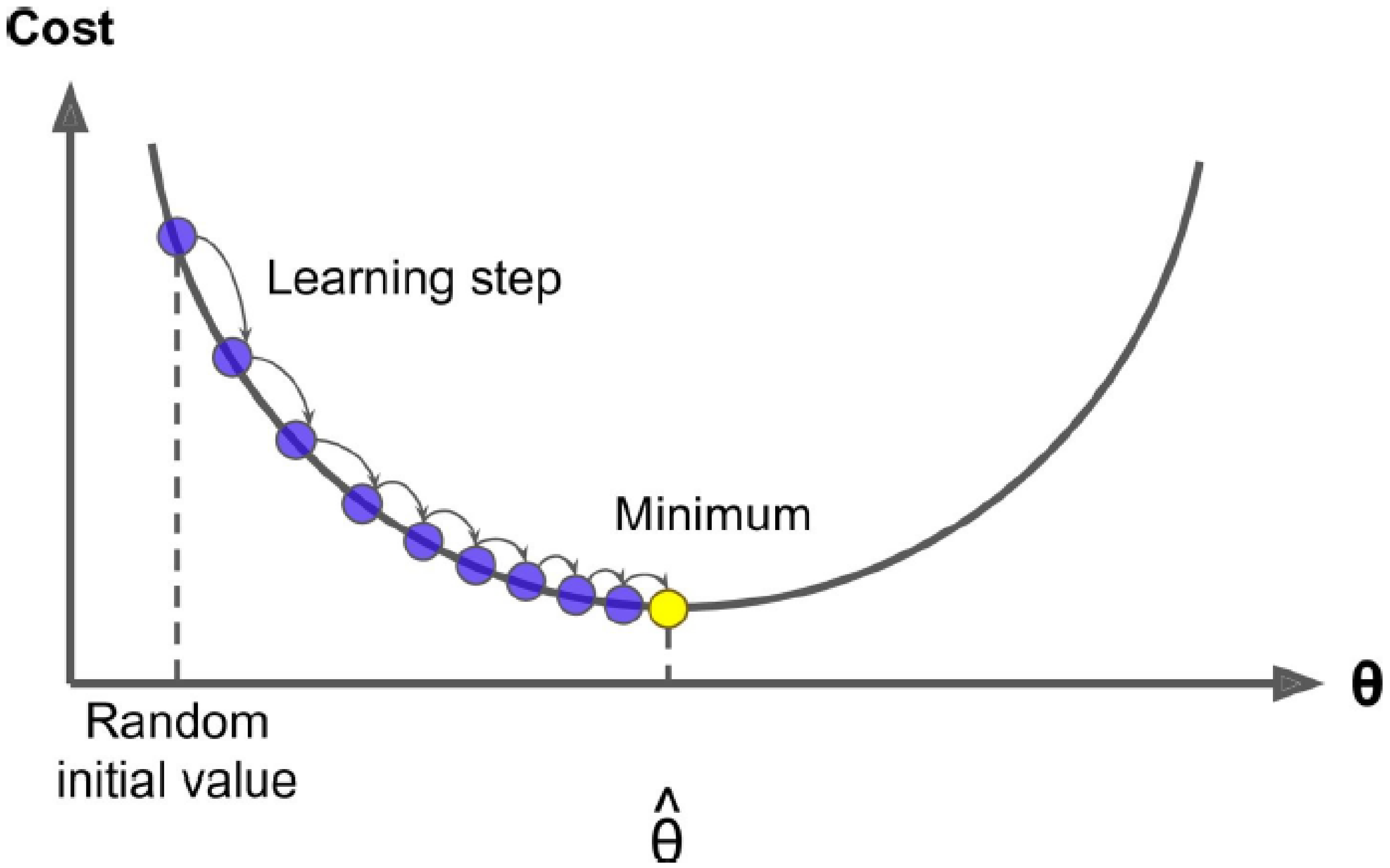
Image Credit : "Hands-on Machine Learning with Scikit-Learn and TensorFlow " by Aurelien Geron

Learning the Parameters

LMS ALGORITHM: GRADIENT DESCENT LEARNING

Normal Equation

LMS Algorithm



The graph shows how the weight adjustment with each learning step brings down the cost or the loss function until it converges to a minimum cost.

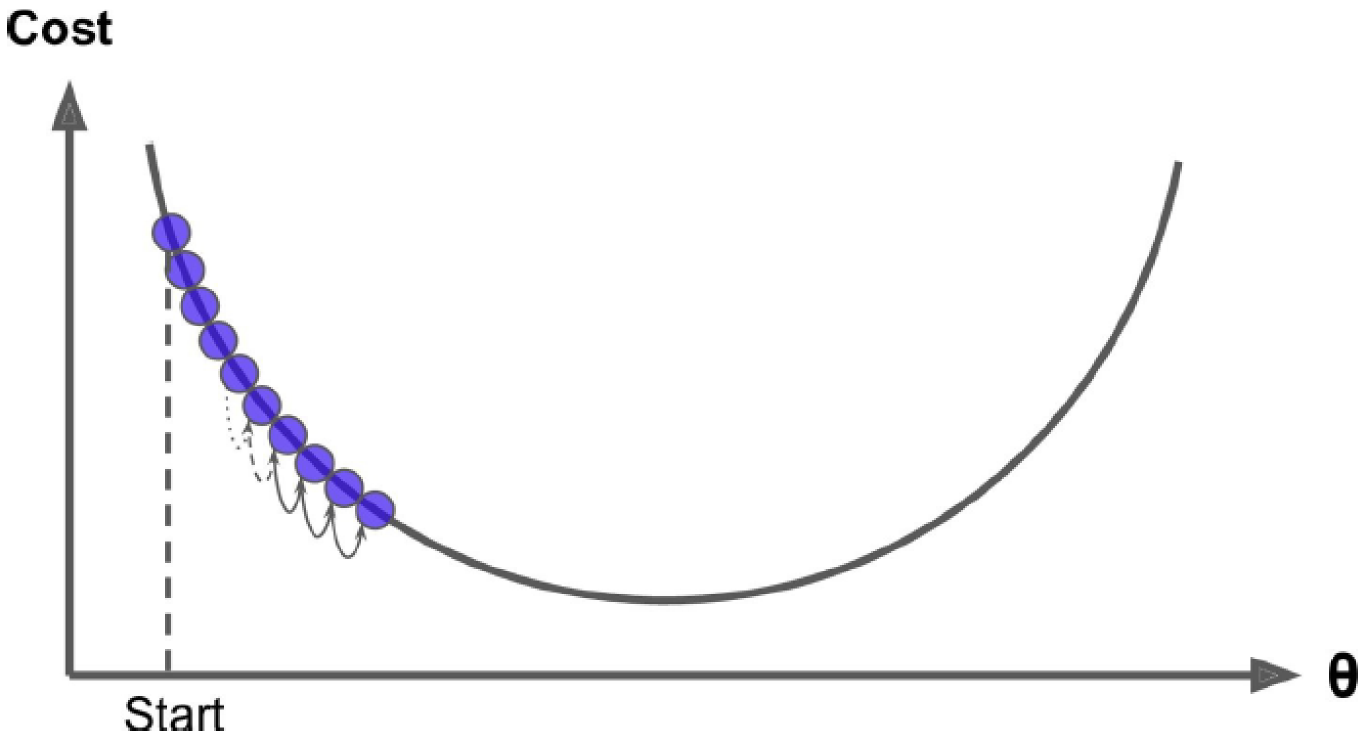
Image Credit : "Hands-on Machine Learning with Scikit-Learn and TensorFlow " by Aurelien Geron

Learning the Parameters

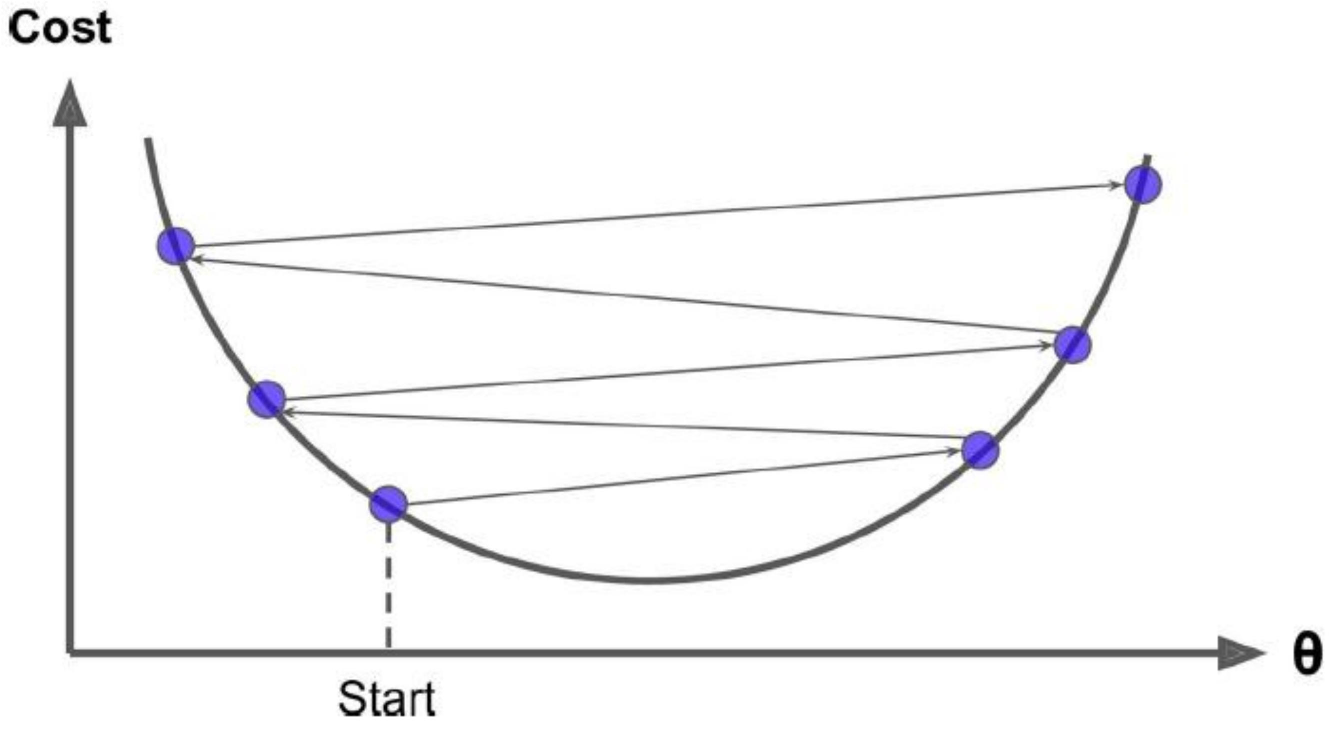
LMS ALGORITHM: GRADIENT DESCENT LEARNING RATE

Normal Equation

LMS Algorithm



Slow learning rate: Converges to minimum but very slowly



Fast learning rate: May not converge to minimum and error might keep increasing with further epochs

An epoch refers to one pass of the model training loop.

Image Credit : "Hands-on Machine Learning with Scikit-Learn and TensorFlow " by Aurelien Geron

Learning the Parameters

LMS ALGORITHM: GRADIENT DESCENT LEARNING RATE

Normal Equation

LMS Algorithm

- Not all cost functions are good bowls. There may be holes, ridges, plateaus and other kinds of irregular terrain.
- In the figure, if random initialization of weights starts on the left, it will stop at a local minimum. If it starts on the right, it will be in a plateau, which will take a long time to converge to the global minimum.
- Fortunately, the MSE cost function for Linear Regression happens to be a convex function with a bowl with global minimum.

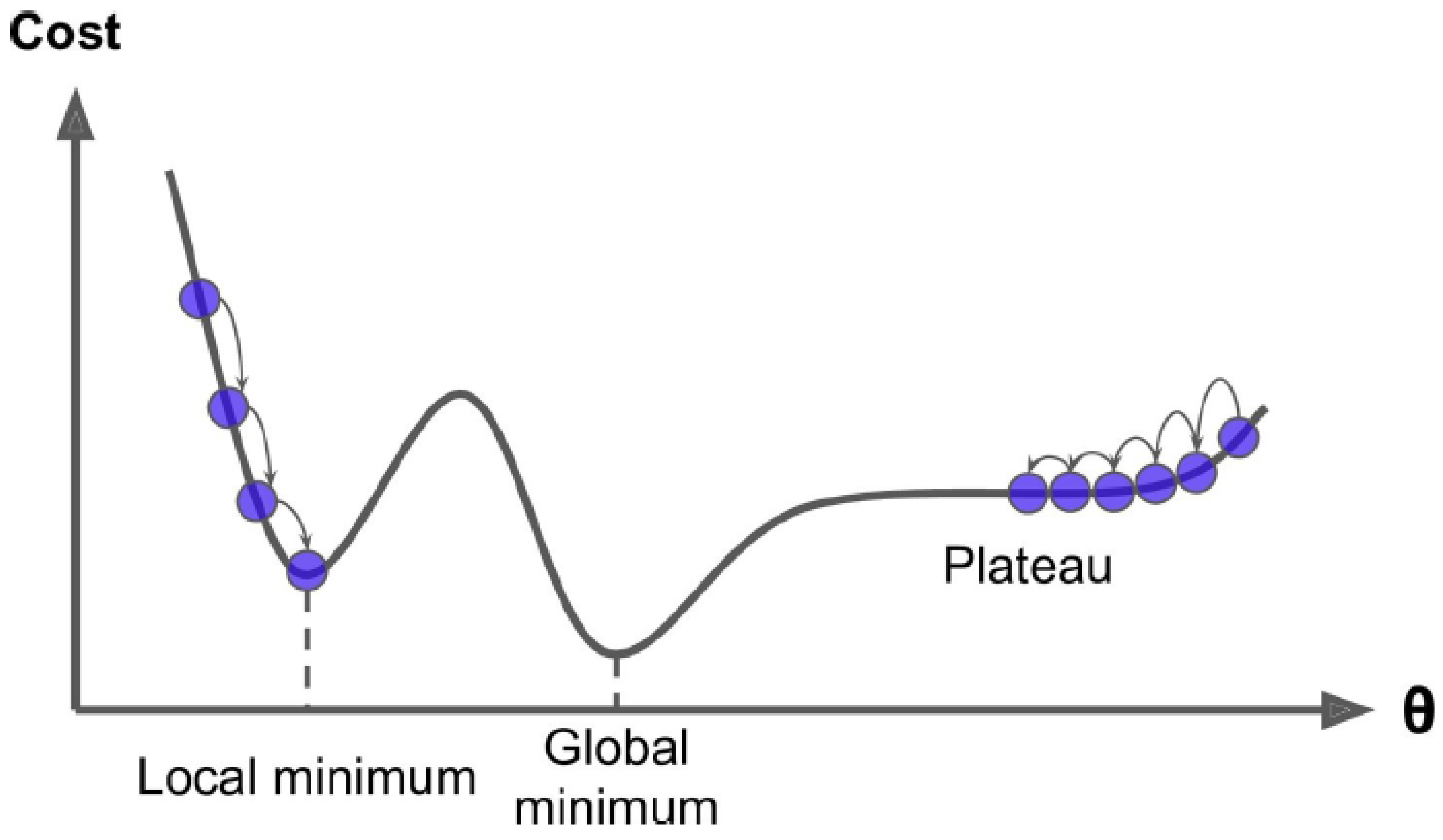


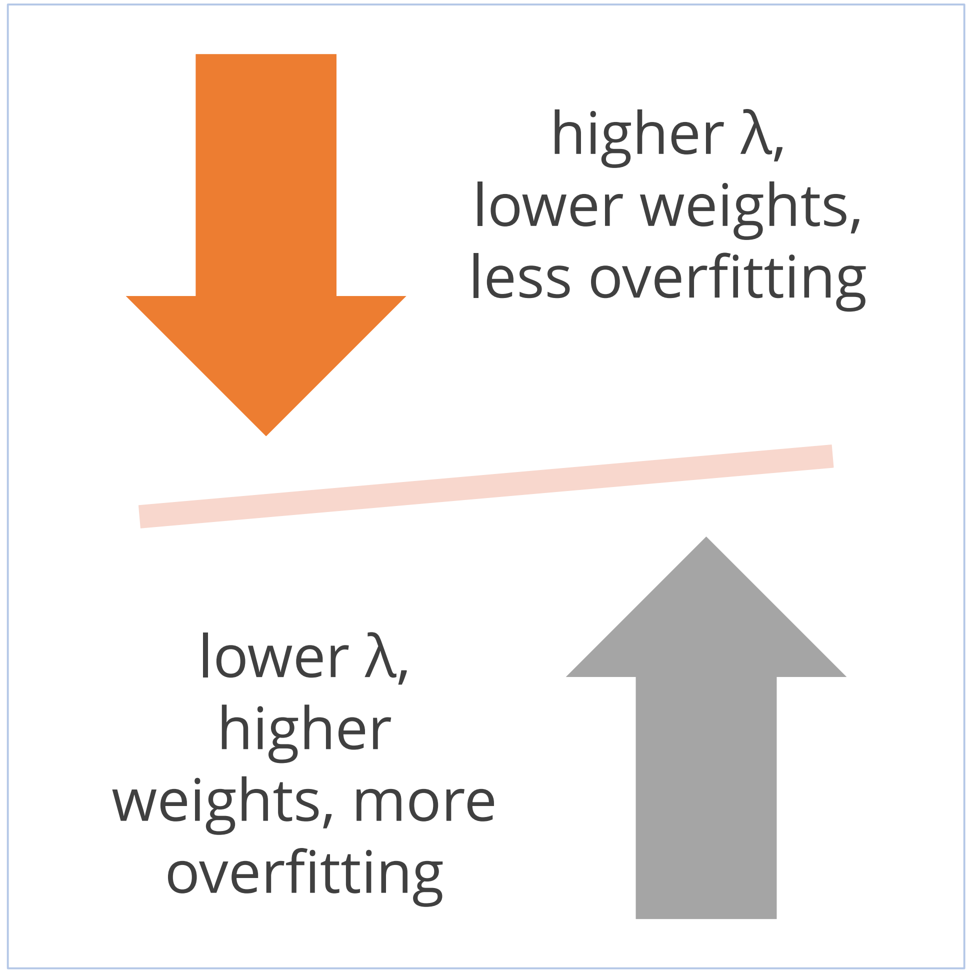
Image Credit : "Hands-on Machine Learning with Scikit-Learn and TensorFlow " by Aurelien Geron

Regularization

- In addition to varying the set of functions or the set of features possible for training an algorithm to achieve optimal capacity, one can resort to other ways to achieve regularization.
- One such method is weight decay, which is added to the Cost function.

$$J(\boldsymbol{w}) = \mathrm{MSE}_{\mathrm{train}} + \lambda \boldsymbol{w}^{\top}\boldsymbol{w},$$

- This approach not only minimizes the MSE (or mean-squared error), it also expresses the preference for the weights to have smaller squared L^2 norm (that is, smaller weights).
- λ is a pre-set value. It influences the size of the weights allowed.
- This works well as smaller weights tend to cause less overfitting (of course, too small weights may cause underfitting).



Regularizing a Model

- To regularize a model, a penalty (to the Cost function) called a Regularizer can be added: Ω(w)
- Hence the Cost function becomes:

  $J(w) = MSE_{train} + \lambda * \Omega(w)$

- In case of weight decay, this penalty is represented by $\Omega(\boldsymbol{w}) = \boldsymbol{w}^\top \boldsymbol{w}$
- In essence, in the weight decay example, you expressed preference for linear functions with smaller weights, and this was done by adding an extra term to minimize in the Cost function. Many other Regularizers are also possible.
- To summarize, the model capacity can be controlled by including/excluding members (that is, functions) from the hypothesis space and also by expressing preferences for one function over the other.

Demo

Linear and Polynomial Regression

Perform various regression techniques on a random dataset. You can perform this in your lab environment using the dataset in the LMS.

Key Takeaways

- ✓ Regression is a Machine Learning technique to predict “how much” of something given a set of variables.
- ✓ The major types of regression are linear regression, polynomial regression, decision tree regression, and random forest regression.
- ✓ Regression uses labelled training data to learn the relation y = f(x) between input X and output Y. It works on linear or non-linear data.
- ✓ Gradient Descent is the most common technique used to train a regression model. It attempts to minimize the loss function to find ideal regression weights.
- ✓ Decision Trees are used for both classification and regression. For regression, Decision Trees calculate the mean value for each leaf node, and this is used as the prediction value during regression tasks.

? Quiz

QUIZ

1

In the equation of a straight line Y = mX + c, the term m is the:

a. Slope

b. Independent variable

c. Dependent variable

d. Intercept

QUIZ 1

In the equation of a straight line Y = mX + c, the term m is the:

a. Slope

b. Independent variable

c. Dependent variable

d. Intercept

The correct answer is **a. Slope**

In the equation of a straight line Y = mX + c, m represents the slope, and c is any constant.

QUIZ

2

The standard error of the estimate is a measure of:

a. explained variation

b. the variation around the regression line

c. the variation of the X variable

d. total variation of the Y variable

QUIZ 2

The standard error of the estimate is a measure of:

a. explained variation

b. the variation around the regression line

c. the variation of the X variable

d. total variation of the Y variable

The correct answer is **b. The variation around the regression line**

The standard error of the estimate is a measure of the variation around the regression line.

Hands-on Assignments

Demo File	Assignment	What it demonstrates?
LinearandPolynomialRegression.py	Modify the degree of polynomial from (PolynomialFeatures(degree = 1)) to 1, 2, 3 and interpret the resulting regression plot. Specify if it is underfitted, right-fitted, or overfitted.	Demonstrate how to reduce data dimensions from 3D to 2D.
	Predict the insurance claim for age 70 with polynomial regression with degree 2 and linear regression.	Modify the code and check the output.
DecisionTreesRegression.py	Modify the code to predict insurance claim values for anyone above the age of 55 in the given dataset.	Predict insurance premium per year based on person's age using Decision Trees.
	Modify the max_depth from 2 to 3, or 4 and observe the output.	Generate random quadratic data and demonstrate Decision Tree regression.
	Modify the max_depth to 20 and observe the output.	

Hands-on Assignments

Demo File	Assignment	What it demonstrates?
DTRegression.py	What is the class prediction for petal_length = 3 cm and petal_width = 1 cm, for the max_depth = 2?	Generate random quadratic data and demonstrate Decision Tree regression.
	Explain the Decision Tree regression graphs produced when max_depth is 2 and 3. How many leaf nodes exist in the two cases? What does average value represent for these two situations?	Modify the code and check the output.
	Explain the Decision Tree regression graphs produced when max_depth is 2 and 3. How many leaf nodes exist in the two cases? What does average value represent for these two situations?	Modify the input variables and check the output.
	Modify the regularization parameter min_sample_leaf from 10 to 6 and check the output of Decision Tree regression. What is the result and why?	Modify the input variables and check the output.
RFRegression.py	What is the output insurance value for individuals with age 60 and n_estimators = 10?	Predict insurance per year based on person's age using Random Forests.
	The program depicts a learning process when the learning rate η is 0.02, 0.1 and 0.5. Give your interpretation of these charts?	Demonstrate various regression techniques over a random dataset.

This concludes “Regression.”

The next lesson is “Classification.”