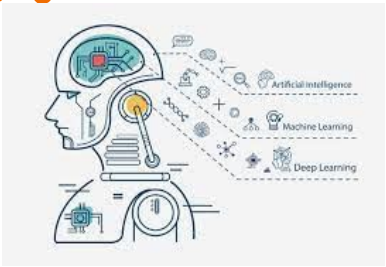
Linear Regression

Implementing Linear Regression Equation using Java

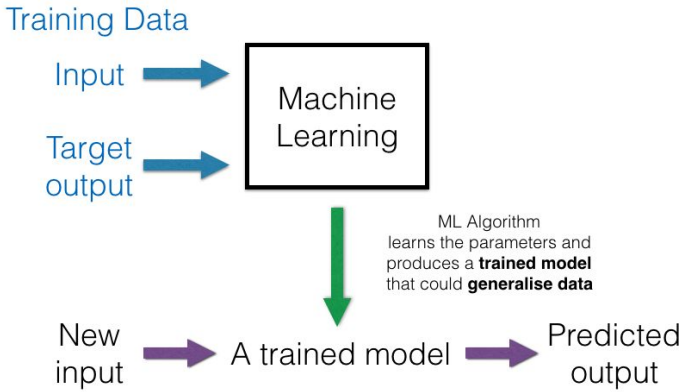


Presented by
Sai Harshinee Roopakula
19577

Table of Contents

Introduction - Machine Learning

- Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed.
- Machine learning focuses on the development of computer programs that can access data and use it to learn for themselves.
- The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide.
- The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly.



Introduction - Machine Learning Types

Machine learning algorithms are often categorized as Supervised Learning, Unsupervised Learning, Semi-supervised Learning and Reinforcement Learning.

1. Supervised machine learning algorithms - Need a lot of labelled data
Supervised learning is where you have input variables (x) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output.

$$Y = f(X)$$

The goal is to approximate the mapping function so well that when you have new input data (x) that you can predict the output variables (Y) for that data.
It is called supervised learning because the process of an algorithm learning from the training dataset can be thought of as a teacher supervising the learning process. We know the correct answers, the algorithm iteratively makes predictions on the training data and is corrected by the teacher. Learning stops when the algorithm achieves an acceptable level of performance.

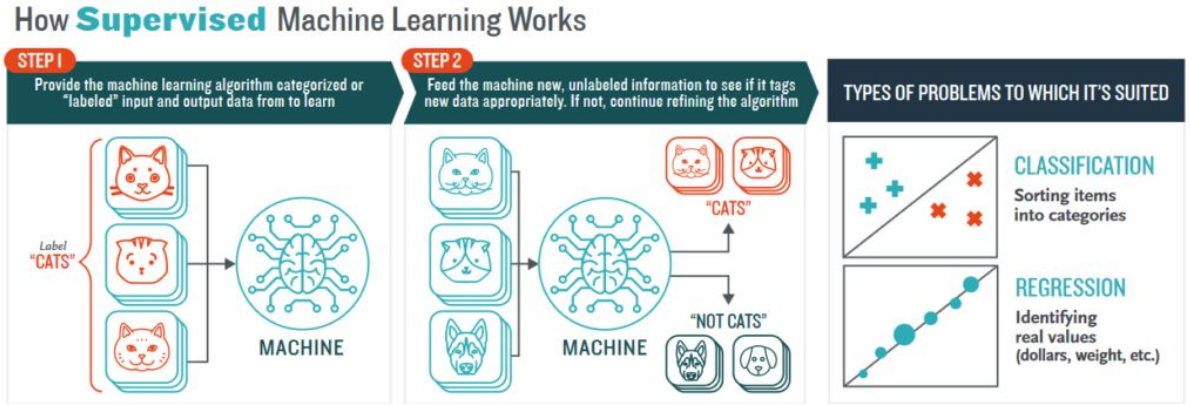
Supervised learning problems can be further grouped into regression and classification problems.

- **Classification**: A classification problem is when the output variable is a category, such as "red" or "blue" or "disease" and "no disease".
- **Regression**: A regression problem is when the output variable is a real value, such as "dollars" or "weight".

Introduction - Machine Learning Types

List of common Supervised Machine Learning Algorithms are:

- Linear Regression
- Logistic Regression
- k-Nearest Neighbors
- Support Vector Machines(SVMs)
- Decision Trees
- Random Forests
- Naive Bayers

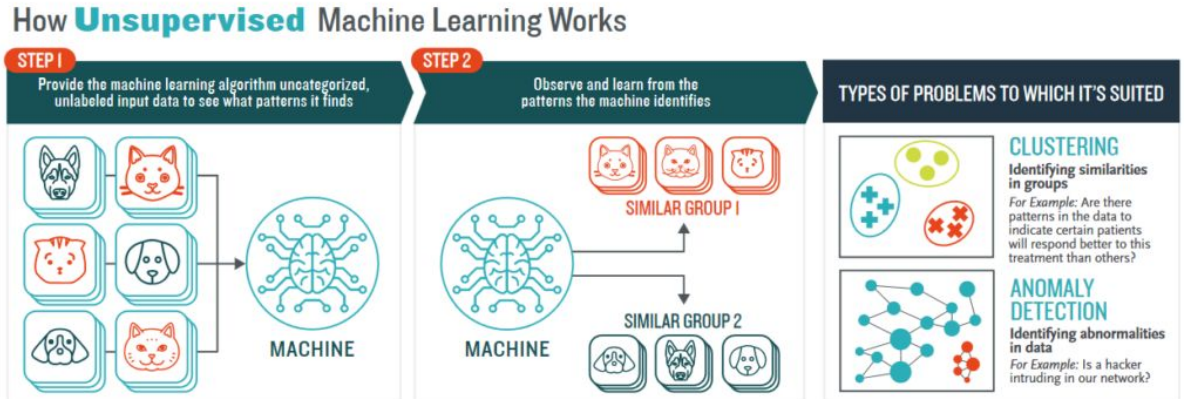


Introduction - Machine Learning Types

2. Unsupervised machine learning algorithms- Need no labelled data.
These are used when the information used to train is neither classified nor labeled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabeled data.
These are called unsupervised learning because unlike supervised learning there is no correct answers and there is no teacher. Algorithms are left to their own to discover and present the interesting structure in the data.

List of common Unsupervised Machine Learning Algorithms are:

- K-means Clustering
- Association Rules



Introduction - Machine Learning Types

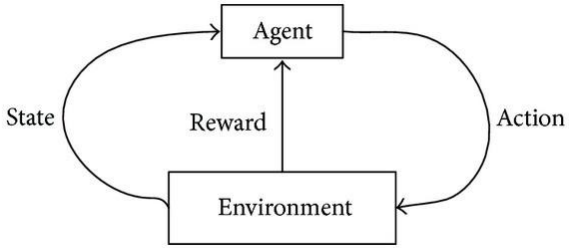
3. Semi-supervised machine learning algorithms
These fall somewhere in between supervised and unsupervised learning, since they use both labeled and unlabeled data for training – typically a small amount of labeled data and a large amount of unlabeled data.
The systems that use this method are able to considerably improve learning accuracy.
A good example is a photo archive where only some of the images are labeled, (e.g. dog, cat, person) and the majority are unlabeled.

4. Reinforcement machine learning algorithms
This method aims at using observations gathered from the interaction with the environment to take actions that would maximize the reward or minimize the risk. Reinforcement learning algorithm (called the agent) continuously learns from the environment in an iterative fashion. In the process, the agent learns from its experiences of the environment until it explores the full range of possible states.

List of common Reinforcement machine Learning Algorithms are:

- Q-Learning
- Temporal Difference (TD)
- Deep Adversarial Networks

Self-driving cars, Robotic hands, Computer played board games (Chess, Go) use Reinforcement machine learning Algorithm.



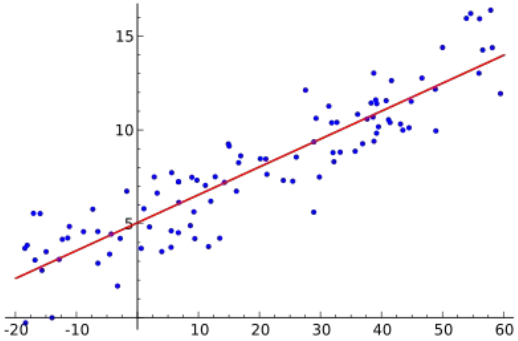
Introduction - Linear Regression

Linear regression is a **linear model**, e.g. a model that assumes a linear relationship between the input variables (x) and the single output variable (y). More specifically, that y can be calculated from a linear combination of the input variables (x).

When there is a single input variable (x), the method is referred to as **simple linear regression**. When there are **multiple input variables**, it is referred to as **multiple linear regression**.

The linear equation assigns one scale factor to each input value or column, called a **coefficient** and represented by the capital Greek letter Beta (B). One additional coefficient is also added, giving the line an additional degree of freedom (e.g. moving up and down on a two-dimensional plot) and is often called the **intercept** or the bias coefficient. For example, in a simple regression problem (a single x and a single y), the form of the model would be:

$$y = B0 + B1*x$$



Introduction - Linear Regression

A regression line is the "best fit" line for the data. You have to draw a line that best represents the data points. It's like an average of where all the points line up. In linear regression, the regression line is a perfectly straight line:

Regression Equation(y) = a + bx
Slope(b) = $(N\Sigma XY - (\Sigma X)(\Sigma Y)) / (N\Sigma X^2 - (\Sigma X)^2)$
Intercept(a) = $(\Sigma Y - b(\Sigma X)) / N$

Regression equations can help you figure out if your data can be fit to an equation. This is extremely useful if you want to make predictions from your data–either future predictions or indications of past behavior.

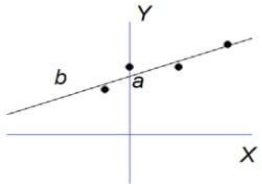
Linear regression equation
(without error)

$$\hat{Y} = bX + a$$

predicted values of *Y*

b = slope = rate of predicted ↑/↓ for Y scores for each unit increase in *X*

Y-intercept = level of *Y* when X is 0



Design-Understanding the project

Implementing Linear Regression Equation using Java.

Step 1: Collect the data.

Read the data from the file “input.text”

Step 2: Create the model.

Finding the values of a and b of a Linear Regression Equation(y) = a + bx" based on the content of "input.txt".

Step 3: Prediction.

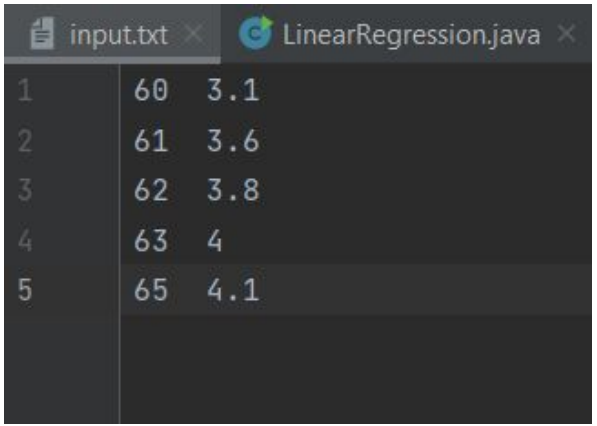
For x = 64, predict the y value.

Design - Input

X values	Y values
60	3.1
61	3.6
62	3.8
63	4
65	4.1

INPUT :The input to the Java program is X and Y values which are stored in the a file - "input.txt"

DESIRED OUTPUT: Predicted value of y for X = 64.



Implementation - Importing necessary libraries

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import java.util.stream.IntStream;
```

Implementation - Code to read the input file

```java
public class LinearRegression {
    private static final String FILE_LOCATION = "src/main/java/input.txt";
    static final List<Float> columnXValues = new ArrayList<>();
    static final List<Float> columnYValues = new ArrayList<>();

    private static void readFile() {
        try {
            BufferedReader bReader = new BufferedReader(new FileReader(LinearRegression.FILE_LOCATION));
            String line = bReader.readLine();
            float x, y;
            System.out.println("X    Y");
            while (line != null) {
                String[] fields = line.split( regex: "\\s+");
                // Get the 1st column
                x = Float.parseFloat(fields[0]);
                // Adding the 1st column values to a list
                columnXValues.add(x);
                // Get the second column
                y = Float.parseFloat(fields[1]);
                // Adding the 2nd column values to a list
                columnYValues.add(y);
                System.out.println(x + " " + y);
                System.out.println();
                // Read next line
                line = bReader.readLine();
            }
        } catch (IOException e) {
            System.out.println("Error reading the file, message: ");
            e.printStackTrace();
        }
    }
```

Here, we are reading the contents of the input.txt file and adding each column value to a list (ColumnXValues, ColumnYValues). So that we can further perform operations on them.

Implementation - Other Util methods necessary

Function to calculate the sum of elements of a List

```
final class Util {
    private Util() {
    }

    static float sum(List<Float> list) {
        float sum = 0;
        for (float i : list) {
            sum += i;
        }
        return sum;
    }
```

This method calculates the sum of elements of a List.

This method is used to calculate:

ΣXY = Sum of the product of first and Second Scores(sumXY)
ΣX = Sum of First Scores (sumX)
ΣY = Sum of Second Scores (sumY)
ΣX^2 = Sum of square First Scores (sumXX)

Implementation - Other Util methods necessary

Function to calculate the product of elements of 2 Lists

```
static List<Float> mul(List<Float> list, List<Float> list1) {
    float[] result = new float[list.size()];
    List<Float> mulResult = new ArrayList<>();

    IntStream.range(0, list.size()).forEach(i -> {
        result[i] = list.get(i) * list1.get(i);
        mulResult.add(result[i]);
    });

    return mulResult;
}
```

This method calculates the product of elements of 2 Lists.

This method is used to calculate:

XY = The product of first and Second Scores(multiplyXY)
X^2 = Square of First Scores (multiplyXX)

Implementation - Other Util methods necessary

Function to calculate b (the slope of the regression line)

```
static double getB(float sumX, float sumY, float sumXY, float sumXX) {
    return Math.round((LinearRegression.columnXValues.size() * sumXY - sumX * sumY) /
            (LinearRegression.columnXValues.size() * sumXX - Math.pow(sumX, 2.0)) * 100.00) / 100.00;
}
```

Function to calculate a (The intercept point of the regression line and the y axis)

```
static double getA(float sumX, float sumY, double b) {
    return Math.round((sumY - b * sumX) / LinearRegression.columnXValues.size() * 1000.00) / 1000.00;
}
```

Implementation - Other Util methods necessary

Function to calculate y predicted.

```
    static double getYpred(double b, double a) {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter the X value for which you want to predict Y");
        float x=sc.nextFloat();
        return Math.round((a + b * x) * 100.00) / 100.00;
    }
}
```

This method is the Output of the program.
It calculates the y predicted value for a given value of X.

Implementation - Code to calculate Linear Regression Equation

```
private static void calculateLinearRegression() {
    System.out.println("The values of X(First Score) are:" + LinearRegression.columnXValues);
    System.out.println("The values of Y(Second Score) are:" + LinearRegression.columnYValues);

    float sumX = Util.sum(LinearRegression.columnXValues);
    System.out.println();
    System.out.println("Sum of first scores:" + sumX);

    float sumY = Util.sum(LinearRegression.columnYValues);
    System.out.println("Sum of second scores:" + sumY);
```

The above code, prints the list of columnXValues and columnYValues.

It also prints the values of ΣX and ΣY that is:
sum of columnXValues(sumX) and
sum of columnYValues(sumY)

Implementation - Code to calculate Linear Regression Equation

```
List<Float> multiplyXY = Util.mul(LinearRegression.columnXValues, LinearRegression.columnYValues);
System.out.println();
System.out.println("Product of first and Second Scores: " + multiplyXY);

float sumXY = Util.sum(multiplyXY);
System.out.println("Sum of the product of first and Second Scores: " + sumXY);
System.out.println();

List<Float> multiplyXX = Util.mul(LinearRegression.columnXValues, LinearRegression.columnXValues);
System.out.println("Square of First Scores: " + multiplyXX);

float sumXX = Util.sum(multiplyXX);
System.out.println("Sum of squares of First Scores: " + sumXX);
System.out.println();
```

The above code, multiples the values of columnXValues and columnYValues and prints it. (multiplyXY).

It also multiples the values of list of columnXValues with itself. (multiplyXX).

It also prints the values of ΣXY and ΣX*X that is:
sum of multiplyXY(sumXY) and
sum of multiplyXX(sumXX)

Implementation - Code to calculate Linear Regression Equation

```
    double b = Util.getB(sumX, sumY, sumXY, sumXX);
    System.out.println("Value of Slope(b) is: " + b);

    double a = Util.getA(sumX, sumY, b);
    System.out.println("Value of Intercept(a) is: " + a);
    System.out.println();
    System.out.println("Regression Equation(y) = a + bx ");
    System.out.println("y=" + a + "+" + b + "x");
    System.out.println("Suppose if we want to know the approximate y value for the variable x = 64. " +
            "Then we can substitute the value in the above equation.");

    double ypred = Util.getYpred(b, a);
    System.out.println("y predicted for value x=64 is " + ypred);
}
```

The above code, calculates the values of Slope(b), Intercept(a).

It also predicts the value of y for a given value of X.

Implementation - Code to calculate Linear Regression Equation

```
public static void main(String[] args) {
    readFile();
    calculateLinearRegression();
}
```

The above code is the main function of the program

Result - Output

```
X    Y
60.0 3.1

61.0 3.6

62.0 3.8

63.0 4.0

65.0 4.1

The values of X(First Score) are:[60.0, 61.0, 62.0, 63.0, 65.0]
The values of Y(Second Score) are:[3.1, 3.6, 3.8, 4.0, 4.1]

Sum of first scores(ΣX):311.0
Sum of second scores(ΣY):18.6

Product of first and Second Scores (XY): [186.0, 219.59999, 235.59999, 252.0, 266.5]
Sum of the product of first and Second Scores (ΣXY): 1159.7

Square of First Scores(X*X): [3600.0, 3721.0, 3844.0, 3969.0, 4225.0]
Sum of squares of First Scores(ΣX*X): 19359.0

Value of Slope(b) is: 0.19
Value of Intercept(a) is: -8.098

Regression Equation(y) = a + bx
y=-8.098+0.19x
Suppose if we want to know the approximate y value for the variable x = 64. Then we can substitute the value in the above equation.
Enter the X value for which you want to predict Y
64
y predicted for value x=64 is 4.06
```

Conclusion

We have implemented Linear Regression equation using Java.

Implementing Linear Regression Equation with Java is easier than manually implementing it, as Manual calculation are cumbersome in nature and has high scope of errors.

Bibliography

https://www.expert.ai/blog/machine-learning-definition/

https://www.hindawi.com/journals/ahci/2019/9610567/

https://npu85.npu.edu/~henry/npu/classes/data_science/algorithm/slide/linear_regression_example.html

https://npu85.npu.edu/~henry/npu/classes/introjava/file/slide/multi-colum_read.html

Link to view the presentation

https://docs.google.com/presentation/d/1Co64mZRfnMap-6ViKiLPtbFtMt6kSQgD_xX5TbOqVpQ/edit?usp=sharing