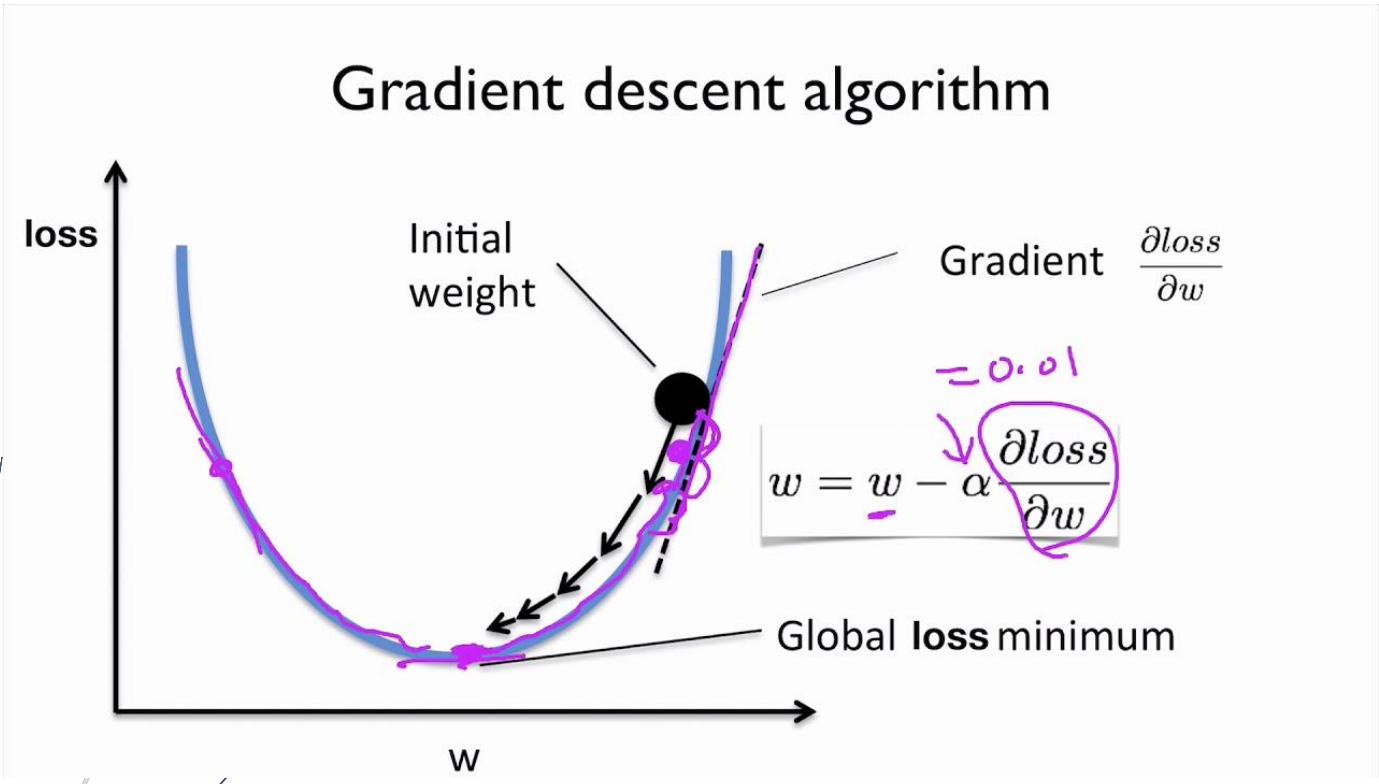
8/25/2019

Gradient Descent Algorithm Implementation

Linear Regression Model



INSTITUTE OF INFORMATION TECHNOLOGY, UNIVERSITY OF DHAKA

BACHELOR OF SCIENCE IN SOFTWARE ENGINEERING

Course Code: SE-837

TYPESCRIPT PROGRAM OF GRADIEN DESCENT

SUBMITTED TO

Dr. B M Mainul Hossain

Associate Professor

Institute of Information Technology

University of Dhaka.

SUBMITTED BY

Tulshi Chandra Das

BSSE0811

TABLE OF CONTENTS

CHAPTER-01: SOURCE CODE

Programming Language: Typescript, Filename: gradientDescent.ts

gradientDescent.ts(1-36)

```typescript
//h(x) = theta0 + theta1*x;

interface point {
    x: number,
    y: number
}

class GradientDescent {
    private sampleSize = 30;
    private learningRate = 0.01;
    private maxIteration = 1000;
    private trainingData: point[] = [];
    private isConverged = false;
    private theta0 = 0;
    private theta1 = 0;
    private threshold = 0.0001;

    constructor(sampleSize = 30, learningRate = 0.01) {
        this.sampleSize = sampleSize;
        this.learningRate = learningRate;
        this.init();
    }

    private init() {
        this.prepareTrainingData();
    }

    public start() {
        let iteration = 1;
        while (!this.isConverged && iteration < this.maxIteration) {
            console.log('\niteration:' + iteration);
            this.iterate();
            iteration++;
        }
    }
```

gradientDescent.ts(37-69)

```
  private iterate() {
      var mse_before = this.J();

      let temp0 = this.theta0 - (this.learningRate *
          this.derivativeTheta0());
      let temp1 = this.theta1 - (this.learningRate *
          this.derivativeTheta1());
      this.theta0 = temp0;
      this.theta1 = temp1;
      console.log("theta0:" + this.theta0);
      console.log("theta1:" + this.theta1);

      this.isConverged = (mse_before - this.J() < this.threshold);
  }

  private J() {
      var sum = 0;
      this.trainingData.forEach((point) => {
          sum += this.squaredError(point);
      });
      return sum / (2 * this.trainingData.length);
  }

  private squaredError(point: point) {
      return Math.pow(this.predictionError(point), 2);
  }

  private predictionError(point: point) {
      return this.h(point.x) - point.y;
  }
  //hypothesis
  private h(x: number) {
      return this.theta1 * x + this.theta0;
  }
```

gradientDescent.ts(70-109)

```
    private derivativeTheta0() {
        var sum = 0;
        this.trainingData.forEach((point) => {
            sum += this.predictionError(point);
        });
        return sum / this.trainingData.length;
    }

    private derivativeTheta1() {
        var sum = 0;
        this.trainingData.forEach((point) => {
            sum += this.predictionError(point) * point.x;
        });
        return sum / this.trainingData.length;
    }

    private prepareTrainingData() {
        for (var i = 0; i < this.sampleSize; i++) {
            var point = this.getRandomPoint(10);
            this.trainingData.push(point);
        }
    }

    private getRandomPoint(max: number): point {
        let slope = 0.5;
        let intercept = 2.5;
        let stddev = 0.9;
        let x = Math.round(Math.random() * max);
        let y = slope * x + intercept + Math.random() * stddev;
        console.log('data points');
        console.log(x + " " + y);
        return {
            x: x,
            y: y
        };
    }
}

 new GradientDescent().start();
```

.

CHAPTER -02: Build And Run

2.1 PREREQUISITE

1. Node Js
2. Typescript

The two above must be installed to build and run the program

2.2 BUILD AND RUN

First open windows CMD or Linux terminal in the directory of gradientDescent.ts

Run the following commands:

```
tsc gradientDescent.ts
```

this will compile and create gradientDescent.js file in same directory

Now run this

```
node gradientDescent.js
```

2.3 HOW PROGRAM WORKS

Hypothesis: h(x) = theta0 + theta1*x;

The steps are below:

1. Prepare training data set
2.
    a. theta0 = theta0 – learningRate*derivativeTheta0(cost function);
    b. theta1 = theta1 – learningRate*derivativeTheta1(cost function);
3. Repeat the step 2 till converged.

Source code url:https://github.com/TulshiDas39/gradientDescent/blob/master/gradient-descent.ts