Agricultural Yield Estimation by Polynomial Regression

Amogh M K (01FB15EEC027)

Guide: Prof. V.K. Agrawal,
Department of Information Science and Engineering,
Crucible of Research and Innovation (CORI),
PES University, Bangalore

PES University
Bangalore-560085

Acknowledgements

On the submission of my project entitled **Agricultural Yield Estimation by Polynomial Regression**, I would like to express my gratitude to the project supervisor **Prof. V.K. Agrawal**, Director Of CORI, PES University for his guidance and inspiration during the course of the project work.

Also, I would like to thank my family and friends for their support and encouragement.

Abstract

The main focus in this project was to gain experience in taking up and solving a regression problem.

This project aims to approximately predict the yield of a crop in a state through polynomial regression using Gradient descent algorithm and normal equation method in order to help the farmers, industries and other associated personnel for future planning.

Table of Contents

1 Introduction

With the impact of climate change in India, majority of the agricultural crops are being badly affected in terms of their performance over a period of last two decades. Predicting the crop yield well ahead of its harvest would help the policy makers and farmers for taking appropriate measures for marketing and storage. Such predictions will also help the associated industries for planning the logistics of their business.

Crop production is a complex phenomenon that is influenced by agro-climatic input parameters. Agriculture input parameters varies from field to field and farmer to farmer. Collecting such information on a larger area is a daunting task. However, the climatic information collected in India at every 1sq.m area in different parts of the are tabulated by Indian Meteorological Department. Also, the yield of every crop in each state is collected and published by the department of agriculture and cooperation every year.

Such data sets are used in this project for predicting the influence on major crops and thus, their yield in a future year.

2 Concepts

2.1 Machine Learning

Machine learning is an emerging technology that can aid in the discovery of rules and patterns in sets of data. It has frequently been observed that the volume of recorded data is growing at an astonishing rate that far outstrips our ability to make sense of it.

Machine Learning has been defined in many ways:
Field of study that gives computers the ability to learn without being explicitly programmed.
- Arthur Samuels
A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance of T, as measured by P, improves with experience E.
- Tom Mitchell

Machine learning algorithms are typically classified into two broad categories, depending on the nature of the learning data available to a learning system. These are:
Supervised Learning: The computer is presented with example inputs and their desired outputs and the goal is to learn a general rule that maps inputs to outputs.
Unsupervised Learning: A computer program interacts with a dynamic environment in which it must perform a certain goal. The program is provided feedback in terms of rewards and punishments as it navigates its problem space.

Another categorization of machine learning tasks arises when one considers the desired output of a machine-learned system:
Classification: Inputs are divided into two or more classes, and the learner must produce a model that assigns unseen inputs to one or more of these classes. This is typically tackled in a supervised way.
Regression: Similar to classification but the outputs are continuous rather than discrete. This is also a supervised problem.

2.2 Polynomial Regression

Regression is a statistical method that allows us to summarize and study relationships between two continuous (quantitative) variables.

If the plot of the training data suggests that there is a linear relationship between the 2 variables, a linear regression model is constructed to fit a line to the set of points.

Sometimes, a plot of the residuals versus a predictor may suggest there is a nonlinear relationship. One way to try to account for such a relationship is through a polynomial regression model.

Such a model for a single predictor, X, is:

$$Y = \theta_0 + \theta_1 X + \theta_2 X^2 + ... + \theta_n X^n \tag{1}$$

n - the degree of the polynomial.
θ - weights or parameters

For lower degrees, the relationship has a specific name (i.e., h = 2 is called quadratic, h = 3 is called cubic, h = 4 is called quartic, and so on).

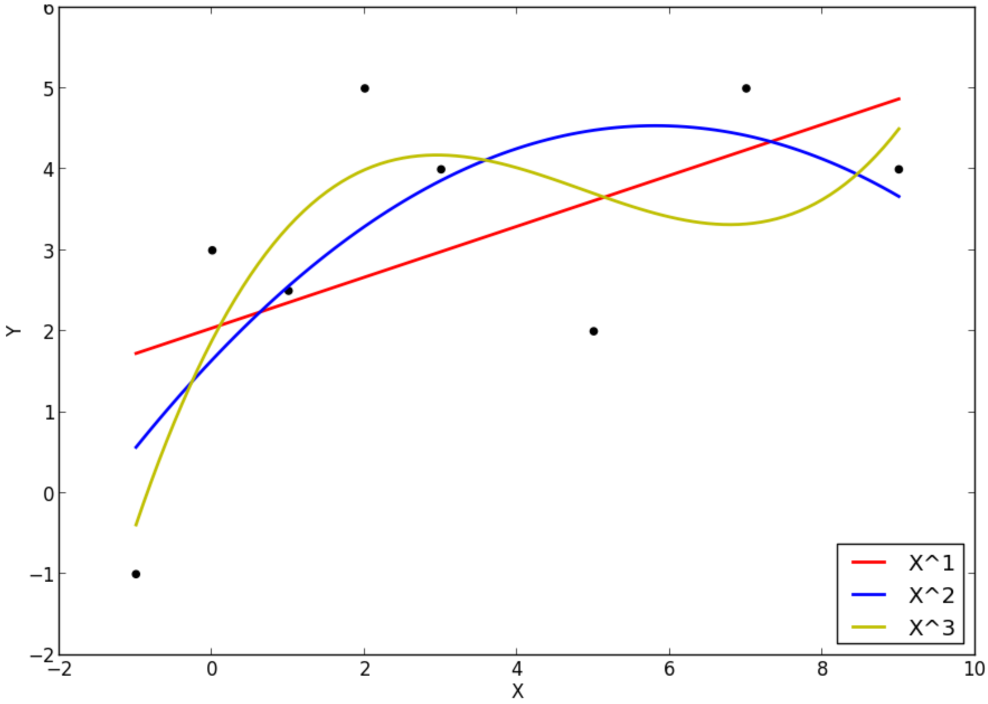

Fig. 1. Linear regression and 2nd, 3rd degree polynomial regression over the same data set

2.3 Cost function

Cost function is a function that maps an event or values of one or more variables onto a real number intuitively representing some "cost" associated with the event. An optimization problem seeks to minimize a cost function. Typically a cost function is used for parameter estimation, and the event in question is some function of the difference between estimated and true values for an instance of data.

Parameter estimation for supervised learning tasks such as regression or classification can be formulated as the minimization of a cost function over a training set in order to find a function that models its input well. The cost function quantifies the amount by which the prediction deviates from the actual values.One such cost function is “Squared error function” which is given below:

$$J(\theta) = (1/2m) \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2 \tag{2}$$

Where, m is the no. of training examples.

The mean is halved as a convenience for the computation of the gradient descent, as the derivative term of the square function will cancel out the 1/2 term.

On minimizing this function, we obtain theta values which in turn gives us the function that models input well.

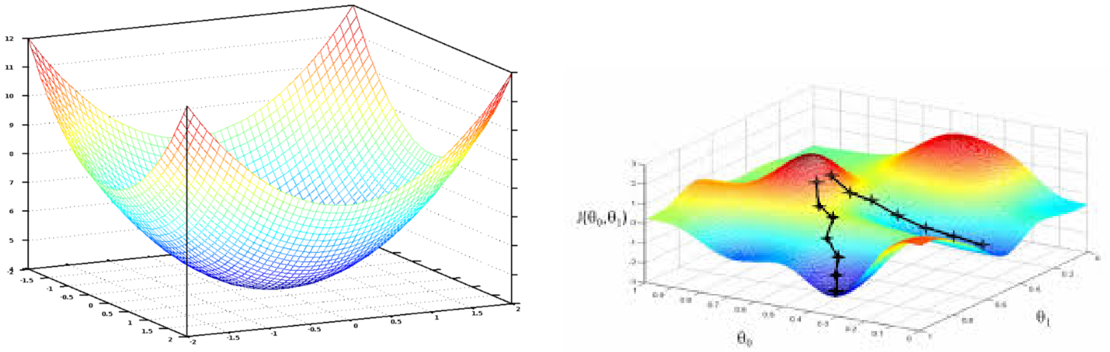

Fig. 2.
Different Cost Functions:
a) Bowl shaped cost function - only one minima.
b) Cost function with one global minima and multiple local minimas.

2.4 Gradient Descent

Gradient descent is one of those “greatest hits” algorithms that can offer a new perspective for solving problems.

Gradient descent is an algorithm that minimizes functions. Given a function defined by a set of parameters, gradient descent starts with an initial set of parameter values and iteratively moves toward a set of parameter values that minimize the function. This iterative minimization is achieved using calculus, taking steps in the negative direction of the function gradient.

Gradient descent is the algorithm used here to effectively minimize the cost function.

The gradient descent algorithm is defined as:

Repeat until convergence:

$$\theta_j = \theta_j - \alpha \frac{\partial(J(\theta))}{\partial\theta} \tag{3}$$

At each iteration j, one should simultaneously update the parameters θ_1,θ_2,...,θ_n. Updating a specific parameter prior to calculating another one on the j(th) iteration would yield to a wrong implementation.

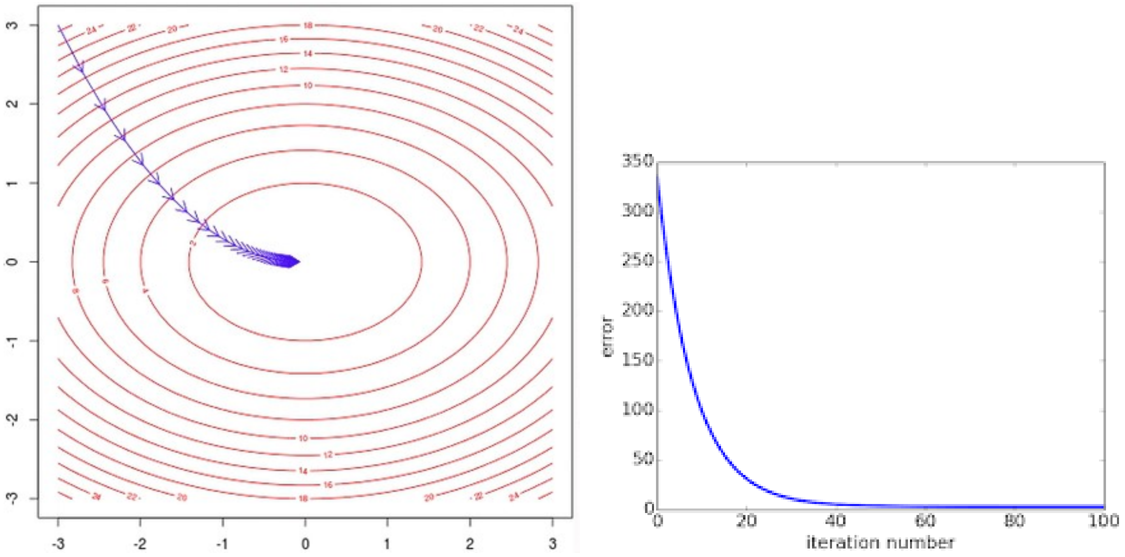

Fig. 3.
a) Working of Gradient Descent to obtain θ values corresponding to minima of cost function.
b)Decrease in cost value after each iteration of Gradient Descent Algorithm.

2.5 Normal Equation

Gradient descent gives one way of minimizing cost function. Normal Equation performs the minimization explicitly and without resorting to an iterative algorithm.

In this method, cost function is minimized by explicitly taking its derivatives with respect to the θ_j's, and setting them to zero. This allows us to find the optimum theta without iteration.

The normal equation formula is given below:

$$\theta = (X^T X)^{-1} X^T Y \tag{4}$$

There is no need to do feature scaling with the normal equation.
The following is a comparison of gradient descent and the normal equation:

Gradient Descent	**Normal Equation**
Need to choose alpha.	No need to choose alpha.
Needs many iterations.	No need to iterate.
Time complexity is $O(kn^2)$.	Time complexity is $O(n^3)$.
Works well when n is large.	Slow if n is very large.

With the normal equation, computing the inversion has complexity O(n3). So if we have a very large number of features, the normal equation will be slow. In practice, when n exceeds 10,000 it might be a good time to go from a normal solution to an iterative process.

3 Problem Statement

4 Implementation and Results

5 Conclusion

Bibliography