REAL-ESTATE-PRICE-PREDICTION

Progress report

OBJECT ORIENTED PROGRAMMING LANGUAGE PROJECT

by

HARSH NAVIN SETA AND KARTIK SHUKLA

(2019BCS-024) (2019BCS-077)

under the supervision of

Dr. VINAL PATEL

ABV-INDIAN INSTITUTE OF INFORMATION TECHNOLOGY AND MANAGEMENT GWALIOR-474 015

CANDIDATE'S DECLARATION

I hereby certify that I have properly checked and verified all the items as prescribed in the check-list and ensure that my project report is in proper format as specified in the guideline for project preparation.

I also declare that the work containing in this report is my own work. I, understand that plagiarism is defined as any one or combination of the following:

1. To steal and pass off (the ideas or words of another) as one's own
2. To use (another's production) without crediting the source
3. To commit literary theft
4. To present as new and original an idea or product derived from an existing source.

I understand that plagiarism involves an intentional act by the plagiarist of using someone else's work/ideas completely/partially and claiming authorship/originality of the work/ideas. Verbatim copy as well as close resemblance to some else's work constitute plagiarism.

I have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results, websites, that are not my original contribution. I have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.

I affirm that no portion of my work is plagiarized, and the experiments and results reported in the report/dissertation/thesis are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, I shall be fully responsible and answerable. My faculty supervisor(s) will not be responsible for the same.

Signature:

Name:HARSH NAVIN SETA,	Name:KARTIK SHUKLA
Roll No.: 2019BCS-024,	Roll No: 2019BCS-077
Date: 16/11/2020	Date: 16/11/2020

ABSTRACT

The real estate market is exposed to many fluctuations in prices because of existing correlations with many variables, some of which cannot be controlled or might even be unknown. Housing prices can increase rapidly, yet the numerous listings available online where houses are sold or rented are not likely to be updated that often. In some cases, individuals interested in selling a house (or apartment) might include it in some online listing, and forget about updating the price.Thus we aim at developing a machine learning application that will help us predict the prices of the houses so that a person can buy the house at a suitable price

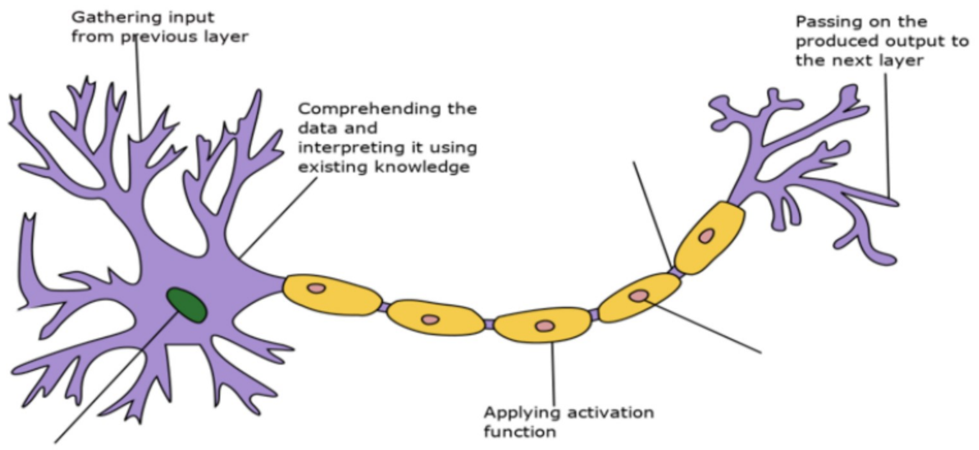
Contents

1 Introduction

1.1 Object Oriented Programming

Object-oriented programming (OOP) is a computer programming model that organizes software design around data, or objects, rather than functions and logic. An object can be defined as a data field that has unique attributes and behavior. OOP focuses on the objects that developers want to manipulate rather than the logic required to manipulate them. This approach to programming is well-suited for programs that are large, complex and actively updated or maintained. The organization of an object-oriented program also makes the method beneficial to collaborative development, where projects are divided into groups. Additional benefits of OOP include code reusability, scalability and efficiency. The first step in OOP is to collect all of the objects a programmer wants to manipulate and identify how they relate to each other – an exercise often known as data modeling. Once an object is known, it is labeled with a class of objects that defines the kind of data it contains and any logic sequences that can manipulate it

1.2 Neural Networks


Gathering input from previous layer
Comprehending the data and interpreting it using existing knowledge
Passing on the produced output to the next layer
Applying activation function

These systems learn to perform tasks by being exposed to various datasets and examples without any task-specific rules. The idea is that the system generates identifying characteristics from the data they have been passed without being programmed with a pre-programmed understanding of these datasets. Neural networks are based on computational models for threshold logic. Threshold logic is a combination of algorithms and mathematics. Neural networks are based either on the study of the brain or on the application of neural networks to artificial intelligence.

Hence, the intelligence brought in by Neural Network to a model has played a major role in the modern day era of Deep learning

Neural networks are based on computational models for threshold logic. Threshold logic is a combination of algorithms and mathematics. Neural networks are based either on the study of the brain or on the application of neural networks to artificial intelligence. The work has led to improvements in finite automata theory.

1.3 Normalization

- Normalization becomes an essential first step when the features in a data set vary in their range. This is because variables that are measured at different scales do not contribute equally to the analysis and might end up creating a bias. Max-min normalization is one of the most common ways to normalize data.

- Formula: X_col[i]=(X_col[i]-X_col.min())/(X_col.max()-X_col.min())

- For every feature, the minimum value of that feature gets transformed into a 0, the maximum value gets transformed into a 1, and every other value gets transformed into a decimal between 0 and 1.

1.4 Randomization of the input data

Before one uses a dataset imported from a source, it is very important to first randomize the data. This is a healthy, essential practice to avoid having any pre-existing pattern in our data-set. Because these unnecessary patterns can be learnt by our model. Our parameters will be wrongly misled, which is not favourable.

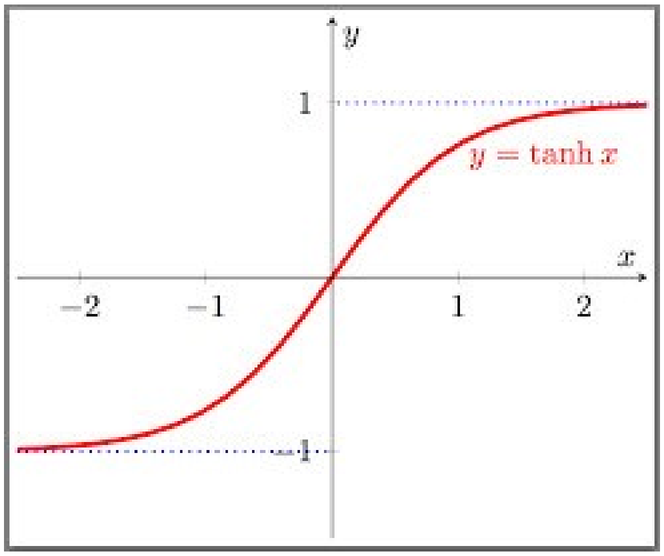
2 Algorithm

In this section we shall provide the algorithm for the implementation of the core concept of our work, namely,Stochastic Gradient Descent.

2.1 Stochastic Gradient Descent

- In Gradient Descent, there is a term called "batch" which denotes the total number of samples from a dataset that is used for calculating the gradient for each iteration. In typical Gradient Descent optimization,like Batch Gradient Descent, the batch is taken to be the whole dataset.
- The word 'stochastic' means a system or a process that is linked with a random probability. Hence, in Stochastic Gradient Descent, one example are selected randomly instead of the whole data set for each iteration. Although,using the whole dataset is really useful for getting to the minima in a less noisy and less random manner, but the problem arises when our datasets gets big.
- Suppose, you have a huge dataset, so if you use a typical Batch Gradient Descent optimization technique, you will have to use all of the one million samples for completing one iteration while performing the Gradient Descent, and it has to be done for every iteration until the minima is reached. Hence, it becomes computationally very expensive to perform. Due to our computational constraint, we've preferred Stochastic Gradient descent algorithm.

2.2 Xaviers Algorithm



- Here we can see that tanh activation function saturates to 1 or -1 very quickly as x gets greater than 1 or -1. Hence we need to prevent the gradients from exploding and reaching very high values.
- This makes gradient descent very slow as parameters get stuck in these saturated regions.

- Xavier initialisation helps us overcome this problem. To avoid the parameters from exploding, we initialise the parameters as
- parameters[i] := parameters [i] * sqrt (2/layer_dims [i-1])
- Hence if number of parameters in a layer is more then the parameters are less in magnitude.
- Therefore , z[i]= dot(A[i-1],W[i-1]) which is sum of multiplication of corresponding Ai's and Wi's elements. Hence z[i] remains just around 1 or -1 preventing the saturation in gradient descent.

2.3 Hyper Parameters

In statistics, hyper-parameter is a parameter from a prior distribution; it captures the prior belief before data is observed. In any machine learning algorithm, these parameters need to be initialized before training a model.

2.3.1 Learning Rate

The amount that the weights are updated during training is referred to as the step size or the "learning rate".The learning rate is a configurable hyper-parameter used in the training of neural networks that has a small positive value.The learning rate controls how quickly the model is adapted to the problem. Smaller learning rates require more training epochs given the smaller changes made to the weights each update, whereas larger learning rates result in rapid changes and require fewer training epochs.

2.3.2 No of Epochs

The number of epochs is the number of complete passes through the training dataset. The size of a batch must be more than or equal to one and less than or equal to the number of samples in the training dataset. The number of epochs can be set to an integer value between one and infinity.

2.3.3 Activation Function

- Activation functions are mathematical functions that determine the output of the layers of a neural network.
- The function is attached to each neuron in the network, and determines whether it should be activated ("fired") or not, based on whether each neuron's input is relevant for the model's prediction.
- ReLU and tanh activation functions are one of the most used and prescribed Regression functions in modern day deep learning era which are both accounted for in our model.

2.3.4 Topology of the Network

The topology of a network defines the 'structure' of a neural network. In simple words, it means the dimensions of layers of our network.

2.4 Propogation algorithms

2.4.1 Forwardpropogation Algorithm

The input data is fed in the forward direction through the network. Each hidden layer accepts the input data, processes it as per the activation function and passes to the successive layer. In order to generate some output, the input data should be fed in the forward direction only. The data should not flow in reverse direction during output generation otherwise it would form a cycle and the output could never be generated. Such network configurations are known as feed-forward network. The feed-forward network helps in forward propagation. At each neuron in a hidden or output layer, the processing happens in two steps:

- 1.Preactivation: it is a weighted sum of inputs i.e. the linear transformation of weights w.r.t to inputs available. Based on this aggregated sum and activation function the neuron makes a decision whether to pass this information further or not.

- 2.Activation: the calculated weighted sum of inputs is passed to the activation function. An activation function is a mathematical function which adds non-linearity to the network. There are four commonly used and popular activation functions — sigmoid, hyperbolic tangent(tanh), ReLU and Softmax.

```
Let Input-> X,Output ->Y
Learning Rate -> (alpha)
CacheLayers ->Z
Neuron Layer->A
Parameter -> w
Activation Function ->g()
No of Layers -> L+1

A[0]:=X
for i from 1 to L:
Z[i]:= dot(A[i-1],w[i-1])
A[i]:=g(Z(i))
Compute Cost:
Total Cost = 1/2(Y-A[L])^2
```

2.4.2 Backpropogation Algorithm

Back-propagation is the essence of neural net training. It is the method of fine-tuning the weights of a neural net based on the error rate obtained in the previous epoch (i.e., iteration). Proper tuning of the weights allows you to reduce error rates and to make the model reliable by increasing its generalization.

Backpropagation is a short form for "backward propagation of errors." It is a standard method of training artificial neural networks. This method helps to calculate the gradient of a loss function with respects to all the weights in the network.

```
Let Input-> X,Output ->Y
Learning Rate ->(alpha)
CacheLayers ->Z
Neuron Layer->A
Parameter -> w
Activation Function ->g()
No of Layers -> L+1

dA[L] := A[L] - Y
for i from L-1 to 1:
dA[i] := dot( dA[i+1]) , w[ i ].transpose)
for i from 0 to L-1:
dW[i]:=dot( dA[i+1]*g'(z[i+1]), A[ i ].transpose)
w[i]:= w[i]- alpha * dw[i]
```

3 Classes Defined

3.1 Neural Network

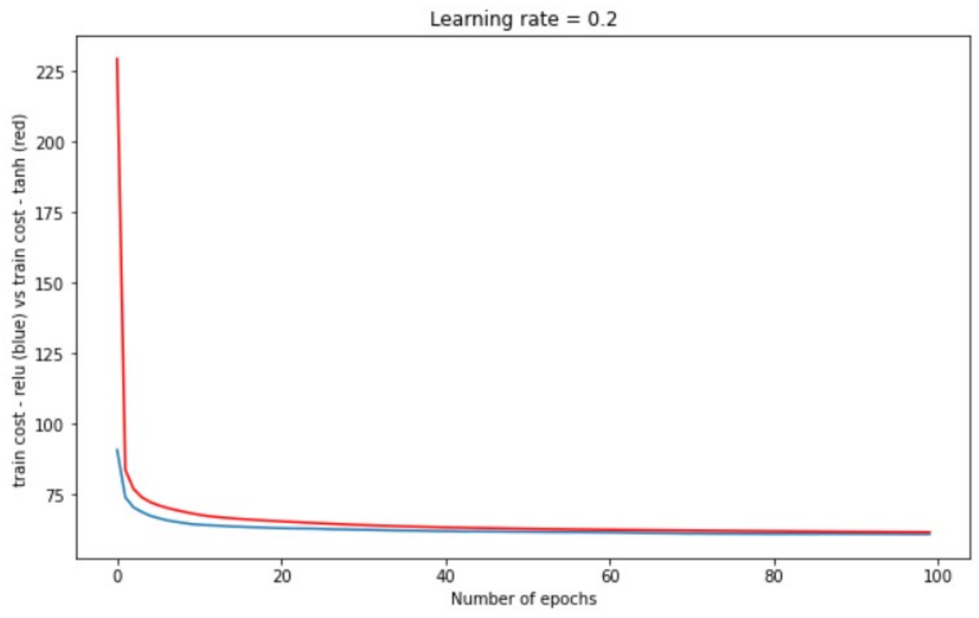
3.1.1 Data Members

- vector<RowVector*> neuronLayers;Stores the different layers of out network.
- vector<RowVector*> cacheLayers;Stores the unactivated (activation fn not yet applied) values of layers.
- vector<RowVector*> deltas;Stores the error contribution of each neurons
- vector<Matrix*> parameters;The parameters of network.
- vector <float> train_pred,test_pred;Stores our prediction.
- vector<float >train_total_cost, train_avg_cost ; stores our total and avg train cost
- vector<float >test_total_cost, test_avg_cost ; stores our total and avg test cost
- vector<int> layer_dims ; dimensions of Neural Network
- float learningRate;
- int num_epochs; set the number of epochs
- string activation_func ; set the activation function to tanh or relu

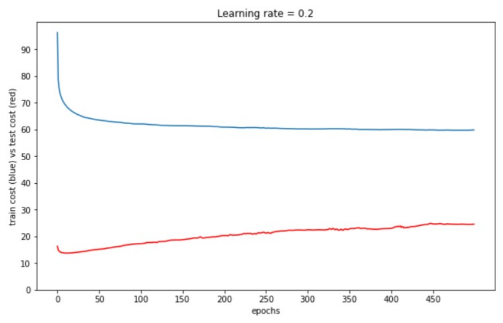
3.1.2 Member Function

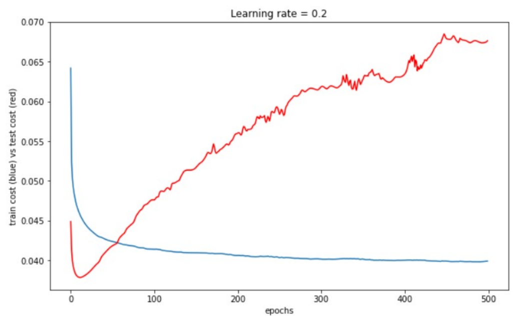
- NeuralNetwork(public; constructor)
- propagateForward(public; void):Function for forward propagation of data.
- propagateBackward(public; void):Function for backward propagation of errors made by neurons.
- calcErrors(public; void):Function to calculate errors made by neurons in each layer
- updateWeights(public; void):Function to update the weights of connections.
- train(public;void):Function to train the neural network give an array of data points
- predict(public; void):Function to make prediction on our test set.

4 Observation



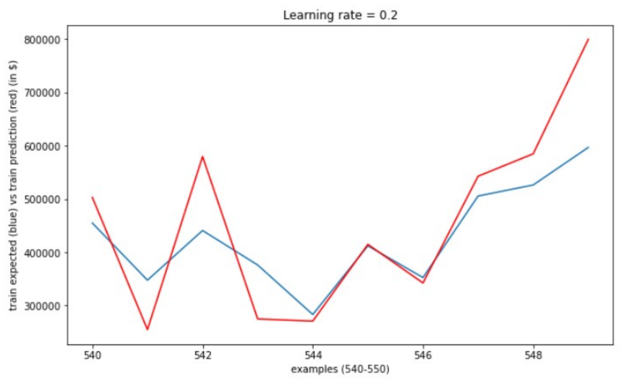
Testing the activation functions relu and tanh we can see that relu is giving us the better performance that tanh activation function as it gives us lower cost.That is why we have included relu as the activation function in our model.

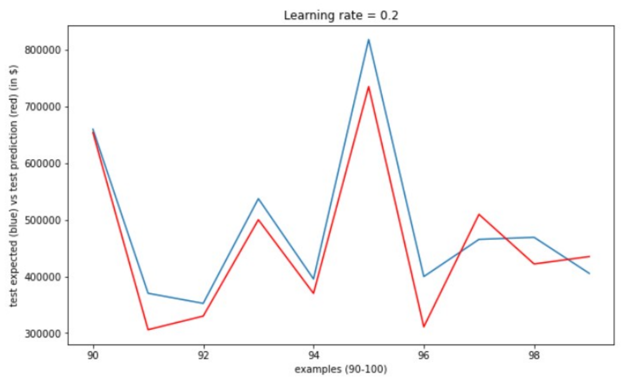




As we can see that after around ten epochs the average and the total test cost increases abruptly,which is not favorable.so we stop at 10 epochs.

5 Result





These are a few random snaps of the graphs that represent the our prediction vs the expected output for training and test data.

6 Conclusion

- Using our Neural Network regression model, we were able to produce a decently accurate predictions on our training and test set.
- This shows how useful Deep learning models is even in real life applications like predicting real estate prices.
- In reality, Modern day deep learning has reached heights like never before. Deep learning is a technology that powers Facebook's news feed, Netflix's recommendation engine and Google's search is radically transforming industries like health care and education.
- Hence we can say that Deep learning has revolutionized the artificial intelligence technologies.

7 References

- https://towardsdatascience.com/forward-propagation-in-neural-networks-simplified-math-and-code-version-bbcfef6f9250
- https://missinglink.ai/guides/neural-network-concepts/7-types-neural-network-activation-functions-right/: :text=Activation
- http://neuralnetworksanddeeplearning.com/chap2.html
- https://prateekvjoshi.com/2016/03/29/understanding-xavier-initialization-in-deep-neural-networks/