**Lactose Intolerance Prediction Using Artificial Neural Networks**

A MINI PROJECT REPORT

Submitted by

BL.EN.U4AIE21006 Akhilesh P

BL.EN.U4AIE21008 Amal Krishna K

BL.EN.U4AIE21112 S Karthick Bharadwaj

**In partial fulfilment for the award of the degree**

**Of**

**BACHELOR IN TECHNOLOGY**

**IN**

COMPUTER SCIENCE AND ENGINEERING (ARTIFICIAL INTELLIGENCE)



AMRITA SCHOOL OF ENGINEERING, BANGALORE

AMRITA VISHWA VIDYAPEETHAM

BANGALORE – 560035

**Abstract**

An artificial neural network for lactose intolerance prediction is presented in this paper. The system input information were the original LCT nucleotide is given as the training data and the output based on that is given which is key predictor of lactose tolerance/intolerance.

The ANN consists of 8 input parameters, each a single nucleotide that feed the training algorithm with information. The aim of ANN presented in this paper is to assist the specialists in lactose intolerance prediction, reducing the time spent.

The model uses the genotype of LCT 13910 C/T and LCT 22018 G/A polymorphisms, which are reliable predictors of lactose tolerance/intolerance, and that information was the output of the neural network.

**Table of Contents**

1. Abstract
2. Lactose intolerance
3. Artificial Neural Networks
4. Program Flow
5. Code
6. Output
7. References

**Lactose Intolerance**

Lactose is a disaccharide that is found in all mammalian milks and is very important for nutrition of new born and infants. In order to be digested, lactose has to be hydrolysed by enzyme lactase (lactase-phlorizin hydrolase, or LPH) into simple sugars, glucose and galactose. Lactase is a trans-membrane glycoprotein of the small intestinal brush border membrane of enterocytes coded, in humans, with LCT gene located on the chromosome 2, long (q) arm at position 21.3. This gene is 49.3 kb in length, consisted of 17 exons and is translated into a 6 kb transcript.

Lactase digestive activity reaches its peak in the first few months of life and decreases after the age of two years. Deficient or absent lactase enzymatic activity in the small intestine results in inability of organism to digest lactose from milk and other dairy products. This condition is called lactose intolerance. Besides the congenital lactase deficiency, which is a very rare condition inherited in an autosomal recessive manner, identified by total lactose intolerance already at infant age, there are three other types of lactose intolerance: primary, secondary, and developmental lactase deficiency. Developmental lactase deficiency and reduced lactase activity is found in infants born before 34 weeks of gestation. Gray was among the first scientists to describe secondary lactase deficiency. Secondary lactase deficiency could occur as a consequence of small intestinal injuries, caused by many different factors such as infections, surgery, chemotherapy, celiac disease, gastroenteritis, prolonged use of antibiotics and other. The most common type of lactose intolerance, which appears in adulthood, is in most of cases characterized with low lactase activity (hypolactasia) leading to primary lactase deficiency.

**Artificial Neural Network**

Machine learning is a field in artificial intelligence and is one of the most rapidly developing subfields of artificial intelligence research. Machine learning enables highly proficient intelligent data analysis. The inexpensive and relatively easy methods developed within the last two decades for collecting and storing data also contributed to making machine learning procedures easier and more consistent. Since the beginning, machine learning was used and implemented within the medical field. Many hospitals and clinics worldwide are monitoring and collecting data which can later be used for machine learning purposes. The machine learning methodology is most convenient for very specific diagnostic problems. Approximations of explanations of certain processes can be considered as the essence of machine learning. Approximations generally do not and cannot explain the whole process, therefore usage of other algorithms would be more convenient. The machine learning process takes into account that the patterns observed within the existing dataset will not change within the future datasets regarding the same problem. In medicine, machine learning programs used for predictions of medical diagnosis are mostly based on concrete biological and physical parameters.

**Program Flow**

We first train the program by giving the actual LCT chromosomes sequence and set a targeted output. We take this chromosome and pass it through different layers called as hidden layers. We take the dot product of the input and weights and we add an offset called bias. We pass this to the activation function which in this case is a sigmoid function which will give the output form the range 0 to 1. We then compare it with the targeted output and if errors are there we update the weights and bias using the gradient function and run it again. After this we train the program so that any input given to it can predict the output and the scope of error will become very low.

After this we pass a new input to the program and compare this prediction with the targeted output.

If the given prediction and the targeted output match we can say that the person is lactose tolerant, if not, then a mutation has occurred which gives us the confirmation that this person is lactose intolerant.

**CODE**

import numpy as np

weights\_1 = np.array([1.45, -0.66])

bias = np.array([0.0])

class NeuralNetwork:

    def \_\_init\_\_(self, learning\_rate):

        self.weights = np.array([np.random.randn(), np.random.randn()])

        self.bias = np.random.randn()

        self.learning\_rate = learning\_rate

    def \_sigmoid(self, x):

        return 1 / (1 + np.exp(-x))

    def \_sigmoid\_deriv(self, x):

        return self.\_sigmoid(x) \* (1 - self.\_sigmoid(x))

    def predict(self, input\_vector):

        layer\_1 = np.dot(input\_vector, self.weights) + self.bias

        layer\_2 = self.\_sigmoid(layer\_1)

        prediction = layer\_2

        return prediction

    def \_compute\_gradients(self, input\_vector, target):

        layer\_1 = np.dot(input\_vector, self.weights) + self.bias

        layer\_2 = self.\_sigmoid(layer\_1)

        prediction = layer\_2

        derror\_dprediction = 2 \* (prediction - target)

        dprediction\_dlayer1 = self.\_sigmoid\_deriv(layer\_1)

        dlayer1\_dbias = 1

        dlayer1\_dweights = (0 \* self.weights) + (1 \* input\_vector)

        derror\_dbias = (

            derror\_dprediction \* dprediction\_dlayer1 \* dlayer1\_dbias

        )

        derror\_dweights = (

            derror\_dprediction \* dprediction\_dlayer1 \* dlayer1\_dweights

        )

        return derror\_dbias, derror\_dweights

    def \_update\_parameters(self, derror\_dbias, derror\_dweights):

        self.bias = self.bias - (derror\_dbias \* self.learning\_rate)

        self.weights = self.weights - (

            derror\_dweights \* self.learning\_rate

        )

    def train(self, input\_vectors, targets, iterations):

        cumulative\_errors = []

        for current\_iteration in range(iterations):

            # Pick a data instance at random

            random\_data\_index = np.random.randint(len(input\_vectors))

            input\_vector = input\_vectors[random\_data\_index]

            target = targets[random\_data\_index]

            # Compute the gradients and update the weights

            derror\_dbias, derror\_dweights = self.\_compute\_gradients(

                input\_vector, target

            )

            self.\_update\_parameters(derror\_dbias, derror\_dweights)

            # Measure the cumulative error for all the instances

            if current\_iteration % 100 == 0:

                cumulative\_error = 0

                # Loop through all the instances to measure the error

                for data\_instance\_index in range(len(input\_vectors)):

                    data\_point = input\_vectors[data\_instance\_index]

                    target = targets[data\_instance\_index]

                    prediction = self.predict(data\_point)

                    error = np.square(prediction - target)

                    cumulative\_error = cumulative\_error + error

                cumulative\_errors.append(cumulative\_error)

        return cumulative\_errors

seq = "AACAGTTC"

A = np.array([[0,0]])

C = np.array([[0,1]])

T = np.array([[1,0]])

G = np.array([[1,1]])

input\_vectors = np.array([[0,0],[0,0],[0,1],[0,0],[1,1],[1,0],[1,0],[0,1]])

targets = np.array([0, 0, 0.3, 0, 1, 0.6, 0.6, 0.3])

learning\_rate = 1

input\_vector = np.array([[0,0]])

sequ = input("enter the lct chromosome(8 characters): ")

for i in sequ:

  if i == 'A':

    input\_vector = np.vstack([input\_vector, A])

  elif i == 'C':

    input\_vector = np.vstack([input\_vector, C])

  elif i == 'T':

    input\_vector = np.vstack([input\_vector, T])

  elif i == 'G':

    input\_vector = np.vstack([input\_vector, G])

input\_vector = np.delete(input\_vector,0,axis = 0)

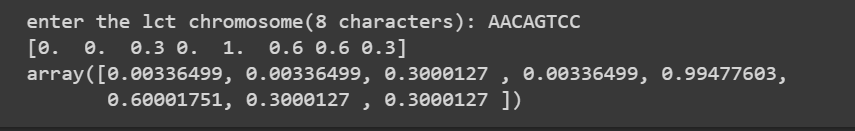
neural\_network = NeuralNetwork(learning\_rate)

training\_error = neural\_network.train(input\_vectors, targets, 100000)

print(targets)

neural\_network.predict(input\_vector)

**OUTPUT**

****

**REFERENCES**

1. Troelsen JT. Adult-type hypolactasia and regulation of lactase expression. Biochim Biophys Acta. 2005;1723:19–32.

2. Rasinperä H, Savilahti E, Enattah NS, et al. A genetic test which can be used to diagnose adult-type hypolactasia in children. Gut. 2004;53:1571–1576.

3. Danielsen, E. M., Skovbjerg, H., Norén, O., & Sjöström, H. (1984). Biosynthesis of intestinal microvillar proteins intracellular processing of lactase-phlorizin hydrolase. Biochemical and biophysical research communications, 122(1), 82-90.

4. Naim, H. Y., Sterchi, E. E., & Lentze, M. J. (1987). Biosynthesis and maturation of lactasephlorizin hydrolase in the human small intestinal epithelial cells. Biochemical Journal, 241(2), 427-434.

5. Spahić, Lemana & Šehović, Emir & Secerovic, Alem & Đozić, Zerina & Smajlović Skenderagić, Lejla. (2020). Lactose Intolerance Prediction Using Artificial Neural Networks. 10.1007/978-3-030-17971-7\_75.