A project report for partial fulfilment of the Degree of Bachelor of Technology
In Electronics and Communication Engineering on

"SUICIDE PREDICTION USING REGRESSION CLASSIFICATION AND NEURAL NETWORKS MODELS"

Submitted by
Avhishek Biswas, Roll – 18700316065, Year – 2019, Dept. - ECE
Deep Bhattacharya, Roll – 18700316058, Year – 2019, Dept. - ECE
Ananya Talukdar, Roll – 18700316081, Year – 2019, Dept. - ECE
Arijit Chowdhury, Roll – 18700316070, Year – 2019, Dept. - ECE

Under the supervision of
Prof. Judhajit Sanyal
Assistant Professor, Department of Electronics and Communication Engineering

Techno International Newtown
New Town, Rajarhat, Kolkata - 700156

CERTIFICATE

This is to certify that “Avhishek Biswas, Deep Bhattacharya Ananya Talukdar and Arijit Chowdhury” of the Department of Electronics and Communication Engineering have successfully completed a project phase 1 on “SUICIDE PREDICTION USING REGRESSION CLASSIFICATION AND NEURAL NETWORKS MODELS” during their Fourth Year in B. Tech, session

2016-20 (under Maulana Abul Kalam Azad University of Technology) under my supervision.

The project is ready for evaluation.

Supervisor(s)	Head of the Dept.
____________________	____________________
____________________	Prof. (Dr) Manabendra Maiti
	Department of Electronics and Communication Engineering

ACKNOWLEDGEMENT

We would like to thank our mentor, Prof. Judhajit Sanyal for his guidance and encouragement that helped us pursue the project. We are also obliged to Prof (Dr.) Manabendra Maity, have spent their valuable time in discussing the required concepts. Without our teachers this attempt would never have been successfully implemented, we owe them a debt of gratitude. Our parents have always been supportive. Their blessings are the roots in our endeavours. Finally, we would like to acknowledge our departmental faculty members who have taught us since first year, as each and every basic concept make a complete block of a specialized subject. More than the references the fact that has been necessary is the motivation to learn more, understand and explore a particular topic with interest and passion, which are built and aided by our teachers.

ABSTRACT

In this project we are creating prediction and classification models using different machine learning methods.

- Linear and Spline Regression
- Neural Networks
- Naïve Bayes

The data set that is being used is from Kaggle that has been produced by the Govt. Of India.

The prediction models will be predicting if a given person in a certain socio-economic category will commit suicide.

REQUIRED LIBRARIES

Python has gathered a lot of interest recently as a choice of language for data analysis.

Can become a common language for data science and production of web-based analytics products.

Needless to say, it still has few drawbacks too:

It is an interpreted language rather than compiled language – hence might take up more CPU time.

NumPy stands for Numerical Python. The most powerful feature of NumPy is n-dimensional array. This library also contains basic linear algebra functions, Fourier transforms, advanced random number capabilities and tools for integration with other low-level languages like Fortran, C and C++.

SciPy stands for Scientific Python. SciPy is built on NumPy. It is one of the most useful libraries for variety of high-level science and engineering modules like discrete Fourier transform, Linear Algebra, Optimization and Sparse matrices.

Matplotlib for plotting vast variety of graphs, starting from histograms to line plots to heat plot. You can use Pylab feature in ipython notebook (ipython notebook –pylab = inline) to use these plotting features inline. If you ignore the inline option, then pylab converts ipython environment to an environment, very similar to Matlab. You can also use Latex commands to add math to your plot.

Pandas for structured data operations and manipulations. It is extensively used for data munging and preparation. Pandas were added relatively recently to Python and have been instrumental in boosting Python's usage in data scientist community.

Scikit-Learn . Built on NumPy, SciPy and matplotlib, this library contains a lot of efficient tools for machine learning and statistical modelling including classification, regression, clustering and dimensionality reduction.

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

CLEANING AND PREPARING THE DATA

One Hot Coding of Gender

```
def replace_gender(val):
    if val == "Female": return 1
    else: return 0

new_ds['Gender'] = new_ds['Gender'].apply(replace_gender)
```

Level Coding of Social-Type

```
new_ds['Type'].unique()
array(['Seperated', 'Widowed/Widower', 'Married', 'Divorcee',
       'Never Married'], dtype=object)
new_ds['Type'].replace({
    'Seperated' : 2,
    'Widowed/Widower' : 4,
    'Married' : 1,
    'Divorcee' : 3,
    'Never Married' : 0
}, inplace = True)
```

Function to Categorize data

```
def val(x):
    employed = ['Professional Activity','Service (Private)','Self-empl
oyed (Business activity)',
                'Service (Government)','Public Sector Undertaking','Fa
rming/Agriculture Activity']
    values = ['Employed','Retired','Unemployed']
    if(x in employed):
        return values[0]
    elif(x == 'Retired Person'):
        return values[1]
    else : return values[2]
```

Cataegorizing Education Level

```
def val(x):
    if(x == 'No Education'):
        return 0
    elif(x == 'Primary'):
        return 1
    elif(x == 'Middle'):
        return 2
    elif(x == 'Matriculate/Secondary'):
        return 3
    elif(x == 'Hr. Secondary/Intermediate/Pre-Universit'):
        return 4
    elif(x == 'Diploma'):
        return 5
    elif(x == 'Graduate'):
        return 6
    elif(x == 'Post Graduate and Above'):
        return 7
    else: return 8

cataegory = pd.Series([])
for ind,row in ds.iterrows():
    ds.loc[ind,"Cataegory"] = val(ds.loc[ind,"Type"])
ds = ds.astype({"Cataegory": int})
```

LINEAR REGRESSION

In linear regression, the relationships are modelled using linear predictor functions whose unknown model parameters are estimated from the data. Such models are called linear models.[3] Most commonly, the conditional mean of the response given the values of the explanatory variables (or predictors) is assumed to be an affine function of those values; less commonly, the conditional median or some other quantile is used. Like all forms of regression analysis, linear regression focuses on the conditional probability distribution of the response given the values of the predictors, rather than on the joint probability distribution of all of these variables, which is the domain of multivariate analysis.
Linear regression has many practical uses. Most applications fall into one of the following two broad categories:

If the goal is prediction, or forecasting, or error reduction,[clarification needed] linear regression can be used to fit a predictive model to an observed data set of values of the response and explanatory variables. After developing such a model, if additional values of the explanatory variables are collected without an accompanying response value, the fitted model can be used to make a prediction of the response.

If the goal is to explain variation in the response variable that can be attributed to variation in the explanatory variables, linear regression analysis can be applied to quantify the strength of the relationship between the response and the explanatory variables, and in particular to determine whether some explanatory variables may have no linear relationship with the response at all, or to identify which subsets of explanatory variables may contain redundant information about the response.

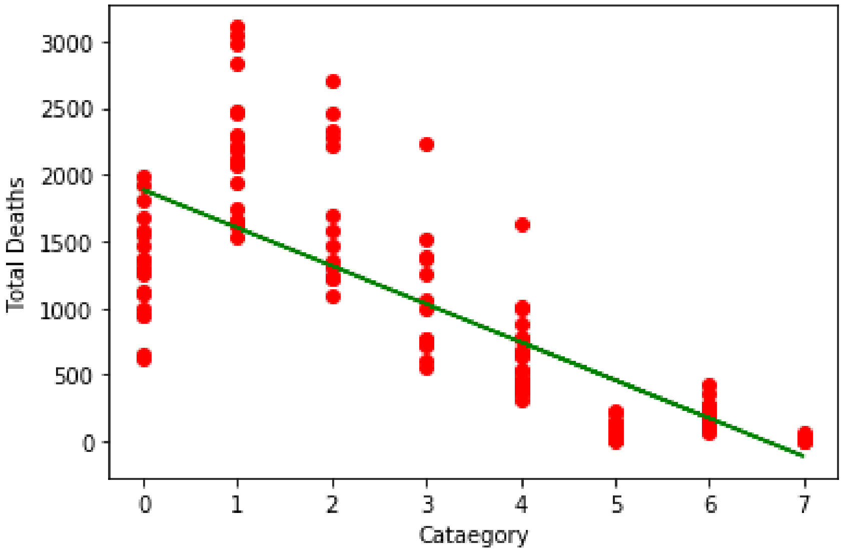
We are creating Linear Regression models on :

- Gender vs Total amount of Deaths for different Social Status
- Category for Education vs Total number of Deaths

As a result we have divided the data into training and testing .The outputs are shown in the following pages.

Creating Regression Model in Educational Status(Cataegory vs Total)

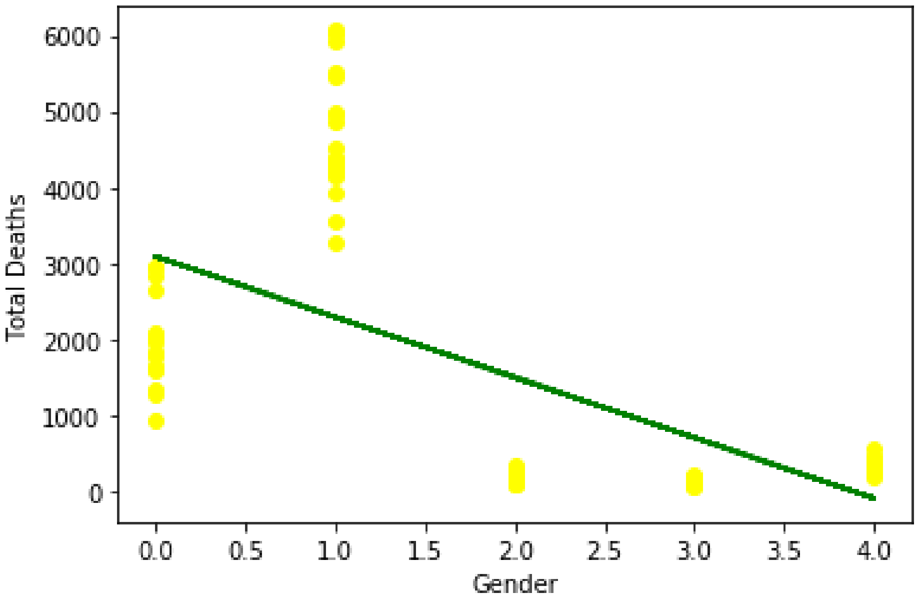
```
model = LinearRegression()
X = useful_Data.drop(['Total'],axis = 'columns')
Y = useful_Data.Total
x_train, x_test, y_train, y_test = train_test_split(X,Y,test_size=1/4)
model.fit(x_train,y_train)
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normaliz
e=False)
plt.scatter(x_train,y_train,color = 'red')
plt.plot(x_train,model.predict(x_train),color='green')
plt.xlabel("Cataegory")
plt.ylabel("Total Deaths")
plt.show()
```



Linear Regression on Social Status for Gender vs Total

```
model = LinearRegression()
X = new_ds.drop(['Gender','Total'],axis = 'columns')
Y = new_ds.Total

x_train, x_test, y_train, y_test = train_test_split(X,Y,test_size=1/3)
model.fit(x_train,y_train)
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normaliz
e=False)
plt.scatter(x_train,y_train,color = 'yellow')
plt.plot(x_train,model.predict(x_train),color='green')
```



Total Deaths
6000
5000
4000
3000
2000
1000
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Gender

SPLINES

What is a spline

- An interval [a..b] is subdivided into sufficiently small intervals [ξ_j.. ξ_{j+1}], with a=ξ_1<…<ξ_{l+1}=b,
- On each such interval, a polynomial p_j of relatively low degree can provide a good approximation to g.
- This can even be done in such a way that the polynomial pieces blend smoothly, i.e. so that the resulting composite function s(x) that equals $p_j(x)$ for $x \in [\xi_j\ \xi_{j+1}]$, all j , has several continuous derivatives.
- Any such smooth piecewise polynomial function is called a *spline*.

Problem splines solves:

- In the simplest situation, one is given points (t_i, y_i) and is looking for a piecewise polynomial function f that satisfies $f(t_i)=y_i$, all i, more or less.

Exact fit → Interpolation

Approximate fit → least squares approximation or smoothing splines

- Or we have a complex function g that we want to approximate with a piecewise polynomial function f that satisfies $f(t_i)=g(t_i)$, all i, more or less.

Piecewise polynomials:

- Jackson's Theorem: If g has r continuous derivatives on [a..b] and n> r+1, then

$$g \in C^{(r)}[a..b] \wedge (n > r+1) \Rightarrow dist(g, \Pi_{<n}) \leq const_r \left(\frac{b-a}{n-1}\right)^r .w(g^{(r)}; \frac{b-a}{2(n-1-r)})$$

$$w(g;h) := \max\left\{| g(x) - g(y) | : x, y \in [a..b], | x-y | \leq h\right\}$$

- The only way to make the error small is to make (b-a)/ (n-1) small → partitioning [a..b] → using piecewise polynomial approximation.

- It is usually much more efficient to make (b-a) small than to increase n. This is the justification for piecewise polynomial approximation.

NAÏVE BAYES

Naive Bayes classifiers are a collection of classification algorithms based on Bayes’ Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.
Bayes’ Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes’ theorem is stated mathematically as the following equation:

P(A|B) = \frac{P(B|A) P(A)}{P(B)}

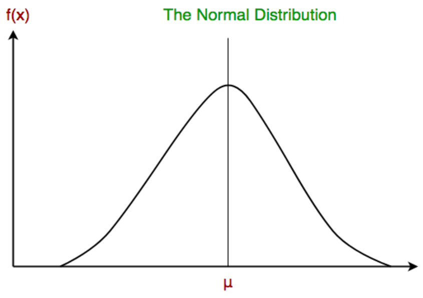
where A and B are events and P(B) ? 0.

Basically, we are trying to find probability of event A, given the event B is true. Event B is also termed as evidence.
P(A) is the priori of A (the prior probability, i.e. Probability of event before evidence is seen). The evidence is an attribute value of an unknown instance(here, it is event B).
P(A|B) is a posteriori probability of B, i.e. probability of event after evidence is seen.
Now, with regards to our dataset, we can apply Bayes’ theorem in following way:

P(y|X) = \frac{P(X|y) P(y)}{P(X)}

where, y is class variable and X is a dependent feature vector (of size n) where:

X = (x_1,x_2,x_3,.....,x_n)
In Gaussian Naive Bayes, continuous values associated with each feature are assumed to be distributed according to a **Gaussian distribution**. A Gaussian distribution is also called Normal distribution. When plotted, it gives a bell shaped curve which is symmetric about the mean of the feature values as shown below:



The likelihood of the features is assumed to be Gaussian, hence, conditional probability is given by:

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp\left(-\frac{(x_i-\mu_y)^2}{2\sigma_y^2}\right)$$

Other popular Naive Bayes classifiers are:

- Multinomial Naive Bayes: Feature vectors represent the frequencies with which certain events have been generated by a multinomial distribution. This is the event model typically used for document classification.

- Bernoulli Naive Bayes: In the multivariate Bernoulli event model, features are independent booleans (binary variables) describing inputs. Like the multinomial model, this model is popular for document classification tasks, where binary term occurrence(i.e. a word occurs in a document or not) features are used rather than term frequencies(i.e. frequency of a word in the document).

NEURAL NETWORKS

Neural networks are artificial systems that were inspired by biological neural networks. 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. The work has led to improvements in finite automata theory.

Components of a typical neural network involve neurons, connections, weights, biases, propagation function, and a learning rule. Neurons will receive an input from predecessor neurons that have an activation, threshold, an activation function f, and an output function.

Connections consist of connections, weights and biases which rules how neuron which transfers output to neuron. Propagation computes the input and outputs the output and sums the predecessor neurons function with the weight. The learning rule modifies the weights and thresholds of the variables in the network.

Supervised vs Unsupervised Learning:

Neural networks learn via supervised learning; Supervised machine learning involves an input variable *x* and output variable *y*. The algorithm learns from a training dataset. With each correct answers, algorithms iteratively make predictions on the data. The learning stops when the algorithm reaches an acceptable level of performance.

Unsupervised machine learning has input data X and no corresponding output variables. The goal is to model the underlying structure of the data for understanding more about the data. The keywords for supervised machine learning are classification and regression. For unsupervised machine learning, the keywords are clustering and association.

Types of Neural Networks

There are *seven* types of neural networks that can be used.

- The first is a multilayer perceptron which has three or more layers and uses a nonlinear activation function.
- The second is the convolutional neural network that uses a variation of the multilayer perceptrons.
- The third is the recursive neural network that uses weights to make structured predictions.
- The fourth is a recurrent neural network that makes connections between the neurons in a directed cycle. The long short-term memory neural network uses the recurrent neural network architecture and does not use activation function.
- The final two are sequence to sequence modules which uses two recurrent networks and shallow neural networks which produces a vector space from an amount of text. These neural networks are applications of the basic neural network demonstrated below.

Limitations:

The neural network is for a supervised model. It does not handle unsupervised machine learning and does not cluster and associate data. It also lacks a level of accuracy that will be found in more computationally expensive neural network. Based on Andrew Trask's neural network. Also, the neural network does not work with any matrices where X's number of rows and columns do not match Y and W's number of rows.

The next steps would be to create an unsupervised neural network and to increase computational power for the supervised model with more iterations and threading.