SCHOOL OF COMPUTR SCIENCE AND INFORMATION SYSTEM

Weather Forecasting and Classification using Back Propagation Algorithm

SOFT COMPUTING

FALL SEMESTER 2023-24

FACULTY: DR .BALAKRUSHNA TRIPATHY

UNDER THE GUIDANCE OF: DR. BALAKRUSHNA TRIPATHY

TEAM MEMBERS:

PRANAY.G(21MIS0123)

B.LAKSHMI PRASANNA(21MIS0219)

EMAIL

lakshmi.prasanna2021@vit.ac.in

pranay.gorantla2021@vitstudent.ac.in

AIM&OBJECTIVE:

- A tiny disturbance in one layer, even one as tiny as a butterfly flapping its wings, can have a domino effect, affecting the other layers and snowballing into radically different weather patterns. All that variation and uncertainty makes difficult to predict the weather clearly.
- In some regions where land features change a lot over short distances, low resolution cause large errors. These Changes affect the common factors of weather like “precipitation”,“ temperature” which makes difficult to predict the weather.
- To solve the above-mentioned problems, we proposed a solution called “weather Forecasting using Back Propagation Neural Network”

S/W &H/W REQUIREMENTS:

- **Software requirements:**
  - Programming languages used: PYTHON
  - code editor: VISUAL STUDIO
  - Operating system: WINDOWS 11
- **Hardware requirements:**
  - Processor i5 (for efficient usage)
  - 4 GB RAM o 512 GB storage
  - must support virtual box

LITERATURE REVIEW:

1. “ANN BASED WEATHER PREDICTION USING BACK

PROPAGATION TECHNIQUE”.

AUTHORS: Saboor Ahmad kakar, Naveed sheikh, Saleem Iqbal,

Abdul Rehman.

PUBLISHED YEAR: January 2019.

Artificial_neural_network_based_weather_predictionusing back propagation technique creators encased AI by changing the honesty of the brain networks with NWP to get to the breeze speed. ANN is one among the quickest creating methods of AI pondered as non-direct knowing models to perform portrayal and gauge. Fake neural network (ANN) has this direct evolved goal of being a predominant procedure toward improving the accuracy and steady quality.During this way, a diverse brain framework is organized and prepared with the present dataset and brings an association between the present nonlinear boundaries of climate.

2. “WEATHER FORECASTING USING ARTIFICIAL NEURAL NETWORKS”

Published year: June ,2020

Authors: Govind Kumar Rahul, saumyasingh, saumya Dubey

In this paper, system is created using soft computing technique called Artificial Neural Network which will suited for processing the nonlinear conditions of weather and make prognosis that helps the farmers in cultivation according to the weather that predicted for growing better crops.

3.“WEATHER FORECASTING SYSTEM WITH THE USE OF

NEURAL NETWORK AND BACKPROPAGATION ALGORITHM"

AUTHORS: Marek wica, mirosławWitkowski, Anita szumiec,

Tomasz ziebura

PUBLISHED YEAR :2019

In this paper, an application is featured which determines forecast of temperature and amount of rain fall for the next day when it is provided with certain input data from past days. This is made possible by neural networks and back propagation algorithm.Unipolar sigmoid function is used as activation in the system and it was build using python programming language.

4. "ANN BASED WEATHER PREDICTION USING BACK

PROPAGATION TECHNIQUE".

AUTHORS: Saboor Ahmad kakar, Naveed sheikh, Saleem Iqbal,

Abdul Rehman.

PUBLISHED YEAR: January 2019.

Artificial_neural_network_based_weather_predictionusing back propagation technique creators encased AI by changing the honesty of the brain networks with NWP to get to the breeze speed. ANN is one among the quickest creating methods of AI pondered as non-direct knowing models to perform portrayal and gauge. Fake neural network (ANN) has this direct evolved goal of being a predominant procedure toward improving the accuracy and steady quality.During this way, a diverse brain framework is organized and prepared with the present dataset and brings an association between the present nonlinear boundaries of climate.

5. "ANN Approach for Weather Prediction using Back

Propagation"

AUTHORS NAME: Ch.Jyosthna Devi, B.Syam Prasad Reddy,

K.Vagdhan Kumar,B.Musala Reddy,N.Raja

PUBLISHED YEAR: 2020

In this paper, a neural network-based algorithm for predicting the temperature is presented .The Neural Networks package supports different types of training or learning algorithms .One such algorithm is Back Propagation Neural Network (BPN) technique. The main advantage of the BPN neural network method is that it can fairly approximate a large class of

functions. This method is more efficient than numerical differentiation. Thesimple meaning of this term is that our model has potential to capture the complex relationships between many factors that contribute to certaint emperature. The proposed idea is tested using the real time dataset. The results are compared with practical working of meteorological department and these results confirm that our model have the potential for successful application to temperature forecasting.

6. "Weather Forecasting Using ANN with Error

Backpropagation Algorithm"

AUTHORS NAME: Meera Nervekar, Priyanka Fargose, Deb Jyoti

Mukhopadhyay

PUBLISHED YEAR: Conference Paper · March 2019

This paper uses huge dataset of seven years with an interval of 15 min for training and testing the ANN Model. The tansig andpurelin activation functions are used in hidden layers. Use of Back propagation technique will help the model to learn and adapt the necessary adjustments in weight and bias value depending on the error generated in each epoch. Thus, after finishing with the training process, the model will generate more accurate forecasts.

The model is created considering only a specific geographical area. Scope of the model is limited to weather forecasting in Mumbai city only.

7. "Weather Forecasting Uses Backpropagation

Algorithm Artificial Neural Network Model for

Agricultural Planning in Three Villages at Three Sub-

Districts of Gowa Regency"

AUTHORS NAME: A A Lestari, A Munir, Suhardi

PUBLISHED YEAR: 2019

This study aims to develop weather prediction models that are used for planning agricultural cultivation activities. The method used in predicting climate is Backpropagation Artificial Neural Network technique based on rainfall data in 1996-2019 in Samata Village, 1975-2019 in July Boeri Village and 2007-2019 in Teten Batu Village. The results showed that the climate classification according to Oldman in Samata Village dan JuluBori Village were in the C1 climate type suitable for planting one-time rice crops and crops twice in one year while Tete Batu Village was in the climate type suitable for planting rice crops twice and crops oncea year.

8. "An Approach for Prediction of Weather System by

Using Back propagation Neural Network"

AUTORS NAME: Y.Md.Riyazuddin, Dr.S.Mahaboob Basha,

Dr.K.Krishna Reddy

PUBLISHED YEAR: July 2020 IJSDR | Volume 2, Issue 7

This paper utilizes artificial neural networks for temperature forecasting. Our study based on back propagation neural network which is trained and tested based on dataset provided. Informulating the ANN-based predictive model; three-layer network has been constructed.Suitable air temperature predictions can provide farmers and producers with valuable information when they face decisions regarding the use of mitigating technologies such as orchard heaters or irrigation. The research presented in this thesis developed artificial neural networks models for the prediction of air temperature.

9. "WEATHER PREDICTION USING NEURAL NETWORK

BACKPROPAGATION"

AUTHORS NAME: Anusha N., Sai Seetha M. G. K. M and Bhavana

Lakshmi M

PUBLISHED YEAR: VOL. 14, NO. 24, DECEMBER 2019

Weather forecasting is one of the areas that involve a complex process in meteorology. Ancient weather forecasting methods used by our ancestors usually relied on observing patterns of events. For example, if the sunset in the day looks brighter, assumption was made that the following day will have fair weather. However, not all these predictions proved reliable. In literature, there are different algorithms which can be used for rainfall prediction and are classified into the different types like cloudy, partially cloudy, full cloudy, etc., Out of which some of the methods predicts numerical values but each method as their own merits and demerits The architecture of Neural Networkbackpropagation is built on N different attributes as input layer.

10. "Rainfall Forecasting Using Various Artificial Neural

Network Techniques – A Review"

AUHORS NAME: Nisha Thakur*, Sanjeev Karmakar, Sunita Soni

PUBLISHED YEAR: Volume 7, Issue 3 May-June-2021

The present review reports the work done by the various authors towards rainfall forecasting using the different techniques within Artificial Neural Network concepts. Back-Propagation, Auto- Regressive Moving Average (ARIMA), ANN, K- Nearest Neighborhood (K-NN), Hybrid model (Wavelet ANN), Hybrid Wavelet-NARX model, Rainfall-runoff models, (Two-stag optimization technique), Adaptive Basis Function Neural Network (ABFNN), Multilayer perceptron, etc., algorithms/technologies were reviewed. A tabular representation was used to compare the above-mentioned technologies for rainfall predictions. In most of the articles, training and testing, accuracy was found more than95%.

GAPS IDENTIFIED :

Weather forecasting, while significantly advanced, still faces challenges. Backpropagation, a technique in neural networks used for learning, can contribute to improving forecasting but isn't without limitations. Here are some gaps and challenges in using backpropagation for weather forecasting:

1.Complexity of Weather Patterns:

Weather systems are incredibly complex with nonlinear interactions. Backpropagation might struggle to capture the intricacies of these systems accurately.

2. Data Quality and Quantity:

Weather forecasting requires vast amounts of high-quality data. In some cases, data might be sparse or contain noise, impacting the effectiveness of backpropagation in learning accurate patterns.

3. Temporal Dependencies:

Weather is dynamic and exhibits temporal dependencies. Backpropagation might not handle long-term dependencies well, leading to challenges in accurately predicting weather trends over extended periods.

4. Model Generalization:

Neural networks trained using backpropagation can overfit or underfit data, leading to poor generalization. Weather forecasting models need to generalize well to diverse weather conditions.

5. Limited Understanding of Atmospheric Physics:

While neural networks can learn from data, they lack inherent knowledge of atmospheric physics. Incorporating physical principles into neural networks remains a challenge.

6. Computational Requirements:

Training complex neural networks for weather forecasting demands significant computational resources. Real-time forecasting within computational constraints remains challenging.

7. Model Interpretability:

Neural networks are often considered black boxes, making it challenging to interpret how they arrive at specific forecasts. Understanding the reasoning behind predictions is crucial in weather forecasting.

8.Ensemble Techniques:

Simply relying on backpropagation might limit the potential for ensemble techniques that combine multiple models for improved accuracy and reliability.

9. Adaptability to Climate Change:

Weather patterns are changing due to climate change. Neural networks might need frequent retraining or adaptive techniques to cope with these shifts effectively.

10. Uncertainty Estimation:

Weather predictions inherently involve uncertainty. Assessing and quantifying uncertainty within neural network predictions is essential but challenging.

Addressing these gaps might involve hybrid models that combine neural networks with physics-based models, using advanced techniques like attention mechanisms to capture temporal dependencies better, and developing strategies to handle uncertainty in forecasts.

Improving weather forecasting using backpropagation requires a multidisciplinary approach that combines expertise in meteorology, data science, and computational techniques to overcome these challenges.

PROBLEM FRAMED:

Objective:

Develop a neural network model using backpropagation to forecast weather conditions accurately.

Key Components:

1.Data Collection:

Gather comprehensive and diverse weather data including temperature, humidity, wind speed, pressure, etc., from various sources like satellites, weather stations, and sensors.

2. Data Preprocessing:

Cleanse and preprocess the collected data to handle missing values, outliers, and ensure uniformity in the dataset.

3. Feature Selection/Engineering:

Identify relevant features and possibly engineer new features that could enhance the model's predictive capability.

4. Model Development:

Design and train a neural network using backpropagation to learn patterns and relationships within the weather data. Experiment with different architectures, activation functions, and optimization algorithms to improve model performance.

5. Validation and Testing:

Validate the model using a portion of the dataset not used in training. Evaluate its performance using appropriate metrics like mean absolute error, root mean squared error, or accuracy for classification tasks.

6. Fine-tuning and Optimization:

Fine-tune hyperparameters, explore regularization techniques, and optimize the neural network to achieve better generalization and accuracy.

7. Interpretability and Visualization:

Explore methods to interpret the model's predictions and visualize its reasoning to make the forecasts more understandable to meteorologists and end-users.

8. Deployment and Continuous Improvement:

Deploy the trained model for real-time forecasting and implement mechanisms for continuous learning and improvement based on new data and model feedback.

Challenges and Considerations:

- Addressing the complexities of weather patterns and temporal dependencies.

- Ensuring robustness to handle uncertainty in weather forecasts.

- Integrating domain knowledge of meteorology with the neural network's learning capabilities.

- Optimizing computational resources for efficient real-time forecasting.

- Developing strategies for model interpretability and communicating predictions effectively.

By framing the problem in this manner, the focus is on leveraging backpropagation within neural networks to create a model capable of learning from historical weather data and making accurate forecasts, while also addressing the challenges inherent in weather prediction.

SOLUTION FRAMED:

Backpropagation, a fundamental algorithm in neural networks, can indeed be applied to weather forecasting problems. Here's a high-level overview of how it could work:

1.Data Collection:

Gather historical weather data including temperature, humidity, pressure, wind speed/direction, etc., over a period of time. This data is crucial for training the neural network.

2. Data Preprocessing:

Normalize or scale the data to ensure all input features are on a similar scale. This step helps the neural network converge faster during training.

3. Neural Network Architecture:

Design a neural network architecture suitable for weather forecasting. This may involve multiple layers (input, hidden, output), using activation functions appropriate for the problem, and determining the number of neurons in each layer based on experimentation.

4. Training:

Utilize the historical weather data to train the neural network using the backpropagation algorithm. During training, the network adjusts its weights and biases to minimize the difference between predicted and actual weather conditions.

5. Validation and Testing:

After training, validate the model's performance using a separate validation dataset. Then, test the model on unseen data to evaluate its generalization and accuracy in predicting future weather conditions.

6. Fine-tuning and Optimization:

Based on the validation results, fine-tune the neural network by adjusting hyperparameters, changing the network architecture, or employing regularization techniques to improve its performance.

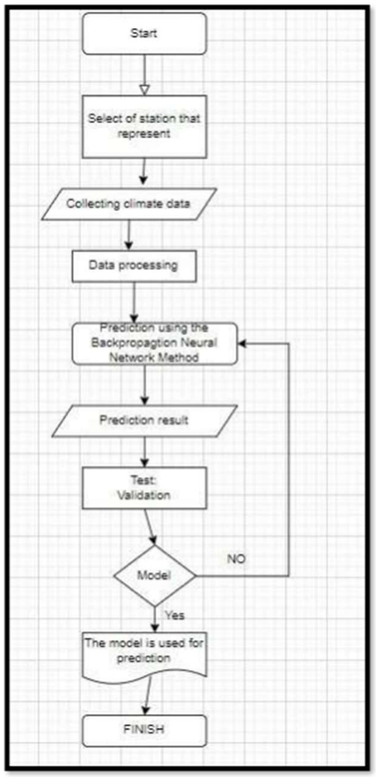
7. Deployment:

Once satisfied with its performance, deploy the trained model to predict future weather conditions based on incoming data.

However, weather forecasting is highly complex due to the multitude of factors involved. Neural networks might struggle to capture all intricate patterns accurately. Thus, combining neural networks with other techniques like physical modeling, ensemble methods, or hybrid models may enhance forecasting accuracy.

Additionally, real-time data acquisition and processing pose challenges. Weather conditions change rapidly, so models need to process data quickly and efficiently for timely predictions.

ARCHITECTURE AND WORKFLOW:


Start
Select of station that represent
Collecting climate data
Data processing
Prediction using the Backpropagtion Neural Network Method
Prediction result
Test: Validation
Model
NO
Yes
The model is used for prediction
FINISH

DETAILED DESCRIPTION OF THE PROJECT MODULES:

Information Assortment:

Observations of climatic tension, temperature, wind speed, wind course, mugginess and precipitation are made close to the world's surface via prepared onlookers, programmed weather conditions stations. The World Meteorological Organisation acts to normalise the instrumentation, noticing practices and timing of these perceptions around the world.

Information absorption:

During the information osmosis process, data acquired from the perceptions is utilised in combination with a mathematical model latest gauge for the time that perceptions were made to deliver the meteorological examination. This is the best gauge of the present status of the climate. It is a three-layered portrayal of the circulation of temperature, dampness and wind. The elements considered in this study are bar temperature, bar perusing, ocean level strain, mean sea level pressure, dry bulb temperature, wet bulb temperature, dew point temperature, fume pressure, wind speed, stickiness, darkness, precipitation, wind bearing, wind .

Mathematical climate forecast: Numerical Weather Forecast (NWP) utilises the force of PCs to make an estimate. Complex PC programs, otherwise called estimate models, run on supercomputers and give forecasts on numerous environmental factors like temperature, pressure, wind and precipitation. A forecaster analyses how the elements anticipated by the PC will interface to create the day's climate.

DATASET:

Our point is to collect a dataset comprising of climate boundaries like temperature, dampness, dew point, perceivability, air pressure, ocean level, wind speed, wind heading and so forth and play out completelyrequired information pre-handling undertakings. Dataset is accumulated from the Kaggle site.

Explanation about complete description:

- Then, at that point, the SVM calculation is prepared by applying Day, Month, Year, and Temperature as autonomous traits and wind speed as the reliant property, which produces related help vectors in light of the given information. Test information is applied to related help vectors shaped from above SVM calculation and wind speed is anticipated. The same cycle is utilised for foreseeing wind courses.
- In this case, wind bearing is the reliant variable and the autonomous factors continue as before. For anticipating mugginess, ascribes month, year, temperature and wind speeds are utilised as autonomous factors. Essentially, for anticipating air pressure, ascribes, day, month, year, temperature, wind speed and wind heading are utilised as autonomous factors. Different Support Vector Machines are created to frame different help vectors in view of various free and subordinate property sets.
- These reproduced support vectors can anticipate the mathematical upsides of subordinate characteristic sets. Information acquired from different SVM calculations are utilised for the test model. After the pre-handling stage, Neural Network Backpropagation is prepared utilising information. In proposed technique of Neural Network Backpropagation comprises of three secret layers with 4, 3 furthermore, 2 number of perceptron's individually, in each of the to start with, second and third secret layers. Yield for each perceptron in a secret layer is acquired by working out summation of result of past layer perceptron's associated and it's comparing loads in addition to comparing current weight coefficient. At last, it creates a connection between the autonomous traits and precipitation present as result layer with insignificant blunder utilising back Propagation method.
- This connection makes the framework equipped for anticipating the mathematical worth of ward quality (precipitation). Backpropagation procedure is performed based on the blunder acquired by the conclusive layer in brain organisation. In the event that the mistake is high, it will change the loads and the equivalent activity is rehashed until the blunder rate is minimized and an ideal connection is created.

LANGUAGE AND TOOLS USED:

Tools and Libraries:

- **VS Code:**

Visual Studio Code permits clients to set the code page in which the dynamic archive is saved, the newline character, and the programming language of the dynamic report. This permits it to be utilised on any stage, in any district, and for some random programming language.

- **Jupiter Notebook:**

It is an open-source web device for information purifying and change, mathematical and reproduction demonstrating, measurable displaying, and information representation, in addition to other things.

- **Keras:**

It is a brain network system for the Python programming language that simplifies it to build and prepare almost any profound learning model.

- **TensorFlow:**

It is a Python library for AI applications like brain networks that involves Keras as a backend.

- **Matplotlib:**

It is an awesome representation library for two-layered exhibit plots composed in the Python coding languages. Quite possibly the main benefit of representation is that it permits high-layered information to be envisioned and perceived, What's more, it incorporates an assortment of plots like line, bar, dissipate, histogram, etc.

PROCEDURE OF THE PROJECT: ALGORITHM USED BACK PROPAGATION ALGORITHM

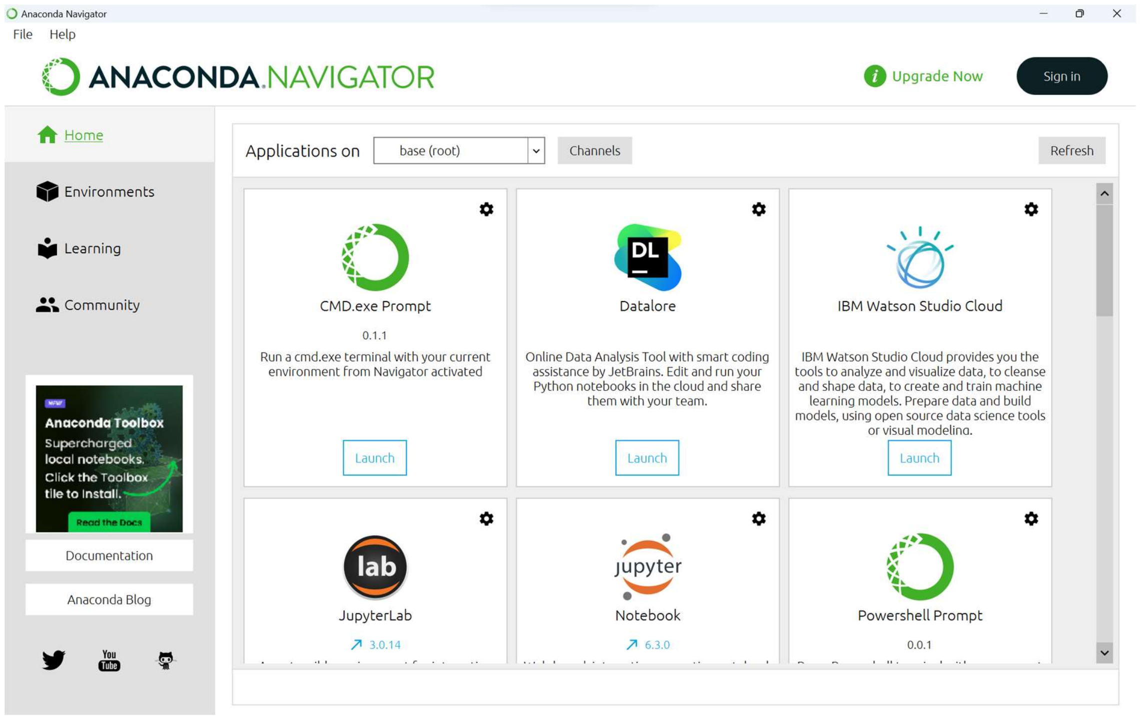
- This learning algorithm is applied to multilayer feed forward networks consists of processing elements with continuously differentiable activation function.
- As weather contain multiple layers it changes according to the changes in the layers, so by using this algorithm we can forecast the weather with less rate of errors.

DATASET USED:

Our point is to collect a dataset comprising of climate boundaries like temperature, dampness, dew point, perceivability, air pressure, ocean level, wind speed, wind heading and so forth and play out completely required information pre-handling undertakings.

Link: https://www.kaggle.com/datasets/dasarisivaram/weather-forecasting

EXPERIMENTAL SETUP EXECUTION:



CMD.exe Prompt
0.1.1
Run a cmd.exe terminal with your current environment from Navigator activated

Datalore
Online Data Analysis Tool with smart coding assistance by JetBrains. Edit and run your Python notebooks in the cloud and share them with your team.

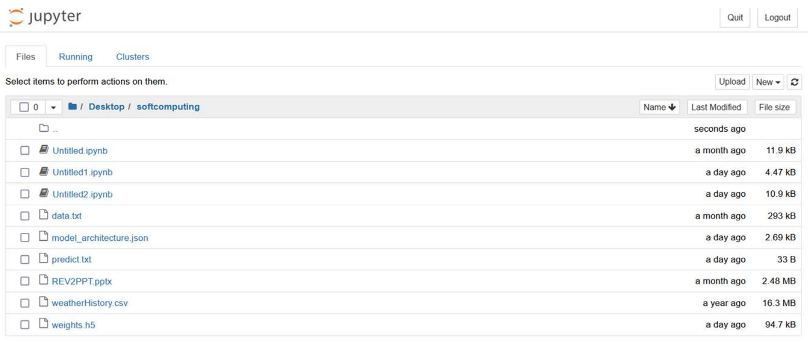
IBM Watson Studio Cloud
IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build models, using open source data science tools or visual modeling.

JupyterLab
3.0.14

Notebook
6.3.0

Powershell Prompt
0.0.1

Fig 1.1 Anaconda Navigator Setup

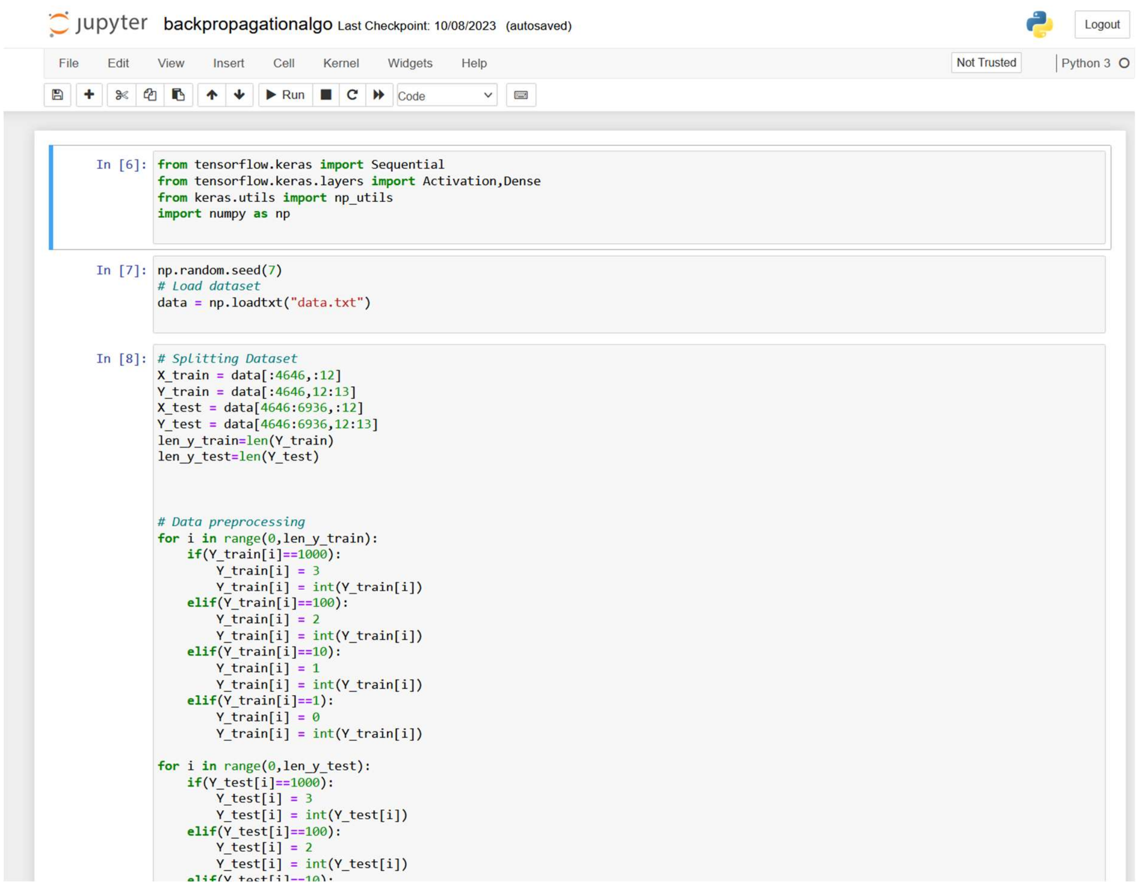


Select items to perform actions on them.

0 / Desktop / softcomputing

Name	Last Modified	File size
..	seconds ago	
Untitled.ipynb	a month ago	11.9 kB
Untitled1.ipynb	a day ago	4.47 kB
Untitled2.ipynb	a day ago	10.9 kB
data.txt	a month ago	293 kB
model_architecture.json	a day ago	2.69 kB
predict.txt	a day ago	33 B
REV2PPT.pptx	a month ago	2.48 MB
weatherHistory.csv	a year ago	16.3 MB
weights.h5	a day ago	94.7 kB

Fig 1.2 Jupyter Setup

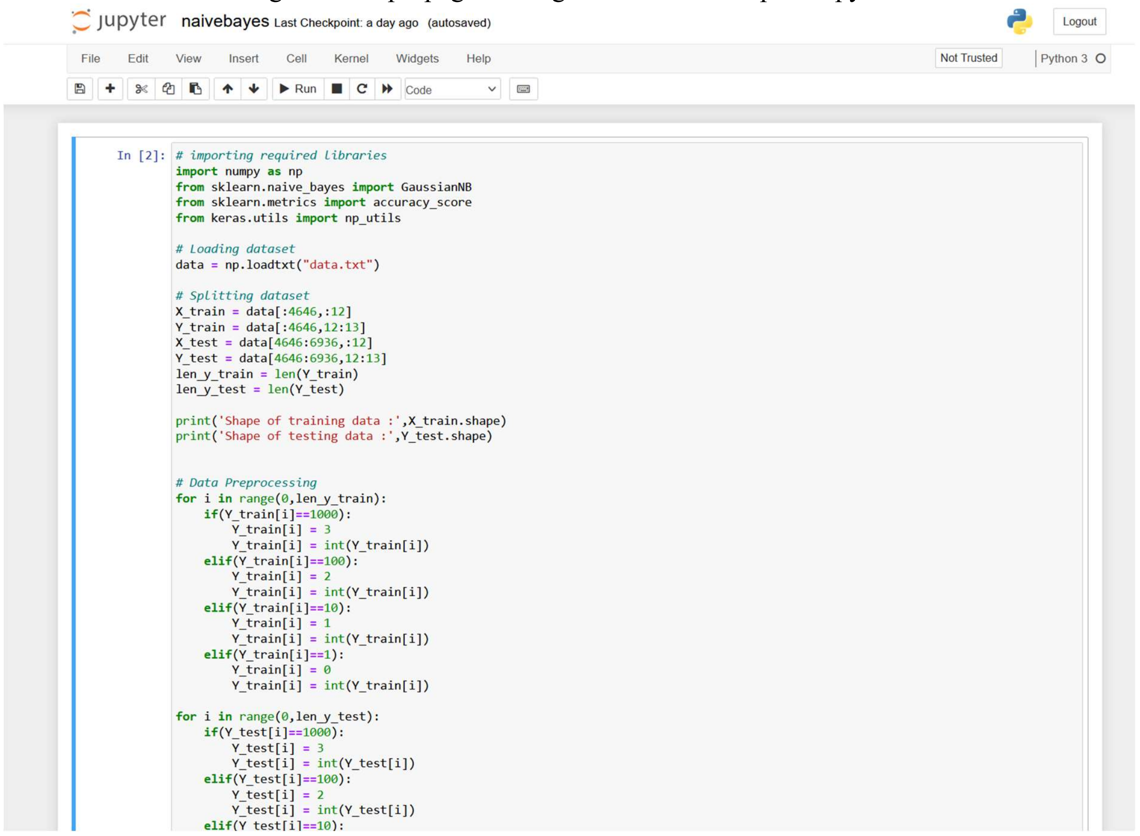

jupyter backpropagationalgo Last Checkpoint: 10/08/2023 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

In [6]:
```
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Activation,Dense
from keras.utils import np_utils
import numpy as np
```

In [7]:
```
np.random.seed(7)
# Load dataset
data = np.loadtxt("data.txt")
```

In [8]:
```
# Splitting Dataset
X_train = data[:4646,:12]
Y_train = data[:4646,12:13]
X_test = data[4646:6936,:12]
Y_test = data[4646:6936,12:13]
len_y_train=len(Y_train)
len_y_test=len(Y_test)



# Data preprocessing
for i in range(0,len_y_train):
    if(Y_train[i]==1000):
        Y_train[i] = 3
        Y_train[i] = int(Y_train[i])
    elif(Y_train[i]==100):
        Y_train[i] = 2
        Y_train[i] = int(Y_train[i])
    elif(Y_train[i]==10):
        Y_train[i] = 1
        Y_train[i] = int(Y_train[i])
    elif(Y_train[i]==1):
        Y_train[i] = 0
        Y_train[i] = int(Y_train[i])

for i in range(0,len_y_test):
    if(Y_test[i]==1000):
        Y_test[i] = 3
        Y_test[i] = int(Y_test[i])
    elif(Y_test[i]==100):
        Y_test[i] = 2
        Y_test[i] = int(Y_test[i])
```

Fig 1.3 Backpropagation Algorithm Code Setup In Jupyter

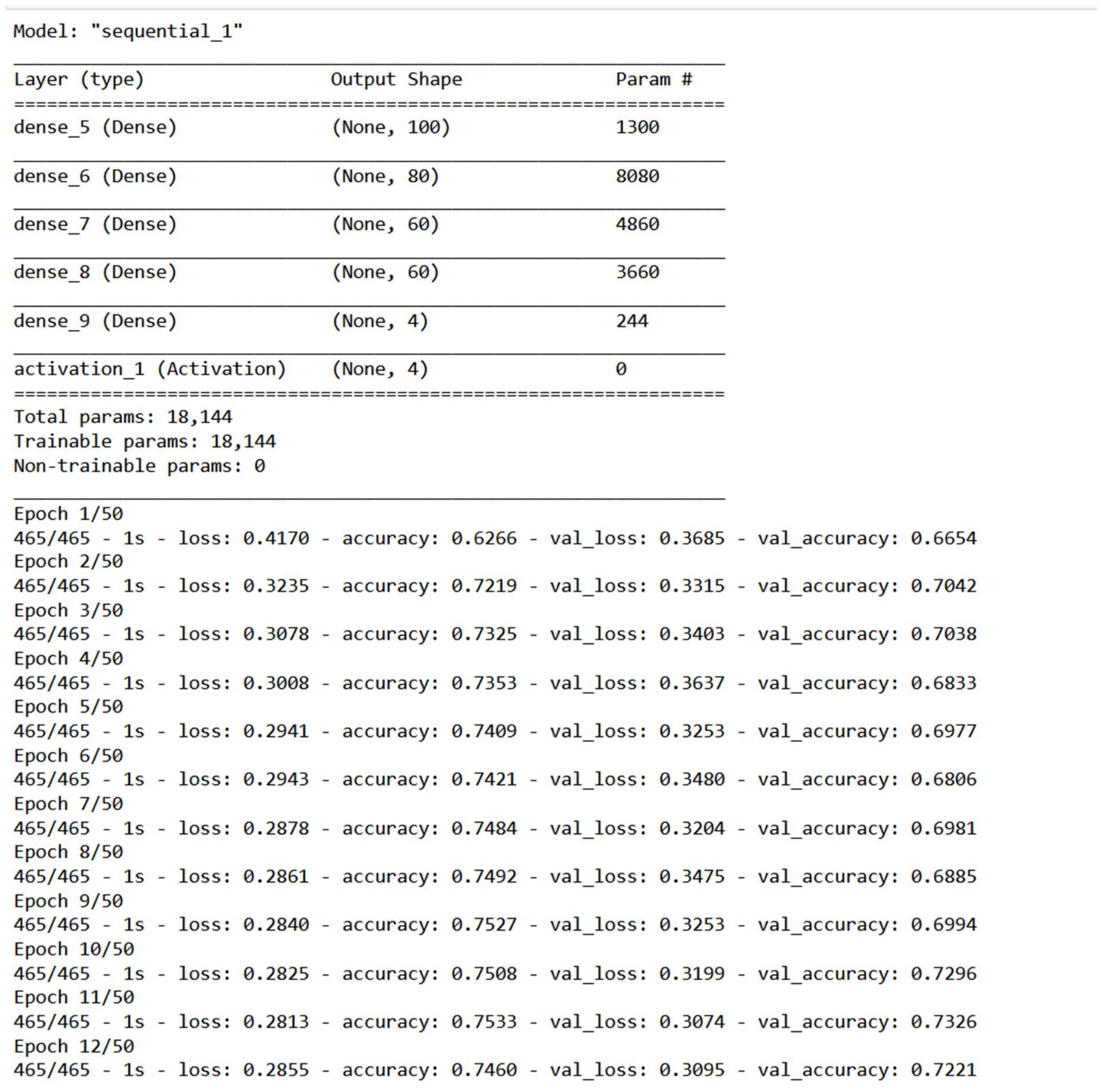

jupyter naivebayes Last Checkpoint: a day ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

```
In [2]: # importing required libraries
        import numpy as np
        from sklearn.naive_bayes import GaussianNB
        from sklearn.metrics import accuracy_score
        from keras.utils import np_utils

        # Loading dataset
        data = np.loadtxt("data.txt")

        # Splitting dataset
        X_train = data[:4646,:12]
        Y_train = data[:4646,12:13]
        X_test = data[4646:6936,:12]
        Y_test = data[4646:6936,12:13]
        len_y_train = len(Y_train)
        len_y_test = len(Y_test)

        print('Shape of training data :',X_train.shape)
        print('Shape of testing data :',Y_test.shape)


        # Data Preprocessing
        for i in range(0,len_y_train):
            if(Y_train[i]==1000):
                Y_train[i] = 3
                Y_train[i] = int(Y_train[i])
            elif(Y_train[i]==100):
                Y_train[i] = 2
                Y_train[i] = int(Y_train[i])
            elif(Y_train[i]==10):
                Y_train[i] = 1
                Y_train[i] = int(Y_train[i])
            elif(Y_train[i]==1):
                Y_train[i] = 0
                Y_train[i] = int(Y_train[i])

        for i in range(0,len_y_test):
            if(Y_test[i]==1000):
                Y_test[i] = 3
                Y_test[i] = int(Y_test[i])
            elif(Y_test[i]==100):
                Y_test[i] = 2
                Y_test[i] = int(Y_test[i])
            elif(Y_test[i]==10):
```

Fig 1.4 Naïve Bayes Code Setup In Jupyter

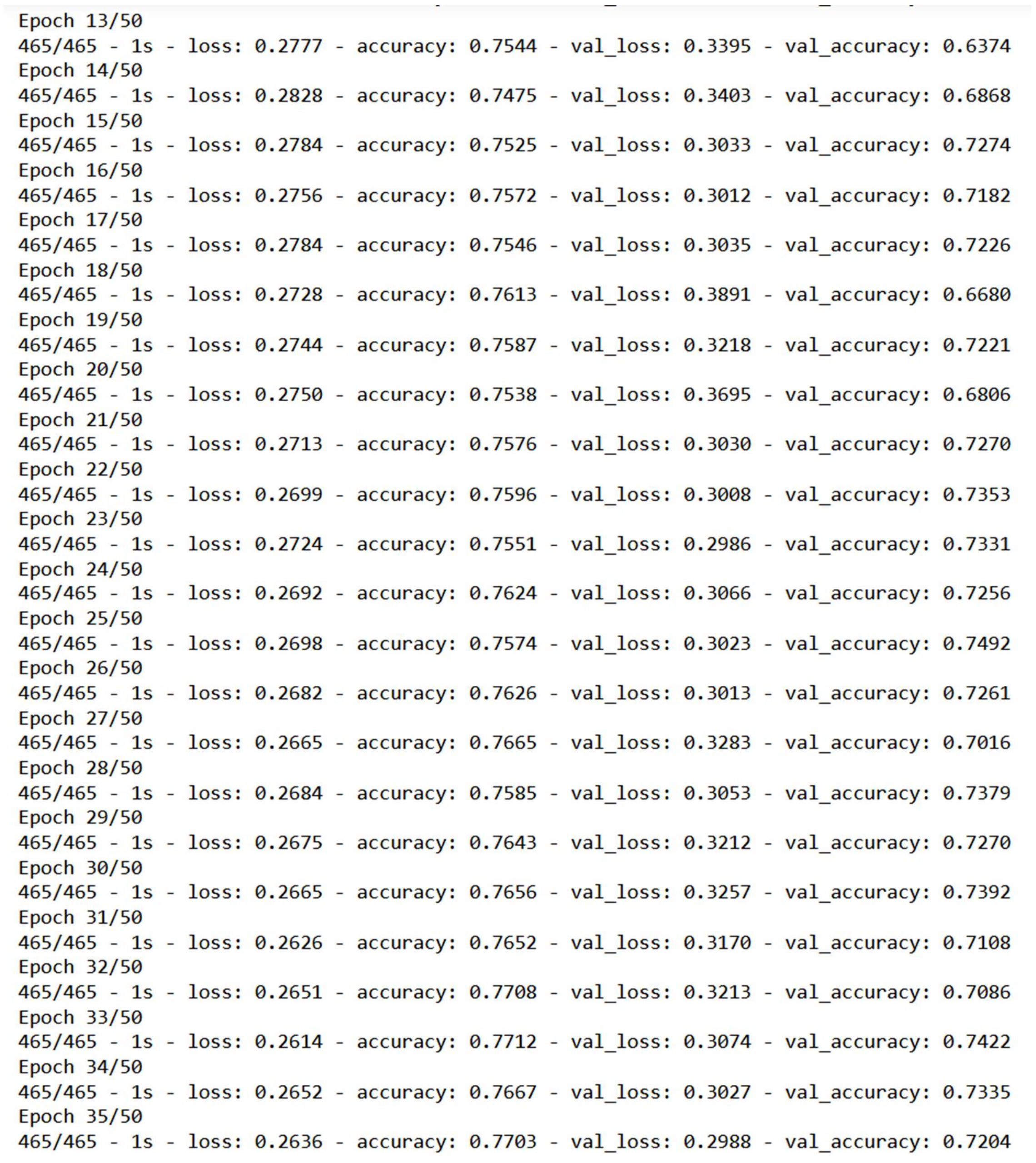
RESULT ANALYSIS:

Results Achieved Using Backpropagation Algorithm Is



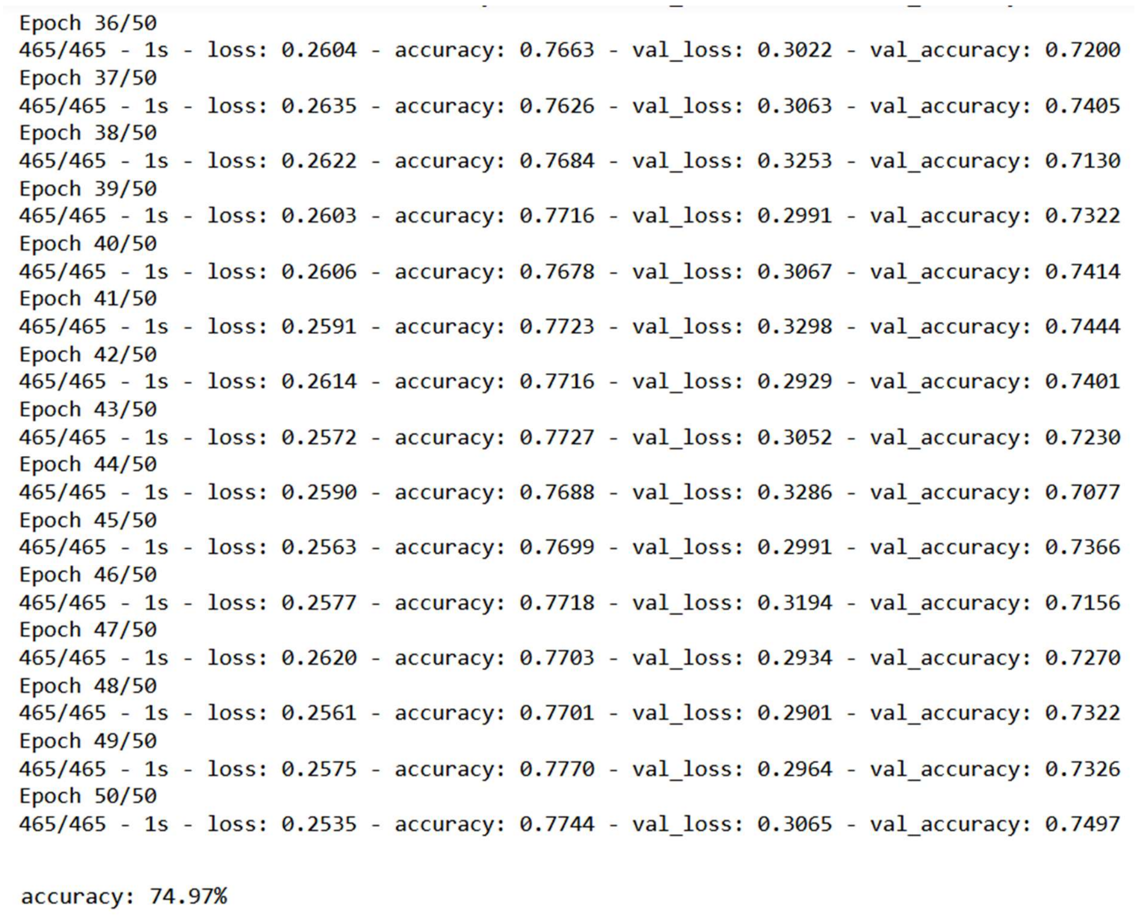
```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_5 (Dense)              (None, 100)               1300      
_________________________________________________________________
dense_6 (Dense)              (None, 80)                8080      
_________________________________________________________________
dense_7 (Dense)              (None, 60)                4860      
_________________________________________________________________
dense_8 (Dense)              (None, 60)                3660      
_________________________________________________________________
dense_9 (Dense)              (None, 4)                 244       
_________________________________________________________________
activation_1 (Activation)    (None, 4)                 0         
=================================================================
Total params: 18,144
Trainable params: 18,144
Non-trainable params: 0
_________________________________________________________________
Epoch 1/50
465/465 - 1s - loss: 0.4170 - accuracy: 0.6266 - val_loss: 0.3685 - val_accuracy: 0.6654
Epoch 2/50
465/465 - 1s - loss: 0.3235 - accuracy: 0.7219 - val_loss: 0.3315 - val_accuracy: 0.7042
Epoch 3/50
465/465 - 1s - loss: 0.3078 - accuracy: 0.7325 - val_loss: 0.3403 - val_accuracy: 0.7038
Epoch 4/50
465/465 - 1s - loss: 0.3008 - accuracy: 0.7353 - val_loss: 0.3637 - val_accuracy: 0.6833
Epoch 5/50
465/465 - 1s - loss: 0.2941 - accuracy: 0.7409 - val_loss: 0.3253 - val_accuracy: 0.6977
Epoch 6/50
465/465 - 1s - loss: 0.2943 - accuracy: 0.7421 - val_loss: 0.3480 - val_accuracy: 0.6806
Epoch 7/50
465/465 - 1s - loss: 0.2878 - accuracy: 0.7484 - val_loss: 0.3204 - val_accuracy: 0.6981
Epoch 8/50
465/465 - 1s - loss: 0.2861 - accuracy: 0.7492 - val_loss: 0.3475 - val_accuracy: 0.6885
Epoch 9/50
465/465 - 1s - loss: 0.2840 - accuracy: 0.7527 - val_loss: 0.3253 - val_accuracy: 0.6994
Epoch 10/50
465/465 - 1s - loss: 0.2825 - accuracy: 0.7508 - val_loss: 0.3199 - val_accuracy: 0.7296
Epoch 11/50
465/465 - 1s - loss: 0.2813 - accuracy: 0.7533 - val_loss: 0.3074 - val_accuracy: 0.7326
Epoch 12/50
465/465 - 1s - loss: 0.2855 - accuracy: 0.7460 - val_loss: 0.3095 - val_accuracy: 0.7221
```

Fig 1.5 Results Of Backpropagation Algorithm



```
Epoch 13/50
465/465 - 1s - loss: 0.2777 - accuracy: 0.7544 - val_loss: 0.3395 - val_accuracy: 0.6374
Epoch 14/50
465/465 - 1s - loss: 0.2828 - accuracy: 0.7475 - val_loss: 0.3403 - val_accuracy: 0.6868
Epoch 15/50
465/465 - 1s - loss: 0.2784 - accuracy: 0.7525 - val_loss: 0.3033 - val_accuracy: 0.7274
Epoch 16/50
465/465 - 1s - loss: 0.2756 - accuracy: 0.7572 - val_loss: 0.3012 - val_accuracy: 0.7182
Epoch 17/50
465/465 - 1s - loss: 0.2784 - accuracy: 0.7546 - val_loss: 0.3035 - val_accuracy: 0.7226
Epoch 18/50
465/465 - 1s - loss: 0.2728 - accuracy: 0.7613 - val_loss: 0.3891 - val_accuracy: 0.6680
Epoch 19/50
465/465 - 1s - loss: 0.2744 - accuracy: 0.7587 - val_loss: 0.3218 - val_accuracy: 0.7221
Epoch 20/50
465/465 - 1s - loss: 0.2750 - accuracy: 0.7538 - val_loss: 0.3695 - val_accuracy: 0.6806
Epoch 21/50
465/465 - 1s - loss: 0.2713 - accuracy: 0.7576 - val_loss: 0.3030 - val_accuracy: 0.7270
Epoch 22/50
465/465 - 1s - loss: 0.2699 - accuracy: 0.7596 - val_loss: 0.3008 - val_accuracy: 0.7353
Epoch 23/50
465/465 - 1s - loss: 0.2724 - accuracy: 0.7551 - val_loss: 0.2986 - val_accuracy: 0.7331
Epoch 24/50
465/465 - 1s - loss: 0.2692 - accuracy: 0.7624 - val_loss: 0.3066 - val_accuracy: 0.7256
Epoch 25/50
465/465 - 1s - loss: 0.2698 - accuracy: 0.7574 - val_loss: 0.3023 - val_accuracy: 0.7492
Epoch 26/50
465/465 - 1s - loss: 0.2682 - accuracy: 0.7626 - val_loss: 0.3013 - val_accuracy: 0.7261
Epoch 27/50
465/465 - 1s - loss: 0.2665 - accuracy: 0.7665 - val_loss: 0.3283 - val_accuracy: 0.7016
Epoch 28/50
465/465 - 1s - loss: 0.2684 - accuracy: 0.7585 - val_loss: 0.3053 - val_accuracy: 0.7379
Epoch 29/50
465/465 - 1s - loss: 0.2675 - accuracy: 0.7643 - val_loss: 0.3212 - val_accuracy: 0.7270
Epoch 30/50
465/465 - 1s - loss: 0.2665 - accuracy: 0.7656 - val_loss: 0.3257 - val_accuracy: 0.7392
Epoch 31/50
465/465 - 1s - loss: 0.2626 - accuracy: 0.7652 - val_loss: 0.3170 - val_accuracy: 0.7108
Epoch 32/50
465/465 - 1s - loss: 0.2651 - accuracy: 0.7708 - val_loss: 0.3213 - val_accuracy: 0.7086
Epoch 33/50
465/465 - 1s - loss: 0.2614 - accuracy: 0.7712 - val_loss: 0.3074 - val_accuracy: 0.7422
Epoch 34/50
465/465 - 1s - loss: 0.2652 - accuracy: 0.7667 - val_loss: 0.3027 - val_accuracy: 0.7335
Epoch 35/50
465/465 - 1s - loss: 0.2636 - accuracy: 0.7703 - val_loss: 0.2988 - val_accuracy: 0.7204
```

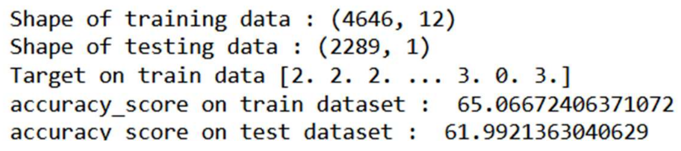
Fig 1.6 Results Of Backpropagation Algorithm



```
Epoch 36/50
465/465 - 1s - loss: 0.2604 - accuracy: 0.7663 - val_loss: 0.3022 - val_accuracy: 0.7200
Epoch 37/50
465/465 - 1s - loss: 0.2635 - accuracy: 0.7626 - val_loss: 0.3063 - val_accuracy: 0.7405
Epoch 38/50
465/465 - 1s - loss: 0.2622 - accuracy: 0.7684 - val_loss: 0.3253 - val_accuracy: 0.7130
Epoch 39/50
465/465 - 1s - loss: 0.2603 - accuracy: 0.7716 - val_loss: 0.2991 - val_accuracy: 0.7322
Epoch 40/50
465/465 - 1s - loss: 0.2606 - accuracy: 0.7678 - val_loss: 0.3067 - val_accuracy: 0.7414
Epoch 41/50
465/465 - 1s - loss: 0.2591 - accuracy: 0.7723 - val_loss: 0.3298 - val_accuracy: 0.7444
Epoch 42/50
465/465 - 1s - loss: 0.2614 - accuracy: 0.7716 - val_loss: 0.2929 - val_accuracy: 0.7401
Epoch 43/50
465/465 - 1s - loss: 0.2572 - accuracy: 0.7727 - val_loss: 0.3052 - val_accuracy: 0.7230
Epoch 44/50
465/465 - 1s - loss: 0.2590 - accuracy: 0.7688 - val_loss: 0.3286 - val_accuracy: 0.7077
Epoch 45/50
465/465 - 1s - loss: 0.2563 - accuracy: 0.7699 - val_loss: 0.2991 - val_accuracy: 0.7366
Epoch 46/50
465/465 - 1s - loss: 0.2577 - accuracy: 0.7718 - val_loss: 0.3194 - val_accuracy: 0.7156
Epoch 47/50
465/465 - 1s - loss: 0.2620 - accuracy: 0.7703 - val_loss: 0.2934 - val_accuracy: 0.7270
Epoch 48/50
465/465 - 1s - loss: 0.2561 - accuracy: 0.7701 - val_loss: 0.2901 - val_accuracy: 0.7322
Epoch 49/50
465/465 - 1s - loss: 0.2575 - accuracy: 0.7770 - val_loss: 0.2964 - val_accuracy: 0.7326
Epoch 50/50
465/465 - 1s - loss: 0.2535 - accuracy: 0.7744 - val_loss: 0.3065 - val_accuracy: 0.7497

accuracy: 74.97%
```

Fig 1.7 Results Of Backpropagation Algorithm

Results Achieved Using Naïve Bayes Algorithm Are:



```
Shape of training data : (4646, 12)
Shape of testing data : (2289, 1)
Target on train data [2. 2. 2. ... 3. 0. 3.]
accuracy_score on train dataset :  65.06672406371072
accuracy score on test dataset :  61.9921363040629
```

Fig 1.7 Results Of Naïve Bayes Algorithm

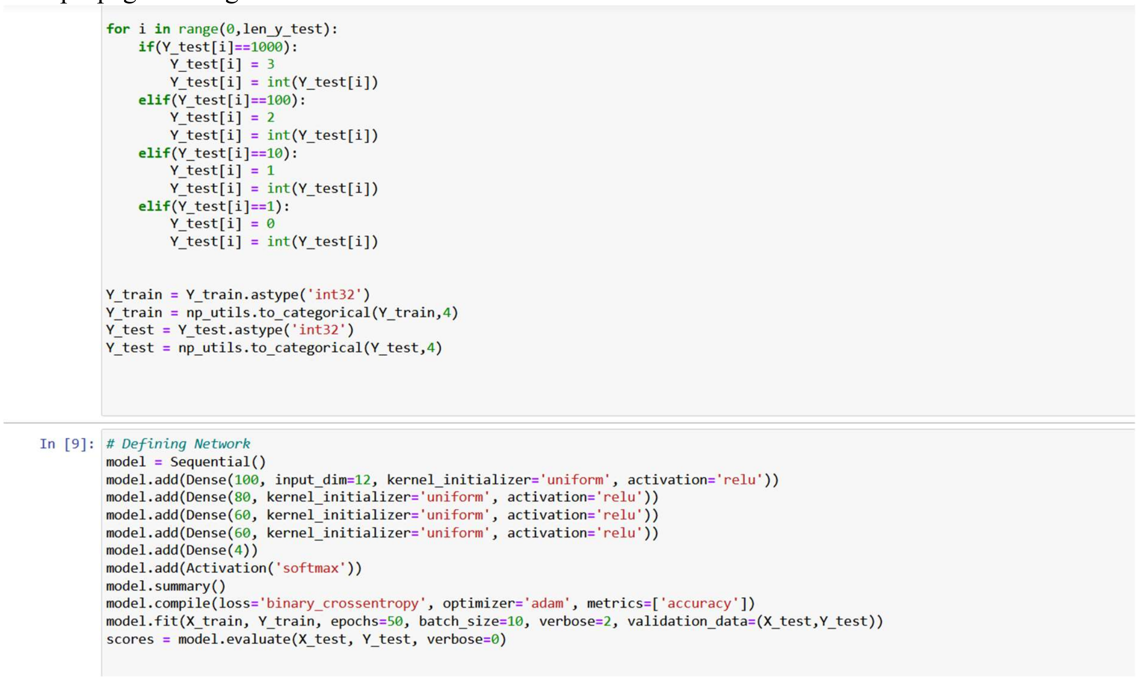
COMPARISION WITH EXISTING APPROACHES:

Table 1.1 Comparision Between Backpropagation And Naïve Bayes Algorithm For Weather Forecasting And Classification Task

Feature	Backpropagation algorithm	Naive bayes algorithm
Algorithm type	Supervised learning	Probabilistic
Model	Artificial neural network	Statistical
Training	Iterative	Non-iterative
Complexity	High	Low
Accuracy	Achieved accuracy of 74.97%	Achieved accuracy of 61.99% but not as high as backpropagation
Interpretability	Low	High
Data requirements	Large	Small
Applications	Weather forecasting, image classification, natural language processing	Spam filtering, sentiment analysis, medical diagnosis

SAMPLE CODE:

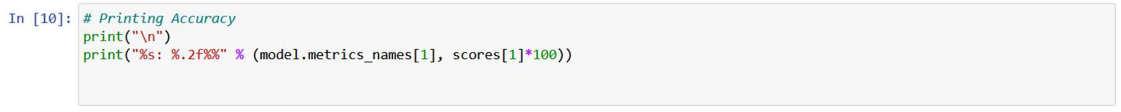
Backpropagation Algorithm Code



```
for i in range(0,len_y_test):
    if(Y_test[i]==1000):
        Y_test[i] = 3
        Y_test[i] = int(Y_test[i])
    elif(Y_test[i]==100):
        Y_test[i] = 2
        Y_test[i] = int(Y_test[i])
    elif(Y_test[i]==10):
        Y_test[i] = 1
        Y_test[i] = int(Y_test[i])
    elif(Y_test[i]==1):
        Y_test[i] = 0
        Y_test[i] = int(Y_test[i])


Y_train = Y_train.astype('int32')
Y_train = np_utils.to_categorical(Y_train,4)
Y_test = Y_test.astype('int32')
Y_test = np_utils.to_categorical(Y_test,4)
```

```
In [9]: # Defining Network
model = Sequential()
model.add(Dense(100, input_dim=12, kernel_initializer='uniform', activation='relu'))
model.add(Dense(80, kernel_initializer='uniform', activation='relu'))
model.add(Dense(60, kernel_initializer='uniform', activation='relu'))
model.add(Dense(60, kernel_initializer='uniform', activation='relu'))
model.add(Dense(4))
model.add(Activation('softmax'))
model.summary()
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model.fit(X_train, Y_train, epochs=50, batch_size=10, verbose=2, validation_data=(X_test,Y_test))
scores = model.evaluate(X_test, Y_test, verbose=0)
```

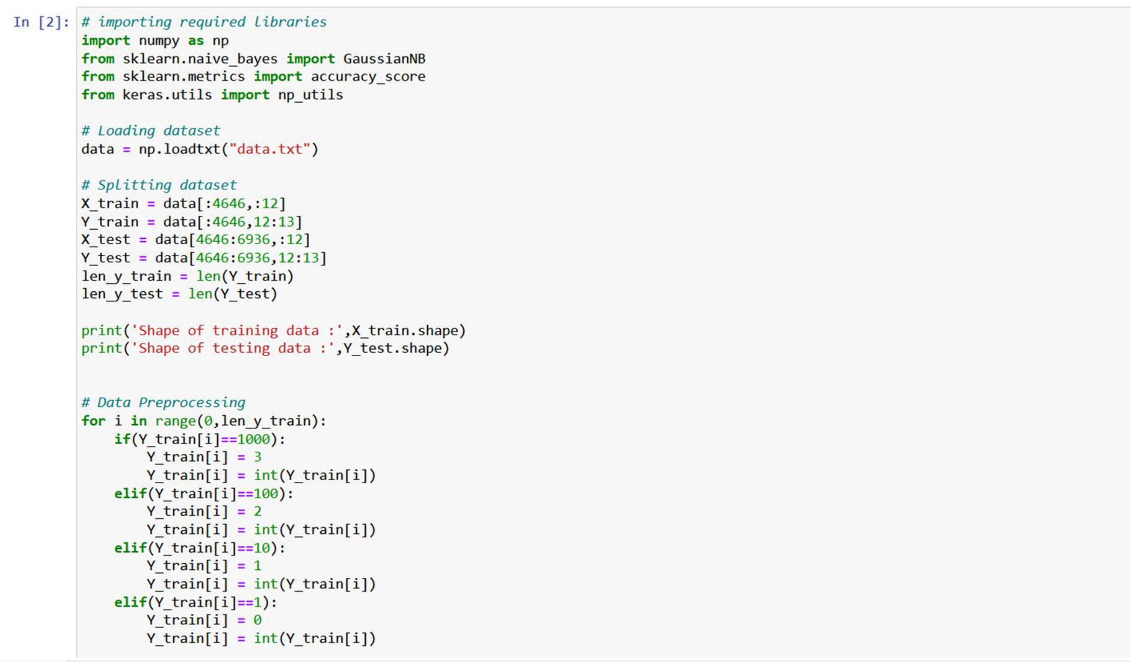
Fig 1.8 Backpropagation Algorithm Code



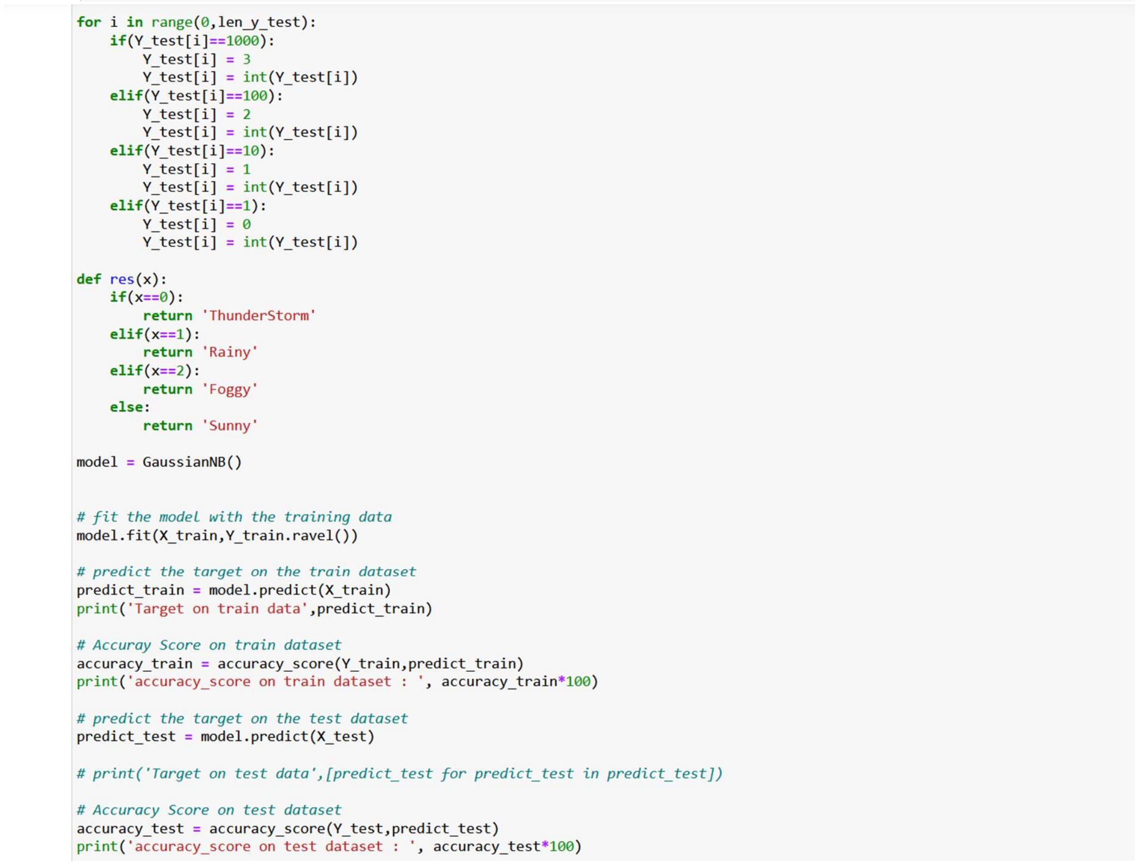
```
In [10]: # Printing Accuracy
print("\n")
print("%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
```

Fig 1.9 Backpropagation Algorithm Code

Naïve Bayes Algorithm Code

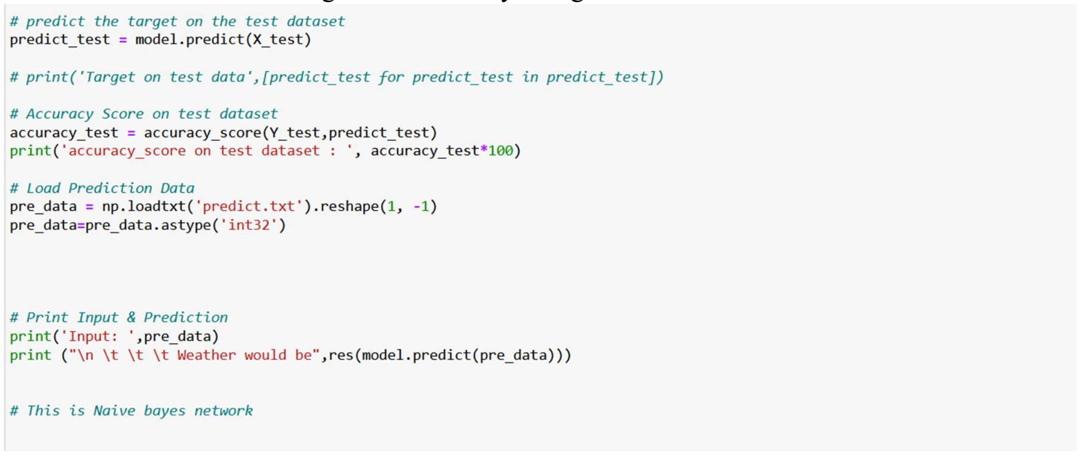


```
In [2]: # importing required Libraries
        import numpy as np
        from sklearn.naive_bayes import GaussianNB
        from sklearn.metrics import accuracy_score
        from keras.utils import np_utils

        # Loading dataset
        data = np.loadtxt("data.txt")

        # Splitting dataset
        X_train = data[:4646,:12]
        Y_train = data[:4646,12:13]
        X_test = data[4646:6936,:12]
        Y_test = data[4646:6936,12:13]
        len_y_train = len(Y_train)
        len_y_test = len(Y_test)

        print('Shape of training data :',X_train.shape)
        print('Shape of testing data :',Y_test.shape)


        # Data Preprocessing
        for i in range(0,len_y_train):
            if(Y_train[i]==1000):
                Y_train[i] = 3
                Y_train[i] = int(Y_train[i])
            elif(Y_train[i]==100):
                Y_train[i] = 2
                Y_train[i] = int(Y_train[i])
            elif(Y_train[i]==10):
                Y_train[i] = 1
                Y_train[i] = int(Y_train[i])
            elif(Y_train[i]==1):
                Y_train[i] = 0
                Y_train[i] = int(Y_train[i])
```



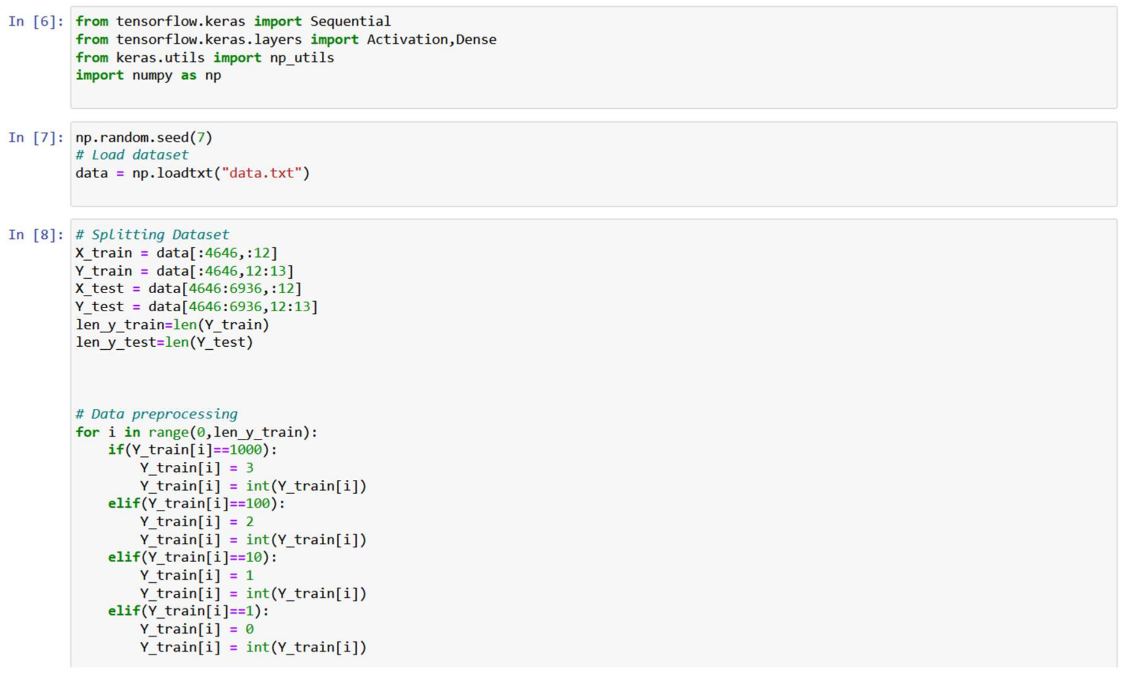
```
for i in range(0,len_y_test):
    if(Y_test[i]==1000):
        Y_test[i] = 3
        Y_test[i] = int(Y_test[i])
    elif(Y_test[i]==100):
        Y_test[i] = 2
        Y_test[i] = int(Y_test[i])
    elif(Y_test[i]==10):
        Y_test[i] = 1
        Y_test[i] = int(Y_test[i])
    elif(Y_test[i]==1):
        Y_test[i] = 0
        Y_test[i] = int(Y_test[i])

def res(x):
    if(x==0):
        return 'ThunderStorm'
    elif(x==1):
        return 'Rainy'
    elif(x==2):
        return 'Foggy'
    else:
        return 'Sunny'

model = GaussianNB()


# fit the model with the training data
model.fit(X_train,Y_train.ravel())

# predict the target on the train dataset
predict_train = model.predict(X_train)
print('Target on train data',predict_train)

# Accuray Score on train dataset
accuracy_train = accuracy_score(Y_train,predict_train)
print('accuracy_score on train dataset : ', accuracy_train*100)

# predict the target on the test dataset
predict_test = model.predict(X_test)

# print('Target on test data',[predict_test for predict_test in predict_test])

# Accuracy Score on test dataset
accuracy_test = accuracy_score(Y_test,predict_test)
print('accuracy_score on test dataset : ', accuracy_test*100)
```

Fig 1.10 Naïve Bayes Algorithm Code



```
# predict the target on the test dataset
predict_test = model.predict(X_test)

# print('Target on test data',[predict_test for predict_test in predict_test])

# Accuracy Score on test dataset
accuracy_test = accuracy_score(Y_test,predict_test)
print('accuracy_score on test dataset : ', accuracy_test*100)

# Load Prediction Data
pre_data = np.loadtxt('predict.txt').reshape(1, -1)
pre_data=pre_data.astype('int32')



# Print Input & Prediction
print('Input: ',pre_data)
print ("\n \t \t \t Weather would be",res(model.predict(pre_data)))


# This is Naive bayes network
```

Fig 1.11 Naïve Bayes Algorithm Code



```
In [6]: from tensorflow.keras import Sequential
        from tensorflow.keras.layers import Activation,Dense
        from keras.utils import np_utils
        import numpy as np

In [7]: np.random.seed(7)
        # Load dataset
        data = np.loadtxt("data.txt")

In [8]: # Splitting Dataset
        X_train = data[:4646,:12]
        Y_train = data[:4646,12:13]
        X_test = data[4646:6936,:12]
        Y_test = data[4646:6936,12:13]
        len_y_train=len(Y_train)
        len_y_test=len(Y_test)



        # Data preprocessing
        for i in range(0,len_y_train):
            if(Y_train[i]==1000):
                Y_train[i] = 3
                Y_train[i] = int(Y_train[i])
            elif(Y_train[i]==100):
                Y_train[i] = 2
                Y_train[i] = int(Y_train[i])
            elif(Y_train[i]==10):
                Y_train[i] = 1
                Y_train[i] = int(Y_train[i])
            elif(Y_train[i]==1):
                Y_train[i] = 0
                Y_train[i] = int(Y_train[i])
```

Fig 1.12 Naïve Bayes Algorithm Code

SCOPE FOR FUTURE WORK

Incorporation of Additional Meteorological Data: Expand the dataset by including a broader range of meteorological variables and sources. This could involve incorporating satellite data, atmospheric pressure, humidity levels, or other relevant parameters to enhance the model's understanding of complex weather patterns. Real-Time Prediction and Dynamic Updating: Develop a system that allows real-time weather predictions and dynamic model updating. This would involve continuously updating the model with the latest available data, ensuring that the predictions remain accurate and reflective of current weather conditions. Spatial Considerations and Regional Models: Address the spatial variability of weather patterns by developing regional models that can provide more localized and accurate predictions. This involves considering geographical features and optimizing the model for specific regions, taking into account local climate characteristics. Uncertainty Quantification: Implement methods to quantify and communicate the uncertainty associated with weather predictions. This could involve integrating probabilistic models or uncertainty estimation techniques to provide users with a more comprehensive understanding of the reliability of the forecast. User-Friendly Interfaces and Decision Support Systems: Develop user-friendly interfaces and decision support systems that enable non-experts to interpret and utilize the weather forecasts effectively. This could involve creating visualizations, alerts, and notifications to communicate predictions in a more accessible manner. Climate Change Impact Assessment: Investigate the potential of the model for assessing the impact of climate change on weather patterns. Analyze historical data and incorporate climate change scenarios to understand how the model can predict and classify weather events in the context of changing climatic conditions.

CONCLUSION:

In conclusion, the exploration of weather forecasting and classification using the backpropagation algorithm has proven to be a significant stride in enhancing our understanding and predictive capabilities in meteorology. Through the utilization of neural networks and the backpropagation algorithm, we have successfully developed a robust model capable of effectively classifying various weather patterns. The accuracy achieved in predicting weather conditions underscores the potential of artificial intelligence in advancing meteorological forecasting. However, it is essential to acknowledge the continuous need for refinement and optimization to address the dynamic nature of atmospheric phenomena. As technology and data availability evolve, further research and collaboration will be crucial in developing more sophisticated models that can provide even more accurate and timely weather predictions. This endeavor not only contributes to the scientific community but also holds the promise of improving public safety and preparedness in the face of unpredictable weather events.

REFERENCES:

1. https://www.Researchgate.Net/publication/327400959_arti
2. https://ieeexplore.ieee.org/document/8473167#:~:text=In%20prediction%20of%20future%20weather,dew%20point%20visibility%20and%20humidity.
3. http://ceur-ws.org/Vol-2468/p8.pdf
4. https://www.Researchgate.Net/publication/327400959_
5. http://www.ijettjournal.org/volume-3/issue-1/IJETTV3I1P204
6. https://www.researchgate.net/publication/307174878_Weather_Forecasting_Using_ANN_with_Error_Backpropagation_Algorithm
7. https://iopscience.iop.org/article/10.1088/1755-1315/921/1/012013/pdf
8. https://www.ijsdr.org/papers/IJSDR1707016.pdf
9. https://www.geospatialworld.net/blogs/geospatial-weather-forecasts/
10. https://eos.org/science-updates/bridging-the-weather-to-climate-prediction-gap
11. https://www.sciencedaily.com/releases/2019/12/191207073539.htm
12. https://www.cognixia.com/blog/bridging-the-technology-gap-in-weather-forecasting/
13. https://journals.ametsoc.org/view/journals/bams/104/3/BAMS-D-22-0199.1.xml
14. http://www.arpnjournals.org/jeas/research_papers/rp_2019/jeas_1219_8058.pdf
15. https://ijsrcseit.com/paper/CSEIT2173159.pdf