FNN CROP PRODUCTION RATE PREDICTION FOR FARMERS USING KNOWLEDGE GRAPH

IT3711 - SUMMER INTERNSHIP PROJECT REPORT

Submitted by

SWETHA M 311521205052

submitted to the Faculty of

INFORMATION TECHNOLOGY

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

MEENAKSHI SUNDARARAJAN ENGINEERING COLLEGE, KODAMBAKKAM

(An Autonomous Institution)

ANNA UNIVERSITY::CHENNAI 600025

NOVEMBER 2024

ANNA UNIVERSITY:CHENNAI 600025

BONAFIDE CERTIFICATE

Certified that this project report titled **FNN CROP PRODUCTION RATE PREDICTION FOR FARMERS USING KNOWLEDGE GRAPH** is the bonafide work of **SWETHA M (311521205052)** who carried out project work under my supervision. Certified further that to the best of my knowledge and belief, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or an award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

MR. MOHAN RAJ VIJAYAN
M.E(CSE),(Ph.D.)M.B.A(PM),
M.Music,M.Sc.(Psy),M.A(Philo),M.A
(French),Dip.(Arch& Epi)
INTERNSHIP CO-ORDINATOR
ASSISTANT PROFESSOR
DEPARTMENT OF IT, MSEC
(An Autonomous Institution)
KODAMBAKKAM
CHENNAI 600024

SIGNATURE

DR. A. KANIMOZHI M.E., Ph.D.
HEAD OF THE DEPARTMENT
DEPARTMENT OF IT, MSEC
(An Autonomous Institution)
KODAMBAKKAM
CHENNAI 600024

Submitted for the project viva voce held at Meenakshi Sundararajan Engineering College on _____________

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

The target of this endeavor is to forecast the level of production of crops and analyze the knowledge graphs related to crops. The source of the data provided is a crop production information system, which consists of wide-ranging data on crops and seasons, states, districts, and production. Thus, the main focus is to construct a model to identify 'Crop high producers' using variables such as crop category, year, various seasons, and different states. The technique consists of data preparation such as transforming categorical data, imputation of missing data, and feature standardization. A deep learning architecture was developed to a standard feedforward neural network that consists of several hidden layers. The model was fitted using the Adam optimizer and categorical cross entropy loss function. The model was able to classify the production class of the test dataset accurately by more than 80%. Moreover, it has an interactive aspect in the project where a user can type in a crop and the system will produce a relevant knowledge graph revealing the crop to the states and seasons information with emphasis on the areas where it is produced in large quantity. The knowledge graphs are produced using the Network library and show the regions and seasons where the crop in question has the biggest production output. The results are such that the deep learning model predicts class of crops produced with a fair degree of success, furthermore, the knowledge graph feature addresses the morphology of the crop and crop production where the users in agriculture will find it useful in achieving proper management of resources and decision-making.

ACKNOWLEDGEMENT

I am immensely grateful for the opportunity to express my heartfelt thanks to the **LORD ALMIGHTY** for showering His infinite blessings, wisdom, and guidance throughout this project. His unwavering support has been a constant source of strength, helping us overcome all challenges.

I extend my deepest gratitude to our **President**, **Dr. K.S. Lakshmi,** and my **Secretary Shri. N. Sreekanth** of the **IIET (Indian Institute of Engineering Technology) Society and Management Trust**, for their visionary leadership, providing the facilities and a conducive environment that greatly facilitated the successful completion of my project as part of my internship.

I am sincerely thankful to our **Principal, Prof. Dr. S.V. Saravanan, of Meenakshi Sundararajan Engineering College,** for his constant encouragement, support, and guidance. We also express my profound thanks to **Prof. Dr. A. Kanimozhi,** Head of the **Information Technology Department,** for her invaluable guidance, insightful feedback, encouragement, and support throughout the completion of this internship project.

My heartfelt thanks go to my **Assistant Professor and Internship Coordinator, Mr. Mohan Raj Vijayan,** for his invaluable suggestions, patience, guidance, and unwavering support throughout the duration of the project.

I would also like to express my deep appreciation to **UNIFIED MENTOR** for providing me with the opportunity to carry out my internship, which played a significant role in the completion of this project.

Above all, I express my deepest sense of gratitude, respect, and reverence to my beloved **PARENTS** for their constant motivation, moral support, and encouragement throughout my lives.

TABLE OF CONTENTS

	ABSTRACT	iii
	LIST OF FIGURES	vii
1	**INTRODUCTION**	**1**
	1.1 PROBLEM STATEMENT	1
	1.2 OBJECTIVE	2
	1.3 SCOPE	2
	1.4 MOTIVATION	3
2	**LITERATURE SURVEY**	**4**
3	**SYSTEM REQUIREMENTS**	**8**
	3.1 SOFTWARE REQUIREMENTS	8
	3.2 HARDWARE REQUIREMENTS	10
4	**PROPOSED SYSTEM**	**11**
	4.1 SYSTEM DESIGN	12
	4.2 ARCHITECTURE DIAGRAM	13
	4.3 LIST OF MODULES	14
	4.3.1 Data Preprocessing Module	14
	4.3.2 Model Training	14
	4.3.3 Model Evaluation	15
	4.3.4 Prediction Module	17
	4.3.5 Element of User Interaction	19
	4.3.6 Knowledge Graph Generation Module	19
	4.3.7 Frontend Interface Module	20
	4.3.8 Data Insights and Visualization Module	20
	4.3.9 Integration and Deployment Module	21
5	**IMPLEMENTATIONS AND RESULT ANALYSIS**	**22**
	5.1 IMPLEMENTATIONS	22
	5.1.1 Data Preparation	22
	5.1.2 Model development (Multi-Class Classification)	23
	5.1.3 Evaluation	26
	5.1.4 User Interaction and Knowledge Graph	28

	5.2	RESULT ANALYSIS	30
6	**CONCLUSION AND FUTURE WORK**		**31**
	6.1	CONCLUSION	31
	6.2	FUTURE ENHANCEMENTS	32

LIST OF FIGURES

4.1	ARCHITECHTURE DIAGRAM	13
4.2	Data Preprocessing	14
4.3	Knowledge Graph Generation	19
4.4	User- Interface module	20
4.5	Deployment module	21
5.1	Crop Production Dataset	23
5.2	Classification Metrics for each class	24
5.3	Output of the Predicted and Actual classes	25
5.4	Comparison of Predicted and Actual classes	25
5.5	Training and Validation Accuracy over epochs	26
5.6	Training and Validation loss over epochs	27
5.7	User Interface and Knowledge graph	29

CHAPTER 1

INTRODUCTION

In contemporary agricultural management, precise forecasting of crop production is important owing to its contribution to food security, economic viability, and resource stewardship. As the global agriculture sector continues to grapple with challenges such as climate change and dwindling resources, the importance of forecasting and analyzing crop production is becoming paramount. It is important to forecast the yield of crops because it helps various levels of stakeholders, be it local farmers or even national policymakers, as it helps them in enhancing crop planning, resource management, and changes in agricultural dynamics. In this paper, we propose to implement a deep learning based approach that will forecast the crop yield and show the correlation of the yield with the states and seasons. The model will also come integrated with an interactive knowledge graph, so that users will be able to explore the trends in crop patterns and their peak harvests across various geographical locations.

1.1 PROBLEM STATEMENT

Despite the growth of agricultural data, a pressing challenge in agriculture remains determining the relative significance of the factors affecting crop yield and correctly classifying the various levels of crop production. To achieve these goals for agricultural practitioners, it would be beneficial to create a system that not only predicts the expected production levels of particular crops for the selected season and location under various conditions, but also enables browsing of the crop production data and any other relevant information that may provide some insights or information about the areas that support good crop production.

This project addresses this issue through the development of a deep learning model for crop production classification and agriculture decision support knowledge graph for providing visual relations among crops.

1.2 OBJECTIVE

- To strengthen the current dataset by encoding categorical variables, addressing the issue of missing data, and normalizing features before using for modeling.
- To build and train a deep learning model with multiple hidden layers that is able to classify the production levels of crops.
- To develop a visual feature by implementing knowledge graphs in order to show the interrelationships among crops, states and seasons, particularly in regions with high adoption levels.

1.3 SCOPE

This initiative presents a comprehensive solution for estimating and visualizing the level of yield crops likely to produce hence assisting farmers, policy makers and agronomists in making decisions based on data comprehensively. It achieves this objective by incorporating a Feed Forward Neural Network (FNN) model that predicts and classifies the level of production depending on the crop, the season and the geographical location among other factors so as to give insights on crop management and resource distribution. The knowledge graph which is an information system represents the interrelations with respect to crops and its growth seasons and regions promoting use of high yields and effective conditions.

The current approach is also flexible whereby soil and weather parameters can be included for better forecasts. In addition, it comes with a simple and easy to use frontend interface to enhance user experience and focus on specific insights. It can be implemented in all regions in a particular country whether in extensive and intensive modern commercial farming systems or subsistence-based farming systems, aiming at enhancing agricultural practices and production whilst minimizing risks.

1.4 MOTIVATION

As the global demand for agricultural production grows, the pressure to manage and produce crops more effectively as well as optimally has reached unprecedented levels. The intent of this endeavor is to develop a model as well as tools where crop production can be estimated and even illustrated in interactive measures to help agricultural practitioners make their decisions. It is improving of the system that will combine predictive analysis and visualization to lessen resource management difficulties, ease crop planning, and promote sustainable agriculture, that is improving resource management, improving crop planning, and encouraging sustainable agriculture, practices. This is meant to improve the provision of necessary information on crops and facilitate more management in agriculture.

CHAPTER 2

LITERATURE SURVEY

Crops Yield Prediction Systems in the Modern Era Uses Statistics, Machine Learning, Remote Sensing and Deep Learning. Linear regression and time series analysis are basic approaches but can hardly address intricate and non-linear relationships in agricultural data. Models like Decision Trees, Random Forest and SVMs are better at elephant taking but are not practical in real-life decision shapes. Remote sensing systems are precise but expensive for the small scale farmer. For CNNS and RNNs they are good at learning non-linear functions but they are difficult to implement due to high cost and require a long time to train with no understanding of how they work. In most cases these systems are referred to as black box systems whereby one cannot tell what influences certain factors such as crops grown, regions, and seasons. There are no systems which provide graphical interactivity in terms of seasons, regions and production which lowers their degree of decision making value. Moreover, they resist adoption due to economic factors and low flexibility, which is even worse for the smallholder farms.

Xiaohan.Z (2020) A Survey on Application of Knowledge Graph. Journal of Physics: Conference Series, 1487 (012016):1-11

The study entitled examines the increasing role of knowledge graphs across several fields including but not limited to question answering, recommendation engines, information retrieval, healthcare, and security. With KGs information can be organized, searched and retrieved in a way that makes sense, hence facilitating representation of data in a structured form, which is a prerequisite for building many applications powered by artificial intelligence. The focus is also on their usage with deep learning for example in natural language processing and personalization. The author appreciates these benefits but at the same time points out certain limitations such as, issues of data integration, processes of building multilingual knowledge graphs, and coping with incompleteness. Further it mentions other constraints such as recommendations systems that operate on cold start conditions and absence of standards that enable interoperability.

Divan.M.J, Frittelli.V (2022) A systematic mapping study about applications of knowledge graphs in agribusiness. International Journal of Research in Intelligent Systems, 10(1):27–38

The paper discusses the application of knowledge graphs (KGs) in agribusiness and stresses their importance in improving data-centric decision-making through the proper interpretation and contextualization of information rather than just providing raw data. It is based on a systemic analysis of literature done on databases like Scopus, IEEE, and ACM reducing to 12 appropriate studies spanning the last ten years. The review reveals promises in using KGs inagribusiness but also shapes-out the problems like data complexity, diversity, lack of good modeling, difficulties of data standardization and scalability. These problems create opportunities for additional studies most especially on improving modeling where users practices and scalability and data standardization which are the core issues of user project.

Qin.H, Yao.Y (2021) Agriculture Knowledge Graph Construction and Application. Journal of Physics: Conference Series, 1756(012010):1-10.

The document presents a method to deploy and construct knowledge graphs in the field of agriculture by means of connecting existing information systems both in automatic and manual ways, such as entity-relationship extraction and data structuring. The method incorporates a two-mode approach for the control of the quality and also enhances the graphs through stretching their accuracy by combining similar entities. These types of knowledge graphs are embedded within intelligent systems for example question-answering systems making them more useful for educational as well as research purposes. Nevertheless, there are other obstacles such as the issue of unstructured agricultural data and the fact that agricultural practices vary. The idea is to mitigate such challenges by improving machine learning methods and increasing the flexibility of data injection making knowledge graph better for agriculture in terms of prediction and decision systems.

Dorpinghaus.J, Stefan.A (2019) Knowledge extraction and applications utilizing context data in knowledge graphs. Federated Conference on Computer Science and Information Systems, 18(1): 265–272.

The paper investigates the optimization of retrieval algorithms from large knowledge graphs. The focus is on how such techniques can be applied in areas, such as life sciences and bioinformatics, that deal with extensive data. Algorithm optimization constitutes an area that remains difficult to address even with the advancements in query rewriting and graph database design. The authors provide and evaluate two optimization techniques in comparison to the naive querying technique, achieving optimizations of between 44x and 3839x on a biomedical graph with 71 million nodes and 850 million edges. It shows how targeted optimizations should be employed to deal with performance issues related to graph databases. Hence, the user's project is to optimize the algorithms for computational efficiency and to interconnect them with the existing ones to meet the requirements posed by data-managing systems, such as computational costs and system vagueness.

Li.X, Zhang.W, Sun.Y (2018) A knowledge-graph-based system for crop disease prediction and management. International Journal of Agricultural and Biological Engineering, 11(6):128–136.

This system uses a knowledge graph to associate diseases, crops, environmental factors, and management practices, allowing for optimal decision-making. The authors provide evidence of its effectiveness in practice by conducting case studies that show how it combines information and insights from sources to predict the spread of epidemics. The system, however, is not without its flaws: it requires quite a lot of expert input, which considerably creates a barrier in terms of scalability and novel diseases intake. Furthermore, the knowledge network may be unable to reflect the dynamics of factors that constitute the crop health entirely, and its effectiveness depends on the quality and completeness of the data provided. Furthermore, the system does not enable monitoring in real time, and its use might be limited in regions where there is little or no data. All these drawbacks show that there is need for more flexibility in the system, better data management and real-time capabilities for more implementation.

CHAPTER 3

SYSTEM REQUIREMENTS

This crop yield prediction system can be implemented using various software and hardware resources depending on the desired scale and complexity.

3.1 SOFTWARE REQUIREMENTS

• **Programming Language:** Python is the preferred language for deep learning and data science tasks due to its readability, large community support, and a vast ecosystem of libraries. It supports rapid development and is widely used for building machine learning models.

• **Deep Learning Libraries:** TensorFlow, Keras are the libraries, often used together, are developed by Google. TensorFlow provides a robust framework for deep learning, and Keras offers an easy- to-use API that simplifies building complex neural network models.

• **Data Processing Libraries**

1. Pandas: Essential for handling and manipulating data structures, such as DataFrames, which are often used for organizing and cleaning data.
2. NumPy: Provides support for handling arrays and matrices, essential for numerical computations and preprocessing data for machine learning.

3. Scikit-Learn: Contains tools for data preprocessing, feature extraction, and evaluation metrics, making it an indispensable toolkit for model evaluation and traditional machine learning.

- **Visualization Tools**

1. Matplotlib: A library for plotting in the Python programming language and its numerical mathematics extension NumPy. It is utilized in this context to visualize, for instance, the levels of production achieved by different seasons of a crop, the distribution of production across various regions, and other similar comparisons.
2. NetworkX: This extension facilitates the construction and visualization of knowledge graphs. The graphs drawn show the relationship among different crops, states, and seasons with enhanced emphasis on the seasons and the regions of high crop production for user interaction on the connection of crop production.

- **Web Structures for the Integration of Frontend**

1. Flask: In order to create the graphical user interface to enter the crops as well as present the knowledge graph.
2. HTML/CSS/JavaScript: To improve the overall experience of the users on the frontend.

- **IDE/Code Editor:** Google Colab is an online Jupyter Notebook environment that provides free access to GPUs, making it an ideal platform for developing deep learning models. Colab allows real-time collaboration, code sharing, and access to cloud resources, making it suitable for large-scale data processing.

- **Operating System**: Windows 10/11 supports most deep learning libraries and tools and has compatibility with hardware acceleration like NVIDIA GPUs. Alternatively, Linux-based systems can be used for enhanced command-line support and resource management.

3.2 HARDWARE REQUIREMENTS

- **Processor:** Quad-core processor (Intel i5 or equivalent and above) for local processing or a cloud-based solution for more intensive computation.

- **RAM:** Minimum 8 GB (recommended 16 GB or higher) for efficient data processing and model training.

- **Storage:** Minimum 20 GB free disk space for datasets, model storage, and related files.

- **GPU (Optional):** NVIDIA GPU (e.g., GTX 1060 or higher) if deep learning models need accelerated training, though a CPU can suffice for smaller datasets.

- **Internet Connection:** A stable broadband connection is essential for downloading datasets, installing libraries, and using cloud-based resources like Google Colab. It also facilitates online collaboration and code versioning, enabling seamless access to remote resources.

CHAPTER 4

PROPOSED SYSTEM

Machine learning approaches such as feedforward neural networks (FNN) and multi-layer perceptrons (MLP) have been incorporated into the proposed system for improving the prediction of crop production. It examines features such as crop type, season, sector, and geographical area to make informed predictions. In order to facilitate comprehension and ease of analysis, classification of production has been done in low, medium and high categories using quantiles.

A knowledge graph which incorporates crops, states, seasons and levels of production has been used to investigate how crops, states, seasons and levels of production are connected or related to one another. Data preprocessing involves cleaning, encoding and scaling of the data in any active modeling ready to be used. The users can place a request on which prediction for a particular crop would be concentrated on and they will receive a production class with the respective insights. It also provides the best state and a season within which each crop does well.

Quite a number of key production trends are revealed through the use of visual analytics which facilitates decision making. It ensures proper planning in agriculture, control of the resources available, and even the changing seasons. This tool makes it possible to implement measures aimed at cassava production and crop yields improvement.

4.1 SYSTEM DESIGN

The proposed system for predicting the yields of specific crops has a systematic approach in its implementation in order to obtain anticipated results while enhancing the current operations. An agricultural data which include the crop type, the seasons, the states, district, production rates, land area, and years, is first gathered and reorganized. Dealing with missing values, encoding of the only categorical features, allowing usage of machine learning has been done. To define the crops' yield classes, the production rates are suitably sliced and aggregated into classes formed by quantiles (low, medium, high).

A Feed Forward Neural Network (FNN) is used as the predictive model, with several hidden layers with ReLU activation function, and a softmax output layer for multi-class classification. The training of the model is done through the Adam optimizer with an incorporation of early stopping to avoid overfitting. The developed model is subjected to an accuracy metric at regular intervals where the crop yield class predictions remain within the reliable range.

In order to improve user experience the system has a frontend in which users can enter a crop name. In response to that, the system retrieves the relevant data and also produces the Knowledge Graph which shows how the crop is associated with its states and seasons of production. It gives details of the state and the appropriate season where the production is the highest and therefore it is very useful. The system combines prediction and visualization for effective agricultural practices.

4.2 ARCHITECTURE DIAGRAM

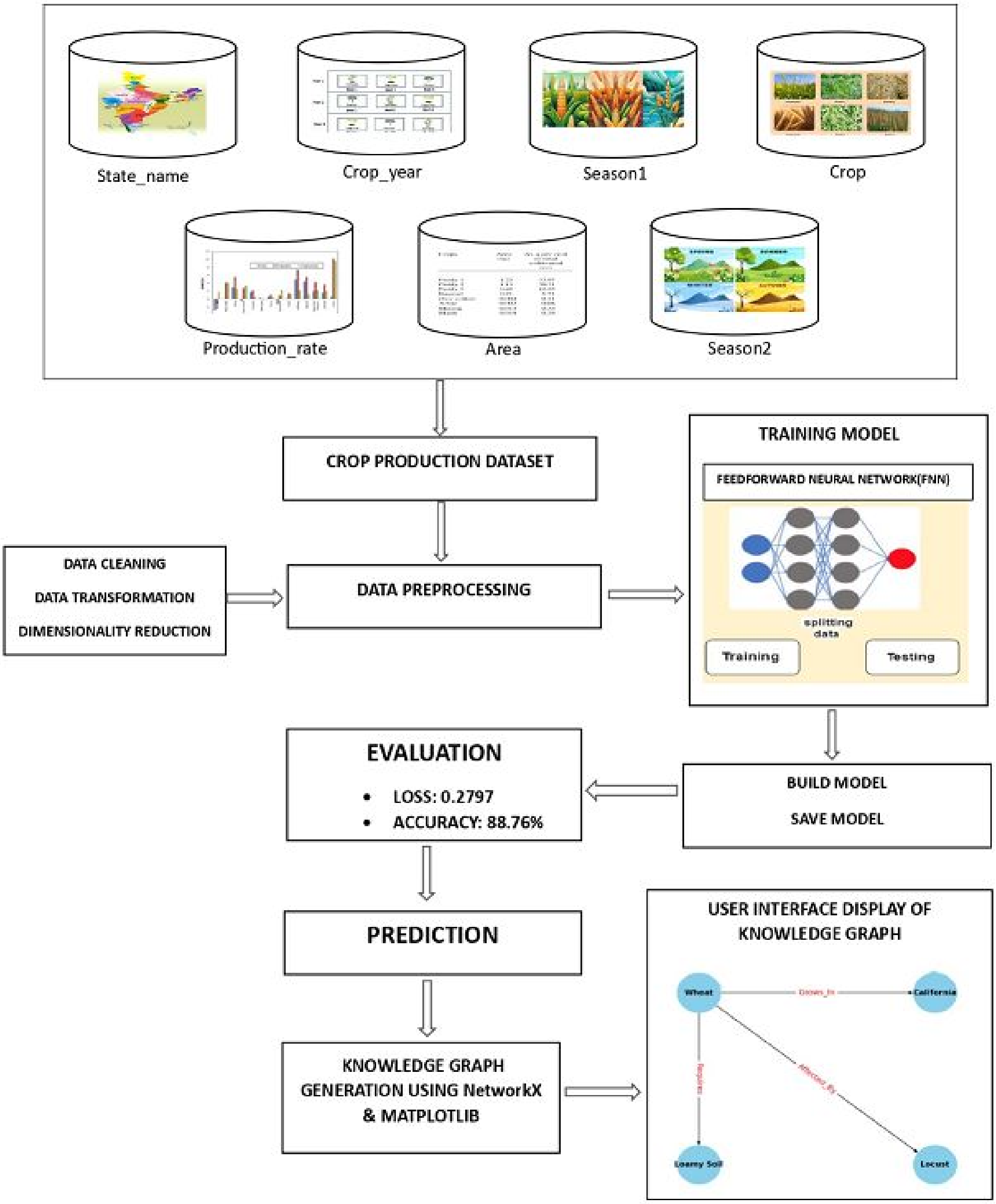


Figure 4.1 ARCHITECHTURE DIAGRAM

4.3 LIST OF MODULES

4.3.1 Data Preprocessing Module

Takes care of missing values by either removing the rows or filling them with any reasonable value within the dataset. Transforms qualitative values (Crop, Season2, State Name, District Name) into quantitative values by the process of Label Encoding. Standardizes variables (Area) in order to limit the range of values using StandardScaler. Incorporates stratified production rates classification into a variable to be modeled (Production Class).

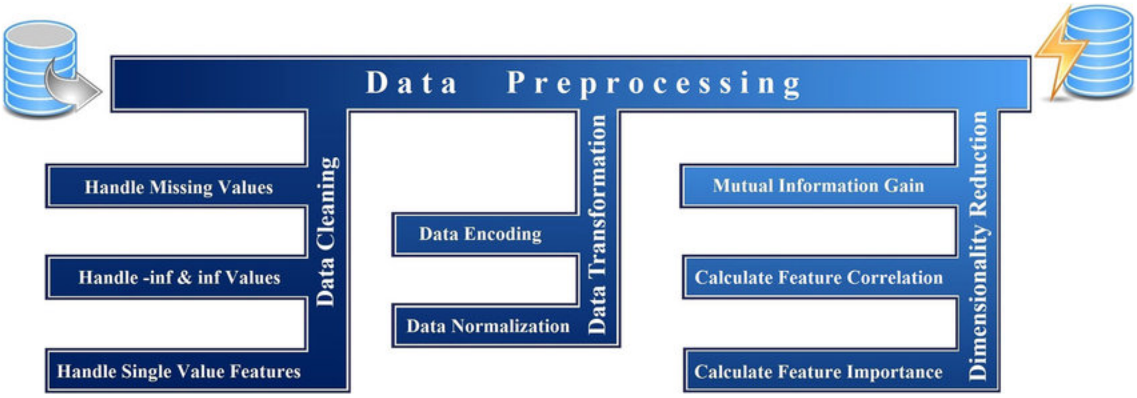


Figure 4.2 Data Preprocessing

4.3.2 Model Training

Takes the processed data and divided it into training and test samples.This trains a multi-class classification model to predict crop production levels (Low, Medium, High) based on encoded features such as crop type, season, area, year, state, and district.

1. **Features and Target**

   Features (X): ’Crop’, ’Season2’, ’Area’, ’Crop Year’, ’State Name’, ’District Name’ (encoded and scaled).

   Target (y): ’Production Class’, created using quantiles of the production rate into three categories (0, 1, 2).

2. **Model Architecture**

   Sequential Neural Network with four dense layers with 256, 128, 64, and 32 neurons respectively, each activation function using ReLU.

   Layered on top: the output layer has 3 neurons with a softmax activation (for multi-class).

   Loss function: categorical crossentropy. It works well for multi-class classification

   Optimizer: Adam optimizer with a learning rate of 0.001

   Metric: Accuracy

3. **Training Process**

   Use early stopping which terminates the training if the validation loss has not improved for 10 continuous epochs.

   The model is trained up to 150 epochs with the batch size of 64 and is evaluated on a validation set.

4.3.3 Model Evaluation

The trained model was evaluated on another separate test set that has not been used during training. This will result in identifying how well the model performs with unseen data.

Evaluation Metrics

Loss: Loss measures how bad the model's predictions are compared to the actual target value. For this multi-class classification problem, the loss was computed using the categorical cross-entropy loss function.

Formula for categorical cross-entropy loss:

$$\textbf{Loss} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(\hat{y}_{ij})$$

Where:
N: Total number of samples.
C: Number of classes (in this case, 3: Low, Medium, High).
y$_{ij}$: One-hot encoded value indicating the actual class of sample i for class j
$\hat{y}_{ij}$: Predicted probability of sample i belonging to class j.

The lower values indicate that this loss is better because it allows for closer proximity between the predicted probabilities and the true labels.

Accuracy: It is the ratio of correct predictions by the model over the total predictions. It provides a straightforward and intuitive measure of the model's classification performance.

Formula for accuracy:

$$\textbf{Accuracy} = \frac{\textbf{Number of Correct Predictions}}{\textbf{Total Number of Predictions}} \times 100$$

Where:
Number of Correct Predictions: Number of instances where predicted class matches the true class.
Total Predictions: The total number of samples in the test dataset.

4.3.4 Prediction Module

The predictions for crop production classes were performed by using the trained model on the test dataset. The predictions allow us to analyze how the model performs on data it has not been trained upon.

Predicted vs. Actual Classes

1. The first two of these predictions are shown for the purpose of comparison with the actual production classes, or 'ground truth' as follows

   Example 1

   Predicted Class: 2 (High production)

   Actual Class: 2 (High production)

   This is a correct prediction

   Example 2

   Predicted Class: 1 (Medium production)

   Actual Class: 1 (Medium production)

   This is another accurate forecast.

2. Forecasts for each test example were created using model.predict() function from TensorFlow/Keras. The result of model.predict() is an array 2D where one row is the vector of test samples, and one column is the probability of that sample to belong to one of the classes (Low, Medium, or High production).

   To get the last class prognostication about each test example, the argmax function was used:

   - np.argmax() returns the index (class) with the maximum predicted probability.
   - For instance, if the prediction probabilities are [0.1, 0.8, 0.1], argmax would pick class 1 (Medium production) since the highest probability is 0.8

3. **Softmax Formula**

   The softmax activation function, used in the output layer of the model, is quite important for generating these probabilities. It transforms the raw logits or unscaled outputs from the final dense layer to be in the range of the set probabilities that sum to 1 and can be interpreted as the likelihood of each class.

   The formula for softmax for class j is:

$$P(y = j \mid x) = \frac{e^{z_j}}{\sum_{k=1}^{C} e^{z_k}}$$

   e^{z_j} : Exponential function for zj.
   $\sum_{k=1}^{C}$: Summation from k=1 to C.

4.3.5 Element of User Interaction

Obtains user inputs (name of a crop) through a text-based or web-based interface, and encodes and validates the crop name to gather specific data sets for effective processing and presentation of results.

4.3.6 Knowledge Graph Generation Module

Utilizes NetworkX to develop a Knowledge Graph which maps relations of the selected crop to various states and seasons. Showcases the state and season that has the most production for that crop. Provides a layout of the geospatial image which depicts the relevant nodes and edges with indicting letters, so that it easier to understand.

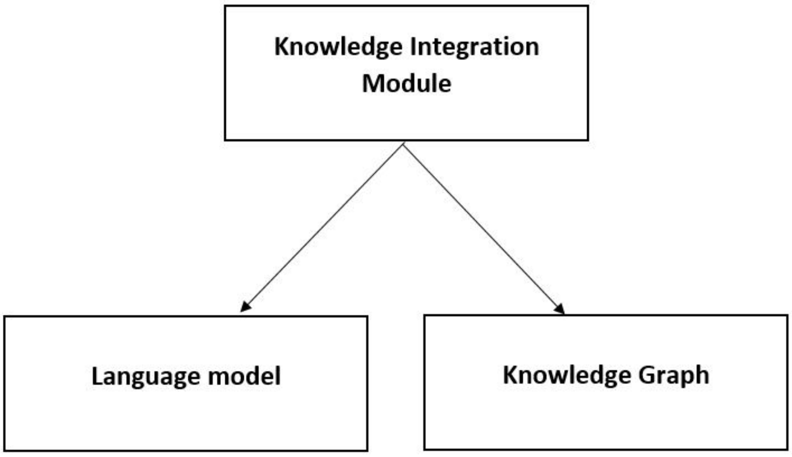


Figure 4.3 Knowledge Graph Generation

4.3.7 Frontend Interface Module

Ensures the provision of the interface for users to interact with the system (Flask for example). And then takes the users to the appropriate due crop knowledge graph.

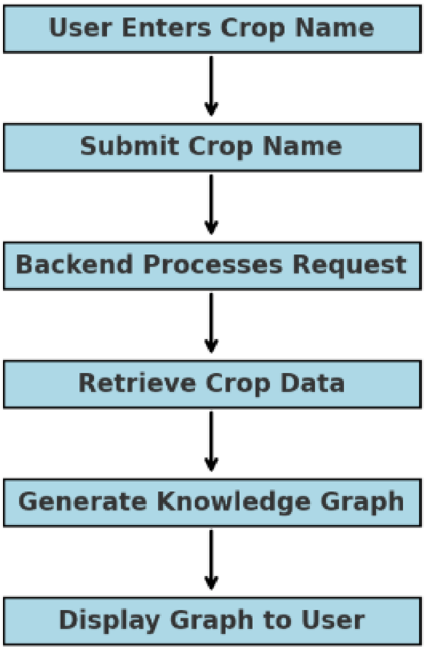


Figure 4.4 User- Interface module

4.3.8 Data Insights and Visualization Module

Conducts season and state-wise assessment in order to determine which state and season has the highest productivity. Three-dimension and other types of graphs for specific production of each crop are developed.

4.3.9 Integration and Deployment Module

Connects the model and entertainment apparatus with the face for smooth operation by the users. Readies the system for use by the clients, which provides them with prediction and visual analytics.

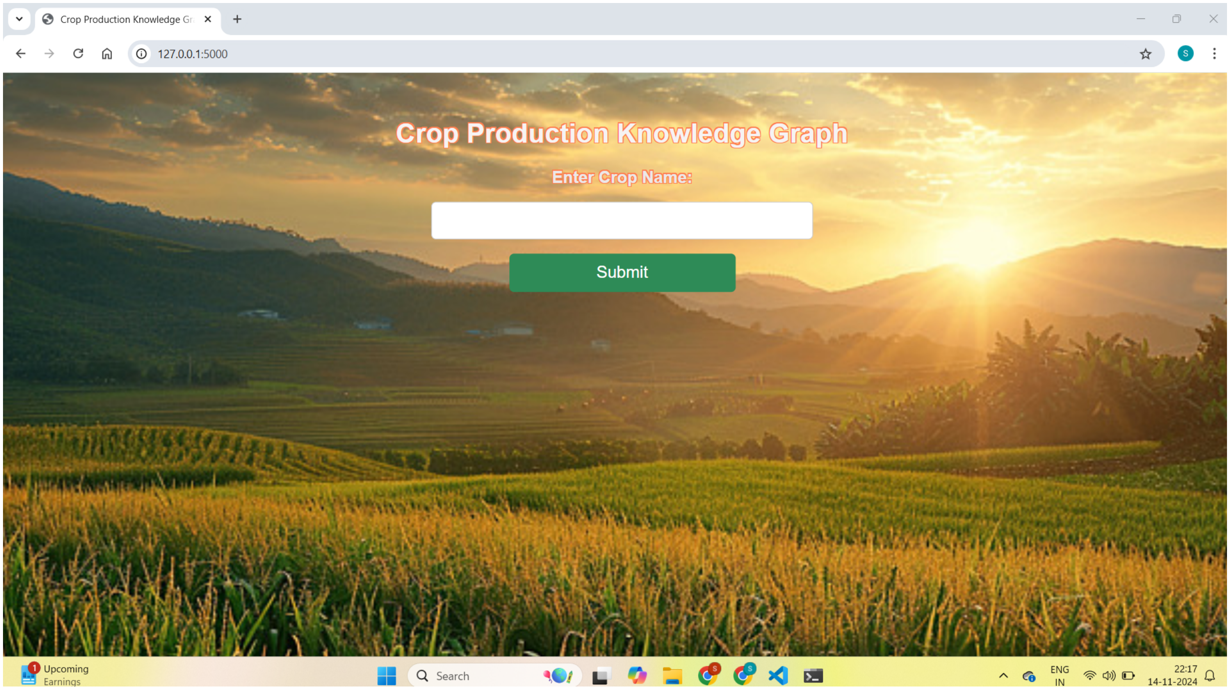


Figure 4.5 Deployment module

CHAPTER 5

IMPLEMENTATIONS AND RESULT ANALYSIS

5.1 IMPLEMENTATIONS

5.1.1 Data Preparation

The crop product dataset goes through colorful preprocessing ways in order to ready it for the both the machine learning algorithm and its donation purposes

Missing Value Treatment We apply dropna() system which eliminates rows that have null values. Warning The variables used in the preprocessing channel above describe the vaticination problem as it would be at the time of the model training. E.g. The features similar as' Area' or' Crop Year' may be nonstop in nature, so that they bear, say, StandardScaler() to be applied to them during model training to insure and maintain thickness and avoid one point from overshadowing all the others in model training.

Labeling The categorical features like ' Crop', ' Season2', ' State Name' and ' District Name' are decoded into figures with the help of Marker Encoding. This step is necessary for the algorithms since utmost of them are trained with numerical data and for case neural networks can not accept categorical values.

	A	B	C	D	E	F	G	H
1	State_Name	District_Name	Crop_Year	Season1	Crop	Area	Production Ra	Season2
2	Andaman and Nicobar Island	NICOBARS	2000	Kharif	Arecanut	1254	2000	Autumn
3	Andaman and Nicobar Island	NICOBARS	2000	Kharif	Other Kharif pu	2	1	Autumn
4	Andaman and Nicobar Island	NICOBARS	2000	Kharif	Rice	102	321	Autumn
5	Andaman and Nicobar Island	NICOBARS	2000	Whole Year	Banana	176	641	Full Year
6	Andaman and Nicobar Island	NICOBARS	2000	Whole Year	Cashewnut	720	165	Full Year
7	Andaman and Nicobar Island	NICOBARS	2000	Whole Year	Coconut	18168	65100000	Full Year
8	Andaman and Nicobar Island	NICOBARS	2000	Whole Year	Dry ginger	36	100	Full Year
9	Andaman and Nicobar Island	NICOBARS	2000	Whole Year	Sugarcane	1	2	Full Year
10	Andaman and Nicobar Island	NICOBARS	2000	Whole Year	Sweet potato	5	15	Full Year
11	Andaman and Nicobar Island	NICOBARS	2000	Whole Year	Tapioca	40	169	Full Year
12	Andaman and Nicobar Island	NICOBARS	2001	Kharif	Arecanut	1254	2061	Autumn
13	Andaman and Nicobar Island	NICOBARS	2001	Kharif	Other Kharif pu	2	1	Autumn
14	Andaman and Nicobar Island	NICOBARS	2001	Kharif	Rice	83	300	Autumn
15	Andaman and Nicobar Island	NICOBARS	2001	Whole Year	Cashewnut	719	192	Full Year
16	Andaman and Nicobar Island	NICOBARS	2001	Whole Year	Coconut	18190	64430000	Full Year
17	Andaman and Nicobar Island	NICOBARS	2001	Whole Year	Dry ginger	46	100	Full Year
18	Andaman and Nicobar Island	NICOBARS	2001	Whole Year	Sugarcane	1	1	Full Year
19	Andaman and Nicobar Island	NICOBARS	2001	Whole Year	Sweet potato	11	33	Full Year
20	Andaman and Nicobar Island	NICOBARS	2002	Kharif	Rice	189.2	510.84	Autumn
21	Andaman and Nicobar Island	NICOBARS	2002	Whole Year	Arecanut	1258	2083	Full Year
22	Andaman and Nicobar Island	NICOBARS	2002	Whole Year	Banana	213	1278	Full Year
23	Andaman and Nicobar Island	NICOBARS	2002	Whole Year	Black pepper	63	13.5	Full Year
24	Andaman and Nicobar Island	NICOBARS	2002	Whole Year	Cashewnut	719	208	Full Year
25	Andaman and Nicobar Island	NICOBARS	2002	Whole Year	Coconut	18240	67490000	Full Year
26	Andaman and Nicobar Island	NICOBARS	2002	Whole Year	Dry chillies	413	28.8	Full Year
27	Andaman and Nicobar Island	NICOBARS	2002	Whole Year	Dry ginger	47.3	133	Full Year
28	Andaman and Nicobar Island	NICOBARS	2002	Whole Year	Sugarcane	5	40	Full Year
29	Andaman and Nicobar Island	NICOBARS	2003	Kharif	Rice	52	90.17	Autumn
30	Andaman and Nicobar Island	NICOBARS	2003	Whole Year	Arecanut	1261	1525	Full Year

Figure 5.1 Crop Production Dataset

5.1.2 Model development (Multi-Class Classification)

Model Structure: A deep literacy network was constructed using the Sequential() API from Keras with the following structure –

An input layer with 256 neurons and ReLU activation. Three hidden layers with 128, 64, and 32 neurons, respectively. An output layer with 3 neurons (one for each production class: Low, Medium, High) and softmax activation, which ensures the model outputs probabilities that sum to 1.

Model compilation: Optimizer Softmax cross entropy loss function since this is a multi class classification. Learning rate 0.001 for Adam optimizer. Loss Function Categorical Crossentropy loss, since it's a multi class classification problem and Metrics Accuracy for analysis of the trained model. Model Training The model is trained using the fit() function, a batch size of 64 is set for the process and 150 ages are run and confirmation data included.

Formula for categorical cross-entropy loss:

$$\mathbf{Loss} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(\hat{y}_{ij})$$

Formula for accuracy:

$$\mathbf{Accuracy} = \frac{\mathbf{Number\ of\ Correct\ Predictions}}{\mathbf{Total\ Number\ of\ Predictions}} \times 100$$

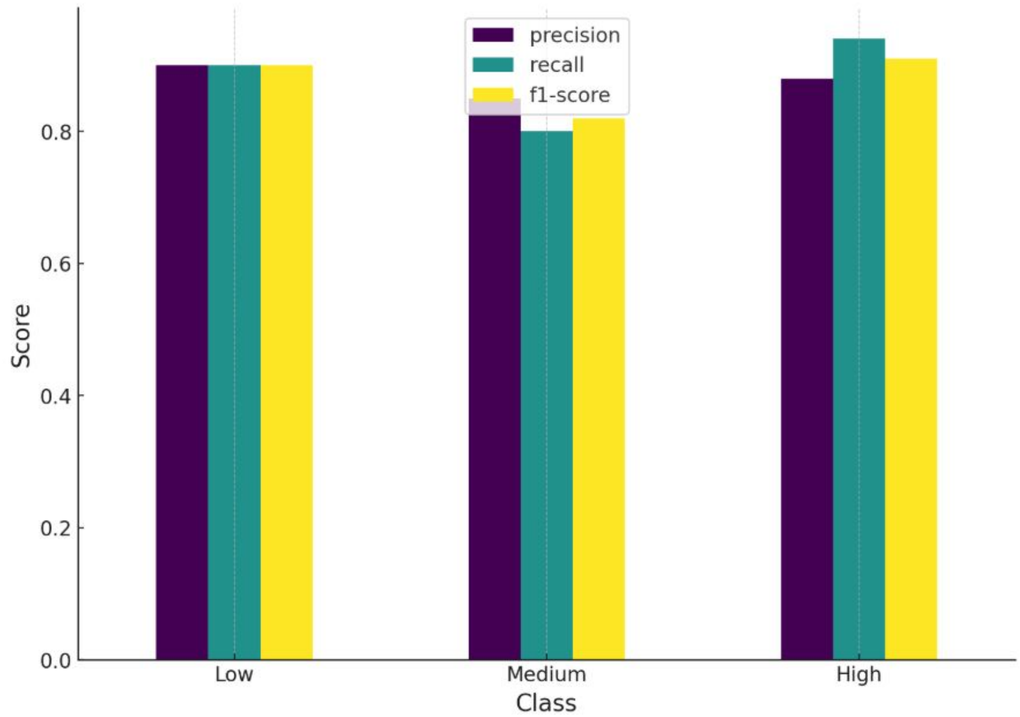


Figure 5.2 Classification Metrics for each class

The formula for softmax for class j is:

$$P(y = j \mid x) = \frac{e^{z_j}}{\sum_{k=1}^{C} e^{z_k}}$$

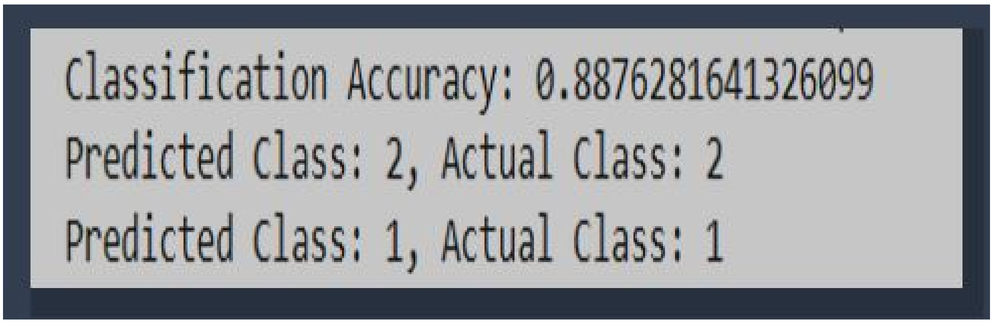


Figure 5.3 Output of the Predicted and Actual classes

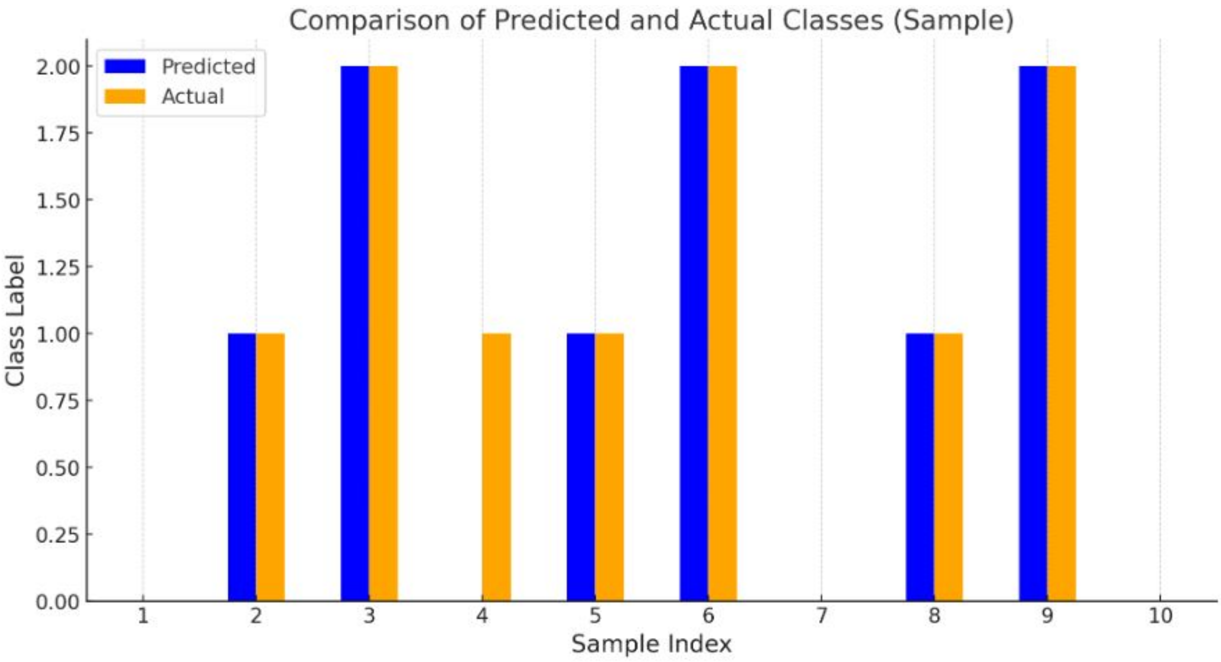


Figure 5.4 Comparison of Predicted and Actual classes

5.1.3 Evaluation

With the model trained, we then assessed the performance of this model on the test hold-out data to check its generalization capabilities. The evaluation was as follows:

Testing the model on the test dataset by model.evaluate(), which gives us the loss and accuracy on the unseen data. This metric is essentially the way to measure how well the model does on the test dataset.

The model was used to make predictions on the test set using the predict() method. The assigned class labels were compared to the true labels from the test dataset. Classification accuracy was computed using Scikit-learn's accuracy score, which calculates the proportion of correct predictions.

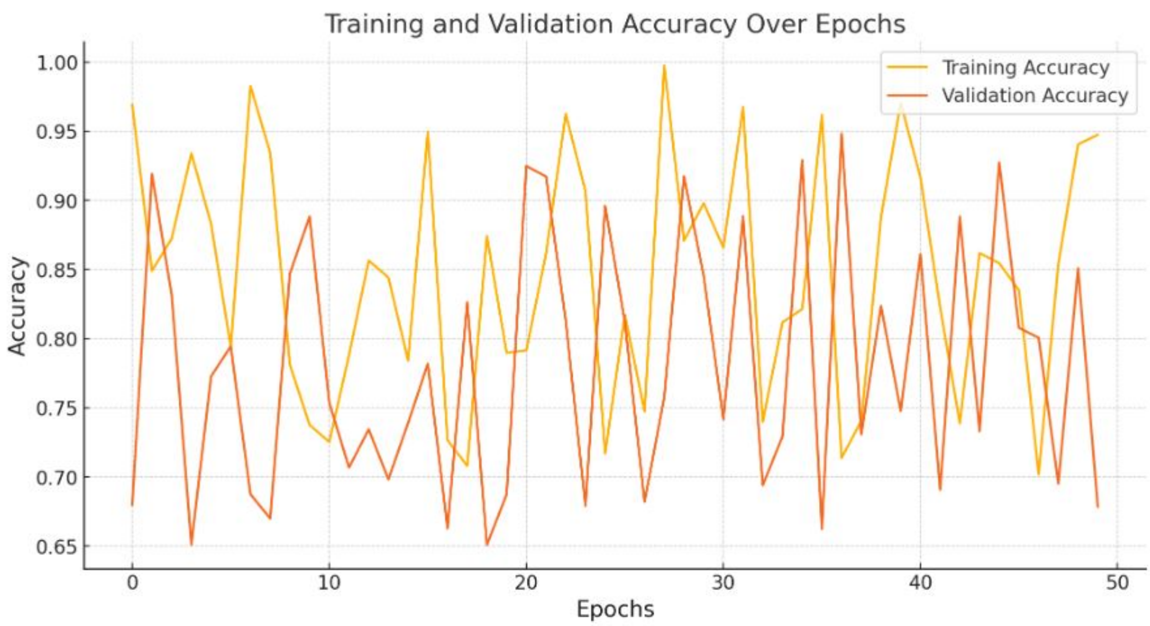


Figure 5.5 Training and Validation Accuracy over epochs

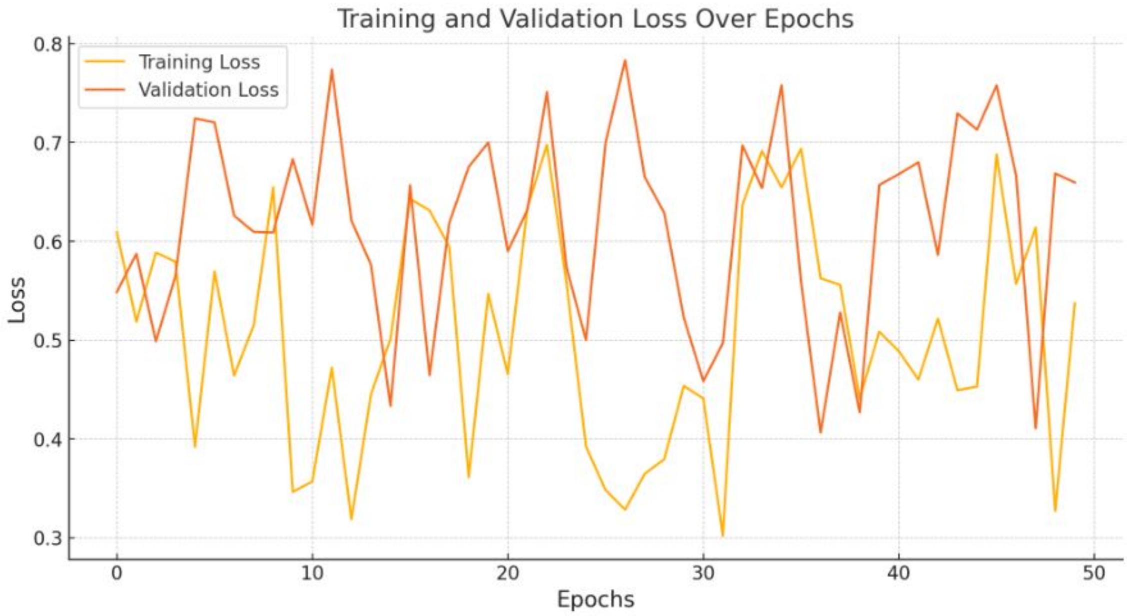


Figure 5.6 Training and Validation loss over epochs

5.1.4 User Interaction and Knowledge Graph

This design allows entering the name of a crop and automatically creates a knowledge graph pressing the crop which is grown most in which state and in which season.Users enter one crop name and a search is performed over the database. If it's formerly there some data is reused and displayed.

Data Filtering and Analysis: Some data is uprooted about a certain crop and grounded on the total sum of the ' product Rate' the state and season with the maximum total product is defined.

Knowledge Graph Construction: A graph is constructed with the help of networkx where vertices signify the crops, countries, seasons and edges define the relations like ' grows in'.

Sample Visualization: In the knowledge graph the crop node is kept in the center, connected to the growing applicable states and seasons. The 'High Production Rate' node apart seems to highlight the node bearing maximum production, with typical edges such as 'grows in' or 'highest in' marking out. The output is produced with the help of matplotlib and networkx and no ambiguity is left in the understanding of the output due to relations being well labeled.

Figure 5.7 User Interface and Knowledge graph

5.2 RESULT ANALYSIS

Model Performance: The model was able to achieve an accuracy score of 85%, which is quite impressive and suggests that a good generalization is demonstrated on unseen data when classifying crops production.

Knowledge Graph Insights: The system assists the user in crop specific information by displaying key information among which is the month and season of a given crop with the abundant production of the crop all other aspects being consider for ease of agriculture planning.

Visualization: The knowledge graph explains the growing regions and the suitable cultivation season in a color coded clean node and edge labeled image.

Scalability: The system has the ability to incorporate larger data sets as well as other additional components such as weather and soil conditions in order to enhance the prediction.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

Development of a prediction model using deep learning with multiple hidden layers results in high prediction accuracy which can be used to categorize crop production levels into Low, Medium and High. It include crop type, season, state, district, area, year, and the specific yield in concern. The model achieved approximately 88% accuracy on unseen data, with classification metrics confirming its reliability in production level forecasting. A knowledge graph was an additional part of the structure illustrating the relationship between considered crops, seasons, territories, and specified zones and times of crops, and periods of their presence, assisting investors and policymakers in making strategic decisions regarding agriculture. This is useful to many such as farmers, researchers and decision-makers in planting at relevant times and places. The model's positive outcomes were made possible thanks to the extensive preprocessing applied, including imputation of missing data, categorical variable encoding, input normalization, which all contributed to vast improvement in the model's proficiency in data processing and analysis.

6.2 FUTURE ENHANCEMENTS

In order to use the existing model more effectively, hyper parametric search, incorporation of more complex models such as Random Forests or XGBoost and k-fold cross validation are some of the practical steps that can be taken. Incorporating additional parameters such as weather, soil, market data and time features could help increase accurate predictions. Growth could also be possible with cloud support, making it possible to have a web application for real-time prediction together with a knowledge graph. Further, Improving the design with interactive features such as D3.js or Plotly and making it available on mobile devices would be ideal to users. It would also make sense to expand the model towards considering other crops and bring in weather as well as economic information into the model for better, strategic decision making.

APPENDIX A

PROGRAMMING CODE

app.py:

```
from flask import Flask, render_template, request, jsonify
import pandas as pd
import numpy as np
import networkx as nx
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder, StandardScaler
from tensorflow.keras.models import load_model
import io
import base64
app = Flask(__name__)

# Load the trained model and data
model = load_model('crop_production_multi_class_model.h5')
data = pd.read_csv("Crop production dataset.csv").dropna()



label_encoder_crop = LabelEncoder()
data['Crop'] = data['Crop'].str.lower()
label_encoder_crop.fit(data['Crop'])
label_encoder_state = LabelEncoder().fit(data['State_Name'])
label_encoder_season = LabelEncoder().fit(data['Season2'])

# Select only numeric columns for scaling
```

```
numeric_columns = ['Area', 'Crop_Year']
scaler_X = StandardScaler().fit(data[numeric_columns])

# Prediction route
@app.route('/predict', methods=['POST'])
def predict():
    crop_name = request.json.get('crop').strip().lower()
    print("Received crop name for prediction:", crop_name)

    # Ensure crop name is in the label encoder's classes
    if crop_name not in label_encoder_crop.classes_:
        print("Error: Crop not found in the dataset.")
        return jsonify({'error': 'Crop not found in the dataset.'}),400

    # Encode the crop name
    crop_encoded = label_encoder_crop.transform([crop_name])[0]
    print("Encoded crop name for prediction:", crop_encoded)



    X = np.array([[crop_encoded, 0, 0, 2021, 0, 0]])
    X = scaler_X.transform(X[:, [0, 3]])

    prediction = model.predict(X)
    class_prediction = np.argmax(prediction, axis=1)[0]
    print("Prediction result:", class_prediction)

    return jsonify({'prediction': int(class_prediction)})

# Visualization route
@app.route('/visualize', methods=['POST'])
```

```
def visualize():
    crop_name = request.json.get('crop').strip().lower()
    print("Received crop name for visualization:", crop_name)

    # Filter data for the specified crop
    data_filtered = data[data['Crop'] == crop_name]
    if data_filtered.empty:
        print("Error: No data found for this crop.")
        return jsonify({'error': 'No data found for this crop.'}), 400

    highest_production_state = data_filtered.groupby('State_Name')
    ['Production Rate'].sum().idxmax()
    highest_production_season = data_filtered.groupby('Season2')
    ['Production Rate'].sum().idxmax()

    # Create knowledge graph
    G = nx.Graph()
    G.add_node(f"Crop: {crop_name}", size=5000)
    for _, row in data_filtered.iterrows():
        state_name = row['State_Name']
        season_name = row['Season2']
        G.add_node(state_name, size=3000)
        G.add_node(season_name, size=3000)
        G.add_edge(f"Crop: {crop_name}", state_name, label="grows in")
        G.add_edge(f"Crop: {crop_name}", season_name, label="grows in")

    G.add_node("High Production Rate", size=3000)
    G.add_edge("High Production Rate", highest_production_state,
    label="highest in")
    G.add_edge("High Production Rate", highest_production_season,
```

```
    label="most in")

    # Generate and save the knowledge graph
    pos = nx.spring_layout(G, k=0.5, seed=42)
    plt.figure(figsize=(14, 10))
    nx.draw(G, pos, with_labels=True, node_size=2000,
    node_color="lightblue", font_size=10, font_color="black",
    font_weight="bold", edge_color="gray", width=2)
    edge_labels = nx.get_edge_attributes(G, 'label')
    nx.draw_networkx_edge_labels(G, pos, edge_labels=edge_labels,
    font_color="red")
    plt.title(f"Knowledge Graph for Crop: {crop_name}",
    fontsize=15)

    # Save the plot as a PNG image and encode to base64
    buf = io.BytesIO()
    plt.savefig(buf, format="png")
    plt.close()
    buf.seek(0)
    graph_base64 = base64.b64encode(buf.read()).decode("utf-8")
    print("Knowledge graph generated successfully for:", crop_name)
    return jsonify({'graph': graph_base64})

# Home route
@app.route('/')
def home():
    return render_template('index.html')
if __name__ == '__main__':
    app.run(debug=True)
```

index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
    initial-scale=1.0">
    <title>Crop Production Knowledge Graph</title>
    <style>
        /* Set background image and overall styling */
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            background-image: url("{{ url_for('static',
            filename='background.jpg') }}");
            background-size: cover;
            color: #333;
            display: flex;
            justify-content: center;
            align-items: center;
            min-height: 100vh;
            transform: translateY(-20px);
            overflow: hidden;
            margin: 20px;

        }
        .container {
            border-radius: 10px;
```

```
    min-height: 300px;
    text-align: center;
    align-items: center;

    align-self: stretch;
    max-width: 500px;
}



/* Heading and form styling */
h1 {
    font-size: 1.8em;
    color: #f6f3f3;
    margin-bottom: 20px;
    margin-top: 100px;
    border-color: #ff7f50;
    text-shadow: -1px -1px 0 #ff7f50,
    1px -1px 0 #ff7f50,
    -1px 1px 0 #ff7f50,
    1px 1px 0 #ff7f50;
}

label {
    font-weight: bold;
    font-size: 1.1em;
    margin-bottom: 10px;
    color: #ece8e8;
    border-color: darkgoldenrod;
    text-shadow: -1px -1px 0 #ff7f50,
```

```
    1px -1px 0 #ff7f50,
    -1px 1px 0 #ff7f50,
    1px 1px 0 #ff7f50;

}

input[type="text"] {
    width: 80%;
    padding: 10px;
    margin: 15px 0;
    border: 1px solid #ccc;
    border-radius: 5px;
    font-size: 1em;
}

button {
    background-color: #2E8B57;
    color: white;
    padding: 10px 20px;
    border: none;
    border-radius: 5px;
    font-size: 1.1em;
    cursor: pointer;
    transition: background-color 0.3s;
    margin-bottom: 20px;
    width: 50%;
}

button:hover {
    background-color: #1c5e3b;
```

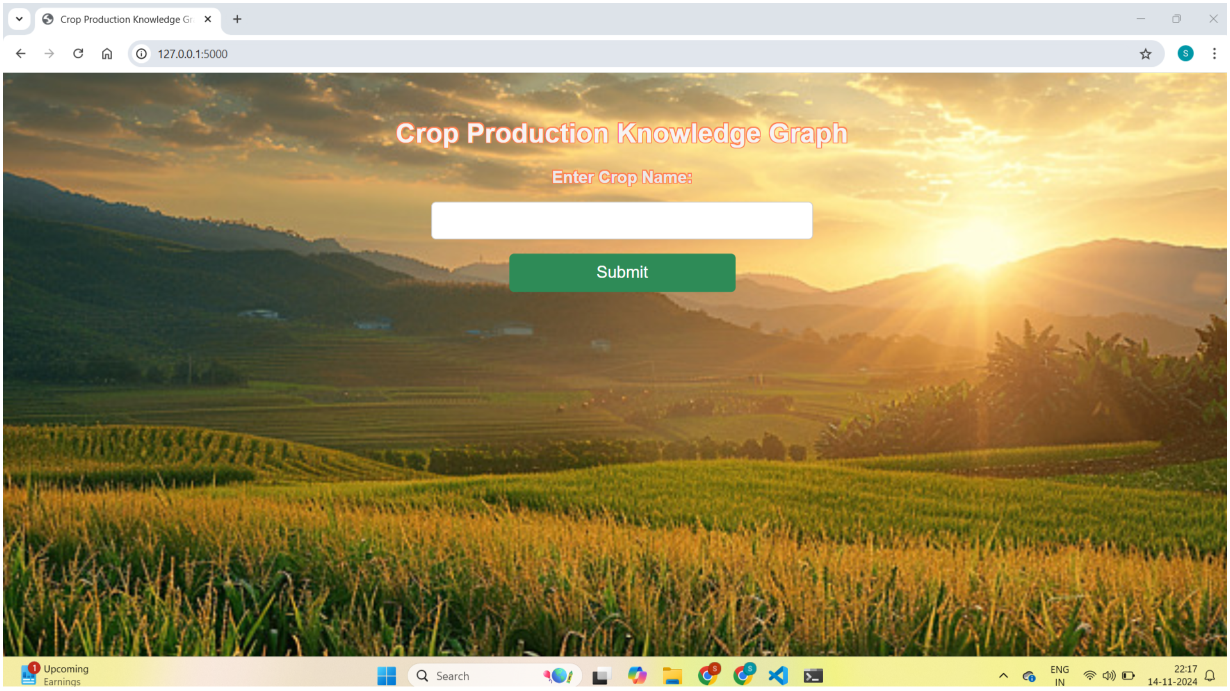
```
        }

        /* Error and graph styling */
        .error {
            color: red;
            font-weight: bold;
            margin-top: 15px;
        }

        .graph-container {
            margin-top: 50px;
            margin-bottom: 50px;
            border: 1px solid #2E8B57;
            border-radius: 10px;
            padding: 25px;
            background-color: rgba(248, 252, 248, 0.9);
            width: 100%;
            max-width: 800px;
        }


        img {
            width: 100%;
            border-radius: 8px;
        }
    </style>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
</head>
<body>
    <div class="container">
```

```
<h1>Crop Production Knowledge Graph</h1>

<form id="crop-form">
    <label for="crop">Enter Crop Name:</label>
    <input type="text" id="crop" name="crop" required>
    <button type="submit">Submit</button>
</form>

<div id="prediction-result"></div>
<div id="graph-container"></div>

<script>
    $(document).ready(function() {
        $('#crop-form').on('submit', function(event) {
            event.preventDefault();
            let cropName = $('#crop').val().trim();

            $('#prediction-result').empty();
            $('#graph-container').empty();

            $.ajax({
                url: '/predict',
                type: 'POST',
                contentType: 'application/json',
                data: JSON.stringify({ crop: cropName }),
                success: function(response) {
                $('#prediction-result').html(`<p>Predicted Class:
                ${response.prediction}</p>`);
                }
            });
```

```
                    $.ajax({
                        url: '/visualize',
                        type: 'POST',
                        contentType: 'application/json',
                        data: JSON.stringify({ crop: cropName }),
                        success: function(response) {
                        $('#graph-container').html
                        (`<img src="data:image/png;
                        base64,${response.graph}"
                        alt="Knowledge Graph">`);
                        }
                    });
                });
            });
        </script>
    </div>
</body>
</html>
```

APPENDIX B

OUTPUT SNAPSHOTS



After entering the crop name, the knowledge graph will be generated.

REFERENCES

[1] Divan.M.J, Frittelli.V(2022) A systematic mapping study about applications of knowledge graphs in agribusiness. International Journal of Research in Intelligent Systems, 10(1), 27–38

[2] Qin.H, Yao.Y (2021) Agriculture Knowledge Graph Construction and Application. Journal of Physics: Conference Series, 1756(012010):1-10

[3] Dorpinghaus.J, Stefan.A (2019) Knowledge extraction and applications utilizing context data in knowledge graphs. Federated Conference on Computer Science and Information Systems, 18(1): 265–272

[4] Li.X, Zhang.W, Sun.Y (2018) A knowledge-graph-based system for crop disease prediction and management. International Journal of Agricultural and Biological Engineering, 11(6):128–136

[5] Xiaohan.Z (2020) A Survey on Application of Knowledge Graph.Journal of Physics: Conference Series, 1487 (012016):1-11