SUMMER INTERNSHIP REPORT

A Report Submitted on Internship at FreeCodeCamp Organization

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING
Rajiv Gandhi University Of Knowledge Technologies, Srikakulam

Submitted by
Palavalasa Jagadeeswari
ID No: S200139

Under the guidance of:
Mr. T. Anil Kumar

Internship Duration: **May 25, 2025 – July 10, 2025**
Submitted on: **July 30, 2025**

FreeCodeCamp.org

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES, SRIKAKULAM

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

SUMMER INTERNSHIP REPORT APPROVAL SHEET

This is to certify that the Internship Project Report titled

"Machine Learning With Python"

Submitted by

Palavalasa Jagadeeswari
ID No: **S200139**

has been evaluated and approved by the department as a partial fulfillment of the requirements of the internship program during the academic year 2025.

Internship Duration: **May 25, 2025 – July 10, 2025**
Date of Submission: **July 30, 2025**

Signature of Guide: _____
Name: **Mr. T.Anil Kumar**
Designation: Internal Guide

Signature of HOD: _____
Name: **Dr./Ms. Laxmi Bala**
Designation: Head of the Department, Computer Science Engineering

Contents

Abstract

This report presents the summary of a 300-hour virtual internship completed at FreeCodeCamp.org from 25th May 2025 to 10th July 2025, as part of the academic curriculum at Rajiv Gandhi University of Knowledge Technologies, Srikakulam. The internship focused on Machine Learning with Python, offering a structured blend of theoretical instruction and hands-on project work aimed at developing real-world skills in the field of Machine Learning

The internship began with a series of **36 lecture videos**, covering core machine learning topics such as supervised and unsupervised learning, Convolutional Neural Networks (CNNs), Natural Language Processing (NLP), and Reinforcement Learning, including Q-learning. These sessions provided foundational knowledge and guided practical application.

Throughout the internship, I applied what I learned by working on five machine learning projects, using tools and libraries such as Pandas, NumPy, Matplotlib, TensorFlow, and Keras. Projects were implemented using Google Colab and Replit, with submission links set for public access. I also showcased my work on LinkedIn, sharing project results and key learnings to demonstrate both technical and communication skills.

This internship gave me practical exposure to real-world machine learning workflows, including data preprocessing, model building, evaluation, and iterative refinement. It strengthened my understanding of Python-based ML tools, improved my problem-solving abilities, and helped me develop confidence in applying machine learning concepts to real datasets.

Company Profile: freeCodeCamp.org

FreeCodeCamp.org is a globally recognized **nonprofit organization** based in the United States, dedicated to making coding education accessible to everyone, everywhere. As a 501(c)(3) public charity, freeCodeCamp delivers high-quality, free learning resources that empower individuals from all backgrounds to develop real-world programming skills and pursue careers in technology.

Founded in **2014** and headquartered in **San Francisco, California,** freeCodeCamp has become a leading platform in the e-learning space, offering thousands of coding challenges, interactive lessons, and hands-on projects. The platform specializes in Full-Stack Web Development and Python-based Software Engineering. It offers hands-on training in technologies like HTML, CSS, JavaScript, React.js, Node.js, MongoDB, and Express.js, alongside Python applications in Data Analysis, Scientific Computing, and Machine Learning using libraries such as Pandas, NumPy, Matplotlib, Scikit-learn, and TensorFlow.

FreeCodeCamp's comprehensive curriculum is designed around a project-based learning approach. Learners can earn certifications in multiple areas like Responsive Web Design, JavaScript Algorithms, Front End Libraries, Machine Learning and Scientific Computing, Data Visualization, and more. **Each certification path includes real-world projects that help students build a strong portfolio, making them job-ready.**

Beyond its curriculum, freeCodeCamp also plays a pivotal role in community-driven learning. It supports a global network of contributors, mentors, and learners through its technical blog, active forums, open-source contributions, and an educational YouTube channel. These resources help learners stay connected, updated, and inspired throughout their coding journey.

FreeCodeCamp continues to **bridge the gap between education and employability by providing practical, industry-aligned experiences at no cost**. Its mission is to help people gain the skills they need to secure jobs in technology and to support lifelong learning in a rapidly evolving digital world.

Introduction

Internships play a crucial role in transforming academic learning into practical experience. They allow students to move beyond classroom theory and gain exposure to real-world tools, technologies, and problem-solving approaches. By working on hands-on tasks and structured projects, students can develop both their technical abilities and essential soft skills, such as time management, analytical thinking, and adaptability—skills that are vital in today's evolving tech landscape.

As a student of Computer Science and Engineering at **Rajiv Gandhi University of Knowledge Technologies**, I completed a **300-hour virtual internship with freeCodeCamp**, focused on **Machine Learning with Python**. The program included **36 instructional videos** (40–45 minutes each) and **five hands-on projects**, combining guided learning with practical application to strengthen core machine learning concepts and promote independent problem-solving.

During this internship, I worked with **Key Python Libraries** such as Pandas, NumPy, Matplotlib, TensorFlow, and Keras. I explored various domains within machine learning, including Convolutional Neural Networks (CNNs) for image recognition, Natural Language Processing (NLP) for text-based tasks, and Reinforcement Learning, where I implemented Q-learning algorithms and LSTM-based models for sequence prediction.

The internship also provided a clearer understanding of real-world machine learning workflows, including data preprocessing, model training, evaluation, and iterative improvement. By working on structured projects, I learned how to handle raw datasets, clean and transform data, choose appropriate models, evaluate performance, and refine results through experimentation—skills that are essential in practical machine learning applications.

In addition to the technical skills, this internship helped me improve my ability to work independently and manage tasks in a remote setting. I learned how to structure machine learning projects, follow best coding practices, use version control tools like GitHub, and document my work in cleared way .Presenting my projects and Certificates on professional platforms like LinkedIn helped me build confidence in communicating technical work effectively.

Overall, this internship with freeCodeCamp has been a transformative experience. It not only enhanced my technical proficiency in machine learning and Python programming but also strengthened my problem-solving approach and project management skills. It served as a bridge between academic learning and real-world application, preparing me to contribute meaningfully to the growing field of artificial intelligence and data science. This experience has further fueled my interest in machine learning and has laid a strong foundation for my future endeavors in the tech industry.

Problem Statement

Many students and aspiring developers often encounter a gap between academic learning and practical implementation. While university courses cover theoretical concepts, they rarely provide the hands-on experience necessary to confidently apply these ideas in real-world scenarios. As a result, learners may struggle to grasp complex topics such as neural networks, data preprocessing, model training, evaluation, and advanced subjects like deep learning and reinforcement learning.

Additionally, students typically lack exposure to widely used tools and environments such as **TensorFlow, Keras, Google Colab,** which are essential in professional machine learning workflows. This lack of practical experience can make it challenging to transition from academic study to contributing to real projects. The Course-based internship offered by **freeCodeCamp**, a free and open-source learning platform,and aslo globally recognized and accessible platform, addresses these gaps through structured video-based lessons and hands-on project work. By combining theory with guided implementation, the program enables students to gain real-world experience in machine learning using Python, better preparing them for future opportunities in the tech industry.

Project Overview

The core project during my internship at freeCodeCamp revolved around a comprehensive, course-based journey in Machine Learning with Python. The program was designed to deliver both theoretical and practical experience through a series of structured video lectures—totaling approximately 40–45 hours across 36 sessions—and hands-on projects. This immersive **300-hour internship** enabled me to apply concepts in real time while working independently on guided tasks.

The internship was centered on five progressive projects, each focusing on a different aspect of machine learning, such as supervised learning models, **Convolutional Neural Networks (CNN)**, **Natural Language Processing (NLP)**, and reinforcement learning techniques like **Q-learning and LSTM.** These projects helped reinforce critical ML workflows such as data preprocessing, model building, training, and evaluation.

All tasks were executed using **Google Colab**, where I documented and shared results through publicly accessible notebooks. Some projects were also built and tested on **Replit**, and final outcomes were shared via LinkedIn for visibility and feedback. Tools like TensorFlow, Keras, NumPy, Pandas, and Matplotlib were extensively used, providing exposure to real-world libraries and best practices in machine learning development.

Overall, this project-based internship gave me a strong practical foundation in machine learning, enhanced my understanding of Python-based data science workflows, and built confidence in tackling complex AI challenges through structured problem-solving.

Technologies Used

During the course of the internship at **FreeCodeCamp**. I worked with the following technologies and tools to design, develop, and optimize projects:

- **Python** – Primary programming language used to implement machine learning workflows and models.
- **TensorFlow (2.x)** – Core deep learning framework used for building and training neural networks, including CNNs and advanced models.
- **Keras (TensorFlow Keras API)** – High-level interface for defining, training, and evaluating neural network models with simplified syntax.
- **TensorFlow Datasets (TFDS)** – Utilized for loading and preparing ready-to-use datasets for training and evaluation.
- **NumPy** – Used for efficient array computations, data manipulation, and matrix operations in ML pipelines.
- **Pandas** – Applied for data preprocessing, loading, cleaning, and exploratory data analysis.
- **Matplotlib** – Used to create visualizations such as accuracy/loss curves and model performance plots.
- **Scikit-learn** – Utilized for train-test splitting, building traditional ML models, and evaluation metrics.
- **Colab** – Google-hosted Jupyter notebook platform that enabled GPU acceleration, code testing, and collaborative development.
- **Replit** – Web-based IDE used for building and testing Python-based logic and interactive scripts outside of notebooks.
- **GitHub** – Employed for version control, code hosting, and showcasing project submissions.
- **pip** – Used to install external libraries and nightly versions of TensorFlow and documentation packages during experimentation.

System Design and Architecture

The internship projects followed a modular and layered architecture that ensured scalability, reusability, and efficient experimentation. Each project was independently structured using the following generalized architecture.

1.Data Acquisition and Preprocessing Layer

- **Description:**This layer handled data collection from public datasets or CSV files, followed by preprocessing tasks like normalization, tokenization, encoding, or data augmentation.
- **Tools and Libraries Used:**
  - Pandas, NumPy – For data manipulation and transformation
  - TensorFlow Datasets, Scikit-learn, Keras Preprocessing – For loading and splitting datasets
  - Matplotlib – For visual inspection of data distribution

2.Model Development Layer

- **Description:**Core machine learning logic was implemented in this layer, utilizing different algorithms based on the nature of the project.
- **Model Architecture Used in Projects:**
  - CNN – Used in Cat and Dog Image Classifier for feature extraction and classification.
  - KNN – Used in Book Recommendation System to compute user-item similarity matrices.

  - Neural Network (LSTM) – Used in SMS Classifier to identify spam/ham messages from text sequences.
  - Linear Regression – Used in Health Insurance Cost Estimator for predicting numerical outputs based on continuous variables.
  - Conditional Logic with Randomization – Used in Rock, Paper, Scissors game to simulate decision-making without ML.

- **Tools and Libraries:**
  - TensorFlow, Keras, Scikit-learn, RMSprop, Adam optimizers
  - Sequential API for model stacking
  - Dropout, Dense, Conv2D, LSTM layers used appropriately per project

3.Training and Evaluation Layer

- **Description:**Models were trained on the preprocessed data with appropriate loss functions and evaluated using accuracy, loss metrics, or confusion matrices.
- **Tools and Techniques Used:**
  - Train/Test Split or Validation Split (e.g., 80/20)
  - Binary Crossentropy, Categorical Crossentropy, or MSE as loss functions
  - Accuracy, Precision, Recall for performance tracking
  - Matplotlib for plotting learning curves

4. Deployment and Interaction Layer

- **Description:**Though formal deployment was not part of the internship, project results were visualized or interacted with using code outputs, notebooks, and video demonstrations.

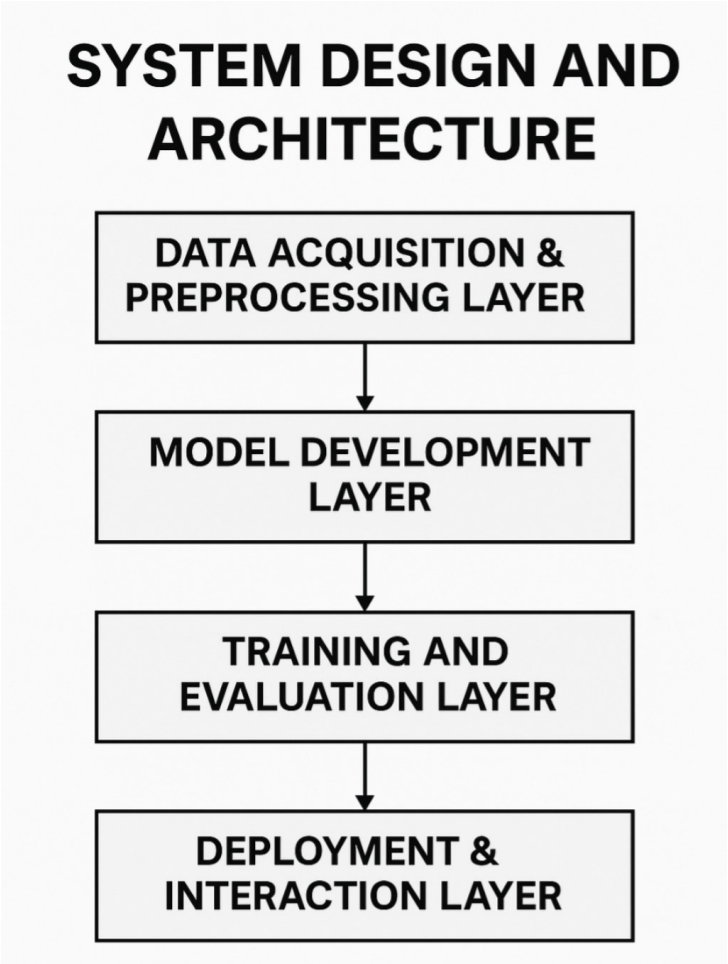

Figure 5.1: System Design and Architecture

- **Tools Used:**
  - Google Colab – Executed training pipelines, visualized outputs, and shared project links
  - Replit – Used for smaller interactive apps like Rock, Paper, Scissors
  - GitHub – Maintained version control and documentation

Module Descriptions

Cat and Dog Image Classifier (CNN)

- Data Collection Module: Loads and preprocesses image data (resize, normalize).
- CNN Model Module: A convolutional neural network built using Conv2D, MaxPooling2D, and Dense layers.
- Training Module: Compiles and trains the model using binary crossentropy loss to differentiate between cats and dogs.
- Prediction Module: Classifies new input images as either a cat or a dog.
- Evaluation Module: Displays model accuracy and loss curves to assess performance.

Book Recommendation System (KNN)

- Data Preprocessing Module: Cleans dataset, removes duplicates, and normalizes book ratings.
- Feature Vector Module: Converts user-item ratings into sparse matrix form using csrmatrix.
- Similarity Calculation Module: Uses NearestNeighbors algorithm from scikit-learn to compute book similarities.
- Recommendation Module: Generates a list of top similar books based on user input.

- Interface Module: Optional CLI or print statements for user interaction.

SMS Spam Classifier (Neural Network)

- Text Preprocessing Module: Tokenizes and pads input SMS messages.
- Embedding Module: Converts words into dense vectors using Embedding layer.
- LSTM Network Module: Uses LSTM and Dense layers for sequential learning and classification.
- Training Module: Trains on labeled messages (spam/ham) using binary crossentropy loss.
- Prediction Module: Predicts whether a new SMS is spam or ham with associated accuracy.

Health Cost Predictor (Linear Regression)

- Data Cleaning Module: Handles missing values and encodes categorical variables.
- Feature Selection Module: Selects inputs like age, BMI, smoking status, and number of children.
- Model Building Module: Implements linear regression using scikit-learn.
- Training and Testing Module: Splits data, trains the model, and calculates RMSE or MAE.
- Prediction Module: Predicts health insurance costs based on new user data.

Rock, Paper, Scissors Game (Python Script)

- User Input Module: Takes user's choice via keyboard input.
- Computer Choice Module: Randomly generates choice using random.choice.
- Game Logic Module: Determines the winner using conditional statements.
- Display Module: Shows both choices and game outcome to the user.
- Loop Module (optional): Allows repeated play or exit from the game.

Code Snippets and Functionality

This section provides code snippets and brief explanations for each of the five projects completed during the internship at **freeCodeCamp**. The projects were focused on **Machine Learning with Python**

Task 1: Cat and Dog Image Classifier

Building a Model

The following model is designed to classify images of cats and dogs. It identifies the category of the input image based on learned features.

```
model=Sequential()
model.add(Conv2D(16, (3, 3), activation='relu', input_shape=(IMG_HEIGHT
model.add(MaxPooling2D(2, 2))
model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(2, 2))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(2, 2))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(2, 2))
model.add(Flatten())
model.add(Dropout(0.5))
model.add(Dense(512, activation='relu'))
model.add(Dense(1,activation='sigmoid'))
model.compile(optimizer='adam',
              loss='binary_crossentropy',
              metrics=['accuracy'])
```

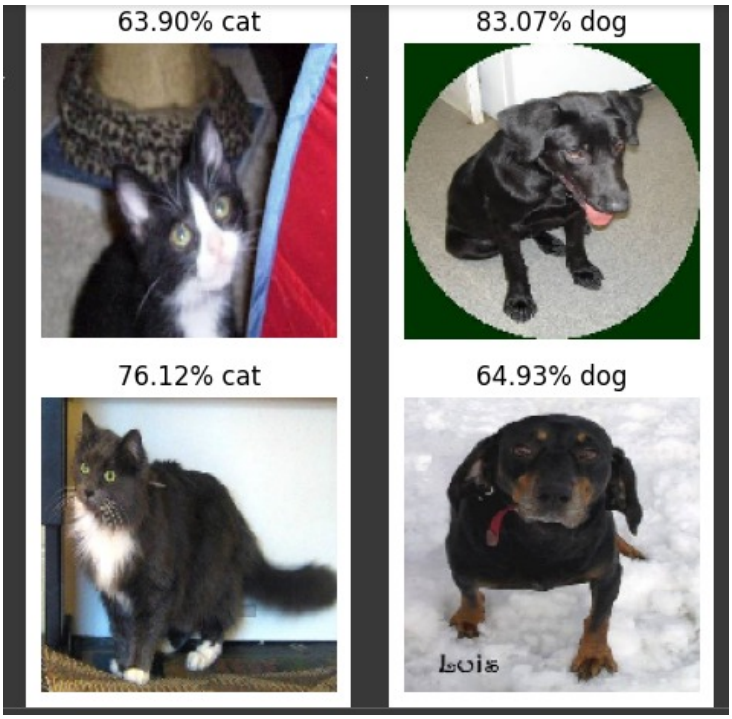

Figure 7.1: Output of Task 1: "A CNN-based model for accurately classifying cat and dog images."

```
model.summary()
```

This CNN model consists of four convolutional-pooling blocks followed by a dense network for binary image classification. It uses ReLU activations, dropout for regularization, and a sigmoid output layer, optimized with Adam and binary cross-entropy.

Task 2: Linear Regression Health Costs Calculator

Building a Model

The following model is designed to predict health insurance costs using Linear Regression. It estimates the cost based on features such as age, BMI, smoking status, and other relevant attributes.

```
1.Create a Model
model = keras.Sequential([
                            normalizer,
                            layers.Dense(2),
```

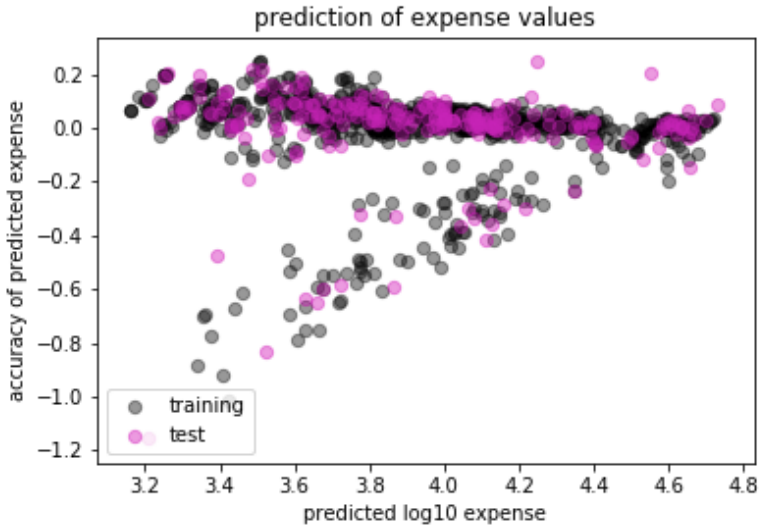

Figure 7.2: Output of Task 2:"A Linear Regression model for predicting health insurance costs based on user-specific features."

```
                                    layers.Dense(1)
])
model.compile(optimizer=keras.optimizers.Adam(learning_rate=0.1),
              loss='mae',
              metrics=['mae', 'mse'])
model.summary()
2.Train a model
model.fit(train_dataset, train_labels, epochs=100, validation_split=0.2
```

The output of a Linear Regression model for a health insurance cost calculator is:

A continuous numerical value representing the predicted health insurance cost (in currency units) for a given individual, based on their input features.

Task 3: Rock Paper Scissors

"A Python-based Rock, Paper, Scissors game where Player 1 competes against multiple opponents, achieving win rates up to 100 percentage through consistent logic and randomized computer choices."

```
def player(prev_play, opponent_history=[], play_order={}):
```

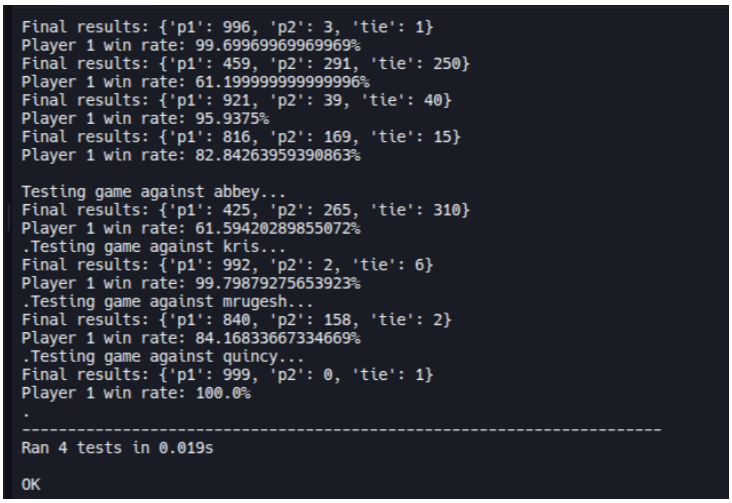

Figure 7.3: Output of Task 3: "A Python-based Rock, Paper, Scissors game that simulates matches against random computer choices with high player win rates."

```
    if not prev_play:
        prev_play = 'R'
    opponent_history.append(prev_play)
    prediction = 'P'
    if len(opponent_history) > 4:
        last_five = "".join(opponent_history[-5:])
        play_order[last_five] = play_order.get(last_five, 0) + 1
        potential_plays = ["".join([*opponent_history[-4:], v])
            for v in ['R', 'P', 'S']]
        sub_order = { k: play_order[k] for k in potential_plays if k in
        if sub_order:
            prediction = max(sub_order, key=sub_order.get)[-1:]
    ideal_response = {'P': 'S', 'R': 'P', 'S': 'R'}
    return ideal_response[prediction]
```

Task 4: Neural Network SMS Text Classifier

A neural network-based SMS text classifier built using TensorFlow and Keras, designed to detect spam messages. The model leverages an Embedding layer and an LSTM unit to learn contextual patterns in text, achieving binary classification with high accuracy.

[11]:
```
# function to predict messages based on model
def predict_message(pred_text):
  class_dict = {
      0 : "ham",
      1 : "spam",
      }
  encoded_message = [one_hot(pred_text, VOCAB_SIZE)]
  padded_message = pad_sequences(encoded_message, maxlen=MAX_LENGTH, padding='post')
  prediction = [model.predict(padded_message)[0][0], class_dict[np.round(model.predict(padded_message)[0][0]
  return prediction

pred_text = "how are you doing today?"

prediction = predict_message(pred_text)
print(prediction)
```

```
[0.0005038009, 'ham']
```

Figure 7.4: Output of Task 4: "The model predicts whether an input SMS is spam or ham, along with the classification accuracy"

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense, Dropout
embedding_dim = 50
max_words = 1000
max_len = 500
1.Define model
model = Sequential()
model.add(Embedding(input_dim=max_words, output_dim=embedding_dim, inpu
model.add(LSTM(64))
model.add(Dense(256, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))
model.build(input_shape=(None, max_len))
2.Compile the model
model.compile(
    loss='binary_crossentropy',
    optimizer='RMSprop',
    metrics=['accuracy']
)
model.summary()
```

Task 5: Book Recommendation Engine Using KNN

A book recommendation engine using KNN finds similar users or books based on ratings. It suggests books liked by users with similar preferences. Simple and effective for personalized recommendations.

```
1.Building a model
model = NearestNeighbors(metric='cosine')
model.fit(df.values)


2.Predict a model
books = get_recommends("Where the Heart Is (Oprah's Book Club (Paperbac
print(books)
def test_book_recommendation():
  test_pass = True
  recommends = get_recommends("Where the Heart Is (Oprah's Book Club (P
  if recommends[0] != "Where the Heart Is (Oprah's Book Club (Paperback
    test_pass = False
  recommended_books = ["I'll Be Seeing You", 'The Weight of Water', 'Th
  recommended_books_dist = [0.8, 0.77, 0.77, 0.77]
  for i in range(2):
    if recommends[1][i][0] not in recommended_books:
      test_pass = False
    if abs(recommends[1][i][1] - recommended_books_dist[i]) >= 0.05:
      test_pass = False
test_book_recommendation()
```

A Book Recommendation Engine using K-Nearest Neighbors (KNN) suggests books based on user preferences or similarity between books. KNN finds the 'k' most similar books (neighbors) using features like user ratings, genres, or descriptions. By identifying books similar to what the user liked before, it recommends titles they're likely to enjoy. It's a simple yet powerful algorithm for collaborative filtering.

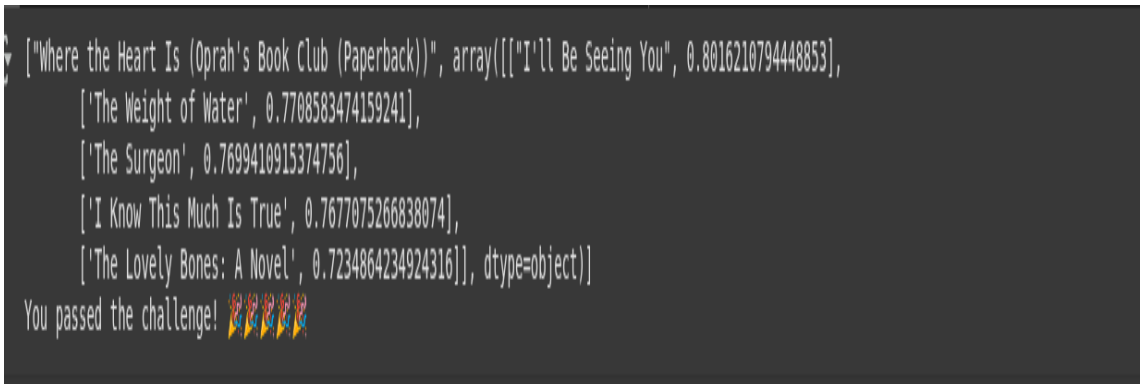

Figure 7.5: Output of Task 5: "KNN-based book recommender suggests books by finding similar users or books based on ratings."

Challenges Faced During Internship

- Understanding core ML concepts initially was challenging.
- Faced issues with data preprocessing and model accuracy.
- Learning TensorFlow and Keras took time.
- Balancing video lectures and project work was tough.
- Debugging and interpreting model results was sometimes difficult.

Learnings and Outcomes

During the course of this internship, I gained both theoretical and practical knowledge in the field of Machine Learning using Python. The structured curriculum offered by freeCodeCamp, which included over 36 hours of video lectures and five comprehensive hands-on projects, helped solidify my understanding of various ML concepts and workflows.

- Gained hands-on experience in building ML models using TensorFlow and Keras.
- Strengthened Python skills and applied them to real-world datasets and projects.
- Learned core concepts of CNNs, NLP, LSTMs, and reinforcement learning.
- Practiced model evaluation techniques like accuracy measurement and loss analysis.
- Used Google Colab and Replit for collaborative, cloud-based development.
- Improved debugging, documentation, and project presentation skills.

Conclusion

The internship with freeCodeCamp provided a comprehensive and practical introduction to machine learning using Python. Through structured video lectures and five hands-on projects, I gained valuable knowledge in core machine learning algorithms, neural networks, CNNs, natural language processing, reinforcement learning, and LSTM models. Utilizing tools such as TensorFlow, Keras, Google Colab, and Replit allowed me to implement real-world solutions effectively. This experience not only strengthened my technical skills but also improved my problem-solving abilities and deepened my understanding of end-to-end machine learning workflows—from data preprocessing to model evaluation. Overall, it was a highly enriching experience that enhanced my readiness for future roles in AI and data science.

References

- https://www.google.com/-Used for researching documentation, troubleshooting, and exploring additional information.
- https://www.freecodecamp.org/-Official website of freeCodeCamp
- https://colab.research.google.com/-Used as the primary environment for coding, training, and testing ML models.
- https://www.tensorflow.org/-Main library used for building and training deep learning models.
- https://github.com/ – Used for version control and sharing code repositories publicly.
- https://replit.com/-Used for writing and running lightweight Python projects online.
- https://chatgpt.com/-Helped in debugging code, clarifying errors, and understanding ML concepts
- https://www.geeksforgeeks.org/-Referred for coding examples, Python concepts, and algorithm explanations.
- https://www.gitpod.io/-Provided as an online IDE by the organization for sharing and running project code through URLs.

Annexure

Annexure A: GitHub Repository Link and Replit Project link

The source code of the Machine Learning with Python Projects completed during this internship is available on the GitHub repository linked below:

- `https://github.com/Jagadeeswari82/My-Internship-Notebooks`
- `https://replit.com/@jagadeeswaripal/RockPaperScissorProject`

Annexure B: Internship Certificate

Below is the internship certificate received upon successful completion of the 300-hours virtual internship at freeCodeCamp.org (freeCodecamp)