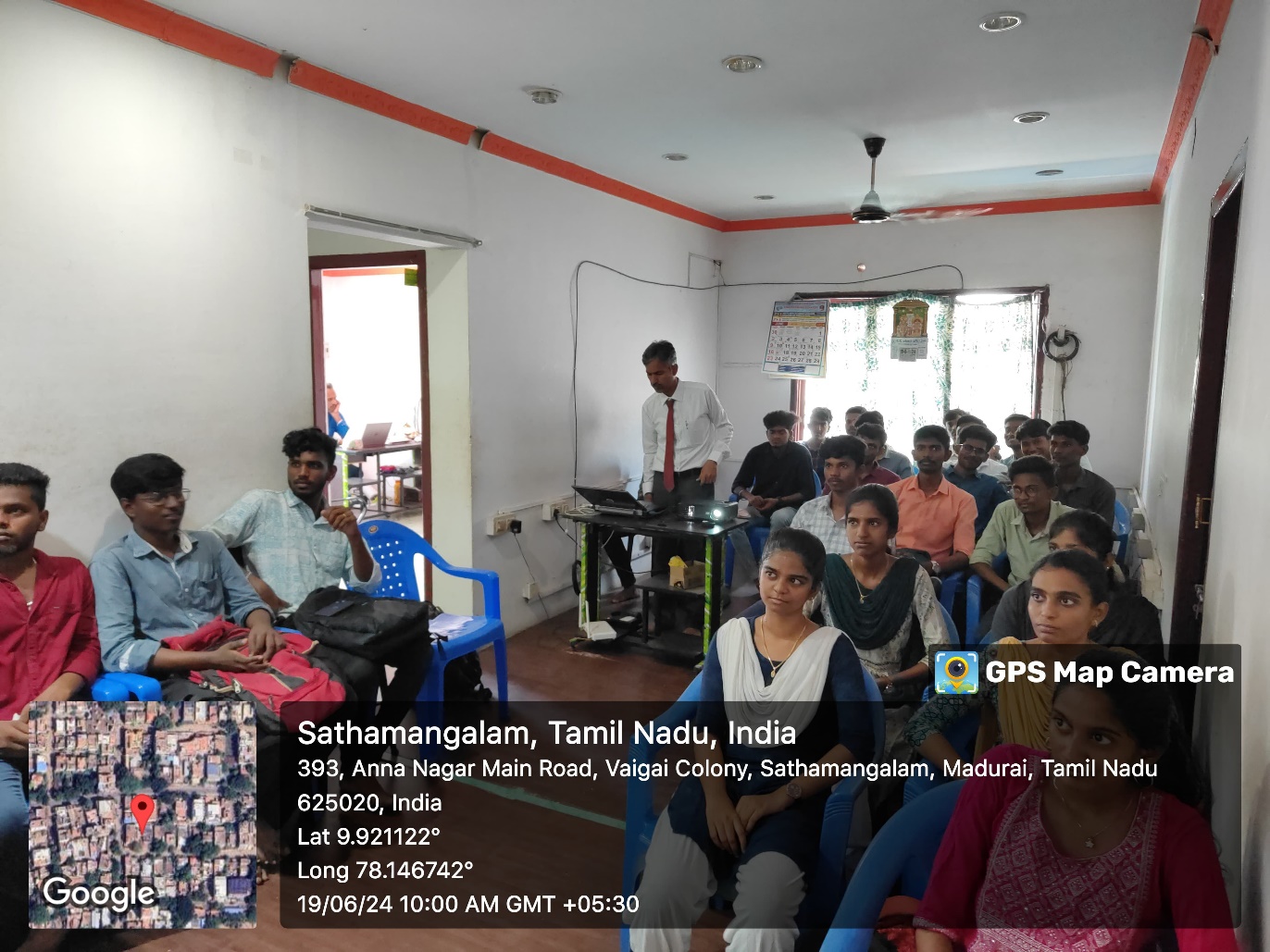
**Machine Learning Internship Report**

**Internship Date:** June 19, 2024 – June 28, 2024  
**Company Name:** Geons Logix  
**Name:** Muthu Bharati S A

**Introduction:**During my internship at Geons Logix, I gained hands-on experience with machine learning concepts and sharpened my skills in applying key algorithms. Throughout the two-week program, I worked on practical applications of machine learning, culminating in a loan approval prediction project. This report summarizes my learning process and the final project.

**Week 1: Introduction to Machine Learning (19/06/2024 – 23/06/2024)**

**Day 1: Fundamentals of Machine Learning**The first day covered the foundational principles of machine learning, including an introduction to supervised, unsupervised, and reinforcement learning. I also explored the significance of data and how different model types are used in these categories, providing a solid base for future applications.



**Day 2: Machine Learning Workflow**  
On the second day, we examined the complete machine learning pipeline, from data collection and preprocessing to model training, evaluation, and deployment. Special attention was given to the importance of clean, structured data in ensuring the accuracy of predictive models.

****

**Day 3: Understanding Linear Regression**  
We took a deep dive into linear regression, focusing on its use for predicting continuous outcomes. A hands-on Python exercise allowed us to build a linear regression model using real-world data, reinforcing the theoretical concepts through practical implementation.

**Day 4: Intro to Decision Trees**  
The fourth day centered around decision trees, an algorithm widely used for both classification and regression tasks. Practical exercises helped clarify how decision trees partition data and make predictions based on feature values.

**Day 5: Machine Learning Applications at Geons Logix**We ended the week by exploring how machine learning is applied at Geons Logix, particularly in the areas of predictive analytics and automation. This session was particularly insightful as it demonstrated how machine learning drives business decisions and optimizes operations.

**Week 2: Loan Approval Prediction Project (24/06/2024 – 28/06/2024)**

**Project Objective:**  
The second week involved working on a project aimed at predicting loan approval statuses. The goal was to build a binary classification model using logistic regression.

**Data Preparation and Preprocessing:**  
I started by cleaning the dataset, handling missing values, and converting categorical data into numerical form using one-hot encoding. I also applied feature scaling to ensure uniform data distribution across all features.

**Model Training and Evaluation:**  
After splitting the dataset into training and test sets (80% training, 20% testing), I trained the logistic regression model. The model achieved an accuracy of around 85%, and further evaluation was done using precision, recall, and F1-score to measure its overall effectiveness.

**Program:**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix, roc\_curve, auc

import matplotlib.pyplot as plt

import seaborn as sns

*# Load the dataset*

loan\_data = pd.read\_csv('loan\_approval\_dataset.csv')

print(loan\_data.head())

*# Data preprocessing*

X = loan\_data.drop(' loan\_status', axis=1) *# Features*

y = loan\_data[' loan\_status'] *# Target variable*

*# Convert categorical variables into dummy/indicator variables*

X = pd.get\_dummies(X)

*# Splitting the dataset into training and testing sets*

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

*# Feature scaling*

scaler = StandardScaler()

X\_train\_scaled = scaler.fit\_transform(X\_train)

X\_test\_scaled = scaler.transform(X\_test)

*# Model training*

model = LogisticRegression()

model.fit(X\_train\_scaled, y\_train)

*# Model evaluation*

y\_pred = model.predict(X\_test\_scaled)

accuracy = accuracy\_score(y\_test, y\_pred)

print("Accuracy:", accuracy)

print("Classification Report:")

print(classification\_report(y\_test, y\_pred))

*# Confusion matrix*

cm = confusion\_matrix(y\_test, y\_pred)

plt.figure(figsize=(8, 6))

sns.heatmap(cm, annot=True, cmap='Blues', fmt='g', cbar=False)

plt.xlabel('Predicted labels')

plt.ylabel('True labels')

plt.title('Confusion Matrix')

plt.show()

*# ROC curve*

y\_test\_binary = y\_test.map({' Approved': 1, ' Rejected': 0})

y\_pred\_binary = pd.Series(y\_pred).map({' Approved': 1, ' Rejected': 0})

fpr, tpr, thresholds = roc\_curve(y\_test\_binary, y\_pred\_binary)

roc\_auc = auc(fpr, tpr)

plt.figure(figsize=(8, 6))

plt.plot(fpr, tpr, color='darkorange', lw=2, label='ROC curve (area = %0.2f)' % roc\_auc)

plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')

plt.xlim([0.0, 1.0])

plt.ylim([0.0, 1.05])

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristic (ROC) Curve')

plt.legend(loc="lower right")

plt.show()

from sklearn.metrics import precision\_recall\_curve, average\_precision\_score

*# Precision-Recall curve*

precision, recall, thresholds = precision\_recall\_curve(y\_test\_binary, y\_pred\_binary)

average\_precision = average\_precision\_score(y\_test\_binary, y\_pred\_binary)

plt.figure(figsize=(8, 6))

plt.step(recall, precision, color='b', alpha=0.2, where='post')

plt.fill\_between(recall, precision, step='post', alpha=0.2, color='b')

plt.xlabel('Recall')

plt.ylabel('Precision')

plt.ylim([0.0, 1.05])

plt.xlim([0.0, 1.0])

plt.title('Precision-Recall Curve: AP={0:0.2f}'.format(average\_precision))

plt.show()

*# Feature Importance Plot*

if hasattr(model, 'coef\_'):

feature\_importance = pd.DataFrame({

'Feature': X.columns,

'Importance': model.coef\_[0]

})

feature\_importance = feature\_importance.sort\_values(by='Importance', ascending=False)

plt.figure(figsize=(10, 6))

sns.barplot(x='Importance', y='Feature', data=feature\_importance)

plt.xlabel('Importance')

plt.ylabel('Feature')

plt.title('Feature Importance')

plt.show()

import seaborn as sns

import matplotlib.pyplot as plt

import pandas as pd

*# Load the wine quality dataset (assuming it's a CSV file)*

loan\_data = pd.read\_csv("loan\_approval\_dataset.csv")

*# Correlation matrix*

correlation\_matrix = loan\_data.corr()

*# Create heatmap*

sns.heatmap(correlation\_matrix, annot=True, cmap='coolwarm', fmt=".2f")

*# Add title*

plt.title('Correlation Heatmap of Wine Quality Dataset')

plt.show()

sns.catplot(x="loan\_id", y=" loan\_amount", kind="bar", data=loan\_data)

plt.show()

**Conclusion:**  
This project provided practical experience in building machine learning models from scratch, focusing on data preprocessing, model training, and evaluation. The loan approval prediction task helped me better understand binary classification techniques and how they apply to real-world problems.

https://github.com/MUTHU-BHARATI/loan-approval/blob/main/ML\_mini.ipynb

**Deep Learning Internship Report**

**Internship Date:** August 20, 2024 – September 3, 2024  
**Company Name:** Phoenix Softech  
**Name:** Muthu Bharati S A

**Introduction:**  
During my deep learning internship at Phoenix Softech, I delved into neural networks and their applications. Over the three-week period, I worked on a character recognition project using Convolutional Neural Networks (CNNs). This report provides a detailed summary of the learning phases and the final project.

**Week 1: Deep Learning Foundations (20/08/2024 – 24/08/2024)**

**Day 1: Introduction to Deep Learning**  
The internship began with a deep dive into the differences between machine learning and deep learning. We explored neural network architecture and how these models are used to solve complex tasks by automatically extracting features.



**Day 2: Artificial Neural Networks (ANNs)**  
On the second day, we built a basic Artificial Neural Network (ANN) in Python, learning how neurons process inputs and generate outputs. The session covered activation functions, weights, and biases in detail.



**Day 3: Deep Neural Networks (DNNs)**  
We discussed Deep Neural Networks (DNNs) on the third day, learning how deeper architectures work and the challenges that arise, such as the vanishing gradient problem. We explored techniques to optimize training in DNNs.



**Day 4: Convolutional Neural Networks (CNNs)**  
On the fourth day, we implemented a simple CNN model for image recognition. The session covered convolution and pooling layers, which are key components of CNNs, particularly in image processing.



**Day 5: Recurrent Neural Networks (RNNs)**  
We concluded the week by exploring Recurrent Neural Networks (RNNs), which are designed to handle sequential data. We built an RNN for a sequence prediction task, understanding how these networks retain memory over time.



**Week 2: Advanced Topics in Deep Learning (25/08/2024 – 29/08/2024)**

**Day 1: Natural Language Processing (NLP)**  
We began the second week with Natural Language Processing (NLP), learning how deep learning techniques are applied to text-based tasks. Word embeddings such as Word2Vec were introduced, and we worked on a text classification task using sequence models.

**Day 2: Generative Adversarial Networks (GANs)**  
On the second day, we explored GANs, learning how generator and discriminator models work together to create new data. I implemented a simple GAN to generate synthetic images.

**Day 3: Deep Reinforcement Learning**  
The focus of the third day was on deep reinforcement learning. We discussed how agents learn by interacting with environments and implemented a basic agent to solve a simple task through trial and error.

**Week 3: Character Recognition Project Using CNNs (30/08/2024 – 03/09/2024)**

**Project Objective:**  
In the final week, I applied my knowledge to build a CNN for recognizing handwritten characters.

**Data Preparation:**  
The data preprocessing involved converting images to grayscale, resizing them to a standard format, and normalizing pixel values.

**Model Training and Evaluation:**  
I built the CNN using multiple convolutional and pooling layers, followed by fully connected layers. The model achieved over 98% accuracy on the test set.

**Conclusion:**  
This project helped me gain hands-on experience with CNNs and deep learning. I learned how to preprocess image data, build and train CNN models, and evaluate their performance.

https://github.com/MUTHU-BHARATI/character-recognition