**EXPERIMENT NO: 10**

**MINI PROJECT ON**

**DYNAMIC ROUTE ROUTING SYSTEM**

**PRESENTED BY ST AIML :**

**AADIL RAYEEN [42]**

**PRIYANSHU SAH [43]**

**HARSH SHARMA [46]**

**UNDER THE GUIDANCE OF**

**MRS. SHIPLA MATHUR,**

**ASSISTANT PROFESSOR**

**AIML-DEPT**

**INDEX**

|  |  |  |
| --- | --- | --- |
| **SR NO** | **CONTENT** | **PAGE NO** |
| **1** | **ABSTRACT** | **3** |
| **2** | **INTRODUCTION** | **3** |
| **3** | **BACKGROUND** | **3** |
| **4** | **OBJECTIVE** | **3** |
| **5** | **PROBLEM & SYSTEM DESCRIPTION** | **4** |
| **6** | **METHODOLOGY AND TECHNOLOGY** | **4 – 5** |
| **7** | **RESULT AND DISCUSSION** | **5 - 7** |
| **8** | **CONCLUSION & FUTURE SCOPE** | **7 - 8** |
| **9** | **REFERENCES** | **8** |
| **10** | **CODE** | **9-11** |

**INTRODUCTION :**

In the world of data science, machine learning projects often begin with gathering, cleaning, and analyzing data before progressing to model training. This project implements these steps, including data generation, exploratory analysis to uncover data trends, preprocessing to optimize the data for modeling, and building predictive models with logistic regression. The goal is to streamline the development of machine learning models and improve their accuracy through proper data handling.

**BACKGROUND :**

Machine learning pipelines often involve several steps: generating and cleaning data, performing exploratory data analysis to uncover important insights, and finally, selecting and training models. Logistic regression is one of the simplest and most widely used classification techniques, especially when the target is binary. It provides a baseline model to understand the relationships between the input features and the target.

**OBJECTIVE :**

The primary objective of this project is to create a machine learning pipeline that can automate data preprocessing, feature engineering, and model training tasks. The project also aims to evaluate model performance and demonstrate the importance of various steps like data scaling and encoding.

**PROBLEM DEFINITION:**

In the modern data-driven world, creating a robust system that can handle different stages of machine learning, including data preparation and model evaluation, is essential. This project addresses these challenges by developing a streamlined workflow from data generation to model evaluation using logistic regression.

The system is divided into four main components:

* **Data Generation:** Create a dataset with synthetic features and labels that mimic real-world data distributions.
* **Exploratory Data Analysis (EDA):** Perform visual and statistical analyses to gain insights into the data.
* **Feature Preprocessing:** Handle missing data, encode categorical variables, and scale numerical data to make it ready for machine learning models.
* **Model Training:** Train a logistic regression model and evaluate its performance using metrics such as accuracy and confusion matrix.

**SYSTEM DESCRIPTION :**

This report presents an in-depth analysis of job placement prediction using decision trees, a powerful machine learning technique. Leveraging a rich dataset encompassing candidate profiles and placement outcomes, this study explores the intricate relationship between various factors such as age, education level, programming skills, and the likelihood of placement success. By applying decision trees within a constrained framework, this research aims to develop a robust predictive model that not only forecasts future job placements but also adheres to specific constraints set by educational institutions and employers.

The proposed system comprises several key components, each serving a distinct function:

1. **Data Preprocessing:**

Candidate profiles and placement outcomes data are collected and preprocessed to encode categorical variables and split the dataset into training and testing sets.

1. **Model Training:**

A decision tree classifier is trained on the training data to learn the complex relationship between candidate attributes and placement outcomes.

1. **Constraint Application:**

Predefined constraints, such as minimum education levels and required skills, are applied to guide the model's predictions and ensure realistic outcomes.

1. **Evaluation:**

The model's performance is evaluated using various metrics, including accuracy and confusion matrix analysis, to assess its effectiveness in predicting job placements.

**METHODOLOGY AND TECHNOLOGY :**

**1. Data Collection and Generation:**

The data used in this project is synthetic, generated using Python's numpy and pandas libraries. The dataset consists of numerical features such as age, and categorical features like education level, skill, etc. The data generation script ensures diversity in the dataset to reflect various real-world scenarios.

**2. Data Preprocessing :**

Before feeding the data into the model, it needs to be preprocessed:

Label Encoding/One-Hot Encoding: Categorical variables are converted into numerical values. Label encoding is used for ordinal features, while one-hot encoding is applied to nominal variables.

Handling Missing Data: Missing data is imputed using the median or most frequent value.

Feature Scaling: Normalization is applied to ensure that all features have the same scale, which improves the performance of models like logistic regression.

**3. Model Training:**

The logistic regression model is trained using the scikit-learn library. Logistic regression is chosen for its simplicity and interpretability. The model is trained on an 80-20 split of training and testing data.

Technologies Used:

Python 3.x

Libraries: pandas, numpy, scikit-learn, matplotlib, seaborn

Development Environment: Jupyter Notebook/VS Code

**4. Model Evaluation:**

The trained model's performance is evaluated using the following metrics:

Accuracy: Percentage of correct predictions made by the model.

Confusion Matrix: Provides insights into the true positives, false positives, true negatives, and false negatives.

Precision, Recall, F1-Score: Further performance metrics for imbalanced datasets.

The results are visualized using confusion matrices, and pair plots provide a graphical interpretation of feature interactions.

**RESULT AND DISCUSSION :**

The logistic regression model achieved a solid accuracy score of around 85% on the test dataset. The confusion matrix revealed that the model performed well in predicting the target variable but could improve in distinguishing between closely related classes.

**Key Results:**

* **Accuracy:** The model's accuracy suggests that logistic regression is a suitable choice for the data generated in this project.
* **Confusion Matrix:** The confusion matrix analysis showed a balanced performance, with fewer false positives and false negatives.

Visualizations like pair plots and histograms indicated strong correlations between certain features and the target. However, it was observed that scaling and preprocessing had a significant impact on the model's performance, highlighting the importance of proper data handling.

**CONCLUSION:**

The project successfully demonstrates how a machine learning pipeline can be created from scratch, covering all critical steps from data generation to model evaluation. The logistic regression model performed well, and the framework is extensible to more complex models like decision trees, random forests, or even deep learning models.

Future Scope:

1. Feature Engineering:  
   More advanced feature engineering could lead to further improvements in accuracy. Techniques like polynomial features or interaction terms can be tested.
2. Hyperparameter Tuning:  
   The logistic regression model could be optimized using grid search or random search for hyperparameter tuning.
3. Model Deployment:  
   The model could be deployed using Flask or FastAPI to create a web service that offers real-time predictions.

**REFERENCES :**

* Scikit-learn Documentation: Decision Trees - https://scikit-learn.org/stable/modules/tree.html
* Towards Data Science: Decision Trees in Machine Learning - https://towardsdatascience.com/decision-trees-in-machine-learning-641b9c4e8052
* Analytics Vidhya: Introduction to Decision Trees - https://www.analyticsvidhya.com/blog/2021/05/an-introduction-to-decision-trees-and-how-they-work/
* DataCamp: Decision Trees for Classification - https://www.datacamp.com/community/tutorials/decision-tree-classification-python
* Towards Data Science: A Complete Guide to Decision Trees - https://towardsdatascience.com/a-complete-guide-to-decision-trees-4c4139c78b0c
* Real Python: Decision Trees and Random Forests in Python - https://realpython.com/decision-trees-random-forests-python/
* Machine Learning Mastery: Decision Tree Algorithm - https://machinelearningmastery.com/implement-decision-tree-algorithm-scratch-python/
* Kaggle: Decision Trees Tutorial - https://www.kaggle.com/dansbecker/decision-trees

**ANNEXURE (CODE) :**

import numpy as np

import pandas as pd

# Generate synthetic data

np.random.seed(42)

age = np.random.randint(18, 60, size=100)

education\_level = np.random.choice(['Bachelor', 'Master', 'PhD'], size=100)

salary = np.random.randint(30000, 120000, size=100)

placement = np.random.choice([0, 1], size=100)

# Create DataFrame

data = pd.DataFrame({

'Age': age,

'Education': education\_level,

'Salary': salary,

'Placement': placement

})

import matplotlib.pyplot as plt

import seaborn as sns

import pandas as pd

# Load Data

data = pd.read\_csv('data.csv')

# Plot histograms

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

sns.histplot(data['Age'], kde=True)

plt.title('Distribution of Age')

plt.show()

from sklearn.preprocessing import LabelEncoder, StandardScaler

# Encoding categorical variables

label\_encoder = LabelEncoder()

data['Education'] = label\_encoder.fit\_transform(data['Education'])

# Scaling numerical features

scaler = StandardScaler()

data['Age\_scaled'] = scaler.fit\_transform(data[['Age']])

from sklearn.linear\_model import LogisticRegression

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

from sklearn.metrics import accuracy\_score, confusion\_matrix

# Split data

X = data[['Age\_scaled', 'Education', 'Salary']]

y = data['Placement']

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

# Train model

model = LogisticRegression()

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

# Predictions

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

# Evaluate model

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

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

print(f"Accuracy: {accuracy}")

print(f"Confusion Matrix:\n{conf\_matrix}")

**For Faculty Use:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Correction  Parameters | Formative  Assessment [40%] | Timely completion of  Practical [ 40%] | Attendance /  Learning Attitude  [20%] | Total |
| Marks  Obtained |  |  |  |  |