**Joint Tech Internship Community Program Organizer**

**Generative AI Consortium (MSME)**

**SystimaNX IT Solutions PVt Ltd.**

**TASK-2 : Predicting Employee Attrition in a Corporate Organization**

**Team Name : NerdHerd**

**Team Member 1: Shree Varshana R**

**GitLink: <https://github.com/ShreeVarshana/Predicting-Employee-Attrition-in-a-Corporate-Organization-Machine-Learning>**

**Team Member 2: Vaishnavi K**

**GitLink: <https://github.com/Vaish228/Predicting_Employee_Attrition.git>**

**Team Member 3: Sowbaraniga K**

**GitLink: <https://github.com/sowbaranigakps-04/Predicting-Employee-Attrition-in-a-Corporate-Organization-Machine-Learning>**

**Employee Attrition Prediction Process**

**1. Data Exploration**

* **Objective:** Get familiar with the dataset to understand its structure and any issues that might need addressing.
* **Steps:**
  + Load the dataset and take a look at the first few rows to see what kind of data we have.

|  |
| --- |
| import pandas as pd  df = pd.read\_csv('/content/employee\_attrition\_dataset.csv')  print(df.head())  print(df.info()) |

* + Check for any missing values to see if we need to clean the data.

|  |
| --- |
| print(df.isnull().any()) |

**2. Data Preprocessing**

* **Objective:** Prepare the data for our machine learning models by cleaning and transforming it.
* **Steps:**
  + Remove columns that won’t help in predicting attrition:

|  |
| --- |
| df.drop('EmployeeID', axis=1, inplace=True)  df.drop('Education', axis=1, inplace=True) |

* + Encode categorical variables so they can be understood by the model. We’ll convert text labels into numbers:

|  |
| --- |
| from sklearn.preprocessing import LabelEncoder  le = LabelEncoder()  df.Gender = le.fit\_transform(df.Gender)  df.MaritalStatus = le.fit\_transform(df.MaritalStatus)  df.Department = le.fit\_transform(df.Department)  df.JobRole = le.fit\_transform(df.JobRole)  df.OverTime = le.fit\_transform(df.OverTime)  df.Attrition = le.fit\_transform(df.Attrition) |

* + Normalize the numerical features to ensure they’re on a similar scale, which can help our models perform better:

|  |
| --- |
| from sklearn.preprocessing import MinMaxScaler  ats = ['Age', 'MonthlyIncome', 'DistanceFromHome']  min\_max\_scaler = MinMaxScaler()  df[ats] = min\_max\_scaler.fit\_transform(df[ats]) |

**3. Model Selection**

* **Objective:** Choose suitable models to predict employee attrition and prepare the data accordingly.
* **Steps:**
  + Define our feature set (what we’ll use to predict) and our target variable (what we’re trying to predict)

|  |
| --- |
| X = df.iloc[:, :14]  y = df.iloc[:, -1] |

* + Split the dataset into training and testing sets so we can evaluate the model’s performance:

|  |
| --- |
| from sklearn.model\_selection import train\_test\_split  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=25, random\_state=46) |

**4. Model Evaluation**

* **Objective:** Check how well our models are performing using accuracy as our main metric.
* **Steps:**
  + Implement a K-Nearest Neighbors (KNN) classifier to see how it performs:

|  |
| --- |
| from sklearn.metrics import accuracy\_score  from sklearn.neighbors import KNeighborsClassifier  knn = KNeighborsClassifier(n\_neighbors=7)  knn.fit(X\_train, y\_train)  y\_pred\_knn = knn.predict(X\_test)  acc\_knn = round(accuracy\_score(y\_pred\_knn, y\_test), 2) \* 100  print("KNN Accuracy:", acc\_knn) |

* + Now, let’s try an XGBoost classifier to see if we can get better results:

|  |
| --- |
| from xgboost import XGBClassifier  xgb = XGBClassifier().fit(X\_train, y\_train)  y\_pred\_xgb = xgb.predict(X\_test)  accuracy\_xgb = accuracy\_score(y\_test, y\_pred\_xgb)  print(f"XGBoost Accuracy: {round(accuracy\_xgb \* 100, 2)}%") |

**5. Visualizations**

* **Understanding Data:** Use visualizations like histograms to get a sense of the distribution of key features, such as age:

|  |
| --- |
| import seaborn as sns  import matplotlib.pyplot as plt  sns.histplot(df['Age'], bins=10, kde=True)  plt.title('Age Distribution')  plt.xlabel('Age')  plt.ylabel('Frequency')  plt.show() |

* **Comparing Models:** Create a bar chart to compare the accuracy of different models, making it easy to see which performed best:

|  |
| --- |
| model\_names = ['KNN', 'XGBoost']  accuracies = [acc\_knn, round(accuracy\_xgb \* 100, 2)]  sns.barplot(x=model\_names, y=accuracies)  plt.title('Model Accuracy Comparison')  plt.ylabel('Accuracy (%)')  plt.ylim(0, 100)  plt.show() |

**Conclusion**

This documentation walks through the steps taken to explore, preprocess, select models, and evaluate performance for predicting employee attrition. The inclusion of visualizations helps in understanding the data and comparing model performances more effectively.