EMPLOYEE ATTRITION PREDICTION PROJECT

Team Name : Model Masters

Team Members

ARUN KARTHIK S 22CSR020

CHANDRU A 22CSR033

ELANKUMARAN R 22CSR053

Introduction

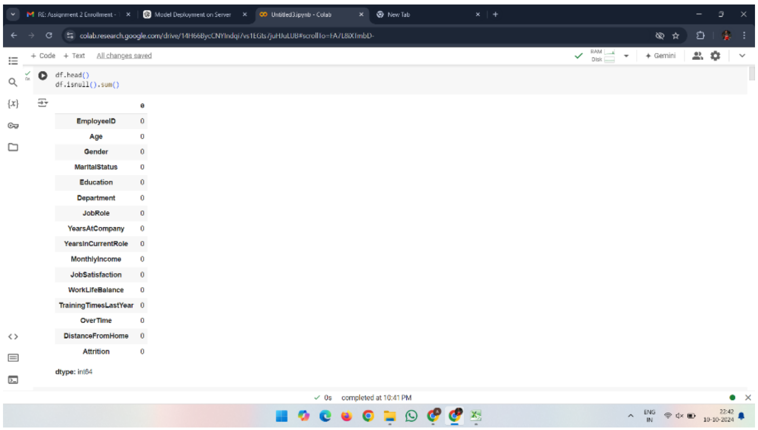
Employee attrition presents challenges for organizations and operational costs. This project utilizes machine learning to predict which employees may be at risk of leaving, allowing the organization to proactively address potential attrition. This predictive insight aims to support retention strategies that improve workforce stability.

Methodology

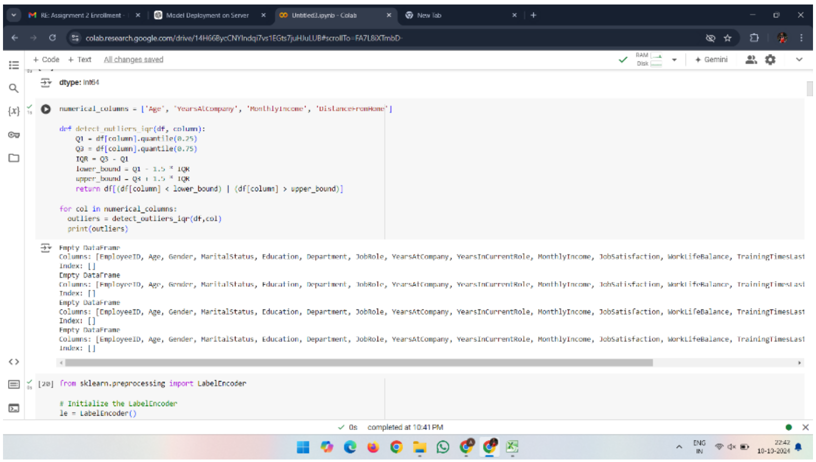
To predict employee attrition, we followed a structured approach, including data preprocessing, feature selection, model training, and evaluation.

1. **Data Collection and Preprocessing**

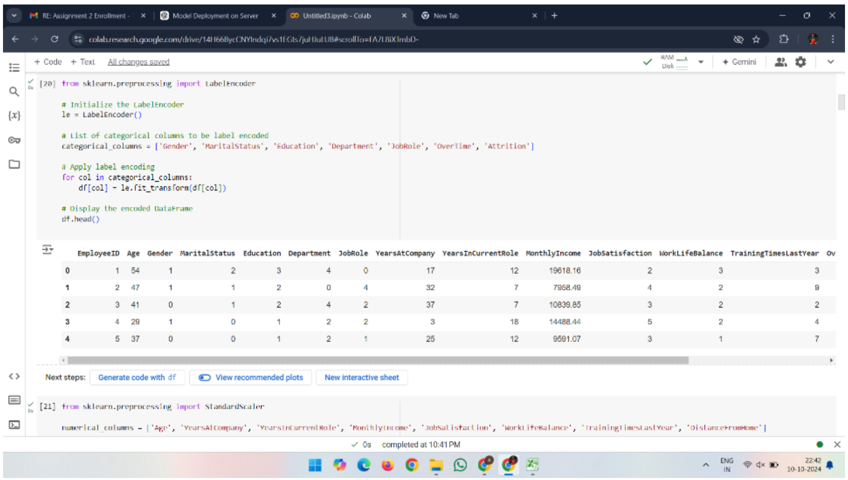
   - The dataset provided a balanced view of attrition, with 2512 instances indicating employees left and 2488 instances indicating retention.
   - Categorical features were converted to numeric form using label encoding. Additionally, key features were normalized to maintain consistency across model inputs.



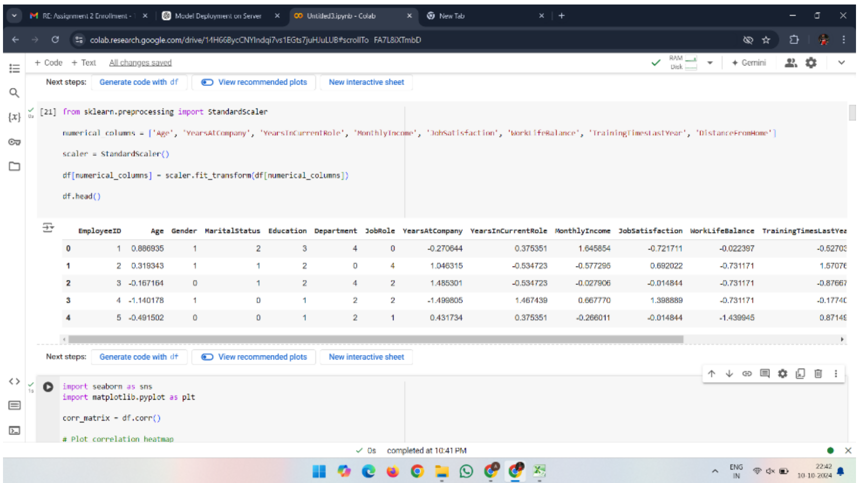
Checking missing values



Checking outliers



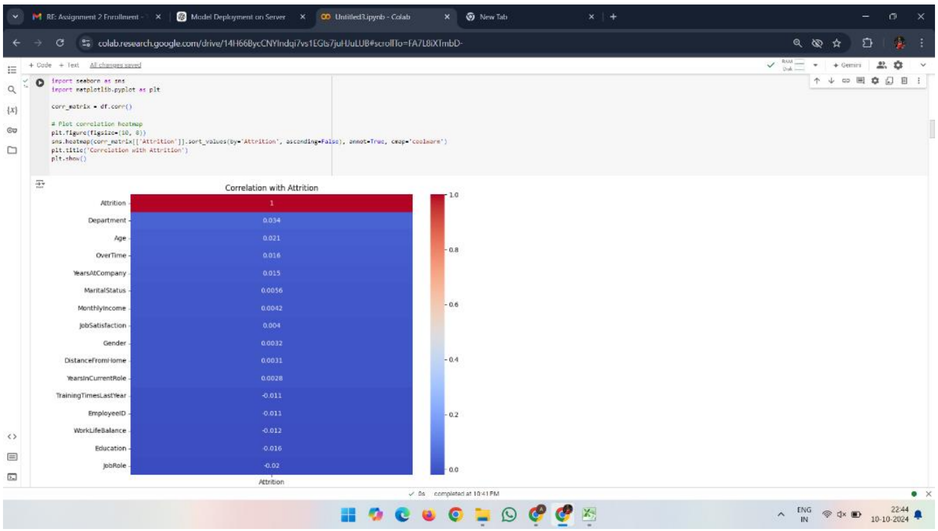
Converting categorical data into values



Normalizing the values

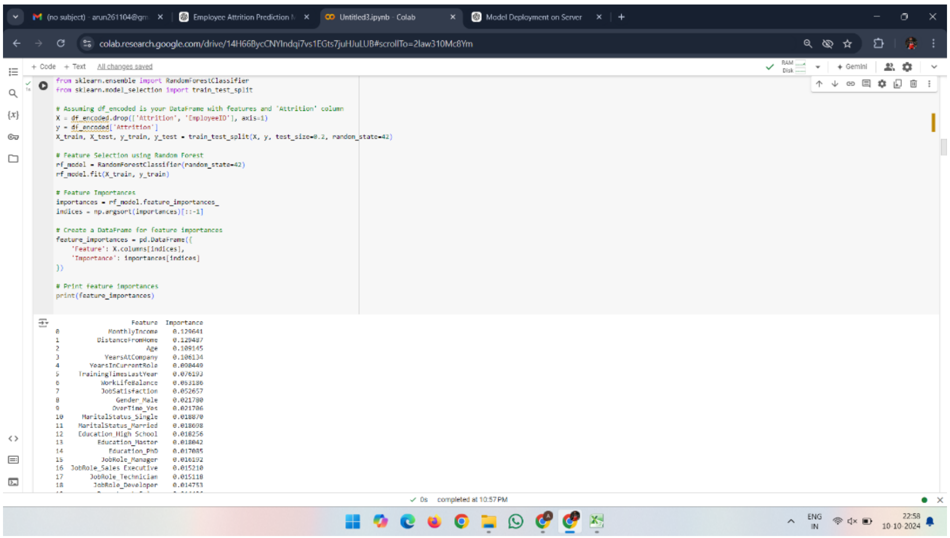
2. **Exploratory Data Analysis (EDA)**

   - Visual and statistical analyses were conducted to uncover patterns in demographics, job roles, and job satisfaction.
   - We identified correlations between attrition and specific features like age, monthly income, job satisfaction, and work-life balance.



3. **Feature Selection**

   - We conducted feature importance analysis using Random Forest, identifying key features contributing to attrition prediction, such as 'DistanceFromHome,' 'MonthlyIncome,' and 'YearsAtCompany,' which appeared to have the most influence on employee attrition.

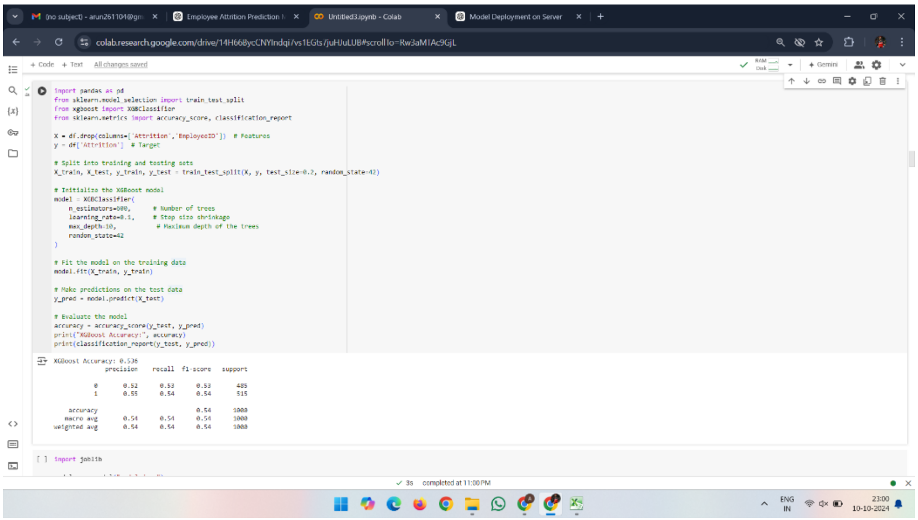


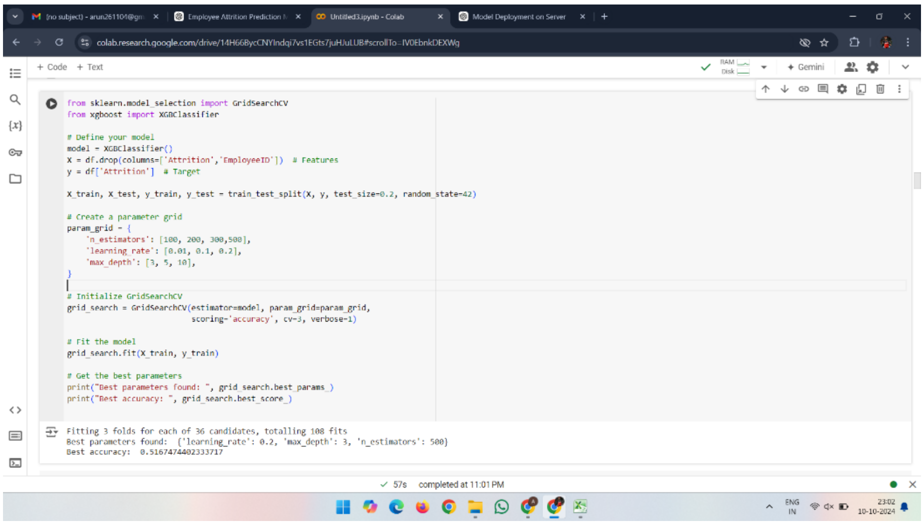
4. **Model Selection and Training**

    - Multiple models were tested, including Logistic Regression, Random Forest, and K-Nearest Neighbors, and were evaluated based on accuracy and other performance metrics.
    - **XGBClassifier** was chosen as the best-performing model after testing, as it demonstrated the ability to handle complex feature interactions.

5. **Model Evaluation**

    - We evaluated the models using accuracy, precision, recall, and F1-score metrics. Our final model, XGBClassifier, achieved an average accuracy of **53.6%**, suggesting that the dataset may lack strong patterns needed for precise prediction.





Findings

- **Key Insights from EDA**
  - **Age and Attrition**: Younger employees exhibited higher attrition rates, indicating a potential need for career development opportunities.
  - **Income and Attrition**: Employees with lower incomes tended to leave more frequently, suggesting that competitive salaries are a significant factor in retention.
  - **Job Satisfaction and Work-Life Balance**: Attrition was higher among employees with lower satisfaction and poor work-life balance, underscoring areas for potential improvement.
- **ModelPerformance**
  Although **XGBClassifier** was the best-performing model, its results were moderate:

- **Overall Accuracy**: **0.536** (53.6%)
- **Precision**:
  - Class 0: **0.52** (52%)
  - Class 1: **0.55** (55%)
- **Recall**:
  - Class 0: **0.53** (53%)
  - Class 1: **0.54** (54%)
- **F1-Score**:
  - Class 0: **0.53** (53%)
  - Class 1: **0.54** (54%)

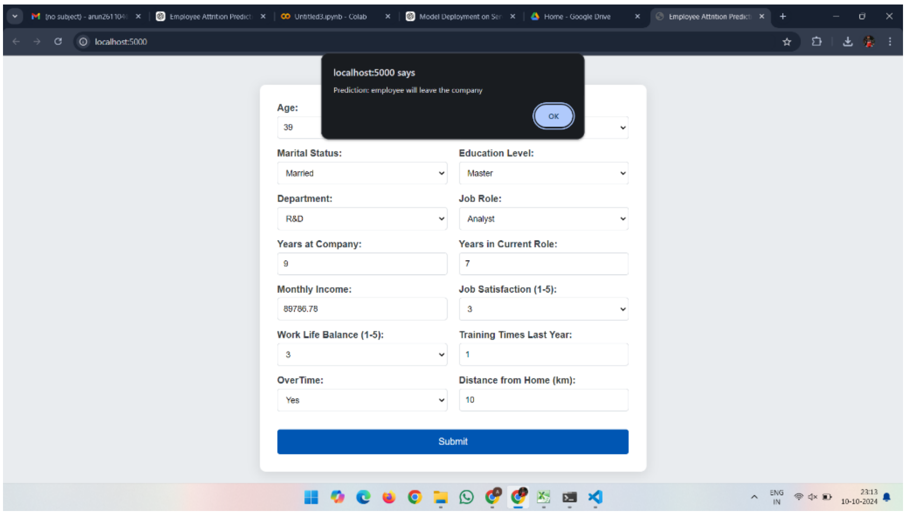
Despite trying different models and extensive tuning, the model's predictive accuracy remained limited. This low accuracy suggests the dataset may lack strong patterns for reliable classification.

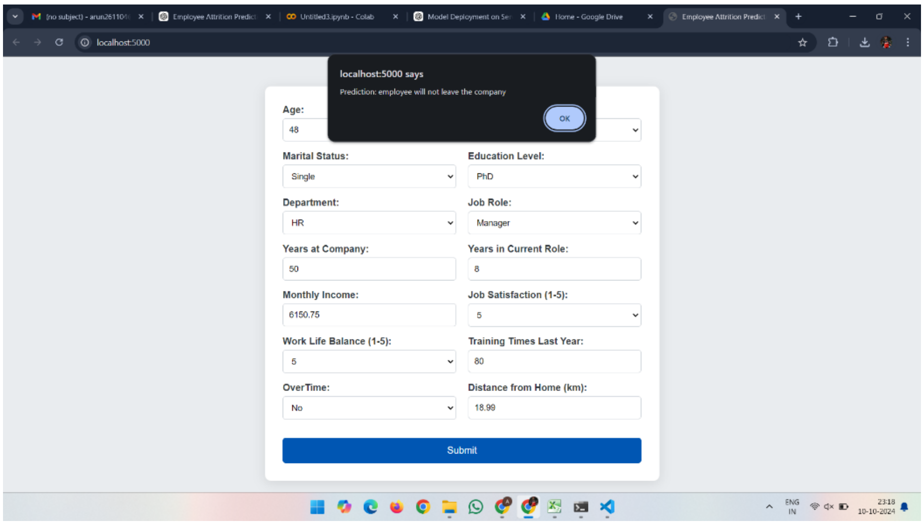
Recommendations

Based on the findings, the following actions are recommended to address areas with higher attrition risks:

- **Career Development for Younger Employees**: Offer clear growth paths, learning opportunities, and mentorship to encourage retention.
- **Enhanced Salary Reviews**: Regularly review and adjust compensation, particularly for lower-paying roles, to stay competitive in the market.
- **Improved Job Satisfaction and Work-Life Balance**: Implement flexible working options and wellness programs, especially for employees in roles or departments reporting lower satisfaction levels.

OUTPUT SCREENSHOTS





Conclusion

This project yielded insights into factors that influence employee attrition, yet the model’s limited performance indicates that the dataset may lack strong, identifiable patterns for precise prediction. For future work, adding new features or integrating external data sources, such as employee feedback or industry trends, may improve accuracy and support more effective retention strategies.

SAMPLE CODE

#SERVER CODE TO INTEGRATE FRONTEND FORM WITH BACKEND AND PREDICT USING MODEL

```
from flask import Flask, request, jsonify, render_template
import pandas as pd
import pickle

# Load the trained model
with open('model.pkl', 'rb') as model_file:
    model = pickle.load(model_file)

app = Flask(__name__)

@app.route('/')
def home():
    return render_template('form.html')

@app.route('/predict', methods=['POST'])
def predict():
    try:
        data = request.json['input'][0]
        print(data)

        # Prepare the input for the model
        input_data = {
```

```
    'Age': int(data['Age']),
    'Gender': int(data['Gender']),
    'MaritalStatus': int(data['MaritalStatus']),
    'Education': int(data['Education']),
    'Department': int(data['Department']),
    'JobRole': int(data['JobRole']),
    'YearsAtCompany': int(data['YearsAtCompany']),
    'YearsInCurrentRole': int(data['YearsInCurrentRole']),
    'MonthlyIncome': float(data['MonthlyIncome']),
    'JobSatisfaction': int(data['JobSatisfaction']),
    'WorkLifeBalance': int(data['WorkLifeBalance']),
    'TrainingTimesLastYear': int(data['TrainingTimesLastYear']),
    'OverTime': int(data['OverTime']),
    'DistanceFromHome': float(data['DistanceFromHome'])  # Change to float
}

# Convert input data to DataFrame
input_df = pd.DataFrame([input_data])

# Make prediction
prediction = model.predict(input_df)
if(prediction[0]==1):
    res='employee will leave the company'
else:
    res='employee will not leave the company'

# Convert prediction to standard Python type
response = {
    'predictions': res  # Convert to int
}
```

```
        return jsonify(response)

    except Exception as e:
        print(f"Error: {e}")  # Log the error message
        return jsonify({'error': str(e)}), 500  # Return the error as a JSON response

if __name__ == '__main__':
    app.run(debug=True, host='localhost', port=5000)
```