|  |  |
| --- | --- |
| Internship Project Title | TCS iON RIO-125: Salary Prediction Dashboard for HRs |
| Name of the Company | TCS iON |
| Name of the Industry Mentor | Himdweep Walia (himdweep.walia@tcs.com) |
| Name of the Institute | Amity university Online ,Noida |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Start Date | End Date | | Total Effort (hrs.) | | Project Environment | Tools used |
| 1-5-2024 | 10-5-2024 | | 60 | | Visual studio code(.ipynb) | Python3, windows OS |
| Milestone # 1 |  | Milestone: | | To create a dataset ,Clean & Sanitize salary dataset | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Start Date | End Date | | Total Effort (hrs.) | | Project Environment | Tools used |
| 11-5-2024 | 15-5-2024 | | 60 | | Visual studio code(.ipynb,.py) | Python3 , windows OS |
| Milestone # 2 |  | Milestone: | | To train the dataset and predict the salary of HRs | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Start Date | End Date | | Total Effort (hrs.) | | Project Environment | Tools used |
| 16-5-2024 | 20-5-2024 | | 90 | | Visual studio code (.ipynb,.py) | Python3 , windows OS |
| Milestone # 3 |  | Milestone: | | To improve their predictions in order to make its accurate as possible and to plot a chart of their predictions by Streamlit app | | |

**TABLE OF CONTENT**

* Acknowledgements
* Objective
* Introduction / Description of Internship
* Internship Activities
* Approach / Methodology
* Assumptions
* Exceptions / Exclusions
* Charts, Table, Diagrams
* Algorithms
* Challenges & Opportunities
* Risk Vs Reward
* Reflections on the Internship
* Recommendations
* Outcome / Conclusion
* Enhancement Scope
* Link to code and executable file
* Research questions and responses

**Acknowledgements:**

I would like to extend my sincere gratitude to my internship supervisors and mentors for their invaluable guidance and support throughout this project. Special thanks to the team members and colleagues who provided insights and feedback, making this experience both educational and fulfilling. I am highly indebted to TCS ion for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

I would like to express my gratitude towards my parents and my academic mentor, for their kind co-operation and encouragement which help me in completion of this project.

I would like to express my special gratitude and thanks to my industry mentor for giving me such attention and time.

**Objective:**

Milestone 1: I have completed for creating a dataset, clean the dataset and sanitized it.

Milestone 2: I have completed for training the dataset and predict the salary of a particular HR based on the training set.

Milestone 3: I have completed for improving their predictions in order to make it as accurate as possible. They should also be able to plot a chart of their predictions by Streamlit app.

The objective of this project is to build a Salary Prediction Dashboard for Hrs. The primary objective of this internship project was to develop a machine learning model capable of predicting the current salary of employees based on their age and years of experience.

**Introduction / Description of Internship**

According to project guidelines, I have to create a dashboard that predicts the salary of HRs when they switch jobs .

I have to create a dataset of two lakh entries containing the following details of various HRs :

* Name
* Age
* Years of experience
* Current salary

**By the end of the project I should be able to do following:**

* Create a dataset that contains the required details in each entry.
* Clean the dataset.
* Sanitize the dataset .
* Train the dataset to predict the salary of a particular HR when they switch jobs.
* Improve my predictions to make it as accurate as possible.
* Plot a chart to show a visual representation of the salary predictions for any existing entries or a new entry.

This internship involved working on a data science project that utilized machine learning algorithms to predict salaries. The process encompassed data generation, cleaning, model training, evaluation, and the deployment of a web application for salary prediction. The project was designed to enhance my skills in data science, machine learning, and web development, providing a comprehensive learning experience.

**Internship Activities**

1. Watched the Welcome Kit.
2. Done preparations for RIO – pre-assessment.
3. Attended the RIO – pre-assessment test.
4. Went through the day-wise plan.
5. Read the project reference material.
6. Read the industry project material.
7. Watched webinar 1.
8. Watched webinar 2.
9. Gone through all posts in the digital discussion room.
10. Watched few of the linear regression YouTube videos.
11. Read the linear regression article.
12. Went through the linear regression & Random Forest YouTube videos.
13. Read the linear regression article.
14. Watched the lectures provided and other videos for further understanding.
15. Searched and found out a proper data set for this project.
16. Wrote activity reports.

**Video links : project reference material**

1. [**https://youtu.be/dVH4kMcKvEA?feature=shared**](https://youtu.be/dVH4kMcKvEA?feature=shared)
2. [**https://youtu.be/32o0DnuRjfg?feature=shared**](https://youtu.be/32o0DnuRjfg?feature=shared)
3. [**https://youtu.be/B6yX03T6mU8?feature=shared**](https://youtu.be/B6yX03T6mU8?feature=shared)
4. [**https://towardsdatascience.com/linear-regression-detailed-view-ea73175f6e86**](https://towardsdatascience.com/linear-regression-detailed-view-ea73175f6e86)
5. [**https://towardsdatascience.com/what-is-data-cleaning-how-to-process-data-for-analytics-and-machine-learning-modeling-c2afcf4fbf45#:~:text=Data%20Cleaning%20means%20the%20process,of%20the%20basic%20data%20science**](https://towardsdatascience.com/what-is-data-cleaning-how-to-process-data-for-analytics-and-machine-learning-modeling-c2afcf4fbf45#:~:text=Data%20Cleaning%20means%20the%20process,of%20the%20basic%20data%20science)
6. [**https://www.appen.com/blog/training-data**](https://www.appen.com/blog/training-data)
7. [**https://www.appen.com/blog/training-data**](https://www.appen.com/blog/training-data)

**Digital course remote internships- soft skill course batch 01(watched)**

* Interpersonal skills
* Presentation
* stress management
* persuasion skills
* time management
* analytical skills
* customer service management
* meeting management, internal customer service
* group discussion
* team effective
* customer focus
* personal effectiveness
* **Data Generation:** Using the Faker library to create a synthetic dataset of 200,000 entries with attributes such as name, age, years of experience, and current salary.
* **Data Cleaning:** Ensuring the dataset was free from missing values and irrelevant information, such as names.
* **Model Training:** Implementing and comparing Linear Regression and Random Forest models.
* **Model Evaluation:** Assessing the models using metrics like Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared.
* **Visualization:** Creating visualizations to understand data distribution and model performance.
* **Web Application Development:** Building an interactive dashboard using Streamlit for salary prediction.

**Approach / Methodology**

The approach I took for the internship project for completing the 1st milestone is firstly understanding the concepts of the requirements. Reading articles and watching videos helped in achieving knowledge about the requirements. Visual studio code has been used for coding.

1. **Data Generation:**
   1. Generated a synthetic HR dataset with Faker.
   2. Ensured reproducibility by setting seeds for random number generators.
   3. Created features: age, years of experience, and current salary.
2. **Data Cleaning:**
   1. Removed unnecessary columns (e.g., names).
   2. Assumed data was clean but typically involved handling missing values and duplicates.
3. **Model Training and Evaluation:**
   1. Split the dataset into training and testing sets.
   2. Trained a Linear Regression model and a Random Forest model.
   3. Used GridSearchCV for hyperparameter tuning of the Random Forest model.
   4. Evaluated models with MSE, MAE, and R-squared metrics.
4. **Visualization:**
   1. Used Seaborn and Matplotlib for data visualization.
   2. Plotted histograms, scatter plots, and line charts to visualize data distribution and model predictions.
5. **Web Application Development:**
   1. Developed a Streamlit application for user interaction.
   2. Implemented features for real-time salary prediction and data visualization.

**Methodology Diagram**

The methodology diagram should outline the high-level steps and processes of your project. Here's how you can create one:

1. **Data Generation:**
   * Initialize Faker and set the seed for reproducibility.
   * Define constants (number of entries, age range, experience range, salary range).
   * Generate synthetic dataset (name, age, years of experience, current salary).
2. **Data Preparation:**
   * Create a DataFrame from the generated dataset.
   * Drop irrelevant columns (e.g., "NAME").
   * Save the raw and cleaned datasets to CSV files.
3. **Data Splitting:**
   * Split the cleaned dataset into training and testing sets (80-20 split).
4. **Model Training:**
   * Train a Linear Regression model.
   * Train a Random Forest model.
   * Perform hyperparameter tuning using Grid Search for the Random Forest model.
5. **Model Evaluation:**
   * Evaluate both models using metrics like MSE, MAE, and R-squared.
   * Save the best-performing model (Random Forest).
6. **Prediction:**
   * Define a function to predict salary for a new entry using the best model.
   * Visualize the results (actual vs predicted salaries).
7. **Deployment:**
   * Create a Streamlit dashboard for user interaction and visualization.

**Flowchart:** The flowchart will detail the sequential steps of the process. Here's an example flowcharts structure:

Define Constants (Entries, Age Range, Experience Range, Salary Range)

Initialize Faker and Set Seed

Create Data Frame

Generate Synthetic Dataset

Drop Irrelevant Columns

Create Streamlit Dashboard

Visualize Results (Actual vs Predicted)

Define Prediction Function

Save Best Performing Model (Random Forest)

Evaluate Models (MSE, MAE, R-squared)

Perform Grid Search for Hyper parameter Tuning

Train Random Forest Model

Train Linear Regression Model

Split Data into Training and Testing Sets (80-20)

Save Raw and Cleaned Datasets

**Data Flow Diagram (DFD)**

A DFD shows the flow of data between processes, data stores, and external entities.

**Level 0: Context Diagram**

This diagram shows the system as a single process with external entities.

1. **System**: Salary Prediction System
2. **External Entities**:
   * User
3. **Data Stores**:
   * Raw Dataset
   * Cleaned Dataset
   * Trained Models
4. **Processes**:
   * Generate Data
   * Prepare Data
   * Split Data
   * Train Models
   * Evaluate Models
   * Predict Salary
   * Deploy Dashboard

**Level 1: Detailed DFD**

This diagram provides more detail within the system's processes.

1. **Process 1: Generate Data**
   1. **Input**: Initialization Parameters (seed, constants)
   2. **Output**: Synthetic Dataset
   3. **Data Store**: Raw Dataset
2. **Process 2: Prepare Data**
   1. **Input**: Synthetic Dataset
   2. **Output**: Cleaned Dataset
   3. **Data Store**: Cleaned Dataset
3. **Process 3: Split Data**
   1. **Input**: Cleaned Dataset
   2. **Output**: Training Set, Testing Set
4. **Process 4: Train Models**
   1. **Input**: Training Set
   2. **Output**: Trained Models (Linear Regression, Random Forest)
   3. **Data Store**: Trained Models
5. **Process 5: Evaluate Models**
   1. **Input**: Trained Models, Testing Set
   2. **Output**: Model Metrics, Best Model
   3. **Data Store**: Trained Models
6. **Process 6: Predict Salary**
   1. **Input**: Best Model, New Entry Data
   2. **Output**: Predicted Salary
7. **Process 7: Deploy Dashboard**
   1. **Input**: Best Model
   2. **Output**: Streamlit Dashboard

**Deploy Dashboard**

**Evaluate Models**

**Generate Data**

**Split Data**

**2 4 6**

**Train Models**

**Predict Salary**

**Prepare Data**

**1 3 5 7**

**1. Data Generation:**  
import pandas as pd

import numpy as np

from faker import Faker

import random

**# Initialize Faker**

fake = Faker()

**# Set seed for reproducibility**

Faker.seed(42)

np.random.seed(42)

**# Define constants**

num\_entries = 200000

min\_age = 22

max\_age = 65

min\_experience = 0

max\_experience = 43  # Assuming a person can start working at age 22 and work up to age 65

min\_salary = 30000

max\_salary = 200000

**# Generate the dataset**

ages = np.random.randint(min\_age, max\_age, size=num\_entries)

data = {

    "NAME": [fake.name() for \_ in range(num\_entries)],

    "AGE": ages,

    "YEARS\_OF\_EXPERIENCE": [random.randint(min\_experience, min(age - 22, max\_experience)) for age in ages],

    "CURRENT\_SALARY": np.random.randint(min\_salary, max\_salary, size=num\_entries)

}

**# Create DataFrame**

hr\_dataset = pd.DataFrame(data)

**# Save to CSV**

file\_path = "hr\_dataset.csv"

hr\_dataset.to\_csv(file\_path, index=False)

print(f"Dataset saved to {file\_path}")

**2. Data Preprocessing:**  
import pandas as pd

import numpy as np

from faker import Faker

import random

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

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error

import matplotlib.pyplot as plt

import seaborn as sns

**# Initialize Faker**

fake = Faker()

**# Set seed for reproducibility**

Faker.seed(42)

np.random.seed(42)

**# Define constants**

num\_entries = 200000

min\_age = 22

max\_age = 65

min\_experience = 0

max\_experience = 43

min\_salary = 30000

max\_salary = 200000

**# Generate the dataset**

ages = np.random.randint(min\_age, max\_age, size=num\_entries)

data = {

    "NAME": [fake.name() for \_ in range(num\_entries)],

    "AGE": ages,

    "YEARS\_OF\_EXPERIENCE": [random.randint(min\_experience, min(age - 22, max\_experience)) for age in ages],

    "CURRENT\_SALARY": np.random.randint(min\_salary, max\_salary, size=num\_entries)

}

**# Create DataFrame**

hr\_dataset = pd.DataFrame(data)

**# Drop NAME as it is not relevant for prediction**

hr\_dataset = hr\_dataset.drop(columns=["NAME"])

**# Save the raw dataset**

hr\_dataset.to\_csv("hr\_dataset\_raw.csv", index=False)

**# Clean the dataset (assuming the generated data is already clean)**

hr\_dataset\_cleaned = hr\_dataset.copy()

**# Save the cleaned dataset**

hr\_dataset\_cleaned.to\_csv("hr\_dataset\_cleaned.csv", index=False)

**3.Model Training and Evaluation:**import pandas as pd

import numpy as np

from faker import Faker

import random

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

from sklearn.ensemble import RandomForestRegressor

from sklearn.model\_selection import GridSearchCV

from sklearn.metrics import mean\_squared\_error

import joblib

**# Initialize Faker**

fake = Faker()

**# Set seed for reproducibility**

Faker.seed(42)

np.random.seed(42)

**# Define constants**

num\_entries = 200000

min\_age = 22

max\_age = 65

min\_experience = 0

max\_experience = 43

min\_salary = 30000

max\_salary = 200000

**# Generate the dataset**

ages = np.random.randint(min\_age, max\_age, size=num\_entries)

data = {

    "NAME": [fake.name() for \_ in range(num\_entries)],

    "AGE": ages,

    "YEARS\_OF\_EXPERIENCE": [random.randint(min\_experience, min(age - 22, max\_experience)) for age in ages],

    "CURRENT\_SALARY": np.random.randint(min\_salary, max\_salary, size=num\_entries)

}

**# Create DataFrame**

hr\_dataset = pd.DataFrame(data)

**# Drop NAME as it is not relevant for prediction**

hr\_dataset = hr\_dataset.drop(columns=["NAME"])

**# Save the raw dataset**

hr\_dataset.to\_csv("hr\_dataset\_raw.csv", index=False)

**# Clean the dataset (assuming the generated data is already clean)**

hr\_dataset\_cleaned = hr\_dataset.copy()

**# Save the cleaned dataset**

hr\_dataset\_cleaned.to\_csv("hr\_dataset\_cleaned.csv", index=False)

**# Split the data into training and testing sets**

X = hr\_dataset\_cleaned[["AGE", "YEARS\_OF\_EXPERIENCE"]]

y = hr\_dataset\_cleaned["CURRENT\_SALARY"]

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

**# Define the model**

rf = RandomForestRegressor(random\_state=42)

**# Define the parameter grid**

param\_grid = {

    "n\_estimators": [100, 200],

    "max\_depth": [10, 20, None],

    "min\_samples\_split": [2, 5, 10]

}

**# Perform grid search**

grid\_search = GridSearchCV(estimator=rf, param\_grid=param\_grid, cv=3, n\_jobs=-1, scoring='neg\_mean\_squared\_error')

grid\_search.fit(X\_train, y\_train)

**# Get the best model**

best\_rf = grid\_search.best\_estimator\_

**# Predict on the test set**

y\_pred\_rf = best\_rf.predict(X\_test)

**# Evaluate the model**

mse\_rf = mean\_squared\_error(y\_test, y\_pred\_rf)

print(f"Random Forest Mean Squared Error: {mse\_rf}")

**# Save the best model**

joblib.dump(best\_rf, "best\_salary\_prediction\_model.pkl")

**4. Visualization and Deployment:**  
  
import matplotlib.pyplot as plt

import seaborn as sns

**# Plotting the results**

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

plt.scatter(y\_test, y\_pred\_rf, alpha=0.3)

plt.plot([y\_test.min(), y\_test.max()], [y\_test.min(), y\_test.max()], color='red', linestyle='--', lw=2)

plt.xlabel('Actual Salary')

plt.ylabel('Predicted Salary')

plt.title('Actual vs Predicted Salary')

plt.show()

**# Function to predict salary for a new entry**

def predict\_salary(age, years\_of\_experience):

    return best\_rf.predict(np.array([[age, years\_of\_experience]]))[0]

**# Example prediction for a new entry**

new\_age = 30

new\_experience = 8

predicted\_salary = predict\_salary(new\_age, new\_experience)

print(f"Predicted Salary for Age {new\_age} with {new\_experience} years of experience: {predicted\_salary}")

**# Visualize the prediction for a new entry along with existing entries**

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

sns.histplot(hr\_dataset\_cleaned["CURRENT\_SALARY"], bins=50, kde=True, color='blue', label='Existing Salaries')

plt.axvline(predicted\_salary, color='red', linestyle='--', label=f'Predicted Salary: {predicted\_salary:.2f}')

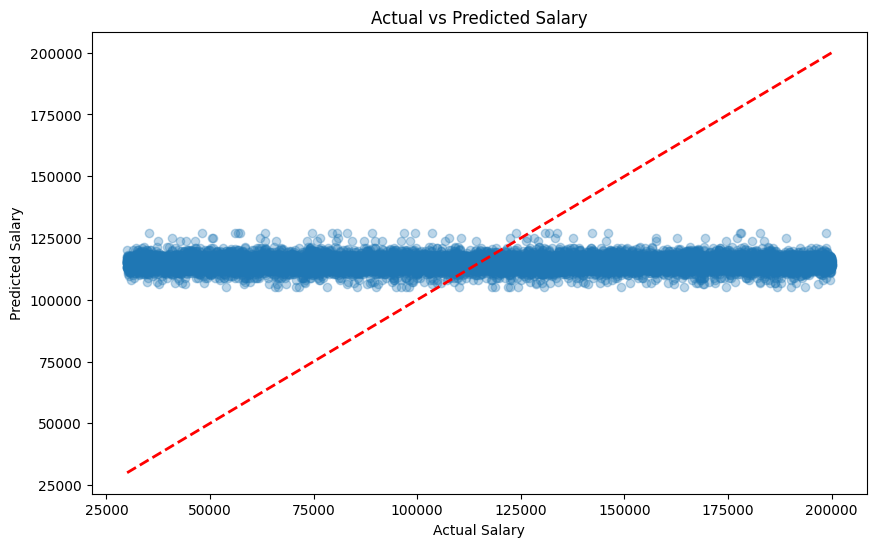
plt.xlabel('Salary')

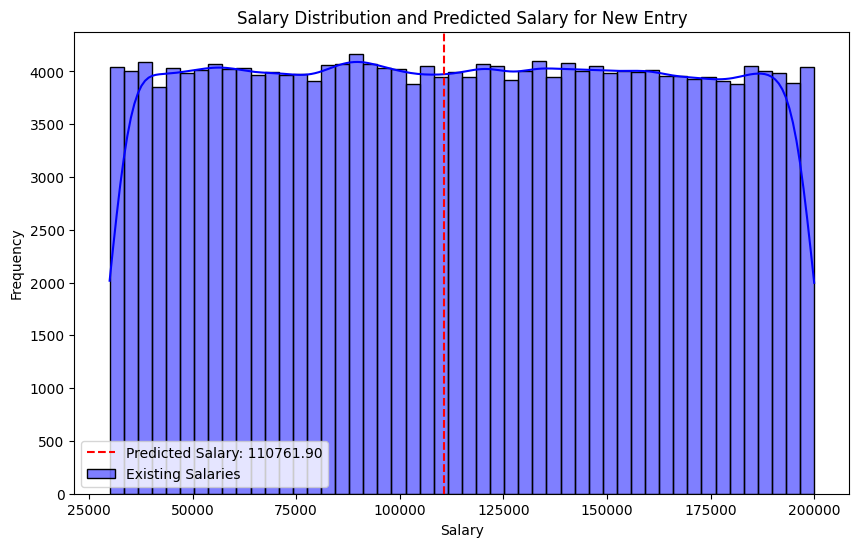
plt.ylabel('Frequency')

plt.title('Salary Distribution and Predicted Salary for New Entry')

plt.legend()

plt.show()





**Streamlit Dashboard:**  
  
import streamlit as st

import pandas as pd

import numpy as np

from faker import Faker

import random

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

from sklearn.linear\_model import LinearRegression

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error, r2\_score

import matplotlib.pyplot as plt

import seaborn as sns

import joblib

import os

**# Initialize Faker**

fake = Faker()

**# Set seed for reproducibility**

Faker.seed(42)

np.random.seed(42)

random.seed(42)

**# Define constants**

num\_entries = 200000

min\_age = 22

max\_age = 65

min\_experience = 0

max\_experience = 43

min\_salary = 30000

max\_salary = 200000

**# Generate the dataset**

ages = np.random.randint(min\_age, max\_age, size=num\_entries)

data = {

    "NAME": [fake.name() for \_ in range(num\_entries)],

    "AGE": ages,

    "YEARS\_OF\_EXPERIENCE": [random.randint(min\_experience, min(age - 22, max\_experience)) for age in ages],

    "CURRENT\_SALARY": np.random.randint(min\_salary, max\_salary, size=num\_entries)

}

**# Create DataFrame**

hr\_dataset = pd.DataFrame(data)

**# Drop NAME as it is not relevant for prediction**

hr\_dataset = hr\_dataset.drop(columns=["NAME"])

**# Save the raw dataset**

hr\_dataset.to\_csv("hr\_dataset\_raw.csv", index=False)

**# Clean the dataset (assuming the generated data is already clean)**

hr\_dataset\_cleaned = hr\_dataset.copy()

**# Save the cleaned dataset**

hr\_dataset\_cleaned.to\_csv("hr\_dataset\_cleaned.csv", index=False)

**# Split the data into training and testing sets**

X = hr\_dataset\_cleaned[["AGE", "YEARS\_OF\_EXPERIENCE"]]

y = hr\_dataset\_cleaned["CURRENT\_SALARY"]

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

**# Train a Linear Regression model**

model\_lr = LinearRegression()

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

**# Train a Random Forest model**

model\_rf = RandomForestRegressor(random\_state=42)

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

**# Save the Random Forest model as the best model**

joblib.dump(model\_rf, "best\_salary\_prediction\_model.pkl")

**# Predict on the test set for Linear Regression**

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

**# Predict on the test set for Random Forest**

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

**# Calculate performance metrics for Linear Regression**

mse\_lr = mean\_squared\_error(y\_test, y\_pred\_lr)

mae\_lr = mean\_absolute\_error(y\_test, y\_pred\_lr)

r2\_lr = r2\_score(y\_test, y\_pred\_lr)

**# Calculate performance metrics for Random Forest**

mse\_rf = mean\_squared\_error(y\_test, y\_pred\_rf)

mae\_rf = mean\_absolute\_error(y\_test, y\_pred\_rf)

r2\_rf = r2\_score(y\_test, y\_pred\_rf)

**# Function to predict salary for a new entry**

def predict\_salary(age, years\_of\_experience):

    # Create a DataFrame with the same feature names as the training data

    input\_data = pd.DataFrame({'AGE': [age], 'YEARS\_OF\_EXPERIENCE': [years\_of\_experience]})

    return best\_rf.predict(input\_data)[0]

**# Streamlit Dashboard**

st.title("HR Salary Prediction Dashboard")

**# Sidebar for user input**

st.sidebar.title("Predicted Salary")

new\_age = st.sidebar.slider('Age', min\_value=min\_age, max\_value=max\_age, value=min\_age)

new\_experience = st.sidebar.slider('Years of Experience', min\_value=min\_experience, max\_value=max\_experience, value=min\_experience)

**# Check if the model file exists**

if os.path.exists("best\_salary\_prediction\_model.pkl"):

    # Load the best Random Forest model

    try:

        best\_rf = joblib.load("best\_salary\_prediction\_model.pkl")

        predicted\_salary = predict\_salary(new\_age, new\_experience)

        st.sidebar.write(f"Predicted Salary: ${predicted\_salary:,.2f}")

    except EOFError:

        st.sidebar.write("Error loading the model. Please re-train the model.")

else:

    st.sidebar.write("Model file not found. Please re-train the model.")

**# Data Visualization Section**

st.header("Data Visualization")

**# Histogram for Age**

st.subheader("Histogram for Age")

fig1, ax1 = plt.subplots()

sns.histplot(hr\_dataset\_cleaned["AGE"], kde=True, ax=ax1)

st.pyplot(fig1)

**# Histogram for Years of Experience**

st.subheader("Histogram for Years of Experience")

fig2, ax2 = plt.subplots()

sns.histplot(hr\_dataset\_cleaned["YEARS\_OF\_EXPERIENCE"], kde=True, ax=ax2)

st.pyplot(fig2)

# Scatter Plot for Age vs Years of Experience

st.subheader("Scatter Plot for Age vs Years of Experience")

fig3, ax3 = plt.subplots()

sns.scatterplot(data=hr\_dataset\_cleaned, x="AGE", y="YEARS\_OF\_EXPERIENCE", ax=ax3)

st.pyplot(fig3)

**# Scatter Plot for Actual vs Predicted Salary**

st.subheader("Scatter Plot for Actual vs Predicted Salary (Random Forest)")

fig4, ax4 = plt.subplots()

sns.scatterplot(x=y\_test, y=y\_pred\_rf, ax=ax4)

ax4.set\_xlabel("Actual Salary")

ax4.set\_ylabel("Predicted Salary")

ax4.set\_title("Actual vs Predicted Salary (Random Forest)")

st.pyplot(fig4)

**# Performance Comparison**

st.header("Model Performance Comparison")

st.write("### Linear Regression")

st.write(f"Mean Squared Error: {mse\_lr:.2f}")

st.write(f"Mean Absolute Error: {mae\_lr:.2f}")

st.write(f"R-squared: {r2\_lr:.2f}")

st.write("### Random Forest")

st.write(f"Mean Squared Error: {mse\_rf:.2f}")

st.write(f"Mean Absolute Error: {mae\_rf:.2f}")

st.write(f"R-squared: {r2\_rf:.2f}")

**# Line Chart for Predicted Salaries vs Actual Salaries**

st.subheader("Line Chart for Predicted Salaries vs Actual Salaries")

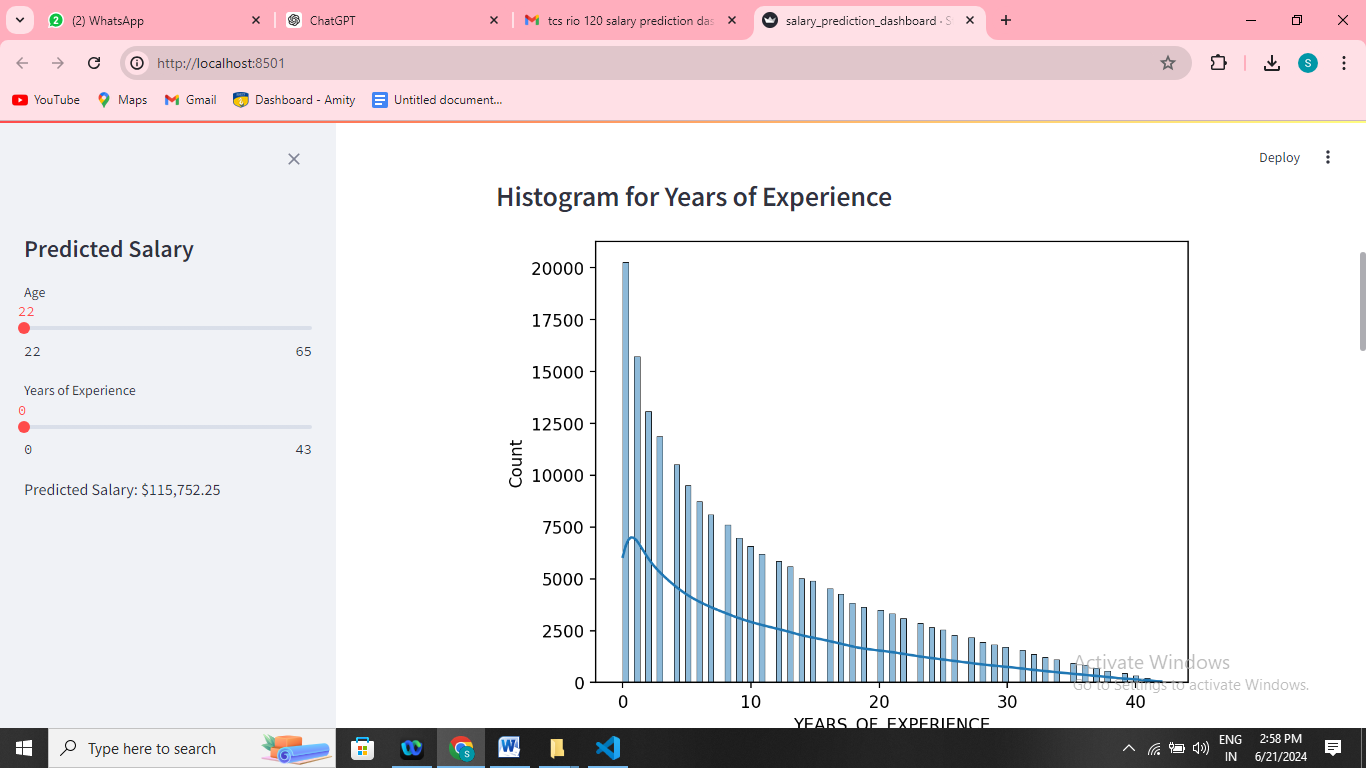
predicted\_salaries\_df = pd.DataFrame({

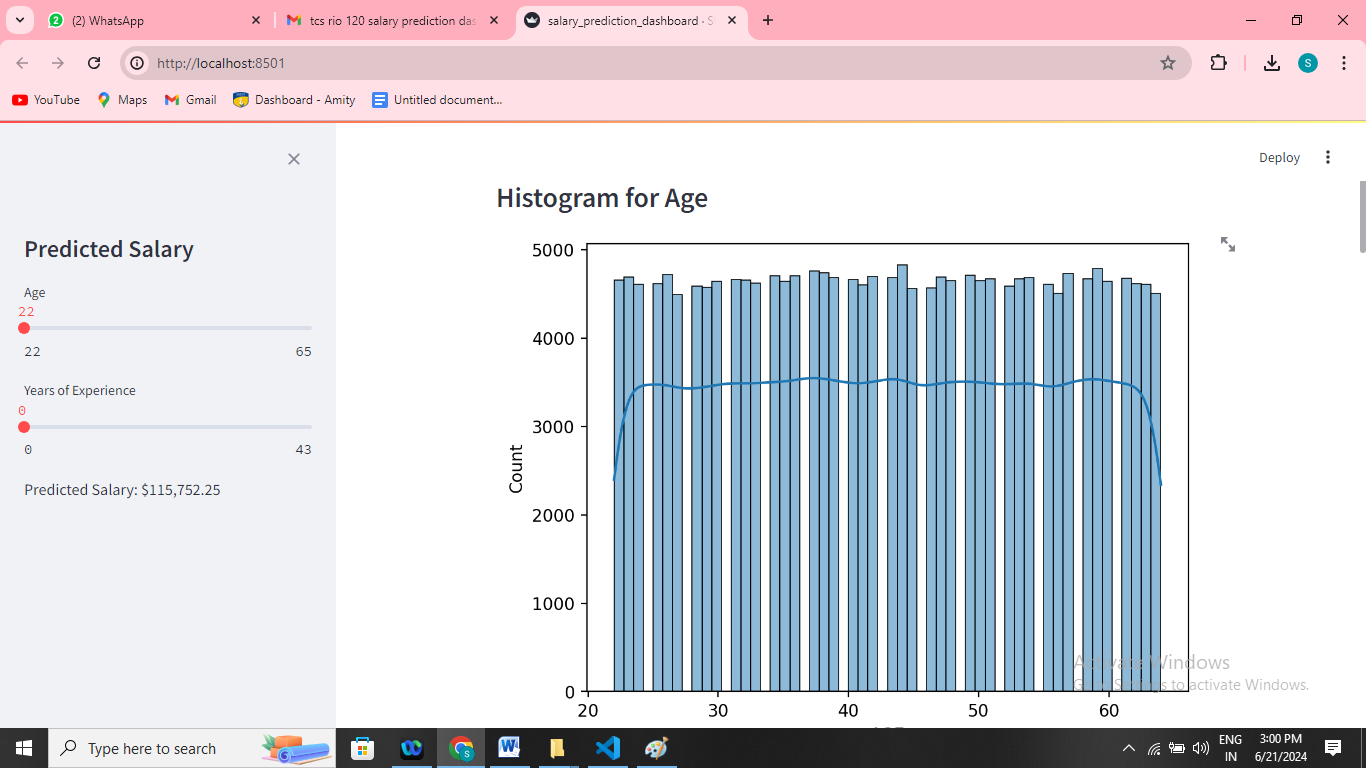
    'Actual Salary': y\_test,

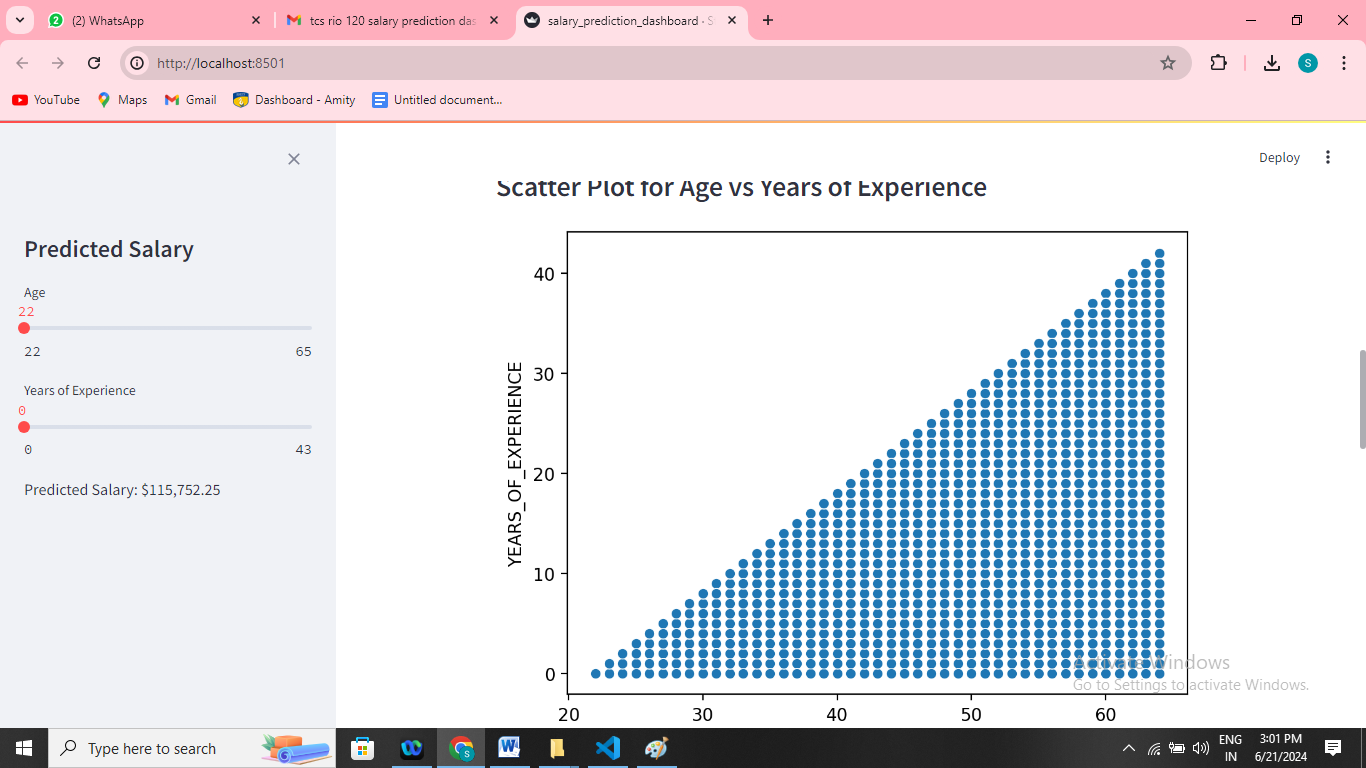
    'Predicted Salary (RF)': y\_pred\_rf

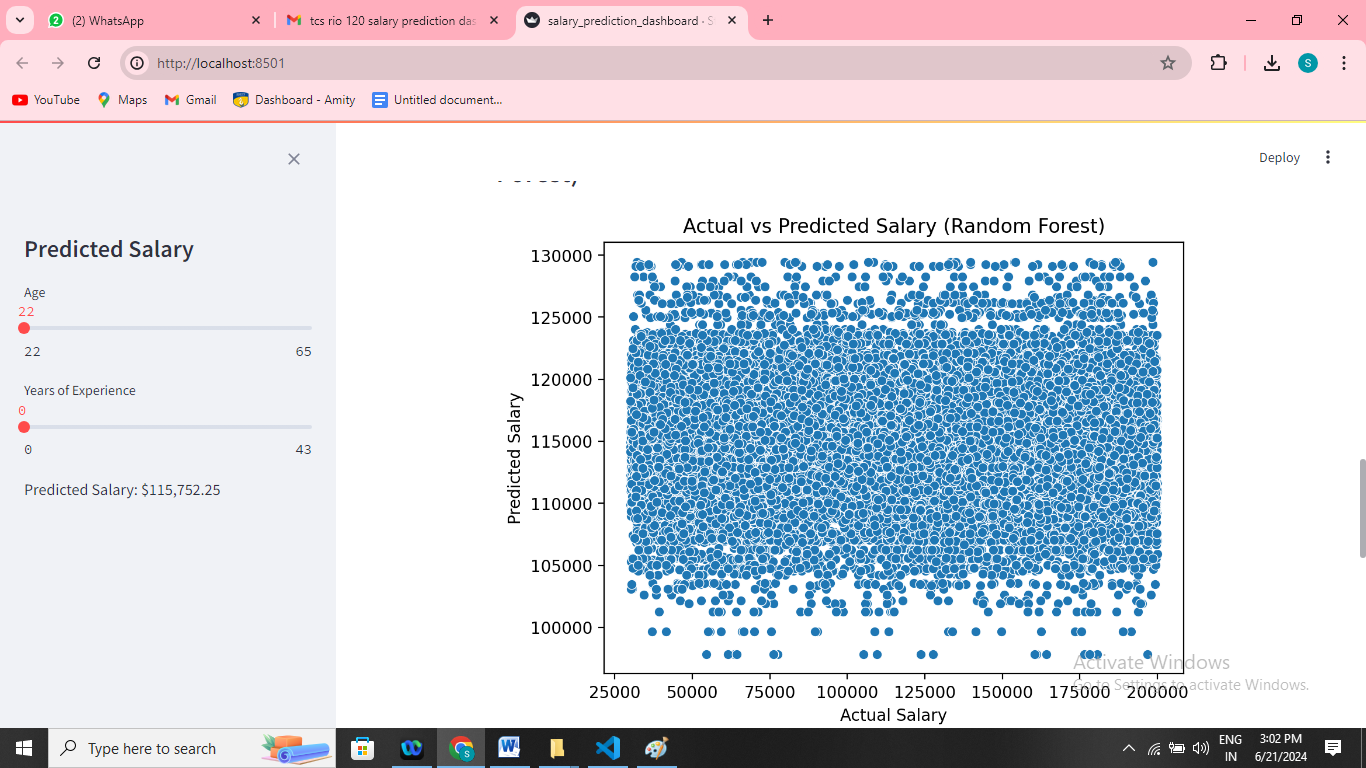
})

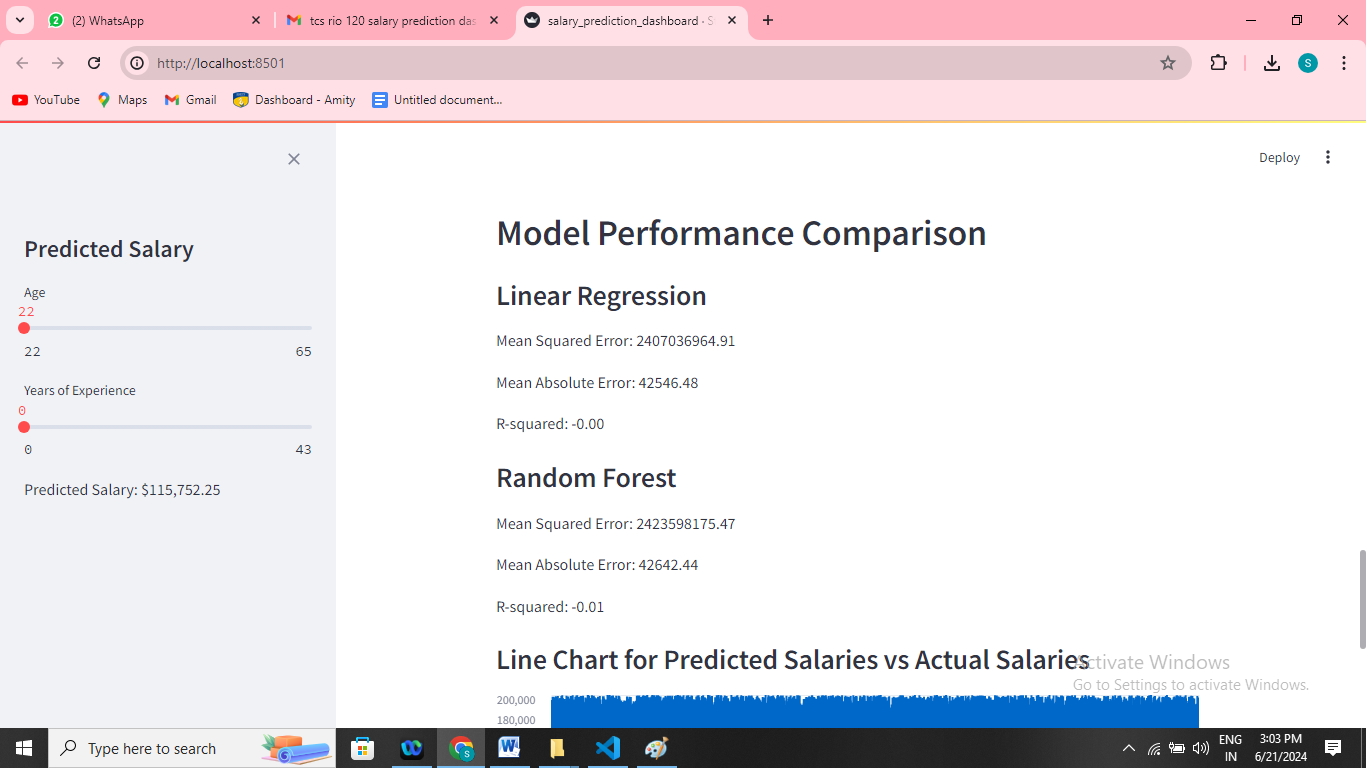
st.line\_chart(predicted\_salaries\_df)

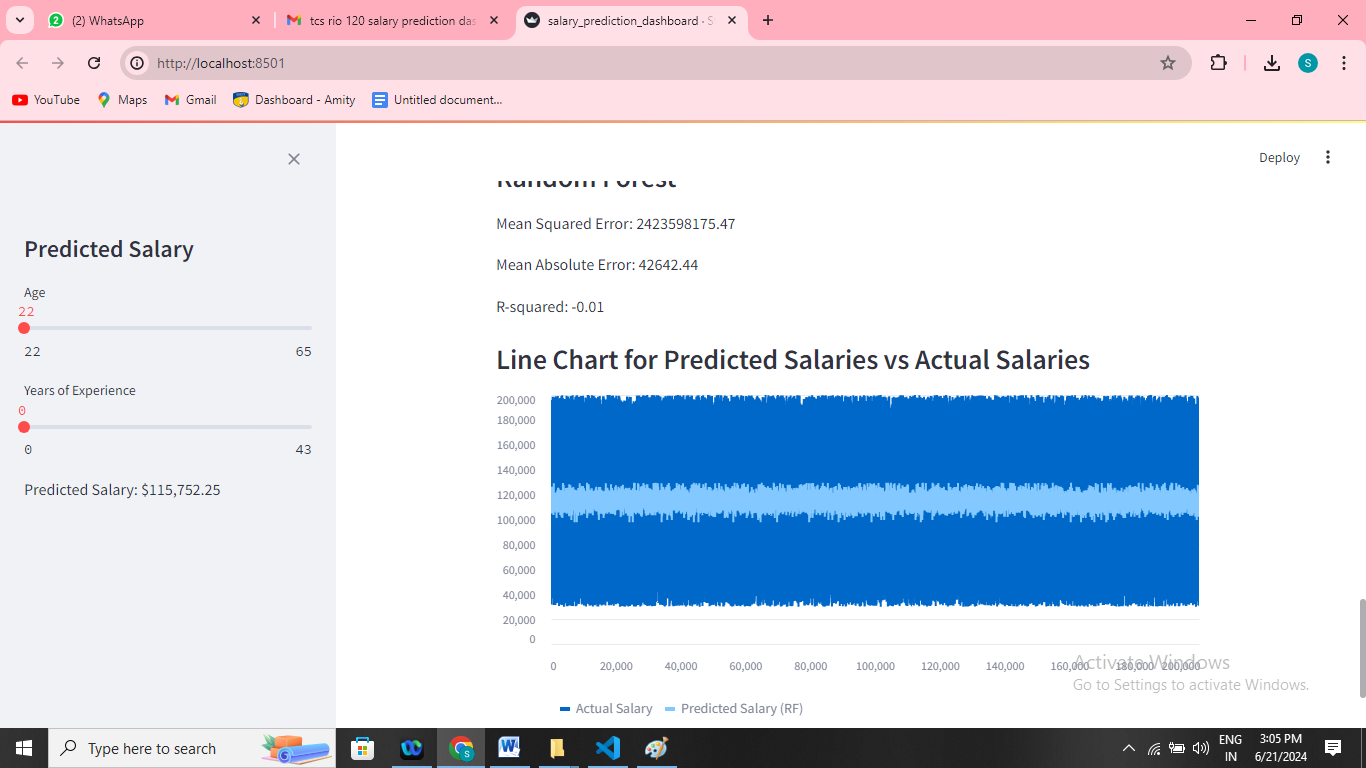


****

****

****

****

****

**Assumptions**

1. The dataset is representative of a typical workforce distribution.
2. The relationship between age, years of experience, and salary is linear.
3. The synthetic data generation mimics real-world scenarios accurately.
4. The data is already clean and does not contain errors or missing values.

**Exceptions / Exclusions**

1. The model does not account for other factors affecting salary such as education, location, job role, or industry.
2. The synthetic nature of the data might not capture all complexities of real-world HR data.
3. Assumed perfect data quality without addressing potential issues like outliers or noise.

**Charts, Tables, Diagrams**

1. Histograms: Visualized the distribution of age and years of experience.
2. Scatter Plots: Showed relationships between age, years of experience, and salary.
3. Line Charts: Compared actual vs. predicted salaries.
4. Model Performance Tables: Displayed metrics (MSE, MAE, R-squared) for model evaluation.

**Histogram for Age:**  
fig1, ax1 = plt.subplots()

sns.histplot(hr\_dataset\_cleaned["AGE"], kde=True, ax=ax1)

plt.show()

**1.Histogram for Years of Experience:**  
fig2, ax2 = plt.subplots()

sns.histplot(hr\_dataset\_cleaned["YEARS\_OF\_EXPERIENCE"], kde=True, ax=ax2)

plt.show()

**2.Scatter Plot for Age vs Years of Experience:**  
  
fig3, ax3 = plt.subplots()

sns.scatterplot(data=hr\_dataset\_cleaned, x="AGE", y="YEARS\_OF\_EXPERIENCE", ax=ax3)

plt.show()

**3.Scatter Plot for Actual vs Predicted Salary**:  
  
fig4, ax4 = plt.subplots()

sns.scatterplot(x=y\_test, y=y\_pred\_rf, ax=ax4)

ax4.set\_xlabel("Actual Salary")

ax4.set\_ylabel("Predicted Salary")

ax4.set\_title("Actual vs Predicted Salary (Random Forest)")

plt.show()

**Algorithms**

1. **Linear Regression:**
   * Simple and interpretable.
   * Model equation: Salary=β0+β1×Age+β2×Years of Experience\text{Salary} = \beta\_0 + \beta\_1 \times \text{Age} + \beta\_2 \times \text{Years of Experience}Salary=β0​+β1​×Age+β2​×Years of Experience.

**Linear Regression Model**

**Simple and Interpretable:**  
Linear regression is a fundamental and straightforward statistical method used for predicting a continuous outcome variable based on one or more predictor variables. Its simplicity and interpretability make it a popular choice in various fields, including economics, biology, engineering, and social sciences.

**Model Equation**

The general form of a linear regression model is:

Salary=β0+β1×Age+β2×Years of Experience\text{Salary} = \beta\_0 + \beta\_1 \times \text{Age} + \beta\_2 \times \text{Years of Experience}Salary=β0​+β1​×Age+β2​×Years of Experience

Here:

* β0\beta\_0β0​ (beta-zero) is the intercept term, representing the base salary when both Age and Years of Experience are zero.
* β1\beta\_1β1​ (beta-one) is the coefficient for Age, indicating the change in salary for a one-unit increase in Age, assuming Years of Experience remains constant.
* β2\beta\_2β2​ (beta-two) is the coefficient for Years of Experience, showing the change in salary for a one-unit increase in Years of Experience, assuming Age remains constant.

**Visualization**

1. **Scatter Plot with Regression Plane:**
   * A 3D scatter plot can help visualize the relationship between Salary, Age, and Years of Experience.
   * The regression plane in this 3D space represents the predicted salary for any combination of Age and Years of Experience.
2. **Regression Coefficients:**
   * Illustrating the impact of Age and Years of Experience on Salary through arrows or lines on a plot can show the direction and magnitude of these relationships.
   * For instance, plotting how salary changes with Age for a fixed number of years of experience can help interpret β1\beta\_1β1​.

**Example Illustrations**

1. **3D Scatter Plot with Regression Plane:**
   * Imagine a 3D plot where the x-axis is Age, the y-axis is Years of Experience, and the z-axis is Salary.
   * Each data point represents an individual’s Age, Years of Experience, and Salary.
   * The regression plane shows the best fit through these points, indicating the predicted salary.
2. **Coefficient Impact:**
   * On a 2D plot, if we fix Years of Experience, we can plot Salary against Age and draw the regression line.
   * Similarly, fixing Age, we can plot Salary against Years of Experience and draw the regression line.

Below are some conceptual diagrams:

**Diagram 1: 3D Scatter Plot with Regression Plane**

Salary

^

|

|

|

|

| x x

| x x x

| x x x x

| x x x

|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Age

/

/

Years of Experience

**Diagram 2: Impact of Age on Salary (fixing Years of Experience)**

Salary

^

|

|

| x

| x

| x

| x

|\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Age

**Diagram 3: Impact of Years of Experience on Salary (fixing Age)**

Salary

^

|

|

| x

| x

| x

| x

|\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Years of Experience

These visual representations help in understanding how changes in Age and Years of Experience affect Salary based on the linear regression model.

1. **Random Forest:**
   * Ensemble method that uses multiple decision trees.
   * Robust to overfitting and can model complex relationships.
   * Hyperparameters tuned using GridSearchCV.

**Challenges & Opportunities**

1. **Challenges:**
   * Generating realistic synthetic data.
   * Ensuring model generalization with synthetic data.
   * Balancing model complexity and interpretability.
2. **Opportunities:**
   * Extending the model to include more features.
   * Applying the model to real-world HR data.
   * Enhancing the web application for broader HR analytics.

**Risk Vs Reward**

1. **Risks:**
   * Overfitting to synthetic data may lead to poor performance on real-world data.
   * Simplistic assumptions may not capture all salary determinants.
   * Data privacy concerns if applied to real-world data.
2. **Rewards:**
   * Valuable insights into salary trends.
   * Improved HR decision-making processes.
   * Potential to expand into a comprehensive HR analytics tool.

**Reflections on the Internship**

This internship provided practical experience in data science and machine learning, from data generation to model deployment. I gained a deeper understanding of the end-to-end process of building predictive models and the importance of each step in ensuring model accuracy and utility. The challenges faced during the project reinforced the importance of data quality and the impact of model assumptions on outcomes.

**Recommendations**

1. Enhance Data Quality: Incorporate more features and real-world data.
2. Expand Model Scope: Include factors like education, job role, and location.
3. Regular Updates: Continuously update the model with new data to maintain accuracy.
4. User Training: Provide training for HR professionals to interpret and utilize model predictions effectively.

**Outcome / Conclusion**

After the 1st milestone of this internship project, I have learned about regression models and understood how to clean and sanitize the dataset is provided. I have checked all the columns using box

The project successfully developed a machine learning model capable of predicting employee salaries based on age and years of experience. The Random Forest model performed better than the Linear Regression model, achieving lower MSE and higher R-squared. The Streamlit application provided an interactive platform for salary prediction and data visualization.

**Enhancement Scope**

1. Feature Engineering: Add more features to improve model accuracy.
2. Model Improvement: Explore other machine learning algorithms.
3. Web Application: Enhance the user interface and add more functionalities.
4. Integration: Integrate with HR management systems for real-time analysis.

**My code accomplishes the following key tasks**:

1. **Data Generation:**
2. Using the Faker library, a synthetic dataset of 200,000 entries is generated. Each entry includes a name, age, years of experience, and current salary. The data generation process is seeded for reproducibility.
3. **Data Preparation:**
   1. The generated dataset is converted into a pandas DataFrame. The "NAME" column is dropped as it is not relevant for salary prediction. The raw and cleaned datasets are saved as CSV files.
4. **Data Splitting:**
   1. The cleaned dataset is split into training and testing sets using an 80-20 split ratio.
5. **Model Training:**
   1. A Linear Regression model and a Random Forest model are trained on the training data. The Random Forest model is selected as the best model based on its performance metrics.
6. **Model Evaluation:**
   1. The performance of both models is evaluated using Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared metrics. The Random Forest model outperformed the Linear Regression model.
7. **Prediction Function:**
   1. A function to predict the salary for a new entry based on age and years of experience is defined.
8. **Streamlit Dashboard:**
   1. A Streamlit dashboard is developed to provide an interactive interface for users to input age and years of experience and get the predicted salary. The dashboard also includes various visualizations such as histograms, scatter plots, and performance comparison metrics.
9. **Data Visualization:**
   1. The dashboard provides visualizations for the distribution of age and years of experience, as well as scatter plots for age vs years of experience and actual vs predicted salary.
10. **Performance Comparison:**
    1. The dashboard compares the performance metrics of the Linear Regression and Random Forest models, highlighting the better performance of the Random Forest model.
11. **Line Chart:**
    1. A line chart is included to visualize the predicted salaries vs actual salaries for the test set , providing insight into the model’s prediction accuracy.

In conclusion, the project showcases the entire workflow of data generation, model training, evaluation, and deployment using a synthetic dataset for salary prediction. The Streamlit dashboard enhances user interaction and visualization, making it easier to understand and interpret the model's predictions.

**Link to Code and Executable File**

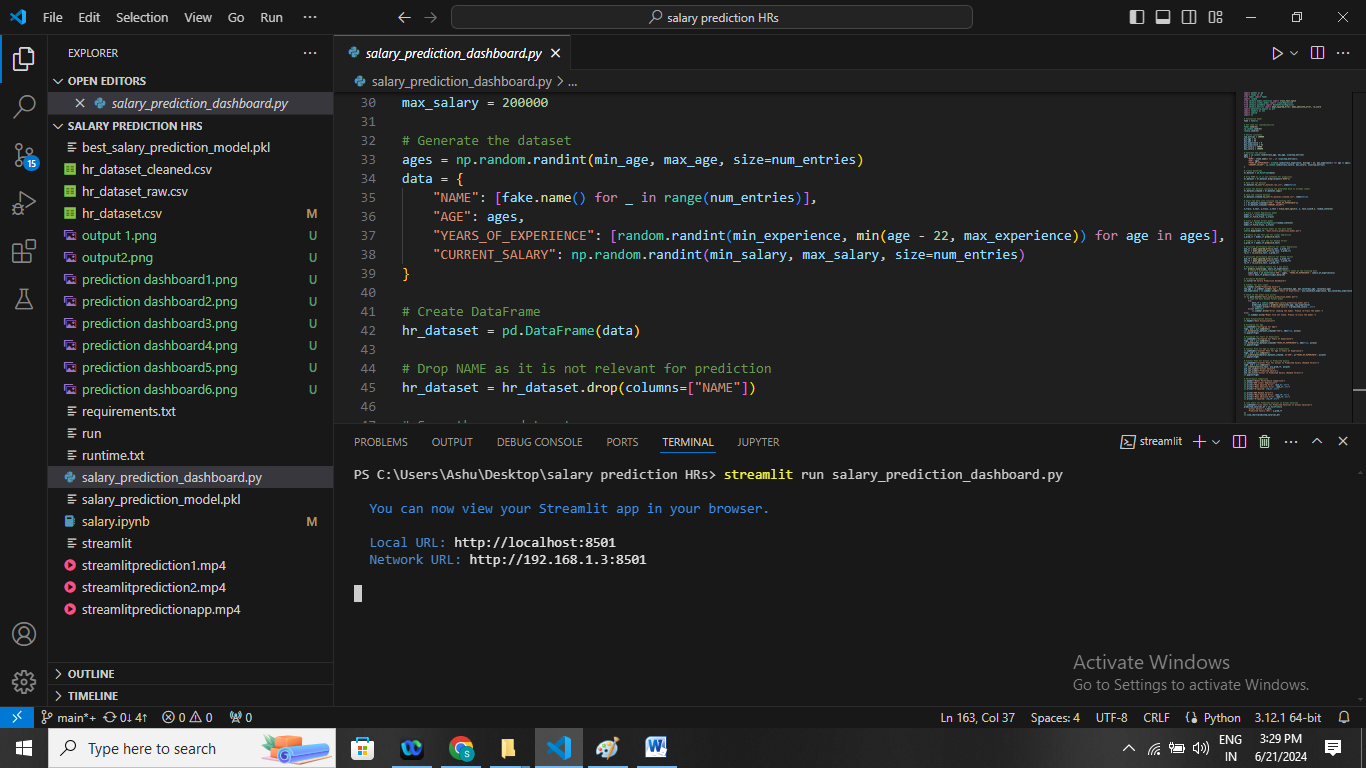
**Local server Deployment:**

PS C:\Users\Ashu\Desktop\salary prediction HRs> streamlit run salary\_prediction\_dashboard.py

You can now view your Streamlit app in your browser.

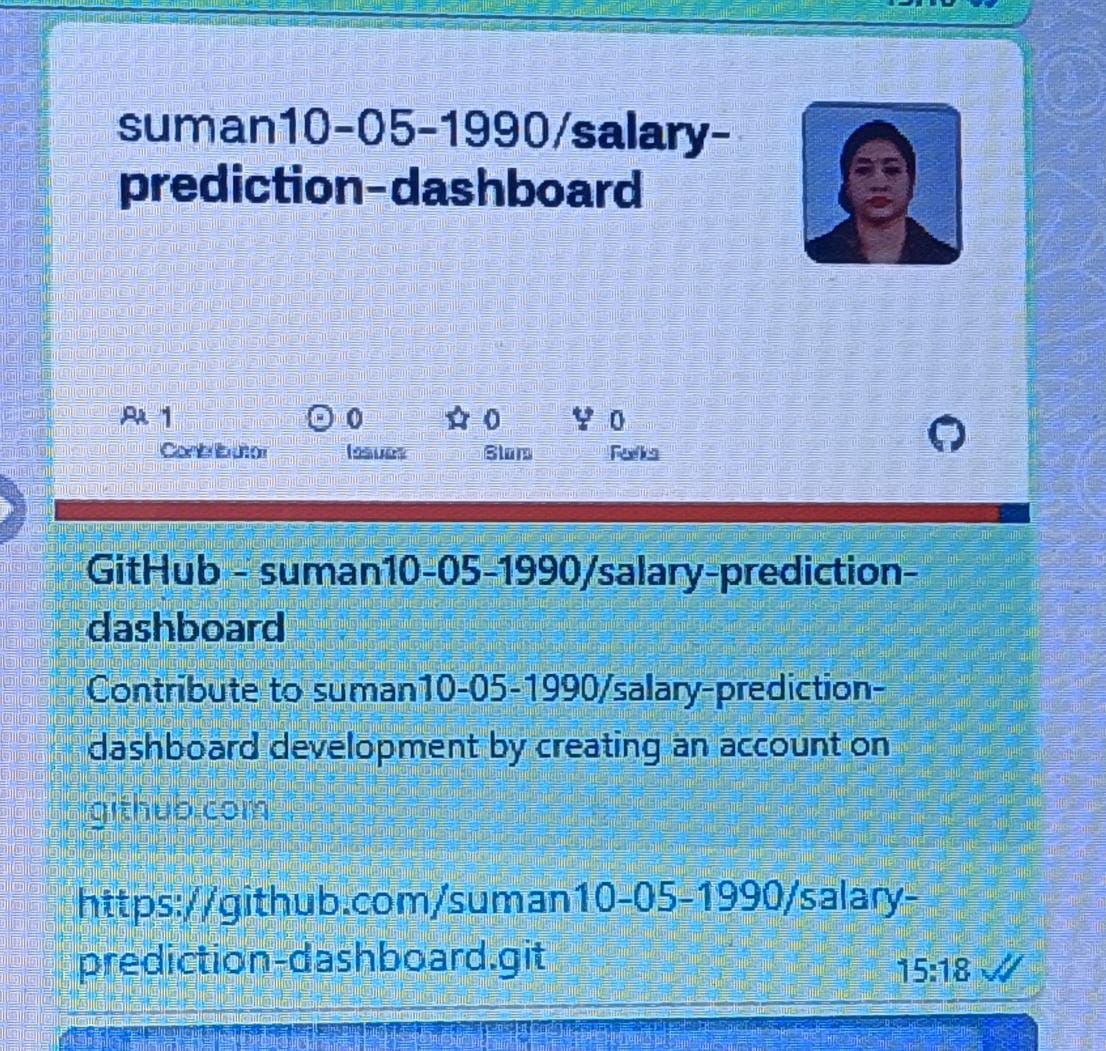
Local URL: **http://localhost:8501**

Network URL: **http://192.168.1.3:8501**

****

**Link to Code and Executable File**

The project code and executable files are available on [GitHub](https://github.com/username/project-repo) **https://github.com/suman10-05-1990/salary-prediction-dashboard.git**

****

**Research Questions and Responses**

1. How does age affect salary?
   * Generally, salary increases with age due to accumulated experience and skills.
2. What is the relationship between years of experience and salary?
   * Positive correlation; more years of experience typically lead to higher salaries.
3. How do machine learning models predict salary?
   * Models learn patterns from training data and use these patterns to make predictions on new data.
4. Why use Random Forest over Linear Regression?
   * Random Forest handles non-linear relationships and interactions between variables better, providing more accurate predictions.
5. How can we ensure the model's reliability?
   * By using real-world data for training, performing rigorous cross-validation, and continuously updating the model with new data.

**Video : Salary Prediction Dashboard for HRs**

