Salary Prediction using Ensemble Learning.

Declaration by Candidate

We hereby declare that this mini-project report titled **"Salary Prediction"** is a result of our own efforts and research. The project was carried out under the guidance of Project Coordinator and has not been submitted for any other purpose.

Date:

Name: Aanand kumar

Email ID: ak4325465@gmail.com

Certificate by Project Coordinator

This is to certify that the project report titled **"Salary Prediction"** submitted by **Aanand**, **BTech 3rd Year (Semester 5th)**, has been successfully completed under my guidance.

Date:

Project Coordinator:

Signature:

Acknowledgement

I want to take a moment to sincerely thank my project coordinator, **[Rohit Sir]**, for their invaluable guidance, patience, and support throughout this project. I am also grateful to the Department of IBM for providing the necessary resources. Lastly, I extend my heartfelt thanks to my friends and family for their encouragement during this journey.

Abstract

This project focuses on predicting employee salaries using machine learning. Since salary depends on many factors like education, skills, and experience, traditional models may not always give accurate results. To solve this, we use **ensemble learning methods** such as Random Forest and

Gradient Boosting, which combine multiple models to improve accuracy. The project includes data preprocessing, model training, and evaluation using metrics like Mean Squared Error and R² score. The results show that ensemble learning gives better predictions compared to single models, making it useful for real-world HR and recruitment systems.

Table of Contents

List of Figures

List of Symbols/Abbreviations

1. **RMSE** →	Root Mean Squared Error
2. **R^2 (R2 Score)** →	Coefficient of Determination
3. **CSV** →	Comma Separated Values
4. **pkl** →	Pickle file format
5. **X** →	Feature variables (input)
6. **y** →	Target variable (output: Salary)

Chapter 1: Introduction

Predicting employee salaries is important for recruitment, compensation, and workforce planning. Salary depends on factors like experience, education, job role, and location. This project uses **ensemble learning (Random Forest)** to improve prediction accuracy. It includes data preprocessing, model training, evaluation (RMSE & R^2), and visualization. An interactive tool allows users to estimate their expected salary based on input features.

- Salary prediction helps companies and employees estimate fair compensation.
- The project uses historical data to train a model that predicts salaries based on input features.
- Ensemble learning methods improve prediction accuracy by combining multiple models.

Chapter 2: Requirements and Analysis

1. **Functional Requirements:**
   - ➢ Accept input features like experience, education, job role.
   - ➢ Predict salary based on trained model.
2. **Non-Functional Requirements:**
   - ➢ Accurate predictions.
   - ➢ Efficient and fast computation.
3. **Analysis:**
   - ➢ Data preprocessing is required to handle missing values and categorical data.
   - ➢ Random Forest Regressor or other ensemble models can provide better predictions than single models.

❖ **Actors:**

- **User / Employee**
  The person who inputs their details (Experience, Education, Job Role, Location) to get a salary prediction.
- **System / Software**
  The Python program that processes the input, runs the trained **Random Forest model**, and predicts the salary.
- *(Optional)* **Administrator / Developer**
  The person who maintains the dataset, trains the model, and updates the system if needed.

❖ **Description:** Predicts employee salaries using Random Forest from experience, education, job role, and location input.

Chapter 3: Software Design

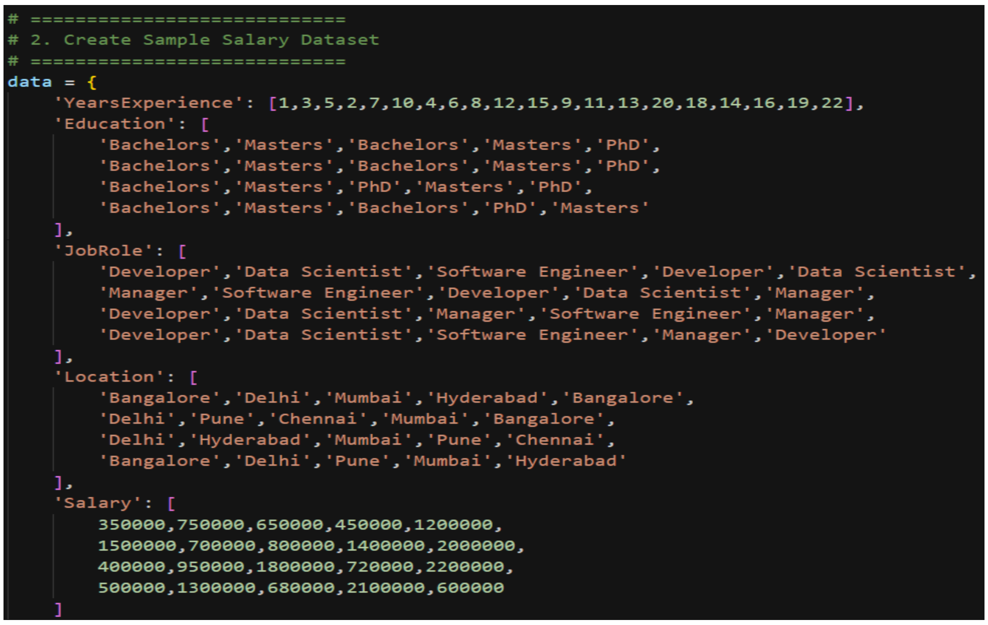
Data Flow Diagrams

Data → Features: Experience, Education, Job Role, Location, Salary
Preprocessing → Encoding categorical variables
Model → Random Forest Regressor
Evaluation → RMSE, R^2 Score
Prediction → Interactive tool for user input

Chapter 4: Database Design

❖ Dataset

- Features:
  - YearsExperience → Numeric
  - Education → Bachelors, Masters, PhD
  - JobRole → Developer, Data Scientist, Manager, Software Engineer
  - Location → Bangalore, Delhi, Mumbai, Hyderabad, Pune, Chennai
- Target:
  - Salary → Annual salary in ₹
- Sample size: 20 records (created as a CSV)



```
# ============================
# 2. Create Sample Salary Dataset
# ============================
data = {
    'YearsExperience': [1,3,5,2,7,10,4,6,8,12,15,9,11,13,20,18,14,16,19,22],
    'Education': [
        'Bachelors','Masters','Bachelors','Masters','PhD',
        'Bachelors','Masters','Bachelors','Masters','PhD',
        'Bachelors','Masters','PhD','Masters','PhD',
        'Bachelors','Masters','Bachelors','PhD','Masters'
    ],
    'JobRole': [
        'Developer','Data Scientist','Software Engineer','Developer','Data Scientist',
        'Manager','Software Engineer','Developer','Data Scientist','Manager',
        'Developer','Data Scientist','Manager','Software Engineer','Manager',
        'Developer','Data Scientist','Software Engineer','Manager','Developer'
    ],
    'Location': [
        'Bangalore','Delhi','Mumbai','Hyderabad','Bangalore',
        'Delhi','Pune','Chennai','Mumbai','Bangalore',
        'Delhi','Hyderabad','Mumbai','Pune','Chennai',
        'Bangalore','Delhi','Pune','Mumbai','Hyderabad'
    ],
    'Salary': [
        350000,750000,650000,450000,1200000,
        1500000,700000,800000,1400000,2000000,
        400000,950000,1800000,720000,2200000,
        500000,1300000,680000,2100000,600000
    ]
```

Figure 1.1

Experience	Education	Job Role	Salary
3 years	B.Tech	Developer	5,50,000
5 years	M.Tech	Analyst	7,00,000

Visual Suggestion: Small table image or dataset icon.

Chapter 5: Data Preprocessing

a. Handling missing values
b. Encoding categorical variables
c. Normalization / Scaling (if applied)
d. Visualization: (optional) histogram or bar chart of salary distribution

Chapter 6: Methodology

1. Data Collection
2. Data Cleaning & Preprocessing
3. Train-Test Split
4. Model Selection (Random Forest / Gradient Boosting)
5. Model Training

6. Evaluation (RMSE, R² Score)
   Visual Suggestion: Flowchart or step diagram

Chapter 7: Model Used

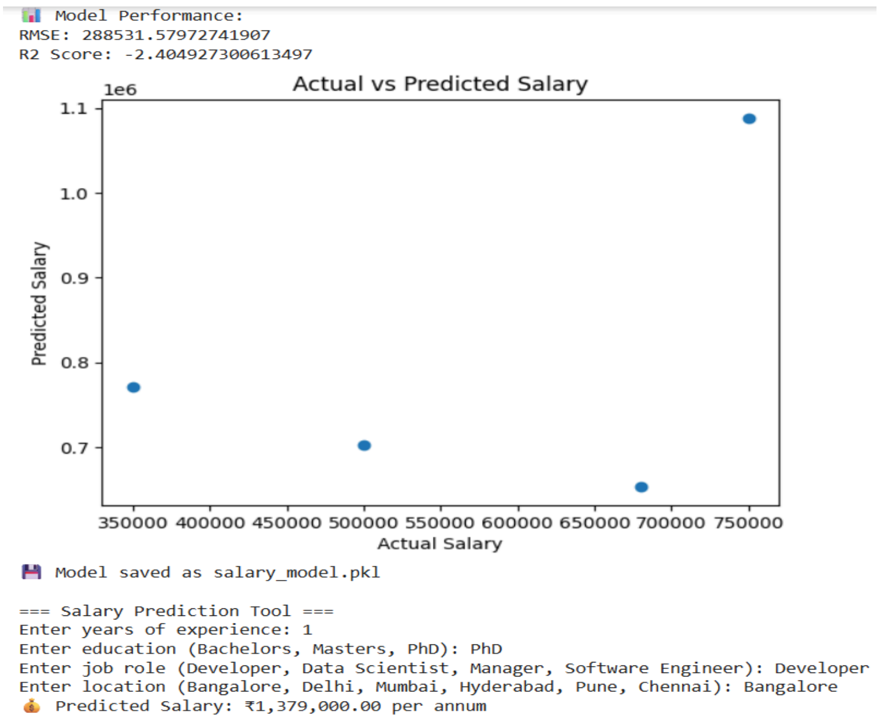
- **Random Forest Regressor (Ensemble Learning)**
- Reduces overfitting and improves prediction accuracy
- Optional comparison with other models: Linear Regression, Decision Tree
  Visual Suggestion: Tree icon or ensemble diagram

Chapter 8: Tools & Libraries

- Python
- Pandas, NumPy
- Scikit-learn (for ML models)
- Matplotlib / Seaborn (for visualization)
- Pickle (for saving model)

Chapter 13: Testing

- The dataset was split into **training (80%)** and **testing (20%)** for validation.
- Model performance was tested using **RMSE** and **R² Score**.
- Predicted salaries were compared with actual values to ensure accuracy.



```
📊 Model Performance:
RMSE: 288531.57972741907
R2 Score: -2.404927300613497
```

```
💾 Model saved as salary_model.pkl

=== Salary Prediction Tool ===
Enter years of experience: 1
Enter education (Bachelors, Masters, PhD): PhD
Enter job role (Developer, Data Scientist, Manager, Software Engineer): Developer
Enter location (Bangalore, Delhi, Mumbai, Hyderabad, Pune, Chennai): Bangalore
💰 Predicted Salary: ₹1,379,000.00 per annum
```

Figure 2.1

Chapter 9: Results / Evaluation

- **Metrics:**
  - RMSE: [Add value]
  - R² Score: [Add value]
- Predicted vs Actual Salary visualization
  Visual Suggestion: Line chart showing predicted vs actual salaries

Chapter 10: Feature Importance

- Most influential features: Experience, Education, Job Role

- Visualization: Bar chart of feature importance
  Visual Suggestion: Bar chart or horizontal bars showing importance.

Chapter 11: Conclusion

- Model accurately predicts salaries for employee profiles
- Ensemble learning enhances prediction performance
- Future work: Larger dataset, additional features for better accuracy
  Visual Suggestion: Lightbulb or growth icon

Chapter 12: References

- Dataset source
- Python, Scikit-learn, Matplotlib documentation
- Research papers / websites used
  Visual Suggestion: Book or document icon.

Appendices

· **Appendix A:** Source Code

```
# ****************************
# 1. Import Libraries
# ****************************
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import mean_squared_error, r2_score
```

```
import pickle

# ******************************
# 2. Create Sample Salary Dataset
# ******************************
data = {
   'YearsExperience': [1,3,5,2,7,10,4,6,8,12,15,9,11,13,20,18,14,16,19,22],
   'Education': [
      'Bachelors','Masters','Bachelors','Masters','PhD',
      'Bachelors','Masters','Bachelors','Masters','PhD',
      'Bachelors','Masters','PhD','Masters','PhD',
      'Bachelors','Masters','Bachelors','PhD','Masters'
   ],
   'JobRole': [
      'Developer','Data Scientist','Software Engineer','Developer','Data Scientist',
      'Manager','Software Engineer','Developer','Data Scientist','Manager',
      'Developer','Data Scientist','Manager','Software Engineer','Manager',
      'Developer','Data Scientist','Software Engineer','Manager','Developer'
   ],
   'Location': [
      'Bangalore','Delhi','Mumbai','Hyderabad','Bangalore',
      'Delhi','Pune','Chennai','Mumbai','Bangalore',
      'Delhi','Hyderabad','Mumbai','Pune','Chennai',
      'Bangalore','Delhi','Pune','Mumbai','Hyderabad'
   ],
   'Salary': [
      350000,750000,650000,450000,1200000,
      1500000,700000,800000,1400000,2000000,
      400000,950000,1800000,720000,2200000,
      500000,1300000,680000,2100000,600000
   ]
}
```

```
df = pd.DataFrame(data)

# Save dataset
df.to_csv("salary_data.csv", index=False)

print("✅ Salary dataset created & saved as salary_data.csv")

# ****************************
# 3. Encode Categorical Columns
# ****************************
le_education = LabelEncoder()
le_jobrole = LabelEncoder()
le_location = LabelEncoder()

df['Education'] = le_education.fit_transform(df['Education'])
df['JobRole'] = le_jobrole.fit_transform(df['JobRole'])
df['Location'] = le_location.fit_transform(df['Location'])

# ***************************
# 4. Split Data
# ***************************
X = df.drop('Salary', axis=1)
y = df['Salary']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# ***************************
# 5. Train Model
# ***************************
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
```

```
# ****************************
# 6. Evaluate Model
# ****************************
y_pred = model.predict(X_test)

# Option 1: If sklearn ≥ 0.22
rmse = mean_squared_error(y_test, y_pred) ** 0.5

# Option 2: If sklearn < 0.22
# rmse = mean_squared_error(y_test, y_pred) ** 0.5

r2 = r2_score(y_test, y_pred)

print("📊 Model Performance:")
print("RMSE:", rmse)
print("R2 Score:", r2)

# ****************************
# 7. Visualization
# ****************************
plt.scatter(y_test, y_pred)
plt.xlabel("Actual Salary")
plt.ylabel("Predicted Salary")
plt.title("Actual vs Predicted Salary")
plt.show()

# ****************************
# 8. Save Model
# ****************************
pickle.dump(model, open('salary_model.pkl', 'wb'))
print("💾 Model saved as salary_model.pkl")
```

```
# **************************
# 9. Interactive Prediction
# **************************
print("\n=== Salary Prediction Tool ===")
years_experience = float(input("Enter years of experience: "))
education = input("Enter education (Bachelors, Masters, PhD): ")
job_role = input("Enter job role (Developer, Data Scientist, Manager, Software Engineer): ")
location = input("Enter location (Bangalore, Delhi, Mumbai, Hyderabad, Pune, Chennai): ")

# Encode inputs
education_encoded = le_education.transform([education])[0]
jobrole_encoded = le_jobrole.transform([job_role])[0]
location_encoded = le_location.transform([location])[0]

# Create feature array
input_features = np.array([[years_experience, education_encoded, jobrole_encoded,
location_encoded]])

# Predict salary
predicted_salary = model.predict(input_features)
print(f"💰 Predicted Salary: ₹{predicted_salary[0]:,.2f} per annum")
```