CAPSTONE PROJECT

EMPLOYEE SALARY PREDICTION USING RANDOM FOREST AND XGBOOST

Presented By:

1. Student Name : Sahana Priya G
2. College Name : Dayananda Sagar University
3. Department : CSE - AIML
4. Email id : sahanacseai@gmail.com
5. AICTE Student Id : STU672101c3f06251730216387

edunet
foundation

OUTLINE

- Problem Statement
- System Development Approach
- Algorithm & Deployment
- Result
- Conclusion
- Future Scope
- References

PROBLEM STATEMENT

In today's competitive job market, determining fair and accurate employee compensation is a significant challenge for HR professionals and hiring managers. Salaries often depend on various factors such as experience, education level, job role, and geographic location. Traditional approaches to estimating salaries are subjective and may lead to inconsistencies or bias. This project aims to develop a machine learning model that can predict employee salaries based on relevant features in a data-driven and consistent manner. The objective is to build a deployable web application that assists recruiters and job seekers in estimating salaries transparently, using real-world data and modern algorithms like Random Forest and XGBoost.

SYSTEM APPROACH

The system follows a structured machine learning pipeline, starting from data ingestion and preprocessing, followed by model training, evaluation, and deployment through a user-friendly interface. The goal is to create a robust and scalable solution that allows real-time salary predictions based on employee features such as experience, education level, job role, and location.

System Requirements

Operating System: Linux, Windows
Development Environment: Google Colab / Jupyter Notebook
Deployment Platform: Hugging Face Spaces
Web Interface: Gradio
Browser: Any modern browser (Chrome, Edge, Firefox)

Libraries Required

pandas – for data manipulation and cleaning
numpy – for numerical operations
scikit-learn – for machine learning models and metrics
xgboost – for high-performance gradient boosting
joblib / skops – for model serialization
Gradio – for creating the web app interface
matplotlib / seaborn – for visualizing data and model results

ALGORITHM & DEPLOYMENT

This section outlines the step-by-step procedure followed to develop and deploy the machine learning model for employee salary prediction :

Step 1: Data Collection & Exploration

- A structured dataset containing features like experience, education, job role, location, and salary was used.
- Initial exploration included checking for missing values, data types, and value distributions.

Step 2: Data Preprocessing

- Categorical variables (e.g., education, job title) were encoded using one-hot encoding.
- Null or inconsistent values were removed.
- Features and target variable (salary) were separated for training.

Step 3: Model Building

- Two models were trained and evaluated:
  - **Random Forest Regressor** – ensemble-based, interpretable
  - **XGBoost Regressor** – high-performance gradient boosting
- The data was split into training and test sets (80/20 split).
- Models were evaluated using **Mean Absolute Error (MAE)** and **R² Score**.

Step 4: Model Selection & Saving

- The model with the best performance (typically XGBoost) was saved using joblib or skops for secure serialization:

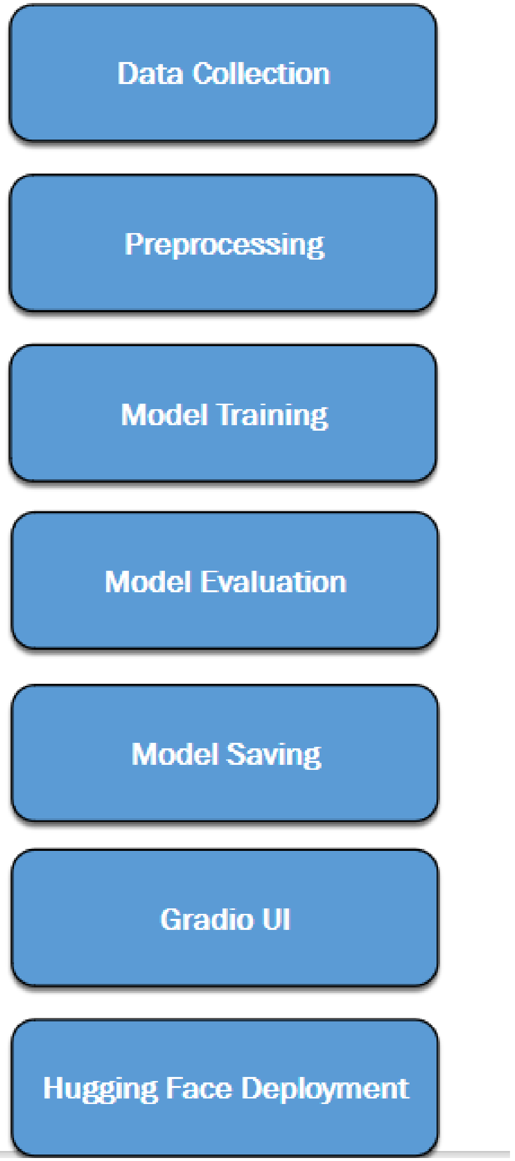
```
Python code
import joblib
joblib.dump(model, "model.pkl")
```

Step 5: Web App Development (Gardio)

- A user interface was built using **Gardio**, allowing users to input employee details (experience, job title, etc.).
- The trained model is loaded, and predictions are made in real-time.

Step 6: Deployment on Hugging Face Spaces

- Files uploaded:
  - app.py
  - Salary prediction.pkl
  - requirements.txt (library list)
- SDK selected: **Gradio**



Data Collection

Preprocessing

Model Training

Model Evaluation

Model Saving

Gradio UI

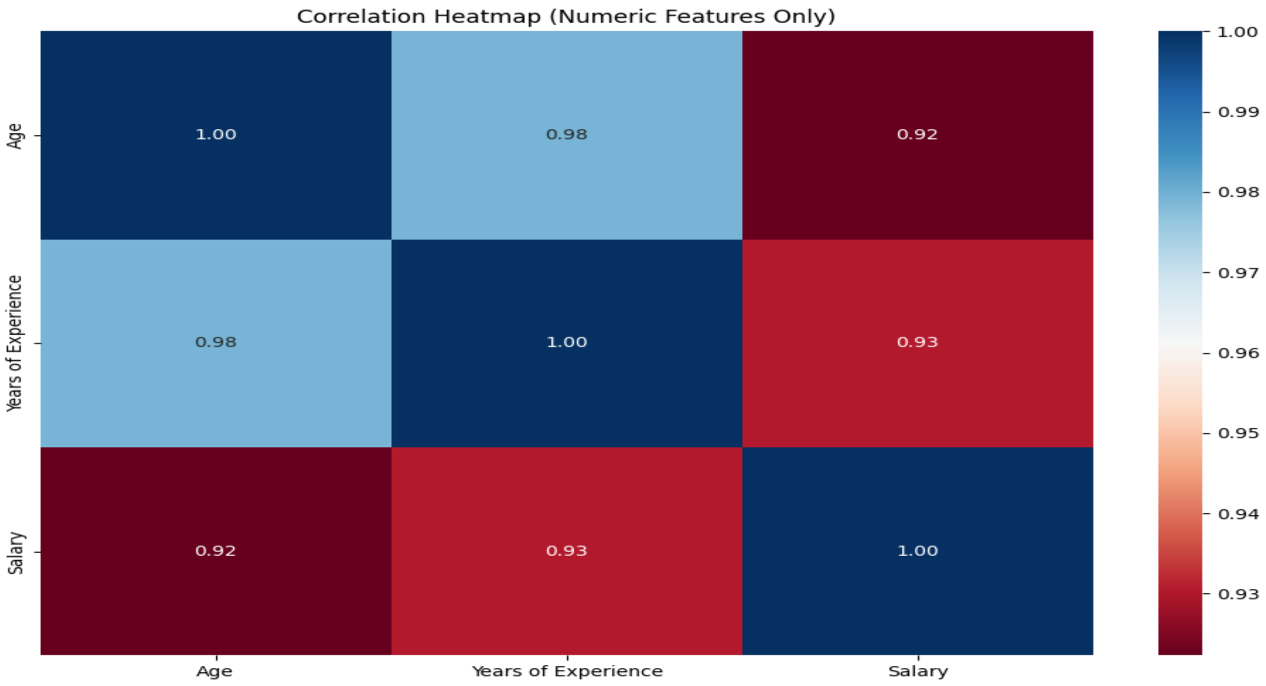
Hugging Face Deployment

RESULT

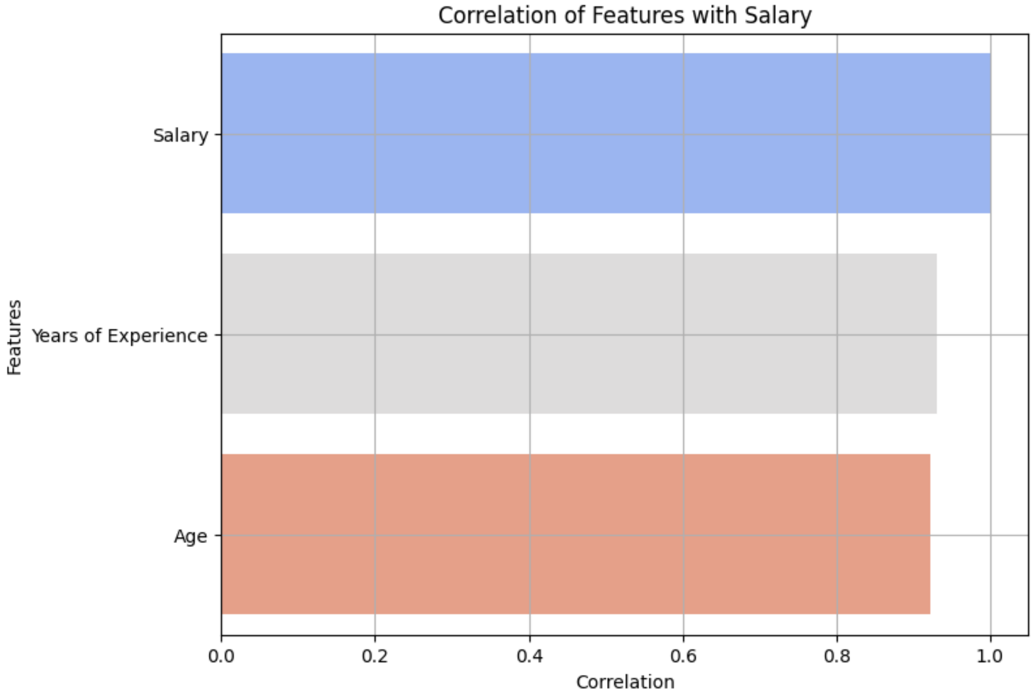
The best model achieved:

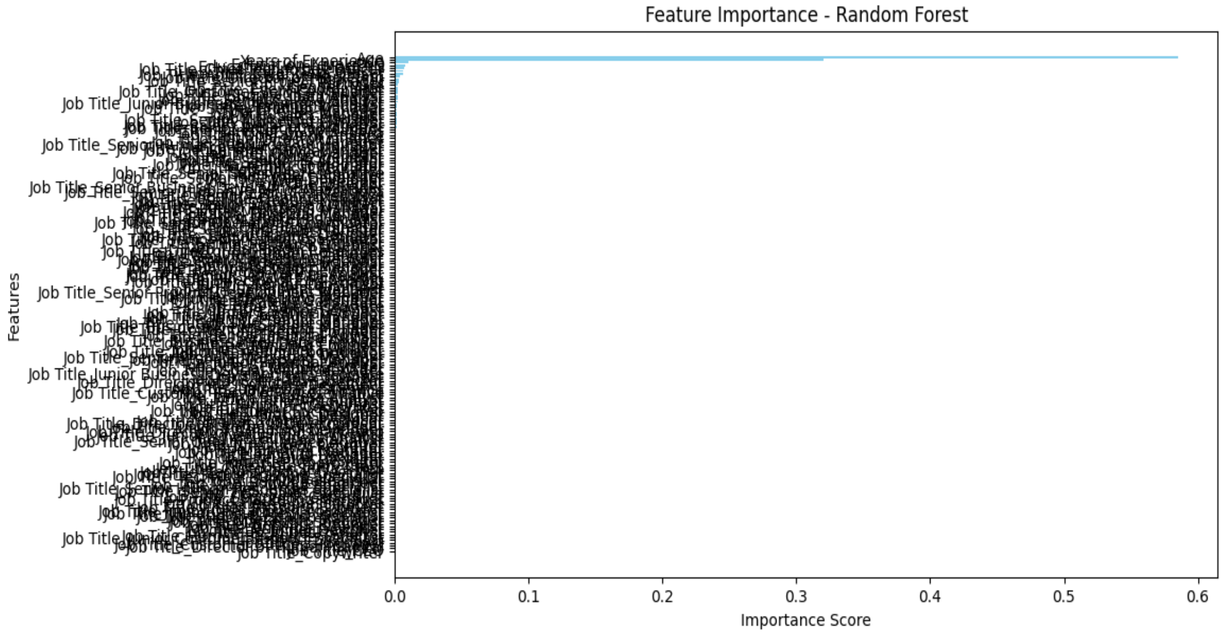
Random Forest
MAE: 9714.666666666666
R² Score: 0.8989615742525804

XGBoost
MAE: 10372.430416666666
R² Score: 0.8845554249375016



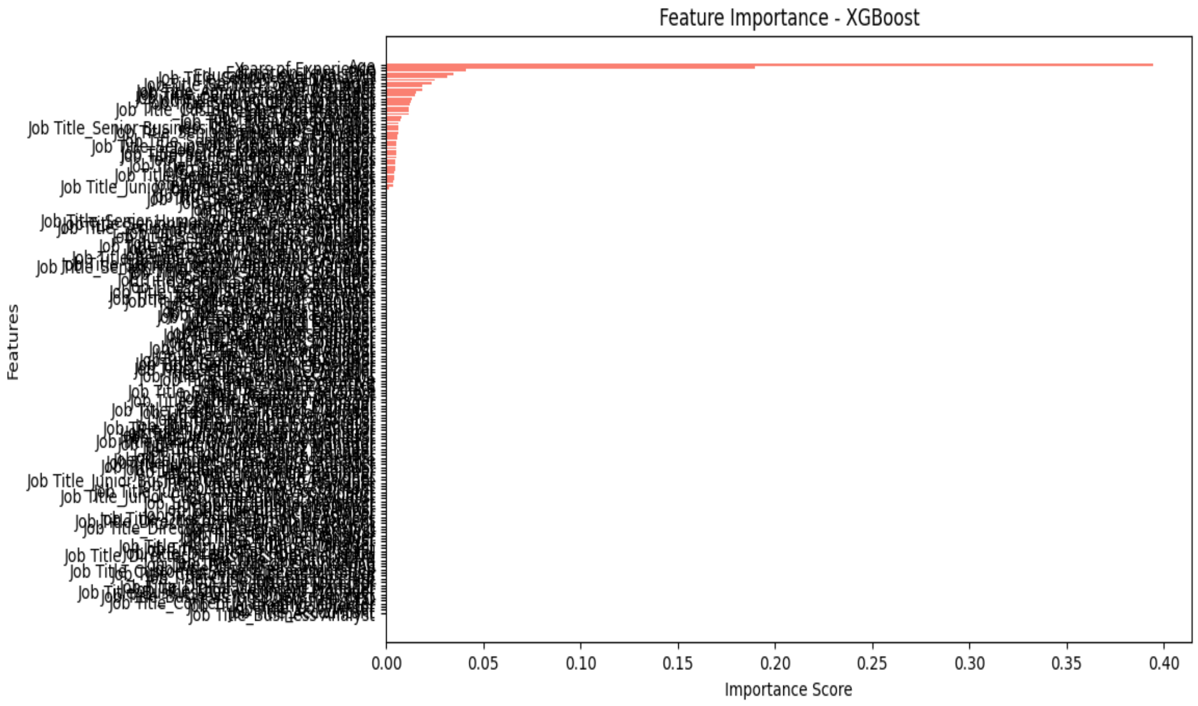
The output is a heatmap showing how strongly each numeric feature is correlated with others, where values near 1 or -1 indicate strong positive or negative relationships.

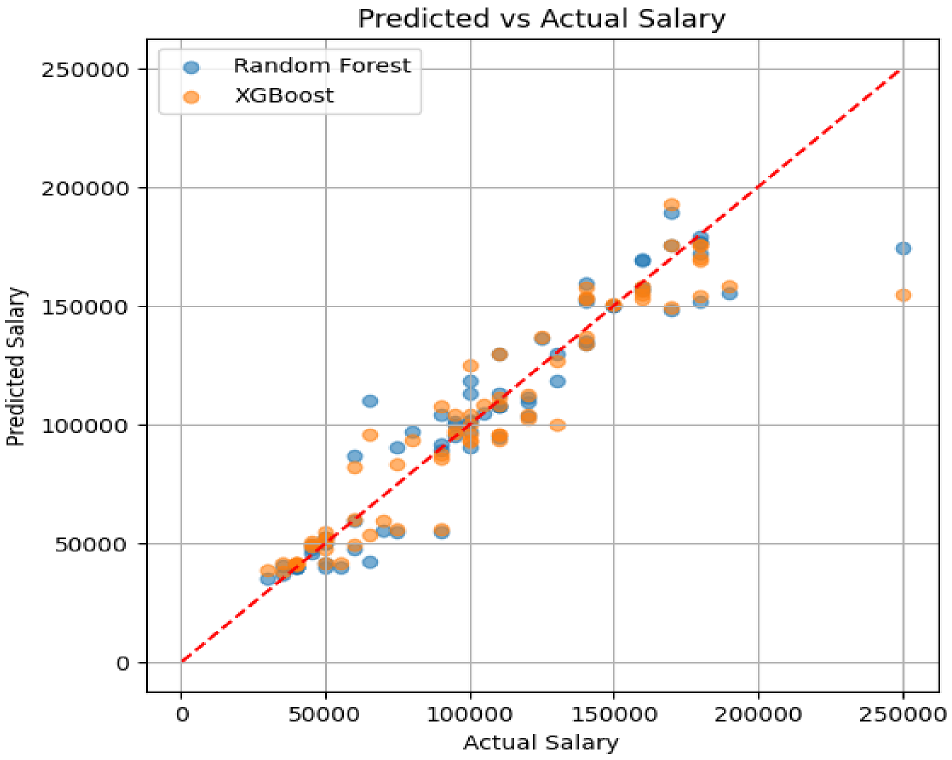




The output is a horizontal bar chart showing how strongly each numeric feature is correlated with the salary, helping identify the most influential predictors.

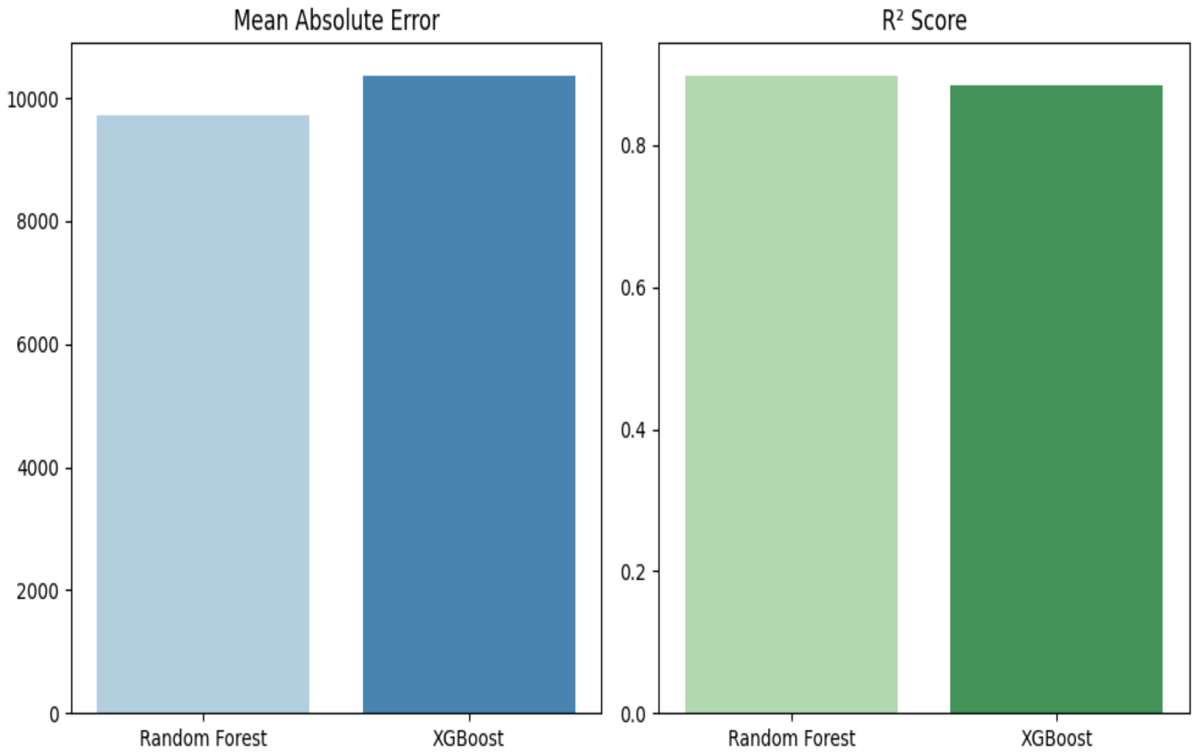
The output is a horizontal bar chart showing how much each feature contributes to the Random Forest model's predictions, with higher scores indicating greater importance.

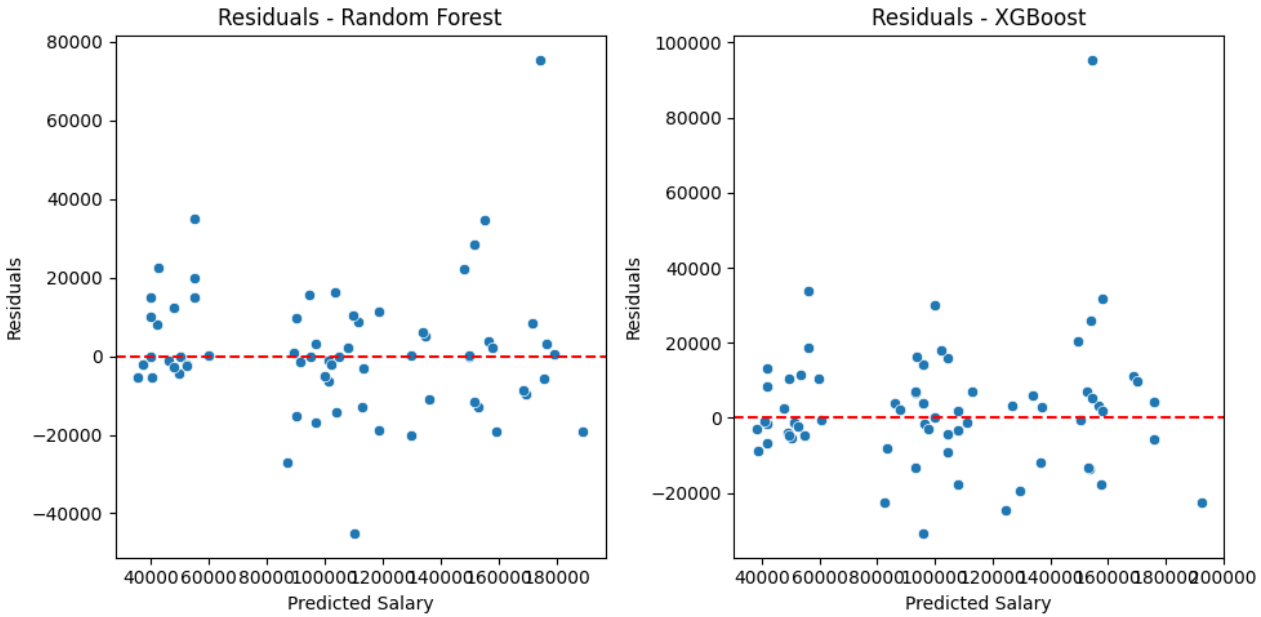




The output is a horizontal bar chart that ranks features based on their contribution to the XGBoost model's predictions, with longer bars indicating higher importance.

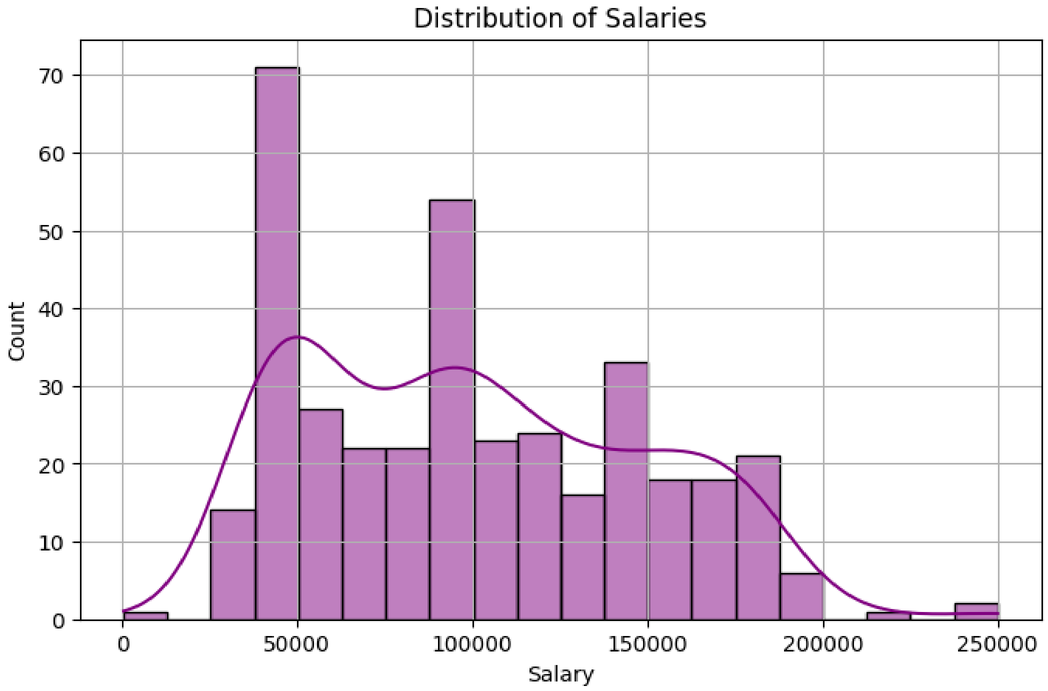
The output is a scatter plot comparing actual vs predicted salaries for both models, where points closer to the red dashed line indicate more accurate predictions.





The output shows side-by-side bar charts comparing the MAE and R² scores of Random Forest and XGBoost models, helping evaluate their prediction accuracy and performance.

The output displays residual plots for both models, showing the difference between actual and predicted salaries, where points scattered closely around the red line indicate better model accuracy and less bias.



The output is a histogram with a KDE curve showing the distribution of employee salaries, helping visualize the spread and skewness of the salary data.

Live Demo (Hugging Face URL) :

https://sahanapriyag-employee-salary-prediction.hf.space/?__theme=system&deep_link=CAvZmJav9UE

GitHub Repository :

https://github.com/SahanaPriyaG/Employee-Salary-Prediction-using-RandomForest-and-XGBoost.git

CONCLUSION

The Employee Salary Prediction system effectively applies machine learning to estimate salaries based on factors such as experience, education, job role, and location. Among the models tested, **XGBoost** and **Random Forest** achieved the best accuracy and performance. The project was successfully deployed using Gradio on **Hugging Face Spaces**, offering a clean and interactive user interface. Key challenges included handling categorical feature encoding and ensuring secure model deployment. Future improvements could involve adding more employee attributes, integrating resume parsing, and enabling batch salary predictions.

FUTURE SCOPE

- **Add More Features**: Include company size, industry type, skills, and certifications to improve prediction accuracy.
- **Real-Time Data Integration**: Connect with APIs (e.g., LinkedIn or Glassdoor) to fetch up-to-date salary trends.
- **Resume Parsing**: Integrate NLP-based resume analysis to auto-fill prediction inputs from uploaded resumes.
- **Batch Prediction**: Allow users to upload CSV files and get multiple salary predictions at once.
- **Globalization**: Extend the model to support multiple countries and currencies.
- **Mobile App Integration**: Develop a mobile-friendly version of the app for recruiters and job seekers on-the-go.

REFERENCES

1. Scikit-learn Documentation – https://scikit-learn.org/
2. XGBoost Documentation – https://xgboost.readthedocs.io/
3. Gradio Documentation – https://gradio.app
4. Hugging Face Spaces – https://huggingface.co/spaces
5. Kaggle Datasets – https://www.kaggle.com/

THANK YOU