CAPSTONE PROJECT

EMPLOYEE SALARY PREDICTION USING LINEAR REGRESSION

Presented By:
Name : Pathi Venkata Nihari
College Name : KKR And KSR Institute Of Technology And Sciences
Department : CSE-Artificial Intelligence
AICTE Internship Student Registration ID :STU669ba2c3e56f51721475779

OUTLINE

- Problem Statement
- System Development Approach
- Algorithm & Deployment (Step by Step Procedure)
- Result
- Conclusion
- Future Scope(Optional)
- References

PROBLEM STATEMENT

- This project aims to develop a model that predicts employee salaries based on various factors.
- Understanding salary trends is crucial for both employers and employees.
- Factors influencing salary include experience, education level, job role, and location.
- Accurate salary predictions can help in fair compensation and budgeting.
- The objective is to create a reliable model that can assist in salary forecasting.

SYSTEM APPROACH

System Requirements

Component	Specification
Operating System	Windows 10 / 11 or Linux / macOS
Processor	Intel Core i5 or above
RAM	Minimum 4 GB (8 GB recommended)
Storage	At least 2 GB of free disk space
Software Environment	Jupyter Notebook / VS Code / Colab
Python Version	Python 3.7 or above

SYSTEM APPROACH

Required Libraries

Library	Purpose
pandas	Data loading and manipulation
numpy	Numerical computations
matplotlib	Data visualization
seaborn	Advanced data visualization
sklearn (scikit-learn)	ML algorithms, preprocessing, model evaluation
joblib or pickle (optional)	To save and load the model

ALGORITHM & DEPLOYMENT

Step 1 – Data Collection

The first step in the Employee Salary Prediction project is acquiring a dataset that includes various employee attributes. The dataset contains features such as:

- Age
- Work class
- Education
- Marital-status
- Occupation
- Relationship
- Race
- Gender
- Capital-gain
- Capital-loss
- Hours-per-week
- Native-country
- Income

The data is loaded using tools like pandas and visualized to understand the structure and distributions.

ALGORITHM & DEPLOYMENT

Step 2 – Data Preprocessing

To prepare the dataset for machine learning, several preprocessing steps were applied:

•**Label Encoding**: All categorical features (workclass, marital-status, occupation, relationship, race, gender, native-country) were encoded using LabelEncoder from sklearn.preprocessing, converting text categories into numerical labels.

•**Feature-Target Split**: The target column income (if categorical) can be converted to binary (0 and 1). If actual salary is provided, that becomes the regression target.

•**Train-Test Split**: Data is split into training and test sets using an 80-20 split to evaluate model performance.

ALGORITHM & DEPLOYMENT

Step 3 – Model Building and Training

After preprocessing, the machine learning model is trained on the cleaned dataset.

Model Comparison:
Along with Linear Regression, models like **Decision Tree Regressor** and **Random Forest Regressor** were also tested.

Observation:
•Linear Regression performed well and gave consistent predictions.
•Decision Tree and Random Forest provided slightly better accuracy but were more complex.
•For simplicity and interpretability, Linear Regression was selected as the final model.

ALGORITHM & DEPLOYMENT

Step 4 – Model Evaluation

After the model is trained, predictions are made on the test set, and performance is evaluated using regression metrics:

•**Mean Absolute Error (MAE)**: Measures average magnitude of prediction errors.

•**Mean Squared Error (MSE)**: Penalizes larger errors more than MAE.

•**R^2 Score**: Indicates how much variance in the target is explained by the model.

These metrics help determine how well the model performs in predicting salary.

ALGORITHM & DEPLOYMENT

Step 5 – Model Deployment

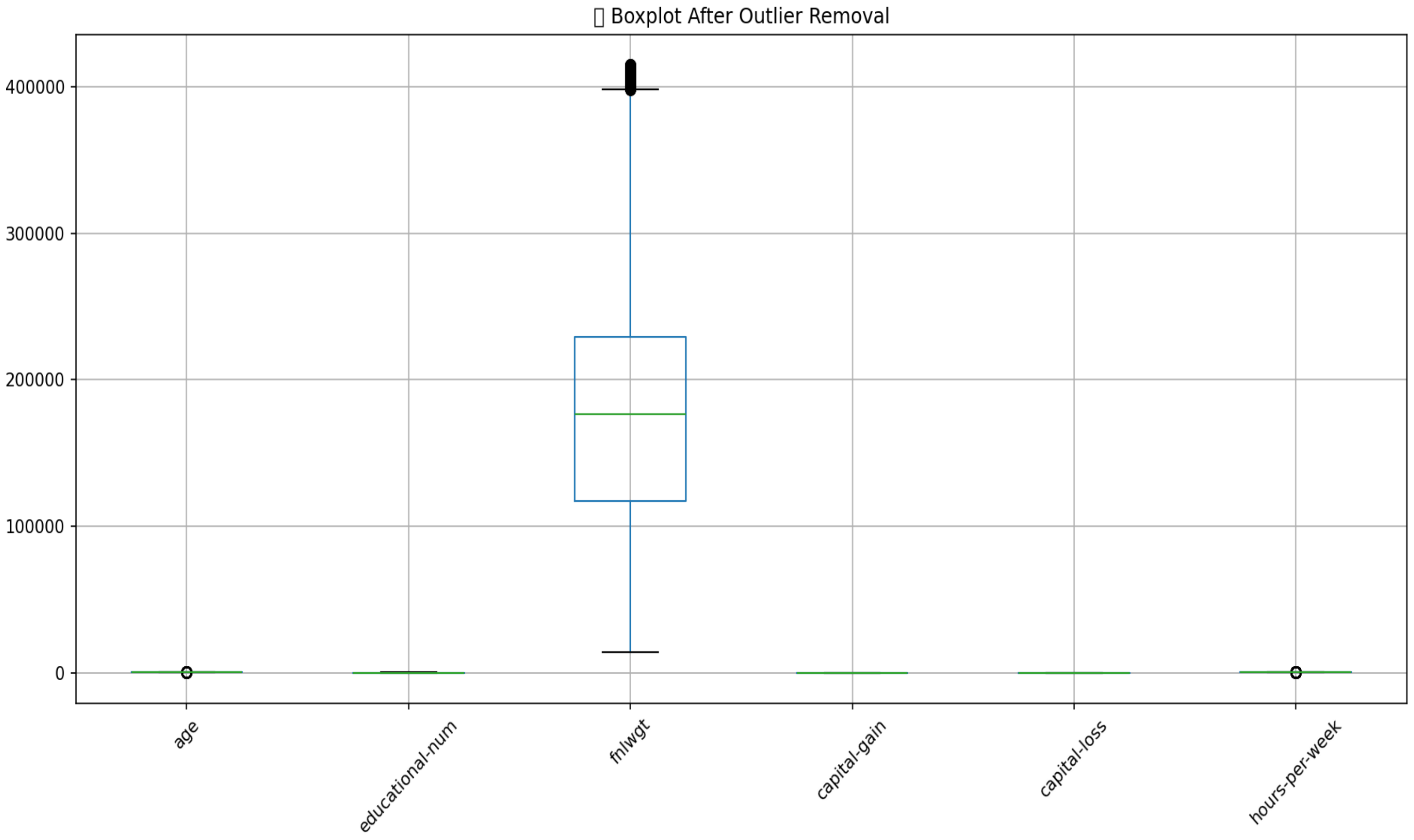
This project uses the trained model to generate salary predictions directly within the development environment.

•The trained model is used to predict salaries on test data.

•Predictions are compared with actual values to check accuracy.

•Results are displayed in tabular format using pandas.

RESULT

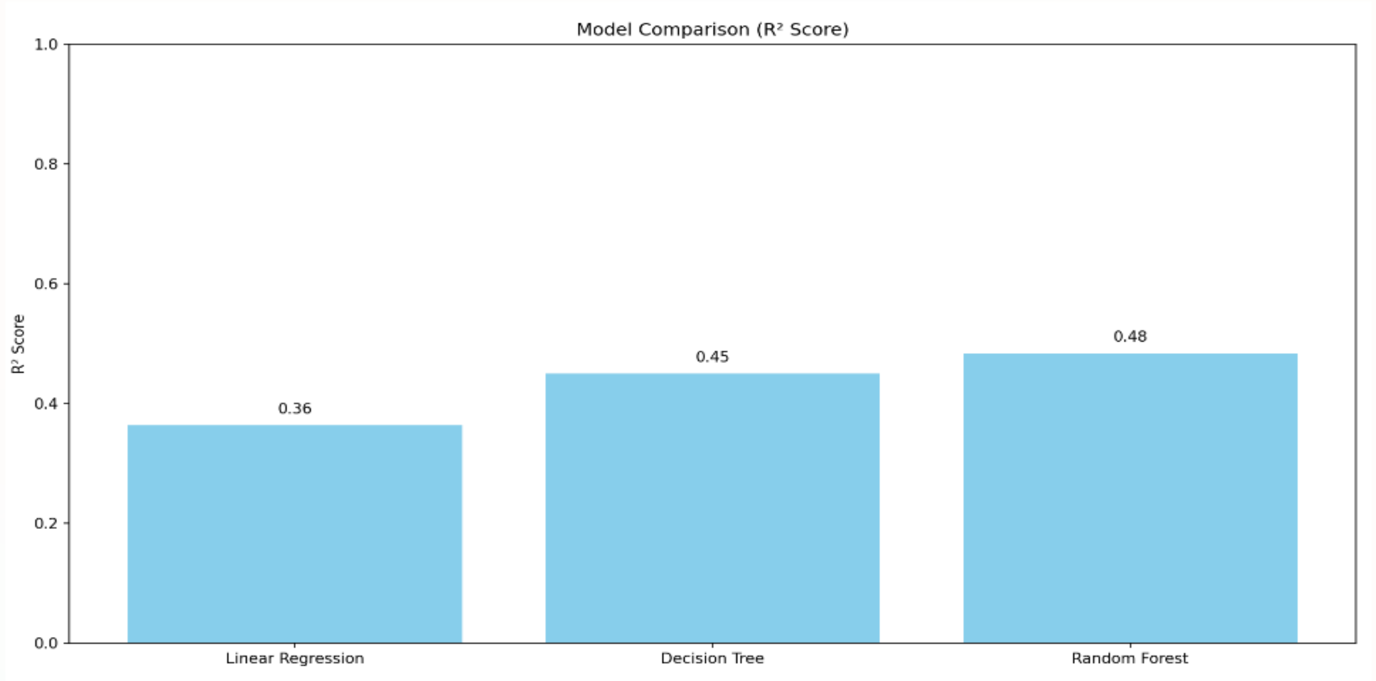


Visualize boxplots after removal of Outliers

RESULT

Model R^2 Score Comparison:

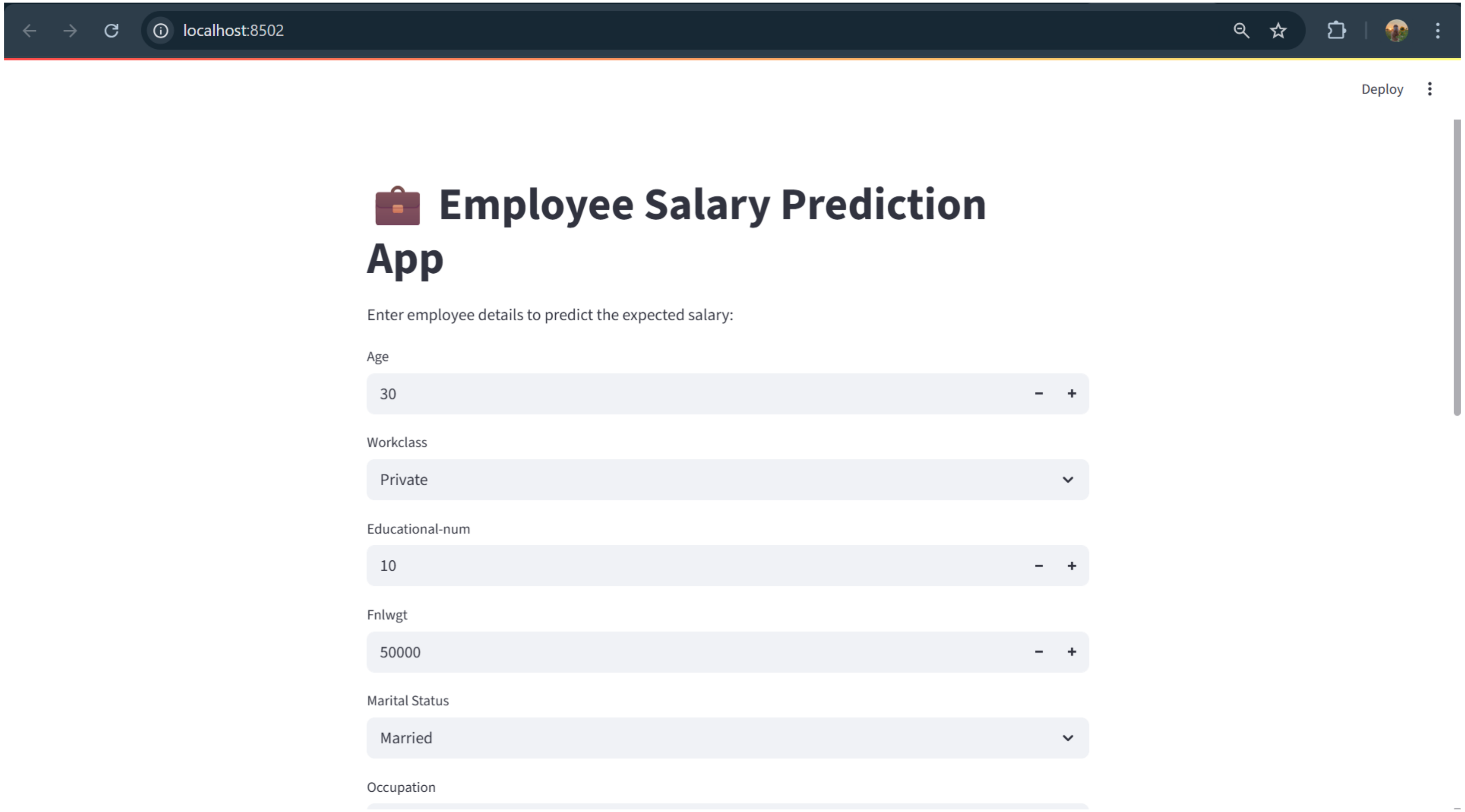
- Linear Regression : R^2 Score = 0.3629
- Decision Tree : R^2 Score = 0.4490
- Random Forest : R^2 Score = 0.4824



Evaluation of the best model: Random Forest Regression

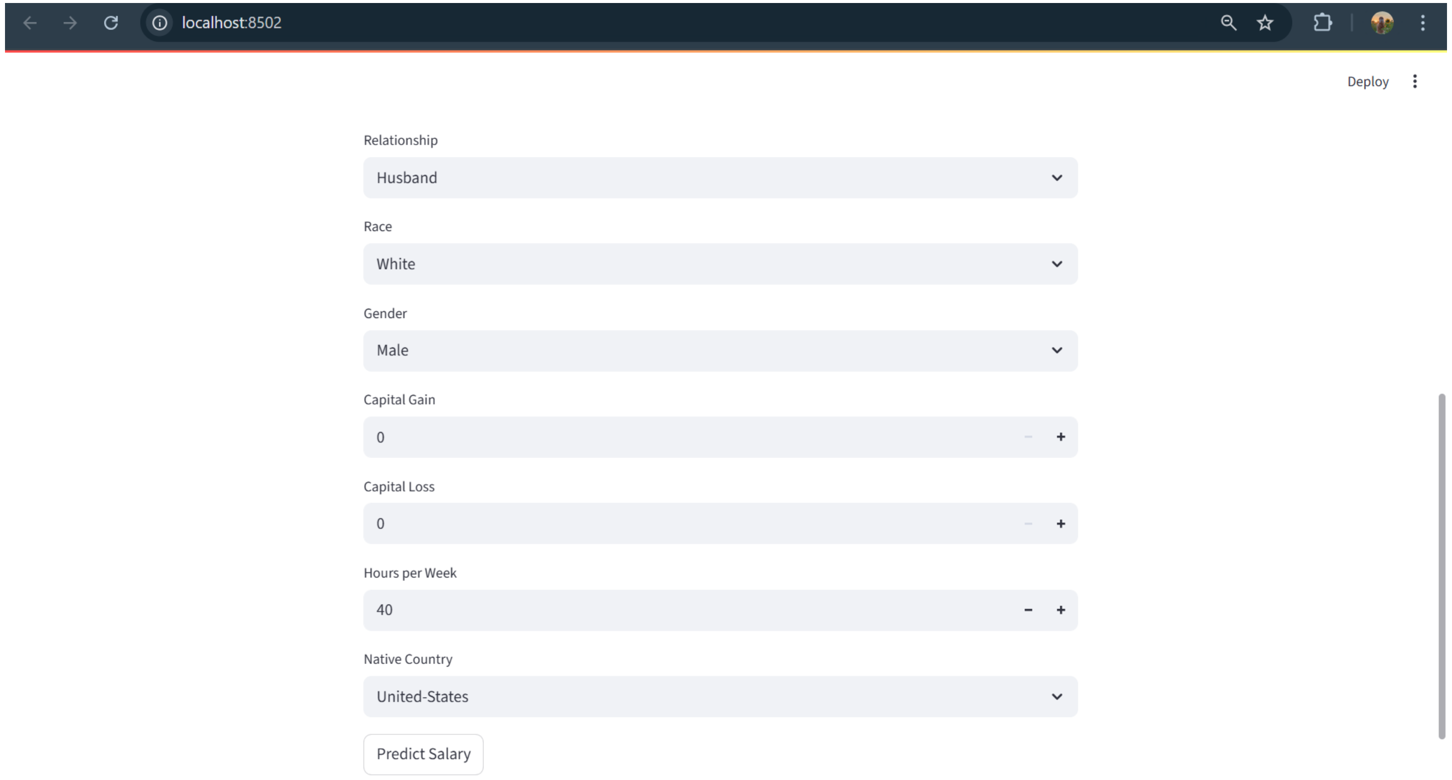
RESULT

STREAMLIT INTERFACE



RESULT

STREAMLIT INTERFACE



GITHUB LINK: https://github.com/nihari06/employee_salary_predict.ipynp

CONCLUSION

- **Random forest regression** gave the best results due to its simplicity and effective prediction performance.

- **Outlier removal (IQR method)** improved model accuracy and data quality.

- The model showed a strong relationship between features and salary, with good R^2 score.

- **Streamlit app** made the solution user-friendly and easy to deploy.

FUTURE SCOPE(OPTIONAL)

•**Add more features** such as experience, location, domain expertise, and performance ratings to improve prediction accuracy.

•**Integrate dynamic datasets** that update automatically as employee records change.

•**Deploy via cloud platforms** (like Heroku, AWS, or Streamlit Cloud) for real-time multi-user access.

•**Enable salary range prediction** using classification (e.g., Low, Medium, High).

•**Incorporate resume parsing** or LinkedIn API to predict salaries from profiles.

•**Build a dashboard** to visualize employee trends, salary distributions, and model metrics.

REFERENCES

1. **Scikit-learn Documentation** – *Used for machine learning models, encoding, and model evaluation*

2. **Pandas Documentation** – *Used for data preprocessing, handling, and transformations*

3. **Streamlit Documentation** – *Used to build and deploy the web application interface*

4. **Matplotlib & Seaborn Libraries** – *Used for visualizations and data plotting*

5. **Kaggle & UCI Repository** – *For reference datasets and feature selection inspiration*

6. **Python Official Documentation** – *Language reference for writing and debugging the code*

THANK YOU