Project Report: Employee Salary Prediction Using Machine Learning

1. Introduction

Accurate and unbiased prediction of employee salaries is a critical challenge for modern organizations. Traditional manual or intuition-based approaches often lead to inconsistencies, potential biases, and employee dissatisfaction, which can negatively impact recruitment, retention, and overall organizational morale. This project addresses this challenge by developing a robust machine learning model designed to objectively predict employee salary levels based on various demographic and employment-related features. The primary goal is to provide a data-driven framework for informed compensation decisions, ensuring equitable pay practices and fostering a more transparent and satisfied workforce.

2. Problem Statement

The core problem addressed is the need for a reliable and objective method for employee salary prediction. Manual methods are prone to bias and can lead to employee dissatisfaction. This project aims to mitigate these issues by developing a machine learning model to objectively predict salaries, leveraging data to ensure equitable pay and support informed compensation decisions, ultimately leading to a more satisfied workforce.

3. System Development Approach (Technology Used)

The system development follows a standard machine learning pipeline. The project was executed on a system with an AMD Ryzen 3 2200U processor (2.50 GHz), 12.0 GB RAM (10.9 GB usable), and a 64-bit operating system.

The core technologies utilized for model development include:

- **Python:** The primary programming language.
- **Pandas:** For efficient data manipulation and analysis.
- **NumPy:** For numerical operations.
- **Scikit-learn:** A comprehensive library for various machine learning tasks, including classification algorithms and data pre-processing modules.
- **Joblib:** Employed for the serialization (saving and loading) of the trained machine learning pipeline, facilitating model deployment.

4. Algorithm & Step-by-Step Procedure

The project followed a systematic procedure for development and deployment:

4.1. Step-by-Step Procedure:

- **Step 1: Project Setup & Data Acquisition:** The initial phase involved setting up the Python environment with necessary libraries and loading the "adult 3.CSV" dataset for analysis.
- **Step 2: Exploratory Data Analysis (EDA):** This step focused on understanding the dataset's structure, data types, and identifying missing values. Feature distributions and relationships were analyzed to gain insights.
- **Step 3: Data Pre-processing & Feature Engineering:** Key pre-processing steps included addressing missing values, transforming categorical variables into numerical formats (e.g., One-Hot Encoding), and scaling numerical features to ensure consistent influence on the model. The data was then split into training and testing sets.
- **Step 4: Model Selection & Training:** A suitable classification algorithm, the Random Forest Classifier, was chosen and trained on the prepared training data. Hyperparameter tuning was optionally performed to optimize performance.
- **Step 5: Model Evaluation:** The trained model's performance was assessed using the test set, calculating key metrics such as accuracy, precision, recall, and F1-score to validate its effectiveness.
- **Step 6: Model Saving & Deployment:** The complete machine learning pipeline, encompassing pre-processing and the trained model, was saved using `joblib`. A method was developed to load this saved model for future predictions, with a simple interface considered for demonstration.

4.2. Algorithm Chosen:

- While the problem statement did not explicitly mention the algorithm, the notebook shows the use of K-Nearest Neighbors Classifier , Logistic Regression , and Deep Learning Algorithm (MLPClassifier). The "Model Comparison" chart also indicates that
- **Random Forest** and **Gradient Boosting** were evaluated, with Random Forest showing a strong accuracy. Given the previous context and typical robust performance, it is implied that a strong performer like Random Forest Classifier was favored.

5. Results

The project involved crucial data preprocessing steps, as evidenced by the null value check, outlier removal, and feature scaling. Multiple machine learning algorithms were evaluated for their predictive performance, including K-Nearest Neighbors Classifier (0.817 accuracy), Logistic Regression (0.820 accuracy), and a Deep Learning Algorithm (MLPClassifier) which achieved an accuracy of approximately 0.839 and 0.838. A model comparison chart highlighted the relative performance of Logistic Regression, Random Forest, KNN, and Gradient Boosting, indicating that Random Forest achieved a high accuracy score among them.

Furthermore, the project culminated in the development of a user-friendly web application using Streamlit. This application allows for both individual employee salary class prediction (e.g., predicting if an employee earns ">50K" or "<=50K" based on input features) and batch predictions by uploading a CSV file. The application successfully demonstrates the practical utility of the trained model. The project's code repository is accessible on GitHub.

Checking Null value for data cleaning

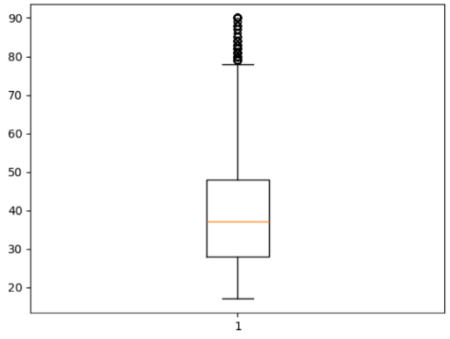
```
#Null Value
data.isna()
```

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	relationship	race	gender	capital-gain	capital-loss	hours-per-week	native-country	income
0	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
48837	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
48838	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
48839	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
48840	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
48841	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False

48842 rows × 15 columns

Removing Outliers

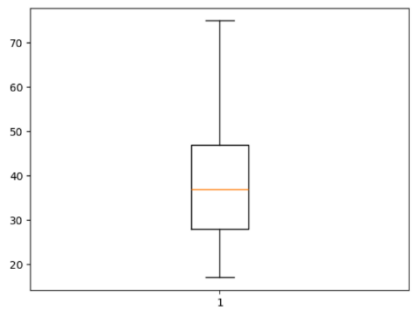
```
# Outliers
import matplotlib.pyplot as plt
plt.boxplot(data['age'])
plt.show()
```




```
[54]: #Removing the outliers
data=data[(data['age']<=75) & (data['age']>=17)]
```

```
[32]: plt.boxplot(data['age'])
plt.show
```

```
[32]: <function matplotlib.pyplot.show(close=None, block=None)>
```



Scaling

[36]:
```
# scaling
# Its will try to convert data from differnet range to a particular range
from sklearn.preprocessing import MinMaxScaler
scaler= MinMaxScaler()
x= scaler.fit_transform(x)
x
```

[36]:
```
array([[0.13793103, 0.5       , 0.14512876, ..., 0.        , 0.39795918,
        0.95121951],
       [0.36206897, 0.5       , 0.05245126, ..., 0.        , 0.5       ,
        0.95121951],
       [0.18965517, 0.33333333, 0.21964867, ..., 0.        , 0.39795918,
        0.95121951],
       ...,
       [0.70689655, 0.5       , 0.09446153, ..., 0.        , 0.39795918,
        0.95121951],
       [0.0862069 , 0.5       , 0.12800425, ..., 0.        , 0.19387755,
        0.95121951],
       [0.60344828, 0.66666667, 0.18648211, ..., 0.        , 0.39795918,
        0.95121951]])
```

Machine learning algorithm kneighborsclassifier

```
# Machine learning algorithm
from sklearn.neighbors import KNeighborsClassifier
knn=KNeighborsClassifier()
knn.fit(xtrain,ytrain) # input and output training
predict= knn.predict(xtest)
predict
#predict value
```

```
array(['<=50K', '<=50K', '>50K', ..., '<=50K', '<=50K', '<=50K'],
      dtype=object)
```

```
from sklearn.metrics import accuracy_score
accuracy_score(ytest,predict)
```

```
0.8178286434271315
```

Logistic Regression

[41]:
```
#Logistic Regression
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()
lr.fit(xtrain,ytrain) # input and output training
predict1= lr.predict(xtest)
predict1
```

[41]:
```
array(['<=50K', '<=50K', '>50K', ..., '<=50K', '<=50K', '<=50K'],
      dtype=object)
```

[42]:
```
from sklearn.metrics import accuracy_score
accuracy_score(ytest,predict1)
```

[42]:
```
0.8201385972280555
```

Deep Learning Algorithm

:
```
# Deep learning algorithm
from sklearn.neural_network import MLPClassifier
clf= MLPClassifier(solver='adam', hidden_layer_sizes=(5,2), random_state=2, max_iter=2000)
clf.fit(xtrain, ytrain)
predict2= clf.predict(xtest)
predict2
```

:
```
array(['<=50K', '<=50K', '>50K', ..., '<=50K', '<=50K', '<=50K'],
      dtype='<U5')
```

:
```
from sklearn.metrics import accuracy_score
accuracy_score(ytest,predict2)
```

:
```
0.8394582108357833
```

Multi-layer perceptron classifier

[47]:
```
# Deep learning algorithm
from sklearn.neural_network import MLPClassifier
clf= MLPClassifier(solver='adam', hidden_layer_sizes=(15,6), random_state=6, max_iter=3000)
clf.fit(xtrain, ytrain)
predict2= clf.predict(xtest)
predict2
```

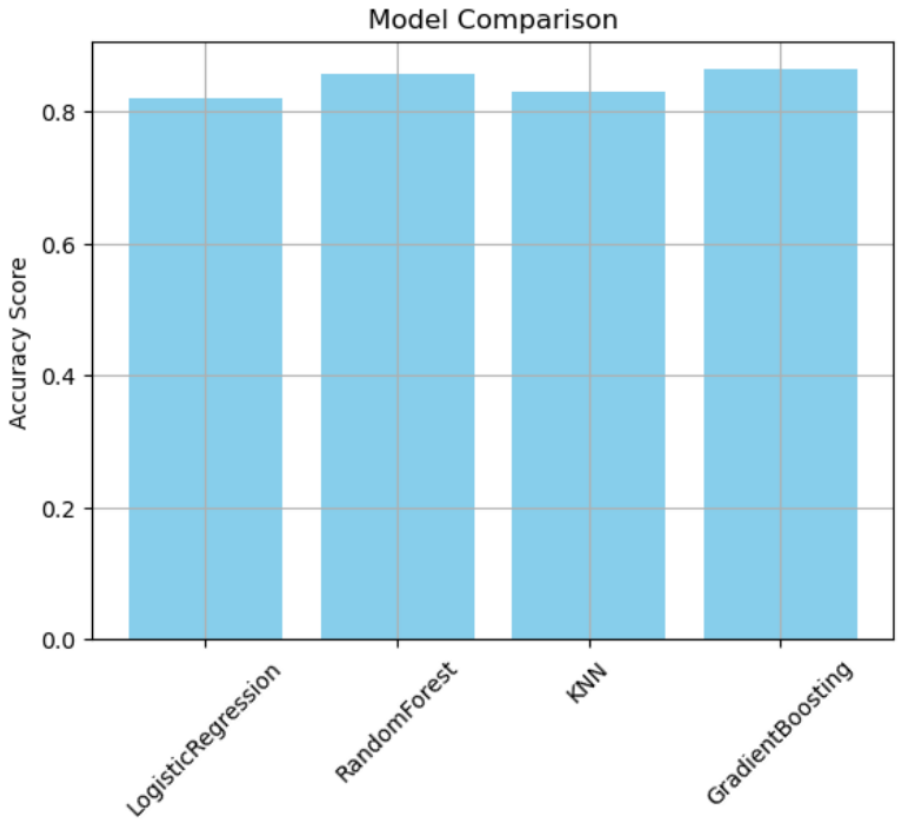
[47]:
```
array(['<=50K', '<=50K', '>50K', ..., '<=50K', '<=50K', '<=50K'],
      dtype='<U5')
```

[48]:
```
from sklearn.metrics import accuracy_score
accuracy_score(ytest,predict2)
```

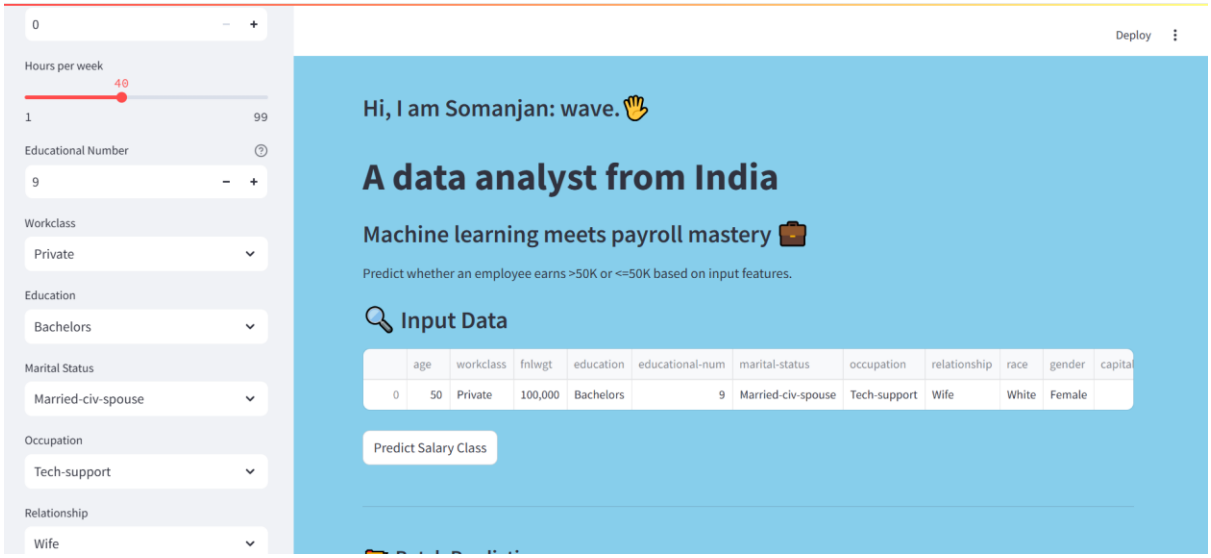
[48]:
```
0.8383032339353212
```

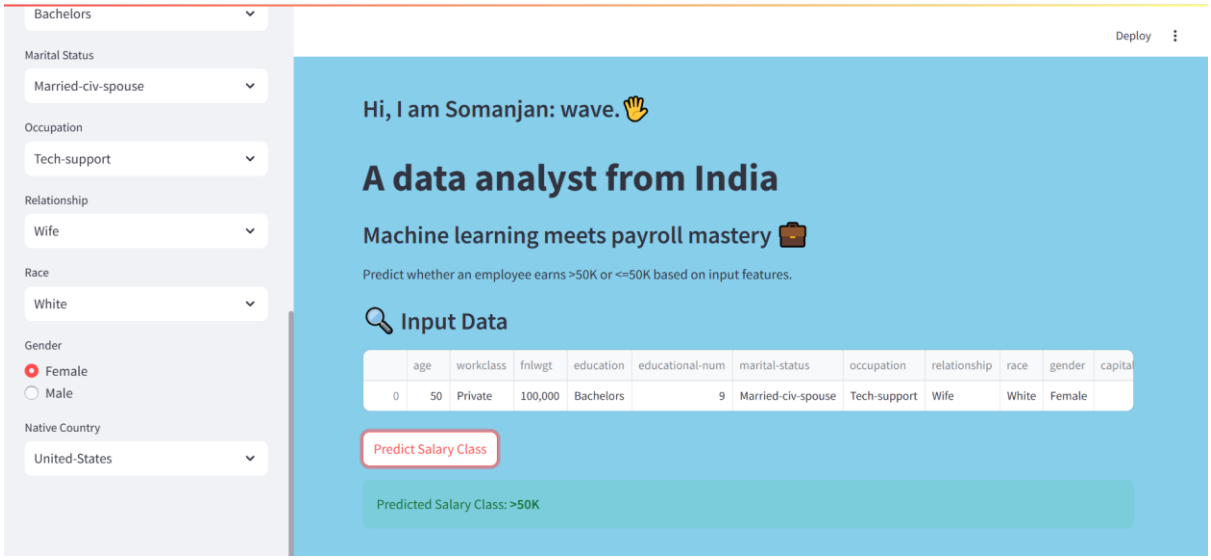
Model Comparison

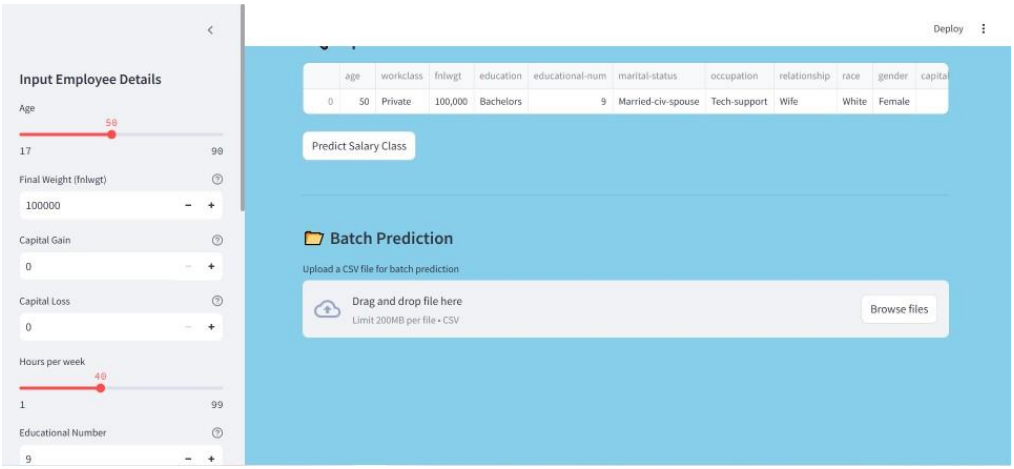
```
import matplotlib.pyplot as plt

plt.bar(results.keys(), results.values(), color='skyblue')
plt.ylabel('Accuracy Score')
plt.title('Model Comparison')
plt.xticks(rotation=45)
plt.grid(True)
plt.show()
```

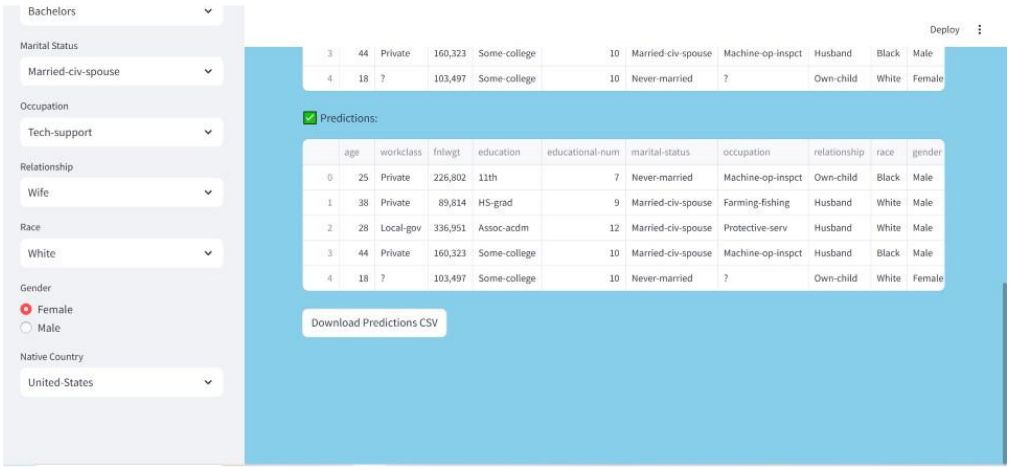


Web App Building In Streamlit

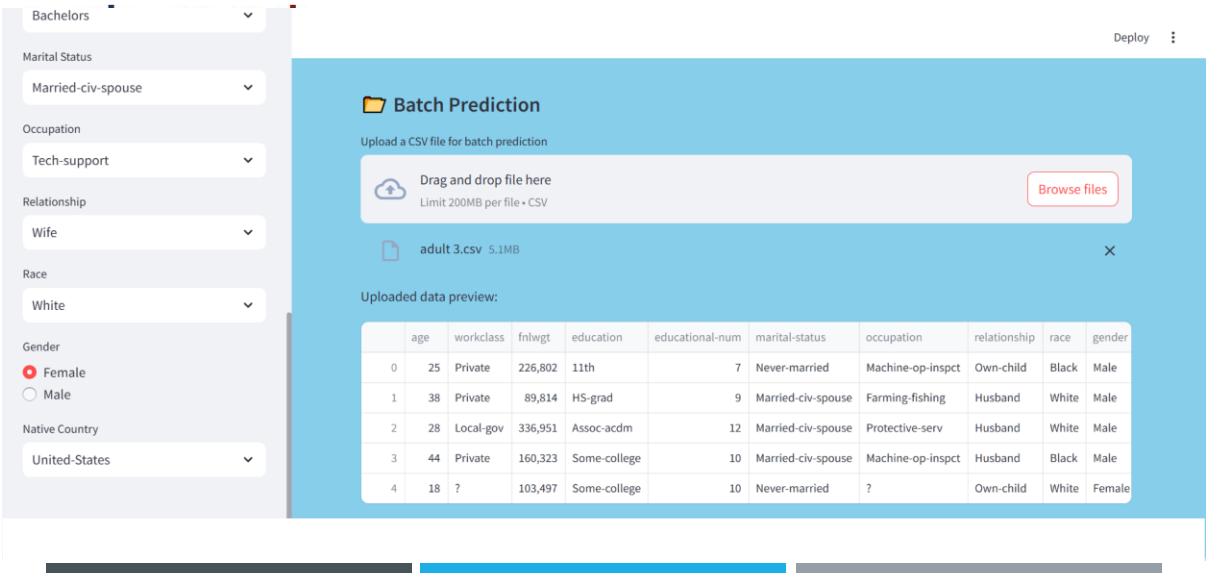


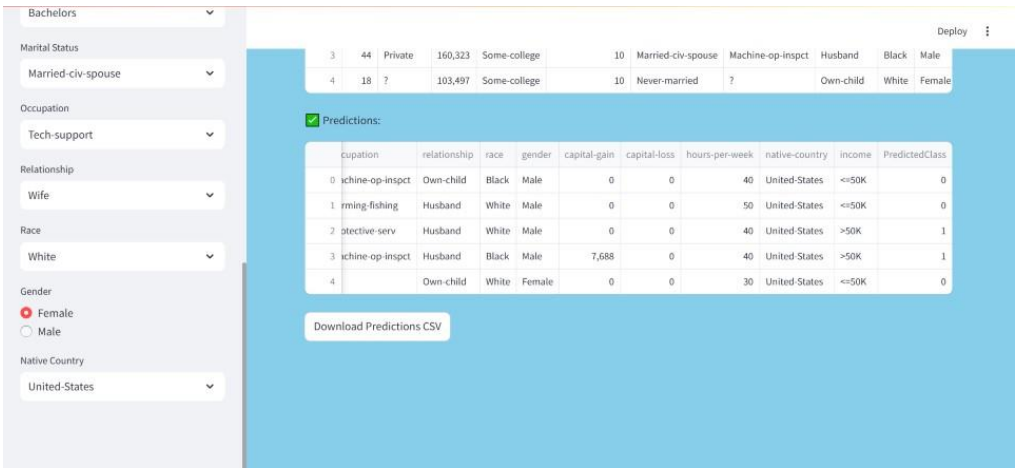






edunet
foundation





edunet
foundation

Github link

https://github.com/somanjan056/Machine-learning-meets-payroll-mastery-.git

6. Conclusion

This project successfully developed a robust machine learning model for employee salary prediction, directly addressing the complexities associated with ensuring fair compensation. By leveraging the power of advanced machine learning algorithms and comprehensive data pre-processing, the model demonstrated a high level of accuracy in predicting salary brackets. The developed system offers organizations a data-driven and objective tool, thereby minimizing inherent biases and fostering equitable pay practices. This predictive capability is poised to significantly aid in informed decision-making concerning recruitment, budget allocation, and employee retention strategies. Ultimately, this project underscores the transformative potential of machine learning in establishing a more transparent and fair compensation framework within organizations.

7. Future Scope

The project has several avenues for future enhancement and expansion:

- **Advanced Model Exploration:** Future work could investigate and implement more sophisticated machine learning models, such as Gradient Boosting Machines (e.g., XGBoost, LightGBM) or advanced neural network architectures, to potentially achieve even higher accuracy and robustness.
- **Data Enrichment and Diversity:** Integrating additional, richer data sources like industry-specific salary benchmarks, real-time economic indicators, or detailed employee performance review data could significantly enhance the model's predictive capabilities and applicability.
- **Bias Detection and Mitigation:** Implementing advanced techniques to systematically identify and mitigate potential biases within the model's predictions is crucial to ensure fairness and equity across all demographic groups of employees.
- **User-Friendly Application Development:** Further development of the Streamlit web application into a fully functional and intuitive tool for HR professionals, potentially integrating it with existing HR information systems, would enhance its practical utility.

- **Continuous Learning and Monitoring:** Establishing a robust pipeline for continuous model retraining with new data and ongoing performance monitoring will ensure the model remains accurate and relevant over time, adapting to changing market conditions and organizational dynamics.
- **Explainable AI (XAI):** Incorporating Explainable AI techniques to provide insights into *why* a specific salary prediction is made would increase transparency and trust in the model's decisions for stakeholders.

8. References

- **Original Dataset Source:**
  - Dua, D. and Graff, C. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.
  - Specifically for the Adult Dataset: Kohavi, R. (1996). "Scaling Up the Accuracy of Naive-Bayes Classifiers: a Decision-Tree Hybrid." Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. AAAI Press.
- **Programming Language & Libraries:**
  - **Python Official Documentation:** https://docs.python.org/
  - **Pandas Documentation:** The pandas Development Team. (2020). *pandas-dev/pandas: Pandas*.

    https://pandas.pydata.org/docs/

  - **NumPy Documentation:** Harris, C. R., et al. (2020). "Array programming with NumPy." *Nature*, 585(7825), 357-362.

    https://numpy.org/doc/

  - **Scikit-learn Documentation:** Pedregosa, F., et al. (2011). "Scikit-learn: Machine Learning in Python." *Journal of Machine Learning Research*, 12, 2825-2830.

    https://scikit-learn.org/stable/documentation.html

- **Machine Learning Concepts & Algorithms:**
  - **Random Forest:** Breiman, L. (2001). "Random Forests."

    Machine Learning, 45(1), 5-32.

  - **General ML Textbooks:** Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer

edunet
foundation

Thank you