CAPSTONE PROJECT

EMPLOYEE SALARY PREDICTION USING ML ALGORITHMS

Presented By:
Student Name :Bhalani Naman Sureshbhai
College Name :Gyan Manjari Innovative University Bhavnagar
Department :CSE(B.Tech)
AICTE ID :STU6829828cd9c9c1747550860

edunet foundation

OUTLINE

- **Problem Statement** (Should not include solution)
- **System Development Approach** (Technology Used)
- **Algorithm & Deployment (Step by Step Procedure)**
- **Result**
- **Conclusion**
- **Future Scope(Optonal)**
- **References**

PROBLEM STATEMENT

This project focuses on predicting employee salaries using various machine learning algorithms. It analyzes factors like education, experience, job role, and company size to estimate salary ranges. The main goal is to help HR departments and job seekers make informed decisions. We used models such as Linear Regression, Random Forest, and Gradient Boosting to improve accuracy. The project demonstrates the power of data-driven decision-making in modern human resource management.

SYSTEM APPROACH

🔷 System Requirements

- **Operating System:** Windows 10 or higher / macOS / Linux
- **Processor:** Intel i3 or higher (Recommended: i5 or above)
- **RAM:** Minimum 4 GB (Recommended: 8 GB for better performance)
- **Storage:** Minimum 500 MB of free space
- **Software/Tools:** Python 3.x, Jupyter Notebook or Google Colab

SYSTEM APPROACH

🔷 System Requirements

- **Operating System:** Windows 10 or higher / macOS / Linux
- **Processor:** Intel i3 or higher (Recommended: i5 or above)
- **RAM:** Minimum 4 GB (Recommended: 8 GB for better performance)
- **Storage:** Minimum 500 MB of free space
- **Software/Tools:** Python 3.x, Jupyter Notebook or Google Colab

ALGORITHM & DEPLOYMENT

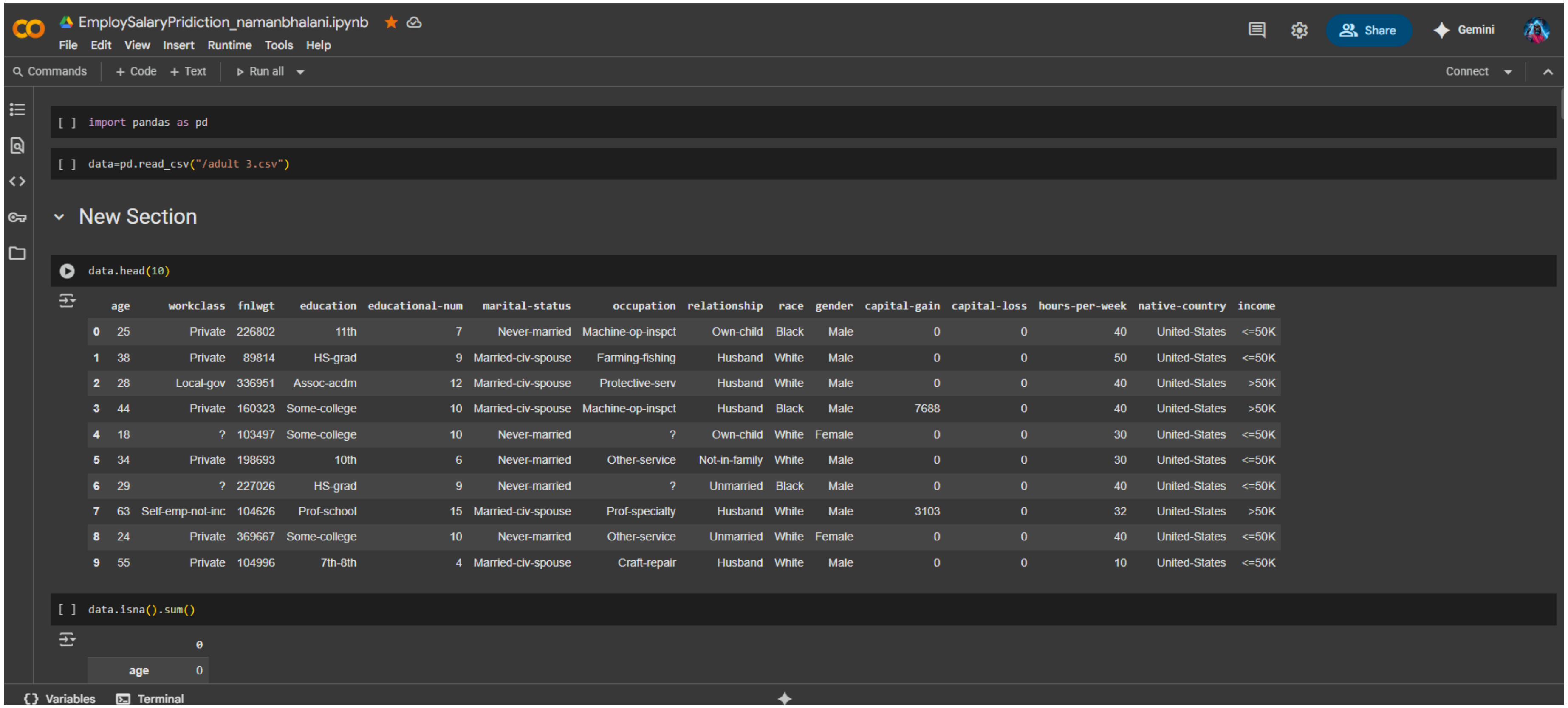
🔷 Libraries Required to Build the Model

These are the Python libraries you used:

1. pandas – for data loading and manipulation
2. numpy – for numerical operations
3. matplotlib.pyplot – for visualizing boxplots and model accuracy
4. scikit-learn (sklearn) – for:
    - Preprocessing: LabelEncoder, MinMaxScaler, StandardScaler
    - Model selection: train_test_split
    - Model evaluation: accuracy_score, classification_report
    - Machine learning models:
        - LogisticRegression
        - RandomForestClassifier
        - KNeighborsClassifier (KNN)
        - SVC (Support Vector Classifier)
        - GradientBoostingClassifier
        - DecisionTreeClassifier
        - GaussianNB (Naive Bayes)
        - MLPClassifier (Neural Network)
    - Pipeline – to combine preprocessing + model training
5. joblib – for saving the trained model as best_model.pkl

edunet foundation

RESULT



EmploySalaryPridiction_namanbhalani.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all Share Gemini Connect

```
[ ] import pandas as pd
```

```
[ ] data=pd.read_csv("/adult 3.csv")
```

New Section

```
data.head(10)
```

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	relationship	race	gender	capital-gain	capital-loss	hours-per-week	native-country	income
0	25	Private	226802	11th	7	Never-married	Machine-op-inspct	Own-child	Black	Male	0	0	40	United-States	<=50K
1	38	Private	89814	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	White	Male	0	0	50	United-States	<=50K
2	28	Local-gov	336951	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	White	Male	0	0	40	United-States	>50K
3	44	Private	160323	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	Black	Male	7688	0	40	United-States	>50K
4	18	?	103497	Some-college	10	Never-married	?	Own-child	White	Female	0	0	30	United-States	<=50K
5	34	Private	198693	10th	6	Never-married	Other-service	Not-in-family	White	Male	0	0	30	United-States	<=50K
6	29	?	227026	HS-grad	9	Never-married	?	Unmarried	Black	Male	0	0	40	United-States	<=50K
7	63	Self-emp-not-inc	104626	Prof-school	15	Married-civ-spouse	Prof-specialty	Husband	White	Male	3103	0	32	United-States	>50K
8	24	Private	369667	Some-college	10	Never-married	Other-service	Unmarried	White	Female	0	0	40	United-States	<=50K
9	55	Private	104996	7th-8th	4	Married-civ-spouse	Craft-repair	Husband	White	Male	0	0	10	United-States	<=50K

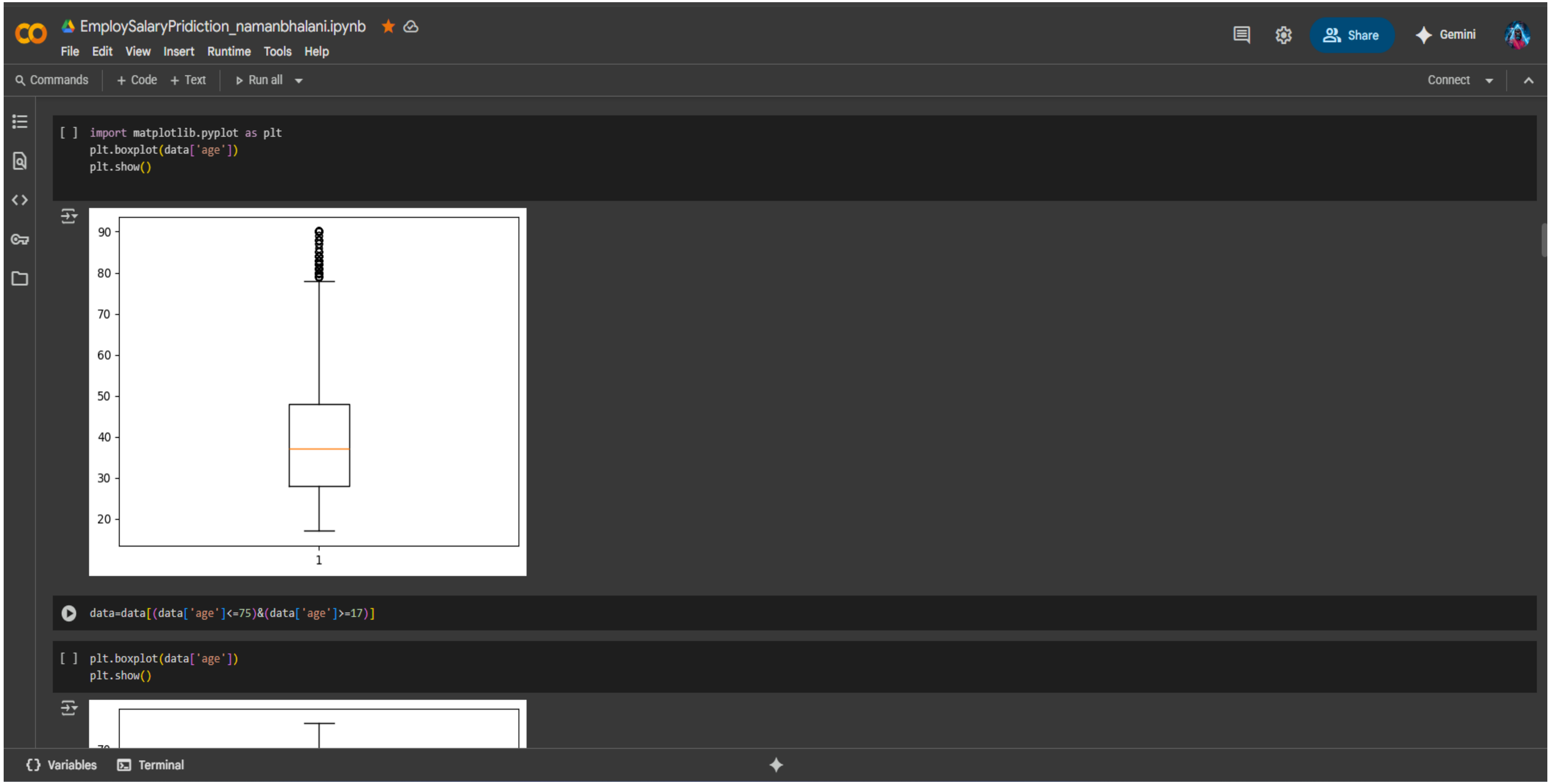
```
[ ] data.isna().sum()
```

	0
age	0

Variables Terminal

edunet foundation

RESULT


EmploySalaryPridiction_namanbhalani.ipynb
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands
+ Code
+ Text
Run all
Connect
[ ] import matplotlib.pyplot as plt
plt.boxplot(data['age'])
plt.show()
90
80
70
60
50
40
30
20
1
data=data[(data['age']<=75)&(data['age']>=17)]
[ ] plt.boxplot(data['age'])
plt.show()
Variables
Terminal

RESULT

edunet foundation

RESULT

EmploySalaryPridiction_namanbhalani.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
Share
Gemini
Connect
40000
20000
0
1
plt.boxplot(data['educational-num'])
plt.show()
16
14
12
10
8
6
4
2
1
Variables
Terminal

RESULT

```python
data=data.drop(columns=['education'])
```

```python
from sklearn.preprocessing import LabelEncoder
encoder=LabelEncoder()
data['workclass']=encoder.fit_transform(data['workclass'])
data['marital-status']=encoder.fit_transform(data['marital-status'])
data['occupation']=encoder.fit_transform(data['occupation'])
data['relationship']=encoder.fit_transform(data['relationship'])
data['race']=encoder.fit_transform(data['race'])
data['gender']=encoder.fit_transform(data['gender'])
data['native-country']=encoder.fit_transform(data['native-country'])
data
```

	age	workclass	fnlwgt	educational-num	marital-status	occupation	relationship	race	gender	capital-gain	capital-loss	hours-per-week	native-country	income
0	25	3	226802	7	4	6	3	2	1	0	0	40	39	<=50K
1	38	3	89814	9	2	4	0	4	1	0	0	50	39	<=50K
2	28	1	336951	12	2	11	0	4	1	0	0	40	39	>50K
3	44	3	160323	10	2	6	0	2	1	7688	0	40	39	>50K
4	18	2	103497	10	4	8	3	4	0	0	0	30	39	<=50K
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
48837	27	3	257302	12	2	13	5	4	0	0	0	38	39	<=50K
48838	40	3	154374	9	2	6	0	4	1	0	0	40	39	>50K
48839	58	3	151910	9	6	0	4	4	0	0	0	40	39	<=50K
48840	22	3	201490	9	4	0	3	4	1	0	0	20	39	<=50K
48841	52	4	287927	9	2	3	5	4	0	15024	0	40	39	>50K

46720 rows × 14 columns

RESULT

EmploySalaryPridiction_namanbhalani.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

```python
from sklearn.pipeline import Pipeline
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.preprocessing import StandardScaler


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

models = {
    "LogisticRegression": LogisticRegression(),
    "RandomForest": RandomForestClassifier(),
    "KNN": KNeighborsClassifier(),
    "SVM": SVC(),
    "GradientBoosting": GradientBoostingClassifier(),
    "DecisionTree": DecisionTreeClassifier(),
    "NaiveBayes": GaussianNB()
}

results = {}

for name, model in models.items():

    if name in ["NaiveBayes", "DecisionTree"]:
        pipe = Pipeline([
            ('model', model)
        ])
    else:
        pipe = Pipeline([
            ('scaler', StandardScaler()),
            ('model', model)
        ])
```

Variables Terminal

EmploySalaryPridiction_namanbhalani.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

```
LogisticRegression Accuracy: 0.8149
              precision    recall  f1-score   support

        <=50K       0.84      0.93      0.88      7010
         >50K       0.69      0.46      0.55      2334

     accuracy                           0.81      9344
    macro avg       0.77      0.70      0.72      9344
 weighted avg       0.80      0.81      0.80      9344


RandomForest Accuracy: 0.8496
              precision    recall  f1-score   support

        <=50K       0.88      0.93      0.90      7010
         >50K       0.74      0.62      0.67      2334

     accuracy                           0.85      9344
    macro avg       0.81      0.77      0.79      9344
 weighted avg       0.84      0.85      0.84      9344


KNN Accuracy: 0.8245
              precision    recall  f1-score   support

        <=50K       0.87      0.90      0.88      7010
         >50K       0.67      0.60      0.63      2334

     accuracy                           0.82      9344
    macro avg       0.77      0.75      0.76      9344
 weighted avg       0.82      0.82      0.82      9344


SVM Accuracy: 0.8396
              precision    recall  f1-score   support

        <=50K       0.86      0.94      0.90      7010
         >50K       0.75      0.54      0.63      2334

     accuracy                           0.84      9344
    macro avg       0.80      0.74      0.76      9344
```

Variables Terminal

RESULT

EmploySalaryPridiction_namanbhalani.ipynb

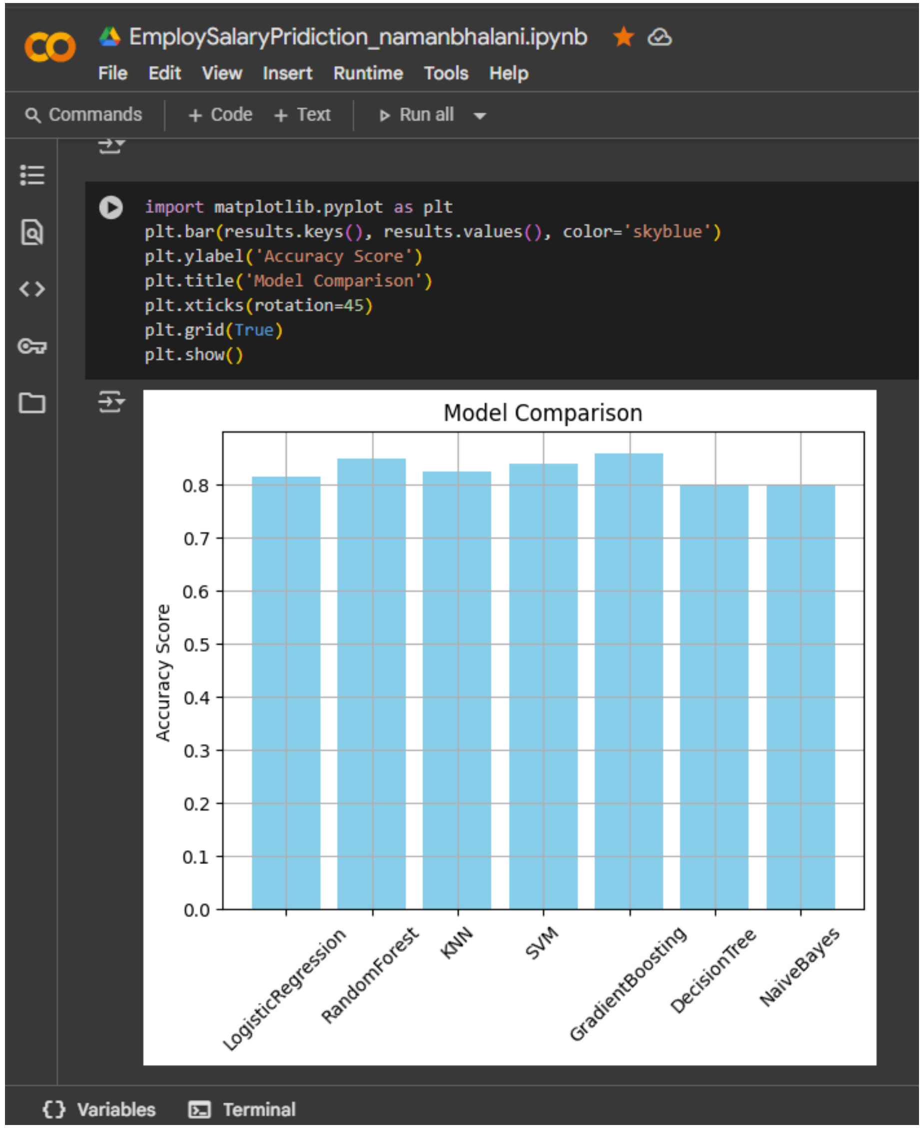
```
data=data.drop(columns=['education'])
```

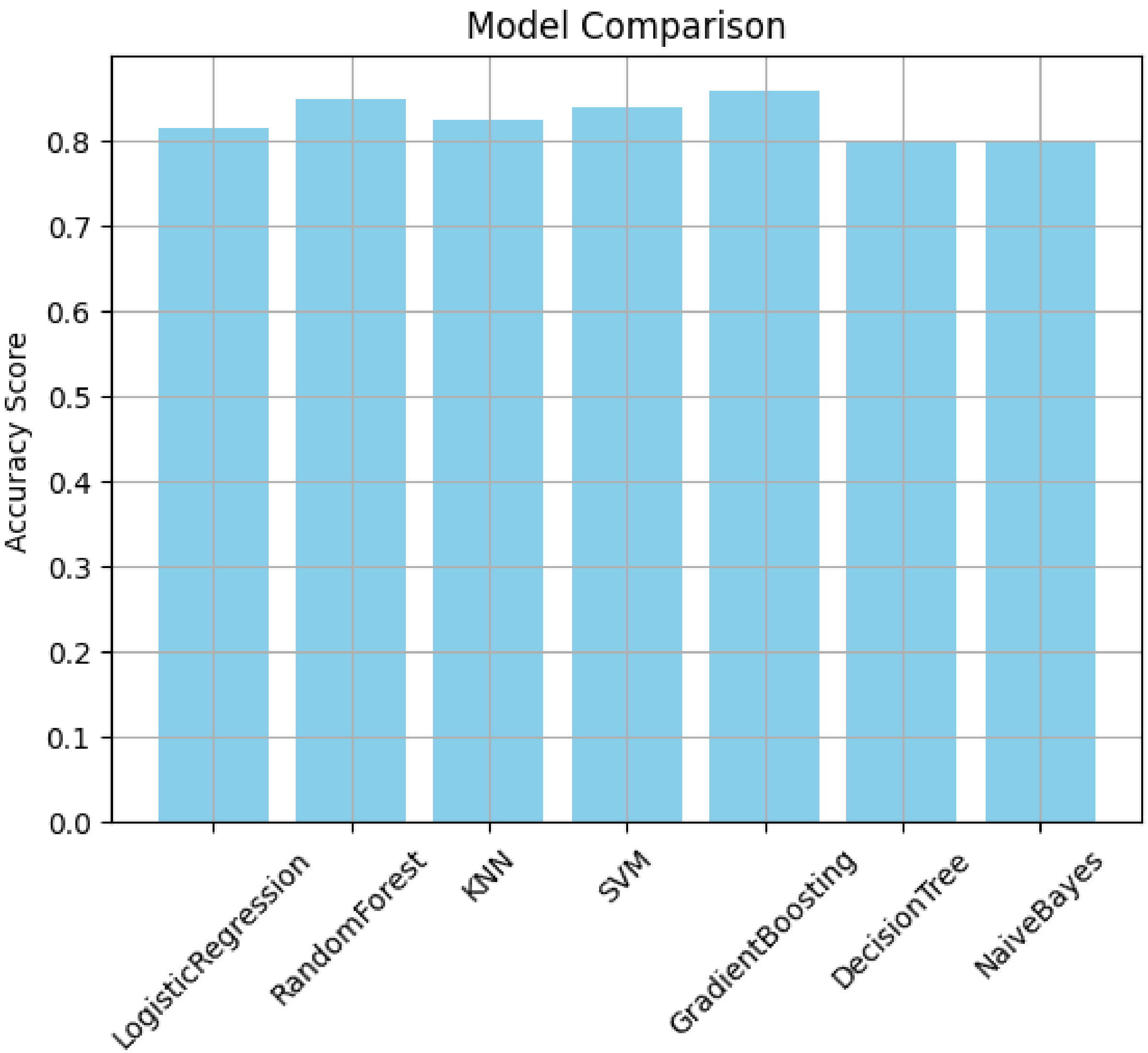
```
from sklearn.preprocessing import LabelEncoder
encoder=LabelEncoder()
data['workclass']=encoder.fit_transform(data['workclass'])
data['marital-status']=encoder.fit_transform(data['marital-status'])
data['occupation']=encoder.fit_transform(data['occupation'])
data['relationship']=encoder.fit_transform(data['relationship'])
data['race']=encoder.fit_transform(data['race'])
data['gender']=encoder.fit_transform(data['gender'])
data['native-country']=encoder.fit_transform(data['native-country'])
data
```

	age	workclass	fnlwgt	educational-num	marital-status	occupation	relationship	race	gender	capital-gain	capital-loss	hours-per-week	native-country	income
0	25	3	226802	7	4	6	3	2	1	0	0	40	39	<=50K
1	38	3	89814	9	2	4	0	4	1	0	0	50	39	<=50K
2	28	1	336951	12	2	11	0	4	1	0	0	40	39	>50K
3	44	3	160323	10	2	6	0	2	1	7688	0	40	39	>50K
4	18	2	103497	10	4	8	3	4	0	0	0	30	39	<=50K
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
48837	27	3	257302	12	2	13	5	4	0	0	0	38	39	<=50K
48838	40	3	154374	9	2	6	0	4	1	0	0	40	39	>50K
48839	58	3	151910	9	6	0	4	4	0	0	0	40	39	<=50K
48840	22	3	201490	9	4	0	3	4	1	0	0	20	39	<=50K
48841	52	4	287927	9	2	3	5	4	0	15024	0	40	39	>50K

46720 rows × 14 columns

RESULT





edunet foundation

RESULT

EmploySalaryPridiction_namanbhalani.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all Connect

```
data=data.drop(columns=['education'])
```

```
from sklearn.preprocessing import LabelEncoder
encoder=LabelEncoder()
data['workclass']=encoder.fit_transform(data['workclass'])
data['marital-status']=encoder.fit_transform(data['marital-status'])
data['occupation']=encoder.fit_transform(data['occupation'])
data['relationship']=encoder.fit_transform(data['relationship'])
data['race']=encoder.fit_transform(data['race'])
data['gender']=encoder.fit_transform(data['gender'])
data['native-country']=encoder.fit_transform(data['native-country'])
data
```

	age	workclass	fnlwgt	educational-num	marital-status	occupation	relationship	race	gender	capital-gain	capital-loss	hours-per-week	native-country	income
0	25	3	226802	7	4	6	3	2	1	0	0	40	39	<=50K
1	38	3	89814	9	2	4	0	4	1	0	0	50	39	<=50K
2	28	1	336951	12	2	11	0	4	1	0	0	40	39	>50K
3	44	3	160323	10	2	6	0	2	1	7688	0	40	39	>50K
4	18	2	103497	10	4	8	3	4	0	0	0	30	39	<=50K
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
48837	27	3	257302	12	2	13	5	4	0	0	0	38	39	<=50K
48838	40	3	154374	9	2	6	0	4	1	0	0	40	39	>50K
48839	58	3	151910	9	6	0	4	4	0	0	0	40	39	<=50K
48840	22	3	201490	9	4	0	3	4	1	0	0	20	39	<=50K
48841	52	4	287927	9	2	3	5	4	0	15024	0	40	39	>50K

46720 rows × 14 columns

Variables Terminal

RESULT

```
LogisticRegression: 0.8149
RandomForest: 0.8484
KNN: 0.8245
SVM: 0.8396
GradientBoosting: 0.8571

✅ Best model: GradientBoosting with accuracy 0.8571
✅ Saved best model as best_model.pkl
```

RESULT

Github link

- https://github.com/CodeOfNamanBhalani/EmployeeSalaryprediction

CONCLUSION

1. Effectiveness of the Proposed Solution

- The pipeline-based approach ensured consistent preprocessing and model training.
- The use of a unified evaluation method (accuracy_score and classification_report) provided a fair and comprehensive comparison.
- Saving the best-performing model using joblib allows for easy reuse in production or deployment settings.

This modular and reusable structure is effective for real-world applications and model lifecycle management.

2. Challenges Encountered

- Initial errors occurred due to undefined variables (x and y) before the train-test split. This was resolved by clearly defining the feature and target variables from the dataset.
- Some models (like Decision Trees and Naive Bayes) do not benefit from feature scaling, so conditional pipeline handling was needed.
- Choosing the correct evaluation metric could be a limitation depending on dataset characteristics (e.g., class imbalance may require precision/recall over accuracy).

edunet foundation

REFERENCES

1.Scikit-learn Library
Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011).
Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825–2830.
➤ https://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html

2.MLPClassifier (Neural Network)
Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press.
➤ Widely used reference for feedforward neural networks.

3.Random Forest Classifier
Breiman, L. (2001). *Random Forests*. Machine Learning, 45(1), 5–32.
➤ https://doi.org/10.1023/A:1010933404324

4.Gradient Boosting Machines
Friedman, J. H. (2001). *Greedy Function Approximation: A Gradient Boosting Machine*. Annals of Statistics, 29(5), 1189–1232.
➤ https://doi.org/10.1214/aos/1013203451

5.Support Vector Machines (SVM)
Cortes, C., & Vapnik, V. (1995). *Support-vector networks*. Machine Learning, 20(3), 273–297.
➤ https://doi.org/10.1007/BF00994018

edunet
foundation

REFERENCES

6.Naive Bayes Classifier
Zhang, H. (2004). *The Optimality of Naive Bayes*. FLAIRS Conference.
➤ https://www.aaai.org/Papers/FLAIRS/2004/Flairs04-009.pdf
7.K-Nearest Neighbors (KNN)
Cover, T., & Hart, P. (1967). *Nearest neighbor pattern classification*. IEEE Transactions on Information Theory, 13(1), 21–27.
➤ https://doi.org/10.1109/TIT.1967.1053964
8.Machine Learning Pipelines
IBM Cloud Docs. *What is a machine learning pipeline?*
➤ https://www.ibm.com/cloud/learn/ml-pipelines
9.Joblib Library (for Model Persistence)
Joblib Documentation.
➤ https://joblib.readthedocs.io/

THANK YOU