Employee Salary Prediction System

Meet The Query Troop

IT22109712 - Fonseka W S M

IT22071934 - Rajana H T R

IT22071316 - Shahaam M

IT22891204 - Wickramaratne A J S de Z

IT22918192 - Rathnayake S J

Background

- Salary prediction is essential for fair compensation and workforce planning.
- Analyzing data on factors such as age and education allows us to predict salary outcomes.
- Accurate predictions enable effective compensation strategies and inform employee expectations.
- Data-driven insights support informed decisions for both employers and employees in the evolving job market.

Technologies

Python

Sci-kit Learn

NumPy

Matplotlib

Pandas

Flask

HTML

Bootstrap

Target & Business Goal

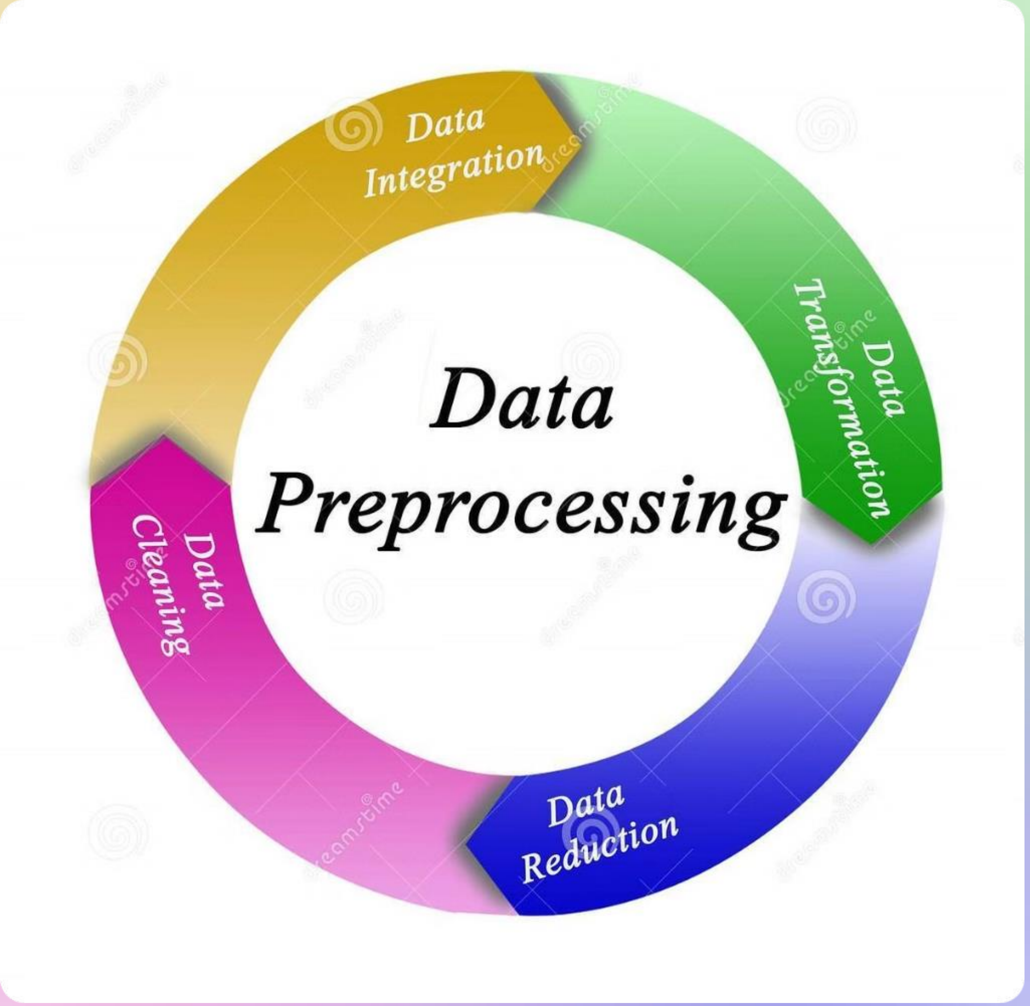
Target

- To create a precise predictive analytics model for forecasting salary outcomes based on employee attributes.

Business Goal

- To enhance compensation strategies and improve employee satisfaction, contributing to talent retention and overall business growth.

Data preprocessing



Handling Null Values

1)find null values

```
df.isnull().sum()
```

```
age                  0
workclass         1836
fnlwgt               0
education            0
education_num        0
marital_status       0
occupation        1843
relationship         0
race                 0
sex                  0
capital_gain         0
capital_loss         0
hours_per_week       0
native_country     584
income               0
dtype: int64
```

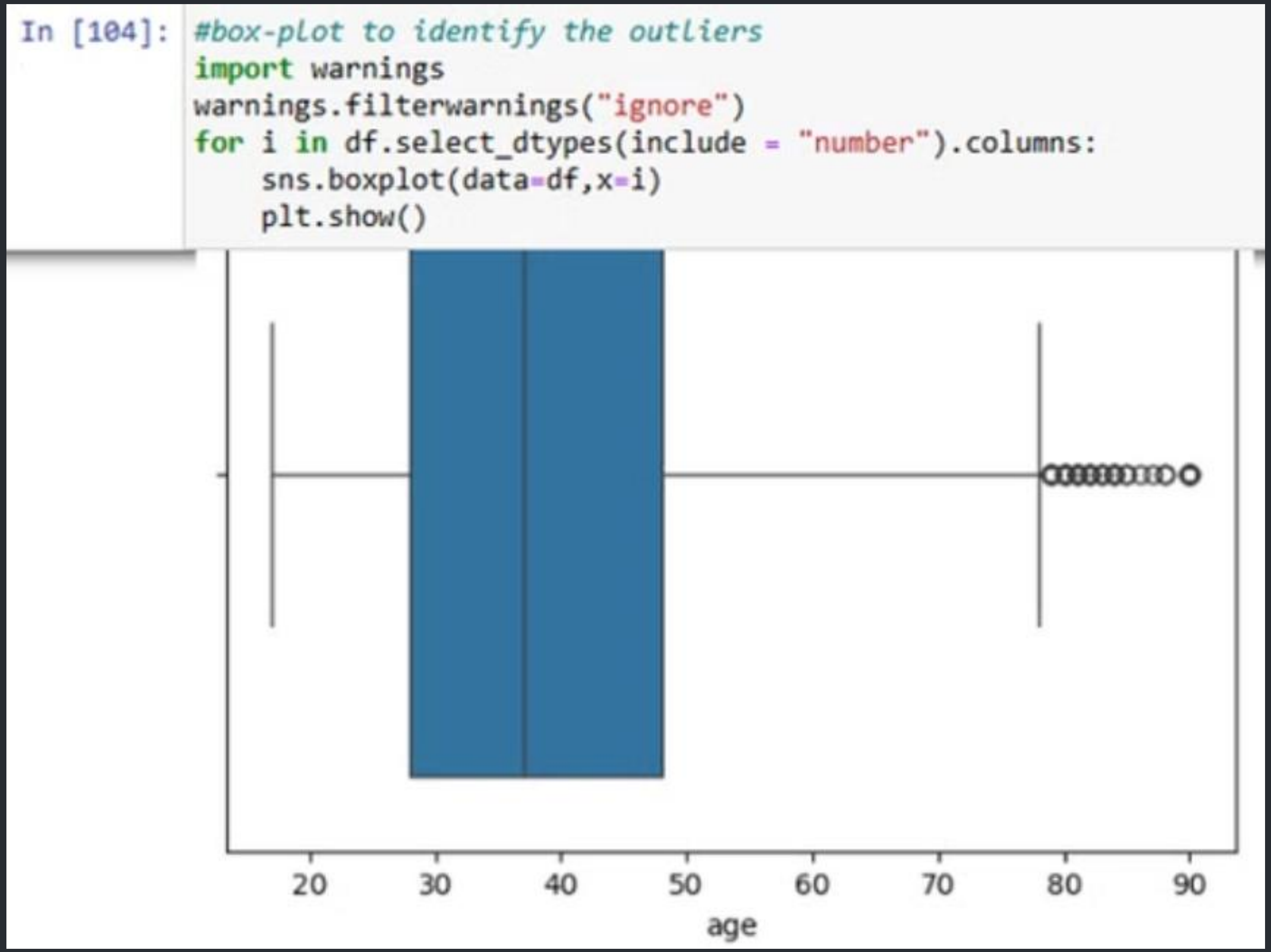
2)Filing missed values with mode

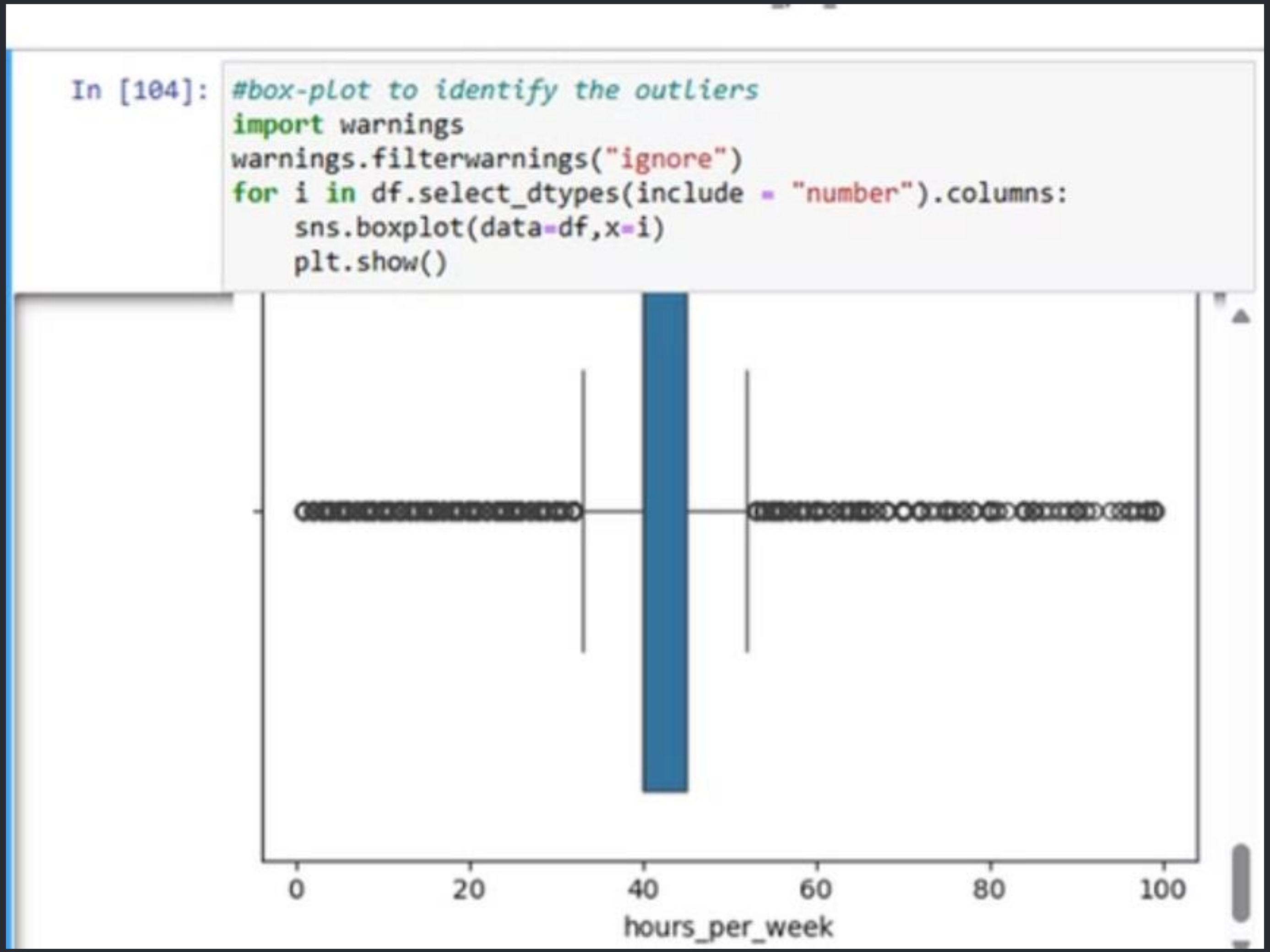
```
111]: for i in ["workclass", "occupation", "native_country"]:
          df[i].fillna(df[i].mode()[0], inplace=True)
```

```
113]: df.isnull().sum()
```

```
113]: age               0
      workclass         0
      fnlwgt            0
      education         0
      education_num     0
      marital_status    0
      occupation        0
      relationship      0
      race              0
      sex               0
      capital_gain      0
      capital_loss      0
      hours_per_week    0
      native_country    0
      income            0
      dtype: int64
```

Handling Outliers





Data Transform

- We grouped the following features into unique categories to reduce the number of unique entries in the dataset: education, work class, marital status, occupation, race, relationship, and native country.

```
In [77]: # Strip leading/trailing spaces from the 'education' column
         df['education'] = df['education'].str.strip()

         # Apply the function to group certain education levels under 'Othe
         def add_education(inpt):
             if inpt in ['10th', '7th-8th', 'Prof-school', '9th', '12th', '
                 return 'Other'
             else:
                 return inpt

         # Apply the function
         df['education'] = df['education'].apply(add_education)

         # Check the value counts
         print(df['education'].value_counts())

         education
         HS-grad         10501
         Some-college     7291
         Bachelors        5355
         Other            4067
         Masters          1723
         Assoc-voc        1382
         11th             1175
         Assoc-acdm       1067
         Name: count, dtype: int64
```

```
In [79]: df['workclass'].value_counts()

Out[79]: workclass
         Private             22696
         Self-emp-not-inc     2541
         Local-gov            2093
         State-gov            1298
         Self-emp-inc         1116
         Federal-gov           960
         Without-pay            14
         Never-worked            7
         Name: count, dtype: int64
```

Encoding Categorical Features

```
In [119]: df = pd.get_dummies(df)
```

```
In [121]: df.head()
```

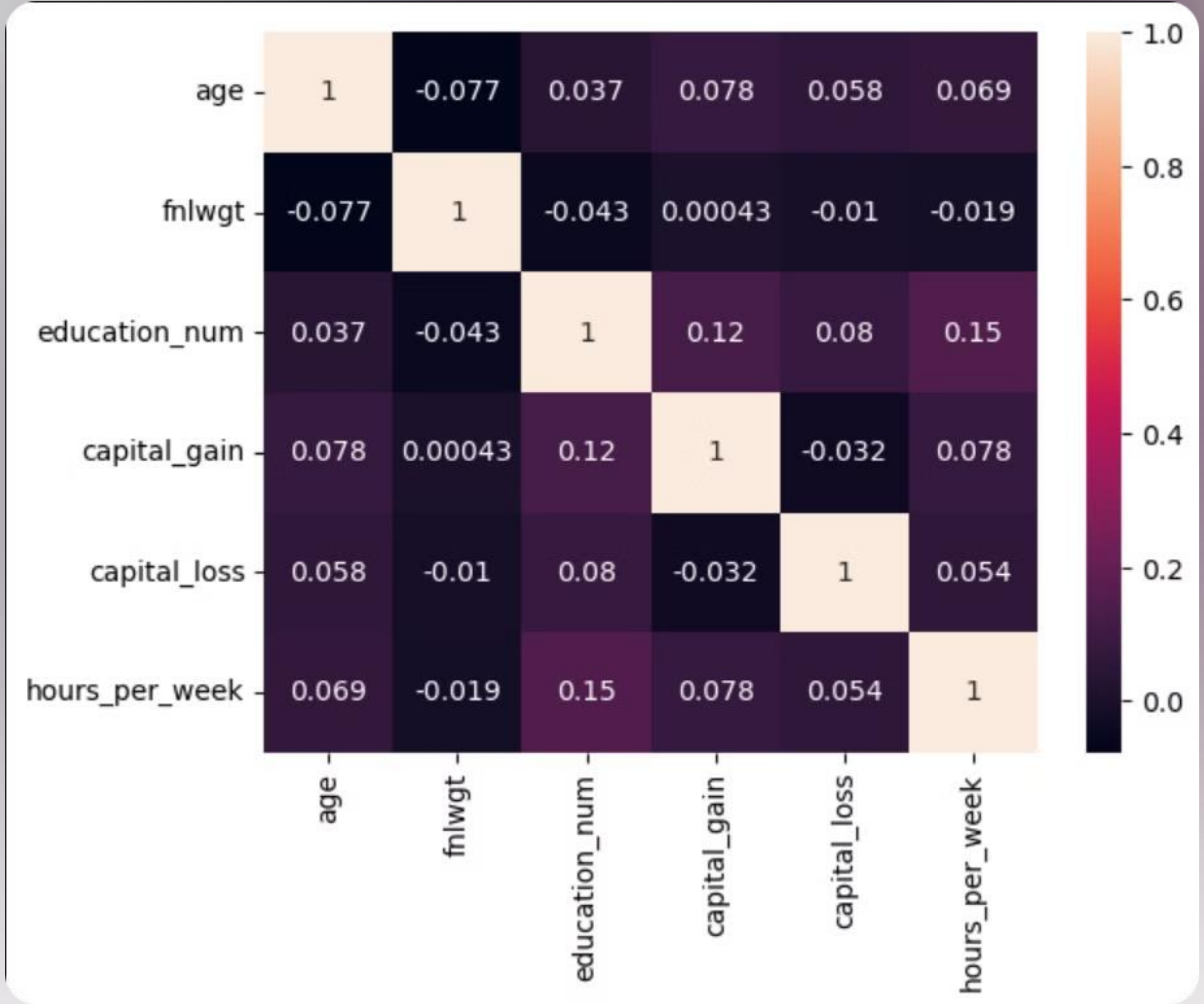
Out[121]:

	age	hours_per_week	workclass_Government	workclass_Other	workclass_Private	workclass_Self Employeed	education_11th	education_Assoc-acdm	education_Assoc-voc	edu
0	39	40	True	False	False	False	False	False	False	
1	50	13	False	False	False	True	False	False	False	
2	38	40	False	False	True	False	False	False	False	
3	53	40	False	False	True	False	True	False	False	
4	28	40	False	False	True	False	False	False	False	

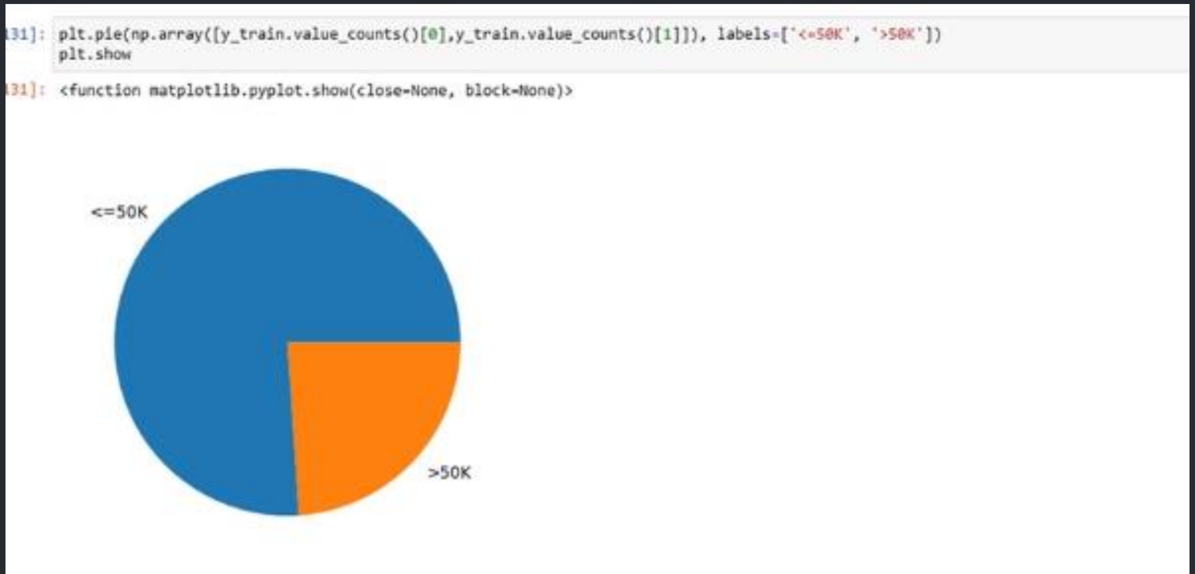
5 rows × 37 columns

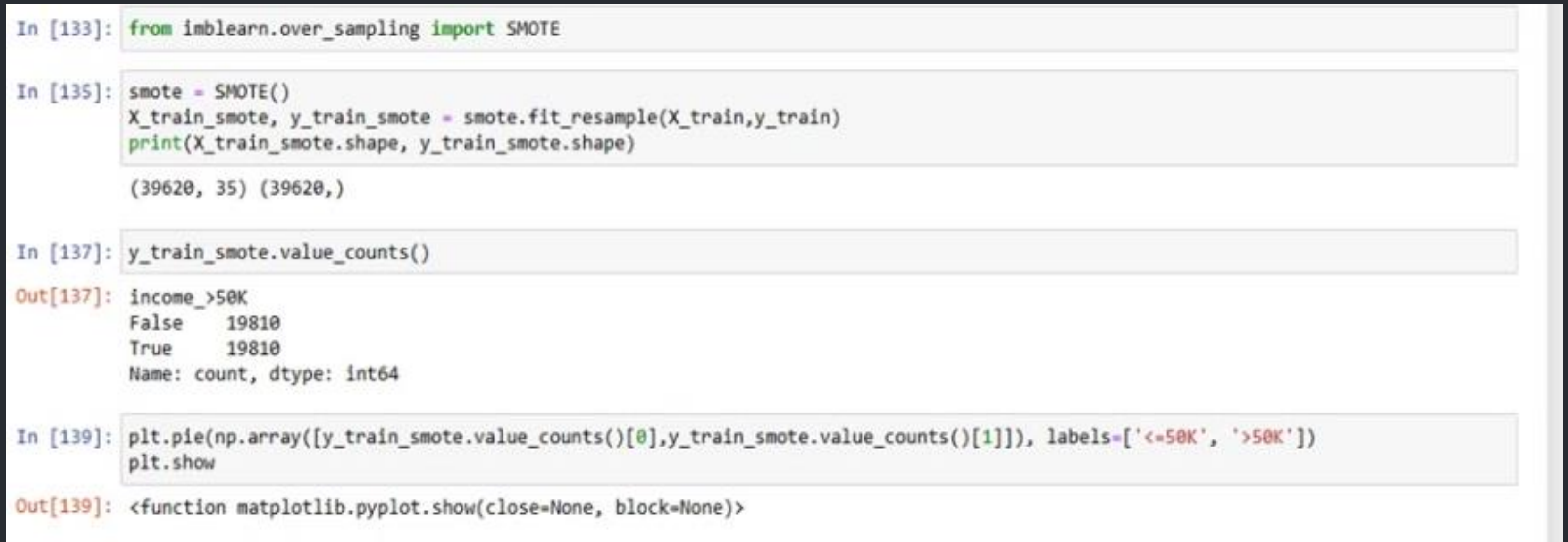
Made with Gamma

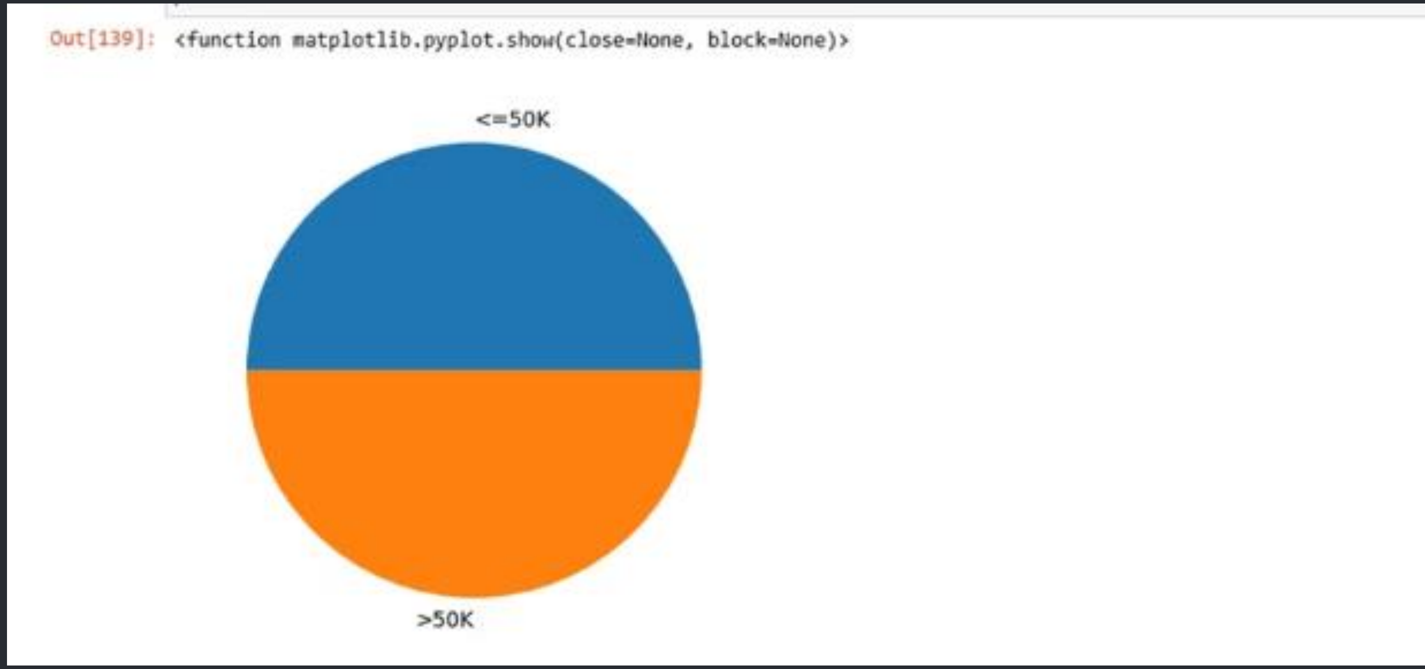
Correlation Matrix



Balance the Dataset







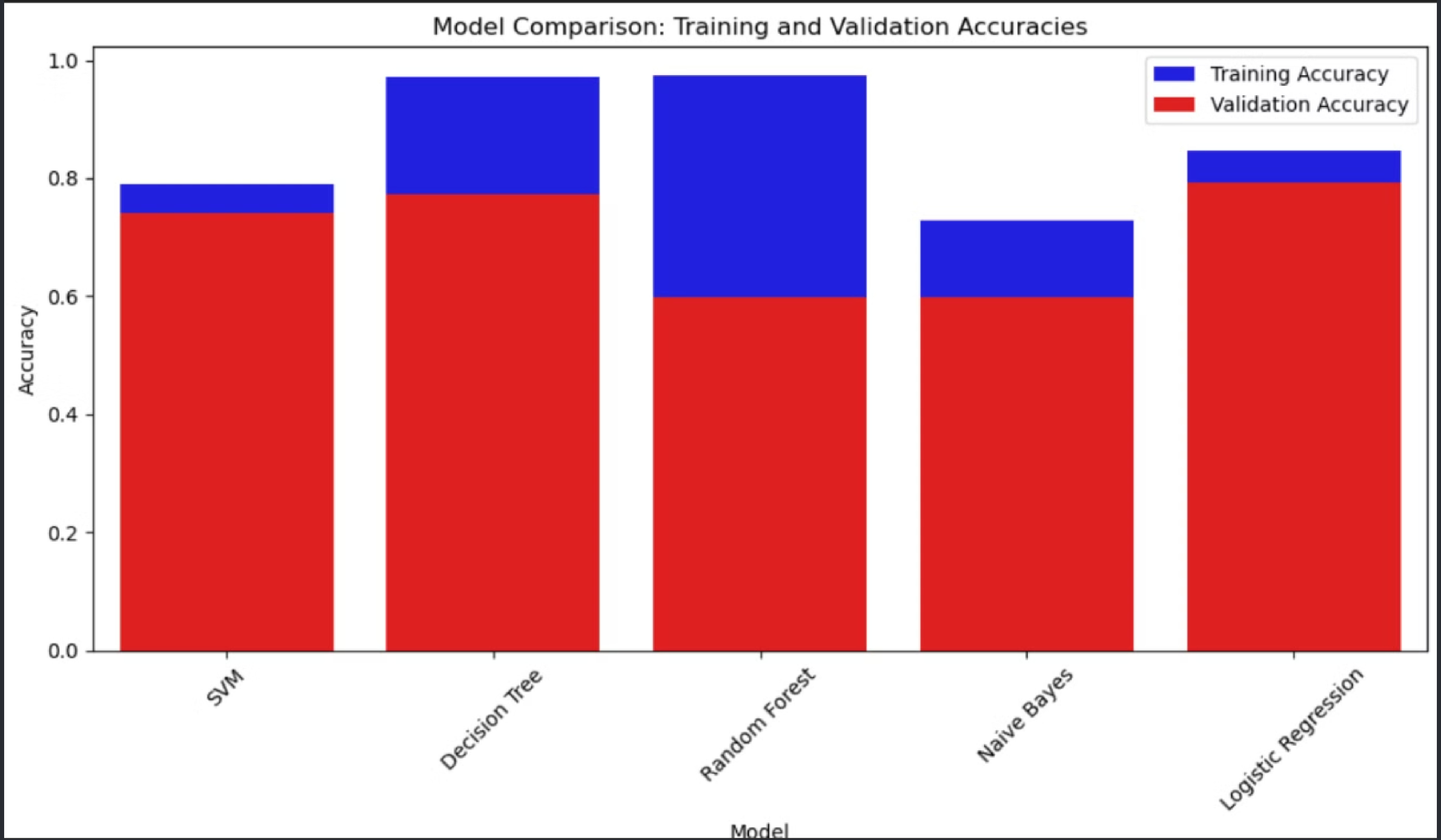
Model Implementation

Model Implementation

1. We used five different models here to predict the customer churn or not.

- Linear Regression
- Random Forest Decision Tree
- Classification
- Naïve Bayes Classification
- Support Vector Classification

2. The Random Forest Model was selected as the most effective in terms of predictive performance.



Made with Gamma

Improving the Accuracy

Improving best model by hyperparameter tuning

```
In [155]: from sklearn.model_selection import GridSearchCV

          parameters = {'n_estimators':[10, 50, 100],
                        'criterion':['gini', 'entropy', 'log_loss']}

          grid_obj = GridSearchCV(estimator=rfc, param_grid=parameters)

          grid_fit = grid_obj.fit(X_train_smote, y_train_smote)

          best_model = grid_fit.best_estimator_

          best_model.score(X_test, y_test)

Out[155]: 0.7911868570551205
```

Final Product

Employee Salary Prediction System

Challenges

- Poor data quality can result in inaccurate salary predictions.
- Certain groups may be underrepresented, causing class imbalance.
- Electing the right machine learning model is essential for accurate predictions.

Solutions

- Clean and validate data for accuracy.
- Apply SMOTE to balance classes.
- Experiment with models like logistic regression and random forests.
- Select the model with the best performance.

Further implementations and developing goals

1. Modify the system to position it as a marketable product.
2. Develop a product tailored to the Sri Lankan job market.
3. Ensure the system addresses the unique dynamics of rapid salary fluctuations in various sectors.

THANK YOU!