Intel Unnati Industrial Training Program- 2024

PROBLEM STATEMENT:

Knowledge Representation and Insight Generation from Structured Datasets

PROJECT REPORT BY: **THE SEMICOLONS**

ARCHIT CHOUDHURY (22053579)
ARNUV RAINA (22052367)
SHREYA MOHAPATRA (22053629)
SHUBHAM DAS (22053632)

Introduction

In a big data-driven age, businesses in many industries regularly deal with enormous volumes of structured data. Under the right conditions, the processing and analysis of this data can produce insightful findings that greatly improve decision-making. Nevertheless, the task is to convey this knowledge and derive valuable insights proficiently. The project "Knowledge Representation and Insight Generation from Structured Datasets" aims to develop an artificial intelligence (AI) solution to turn structured information into knowledge that can be used.

The primary objective of this project is to develop a comprehensive AI-based solution that can efficiently represent knowledge and generate insights from any structured dataset. The solution should be adept at processing and analyzing structured data, identifying patterns, and generating meaningful insights to guide strategic decisions. This involves creating a system that can handle the intricacies of structured datasets, making the data comprehensible and actionable.

The solution empowers users to unlock the value of their data. It starts by meticulously cleaning and preparing the data (think fixing errors and inconsistencies) to ensure reliable analysis. Then, it transforms raw data into clear and informative visualizations, like charts and graphs, making complex trends readily apparent. Powerful machine learning algorithms delve into the data, uncovering hidden patterns and relationships. Based on these patterns, the solution generates actionable insights, translating data into practical knowledge that guides better decision-making. This solution is built to handle a wide range of data sizes and complexities, ensuring its usefulness across various industries. Finally, a user-friendly interface makes interaction effortless, allowing users of all technical backgrounds to navigate and gain insights from their data easily.

This project will demonstrate the practical benefits of the developed AI-based solution through case studies and real-world applications. By transforming structured datasets into valuable knowledge, the project aims to provide a framework and set of best practices that organizations can adopt to harness the full potential of their data resources. The ultimate goal is to empower organizations with the insights needed to make informed, strategic choices that drive innovation, improve efficiency, and enhance overall performance.

The Adult Income Dataset: A Look at Income Prediction

This project uses the Adult dataset, a popular collection of data from the UCI Machine Learning Repository. This dataset, also known as Census Income, is often used for tasks like income prediction.

Source:

The UCI Machine Learning Repository provides a variety of datasets for machine learning research. The Adult dataset comes from the 1994 U.S. Census and is commonly used for income prediction.

What's Included:

The Adult dataset has 48,842 data points (people) and 14 features (excluding the target variable we want to predict). The goal is to predict whether someone makes more than $50,000 a year based on census information. Here's a breakdown of the features:

- **Age** (continuous)

- **Work Type** (e.g., private company, self-employed)

- **Weight** (to adjust for population estimates)

- **Education** (e.g., bachelor's degree, high school diploma)

- **Years of Education** (continuous)

- **Marital Status** (e.g., married, single)

- **Occupation** (e.g., tech support, sales)

- **Family Relationship** (e.g., spouse, child)

- **Race** (e.g., white, Asian)

- **Gender** (male or female)

- **Capital Gains** (amount of money made from selling assets)

- **Capital Losses** (amount of money lost from selling assets)

- **Hours Worked per Week** (continuous)

- **Native Country** (e.g., United States, Canada)

What We Want to Predict:

The target variable is income, a binary variable indicating whether someone makes more than $50,000 a year (>50K or <=50K).

Data Summary:

- **Type:** Multiple variables (multivariate)

- **Task:** Classification (predict a category)

- **Data Points:** 48,842

- **Features:** 14 + 1 target variable

- **Missing Values:** Yes (some data points are missing information for certain features)

How We Used It:

The Adult dataset is ideal for classification tasks, where we want to predict income based on someone's background. It's a great resource for exploring machine learning techniques like data cleaning, feature selection, model training, and evaluation.

In this project, we used the Adult dataset to develop a system for understanding and using data (knowledge representation) and to find interesting trends (insight generation). By analyzing this data, we uncovered patterns that could help in informed decision-making. The variety of features in the Adult dataset made it a strong candidate for showcasing advanced data analysis methods.

Methodology

PREDICTION MODELS:

In machine learning prediction, several powerful models offer distinct advantages and considerations. We tested the accuracies of the following four models to decide which one to use.

- **K Nearest Neighbors (KNN):** KNN makes predictions based on "similarity breeds similarity." For a new data point, it identifies the K most similar data points from the training set (its nearest neighbors). The class label (or value) that is most frequent among these K neighbors is assigned as the predicted class (or value) for the new data point. KNN excels at handling non-linear relationships but can be sensitive to the chosen number of neighbors (K) and the number of features. Additionally, it requires storing the entire training dataset for prediction.

- **Decision Tree:** This model resembles a flowchart, where data is progressively filtered based on conditions. Each branch of the tree represents a specific feature and its possible values. New data points follow the decision tree, eventually reaching a leaf node that represents the predicted class. Decision trees are interpretable (the logic is clear), and handle both numerical and categorical data well, but can overfit if not controlled (become too specific to the training data).

- **Random Forest:** Instead of relying on a single decision tree, a Random Forest builds an entire "forest" of them. Each tree is trained on a random subset of features and data points. When presented with a new data point, the forest takes a majority vote from all the individual trees to determine the final classification. This approach makes Random Forests more robust to overfitting and generally good performers across various datasets. However, understanding the specific reasoning behind predictions can be challenging ("black box" model), and training large datasets can be computationally expensive.

- **Logistic Regression:** Logistic regression is a statistical method for modeling the relationship between features (independent variables) and a binary dependent variable (usually 0 or 1). It estimates the probability of an event (classification) by analyzing how features influence the likelihood of belonging to one class versus the other. This model works well for problems with two possible outcomes (binary classification). While interpretable (coefficients show feature importance), it assumes a linear relationship between features and the target variable. This might not be suitable for complex, non-linear scenarios. Additionally, imbalanced datasets (unequal distribution of classes) can pose challenges for Logistic Regression.
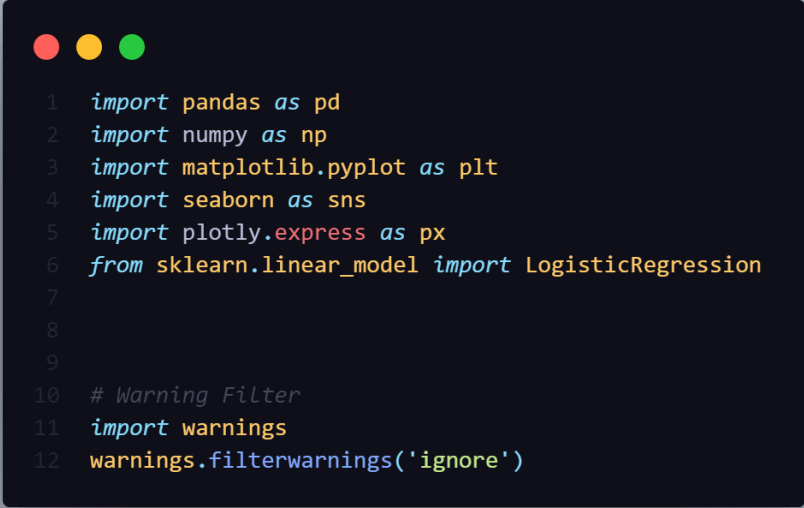
TOOLS USED:

NumPy, Pandas, Matplotlib, and Seaborn are essential tools that were used in this project. Each tool plays a specific role in the data analysis process:

- **NumPy (Numerical Python):** Provides the foundation for numerical computations. It excels at storing and manipulating large datasets of numbers efficiently. NumPy offers mathematical functions, linear algebra operations, random number generation, and more. It's the workhorse for numerical processing in Python.
- **Pandas:** Built on top of NumPy, Pandas specializes in data analysis and manipulation. It offers effective data structures like DataFrames and Series. Pandas allows us to clean, transform, and explore data using various functions and methods. It integrates seamlessly with NumPy and other data science libraries.
- **Matplotlib:** This versatile plotting library allows us to create various visualizations like scatter plots, histograms, bar charts, and more. Matplotlib provides fine-grained control over plot elements like labels, titles, legends, and customization options. It often serves as the foundation for more advanced data visualization libraries, like Seaborn.
- **Seaborn:** Built on top of Matplotlib, Seaborn offers a higher-level interface specifically designed for creating statistical data visualizations. It creates aesthetically pleasing and informative statistical graphics with a focus on clarity. Seaborn provides pre-built themes and functions for specific plots, like violin plots, heatmaps, and distribution plots. By offering easy-to-use functions and a consistent visual style, Seaborn simplifies data visualization tasks for data scientists.

These tools work together seamlessly. NumPy provides the numerical muscle, Pandas structures and manipulates data, Matplotlib offers the core functionalities for visualizations, and Seaborn builds upon Matplotlib for more refined and statistically oriented plots.

0.0.1 Loading Necessary Libraries

```python
1   import pandas as pd
2   import numpy as np
3   import matplotlib.pyplot as plt
4   import seaborn as sns
5   import plotly.express as px
6   from sklearn.linear_model import LogisticRegression
7
8
9
10  # Warning Filter
11  import warnings
12  warnings.filterwarnings('ignore')
```

0.0.2 Loading Dataset

```python
1   df = pd.read_csv('/content/adult.csv')
2   df.head(10)
```

0.0.3 Learning More about Dataset

```
1   df.columns
2   df.shape
3   df.dtypes
4   df.isnull().sum()
5   df.nunique()
6   df.describe().T
```

In the initial data analysis phase, we employed several commands to understand our dataset thoroughly. `df.columns` listed all variables, `df.shape` provided the dataset's dimensions, and `df.dtypes` identified data types. We used `df.isnull().sum()` to locate missing values, `df.nunique()` to assess variable uniqueness, and `df.describe().T` to summarize numerical features.

0.0.4 Value Count Function

```
1   df['workclass'].value_counts() #fillna
2   df['occupation'].value_counts() #fillna
3   df['native-country'].value_counts() #fillna
4   df['income'].value_counts() #0,1
5   df['marital-status'].value_counts()
6   df['gender'].value_counts()
7   df['race'].value_counts()
```

The `value_counts()` function was used to gain initial insights into the data and prepare for further analysis. This function served several purposes that contributed to knowledge representation:

1. **Data Distribution:** It revealed how frequently each unique value (category) appears in each column, allowing us to understand the data's balance and identify any heavily skewed distributions.

2. **Missing Value Exploration:** (`#fillna`) checking for missing values. Categories with missing entries weren't included in the `value_counts()` output, making them easier to detect.

3. **Categorical Data Analysis:** `value_counts()` is particularly valuable for categorical features like workclass or marital status. It provided a quick overview of the categories and their frequencies.

4. **Visualization Preparation:** The output of `value_counts()` was used to create charts (bar charts, pie charts) that effectively represent the distribution of categorical data and any emerging patterns.

0.0.5 Removing rows with '?' Values

```python
# Replacing ? with NaN
df['workclass'] = df['workclass'].replace('?', np.nan)
df['occupation'] = df['occupation'].replace('?', np.nan)
df['native-country'] = df['native-country'].replace('?', np.nan)

# Drop rows with any NaN values
df = df.dropna()

# Check the shape of the DataFrame
df.shape

df2=df.copy()
```

- The character '?' is a potential indicator of missing values. The `replace()` function was used to replace all occurrences of '?' with `np.nan` (Not a Number).
- After replacing '?' with `np.nan`, the `dropna()` function was employed to remove any rows that still contained missing values (represented by `nan`). This ensured that we were working with a dataset where all values were complete and suitable for further analysis.
- `df.shape` prints the dimensions (number of rows and columns) of the DataFrame after these cleaning steps. This would be helpful to verify that the expected number of rows remains after dropping rows with missing values.
- `df2=df.copy()` creates a copy of the cleaned DataFrame to preserve the original dataset in its uncleaned state for later reference or comparison.

0.0.6 Feature Engineering

```
1  # For Education
2  df.education = df.education.replace(['Preschool','1st-4th','5th-6th','7th-8th','9th','10th','11th','12th'],'School')
3  df.education = df.education.replace('HS-grad','High School')
4  df.education = df.education.replace(['Assoc-voc','Assoc-acdm','Prof-school','Some-college'],'Higher-Education')
5  df.education = df.education.replace('Bachelors','Under-Grad')
6  df.education = df.education.replace('Masters','Graduation')
7  df.education = df.education.replace('Doctorate','Doc')
```

- The original education categories in the `'education'` column were consolidated into more meaningful groups. This categorization simplified the education levels while preserving the essential information for analysis. It allowed us to investigate potential correlations between education attainment and income.

```
1  # For Marital status
2  df['marital-status'] = df['marital-status'].replace(['Married-civ-spouse','Married-AF-spouse'],'Married')
3  df['marital-status'] = df['marital-status'].replace(['Never-married'],'Unmarried')
4  df['marital-status'] = df['marital-status'].replace(['Divorced','Separated','Widowed','Married-spouse-absent'],'Single')
```

- The `'marital-status'` column underwent adjustments to create broader categories. This consolidation reduced the number of marital status categories while maintaining the core information relevant to income analysis, helping us explore how marital status might influence income levels.

```
# For Income
df['income'] = df['income'].replace({'<=50K': 0, '>50K':
1})
```
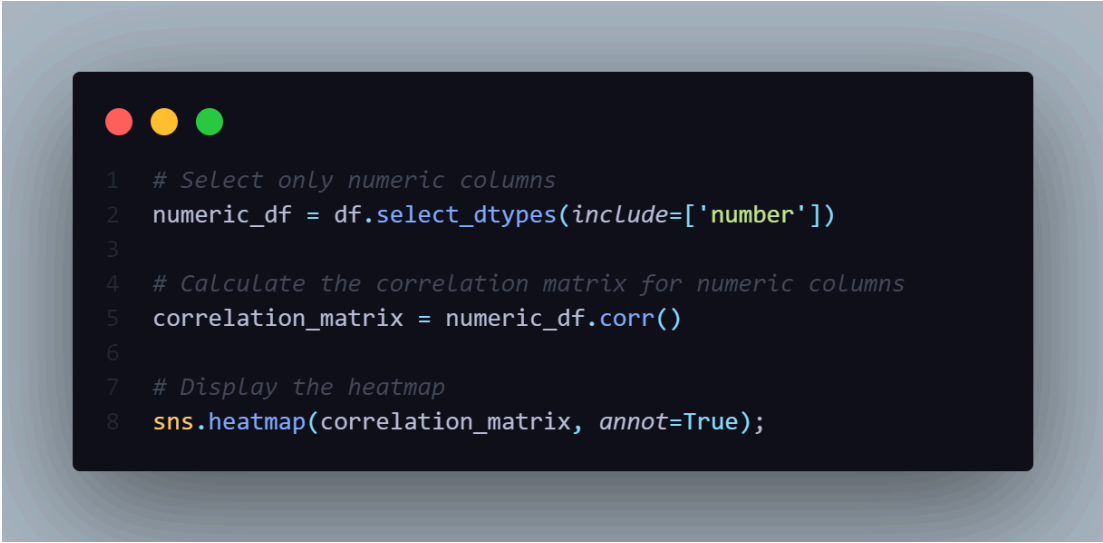
- The original income labels (`'<=50K'` and `'>50K'`) in the `'income'` column were transformed into numerical values (0 and 1, respectively). This conversion simplified calculations and prepared the data for use in machine learning models that often require numerical target variables.

```
df
df2
```

- df= Post Reductions
- df2= Pre Column Reductions (Used in Plots)

0.1.1 Some Graphs and Plots

```python
1   # Select only numeric columns
2   numeric_df = df.select_dtypes(include=['number'])
3
4   # Calculate the correlation matrix for numeric columns
5   correlation_matrix = numeric_df.corr()
6
7   # Display the heatmap
8   sns.heatmap(correlation_matrix, annot=True);
```

- Here we first filtered the dataframe `df` to keep only the columns that contained numeric data (like integers and floats). The result was stored in `numeric_df`. Then we calculated the correlation matrix for the numeric columns in `numeric_df`. The correlation matrix shows how much each numeric column is related to every other numeric column, with values ranging from -1 to 1. A value of 1 means perfect positive correlation, -1 means perfect negative correlation, and 0 means no correlation. Then we created a heatmap of the correlation matrix using Seaborn, a data visualization library. The `annot=True` parameter means that the correlation values will be displayed on the heatmap cells. The heatmap helps to quickly identify which numeric variables are related to each other and the strength of their relationships.

2. INCOME WITH EDUCATION:

```
1  # @title Income with Education
2  sns.countplot(x=df2['income'],palette='magma',hue='educatio
   n',data=df);
```

- The `sns.countplot` function creates a count plot showing the number of observations for each income level. The x-axis represents income levels, set by `x=df2['income']`. The plot differentiates bars by education levels using `hue='education'`. The data source for this plot was `df`.
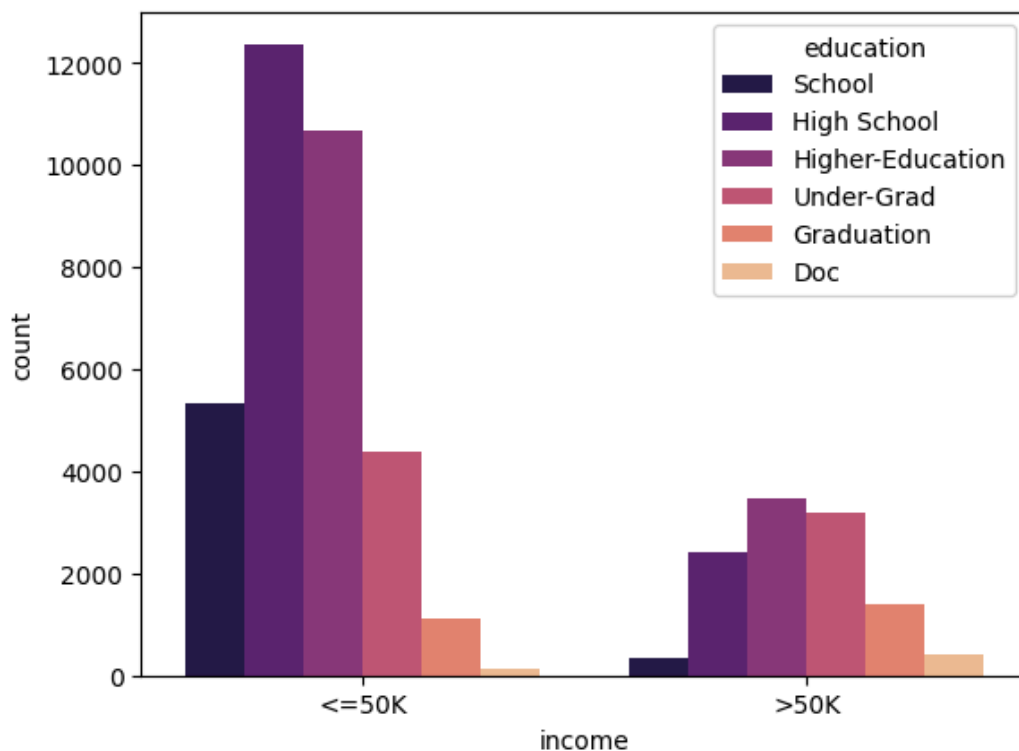
3. INCOME DISTRIBUTION BY WORKCLASS:

```
1   # @title Income Distribution by Workclass
2   plt.figure(figsize=(14,5))
3   sns.histplot(data=df2, x="workclass", hue="income",multiple
    ="stack")
4   plt.title("Income Distribution by Workclass")
5   plt.xlabel("Workclass")
6   plt.ylabel("Count")
7   plt.show()
8   print(df2['workclass'].unique())
```

● A histogram was created to show the distribution of income by workclass using the `sns.histplot` function. It stacked bars colored by income levels, with the x-axis representing workclass and the y-axis representing count. The plot was titled "Income Distribution by Workclass." Finally, the unique values in the workclass column of `df2` were printed.

Income Distribution by Workclass
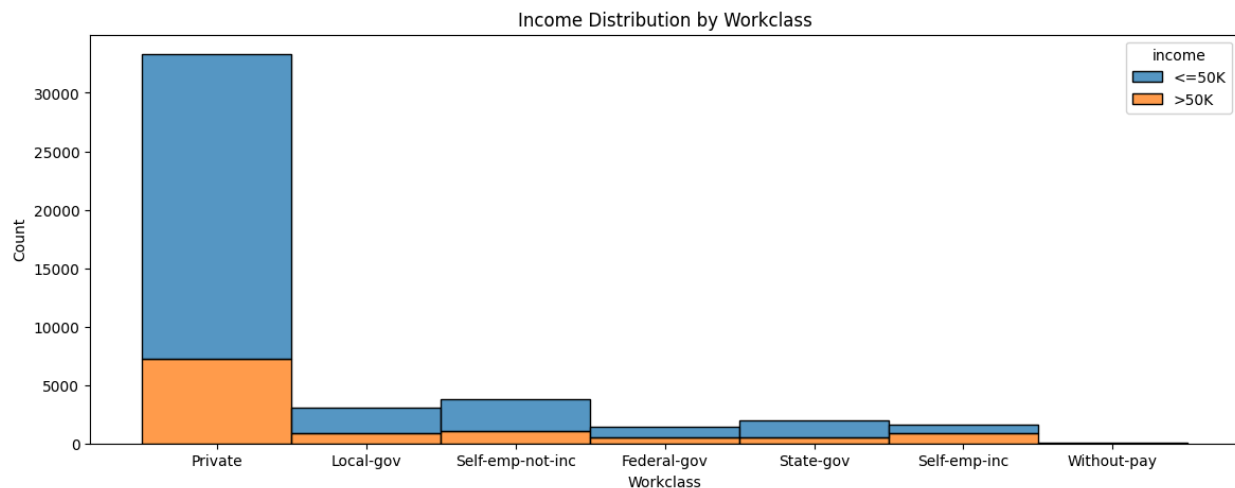
4. INCOME DISTRIBUTION BY WORKCLASS:

```
1  # @title  Income Distribution by Workclass
2  plt.figure(figsize = (15,5))
3  sns.countplot(data = df2, x = 'income',palette='magma', hue
   = 'workclass');
```

- A count plot was created using df2. The x-axis represents income levels. The bars are colored by workclass. This plot illustrates the distribution of income by workclass.
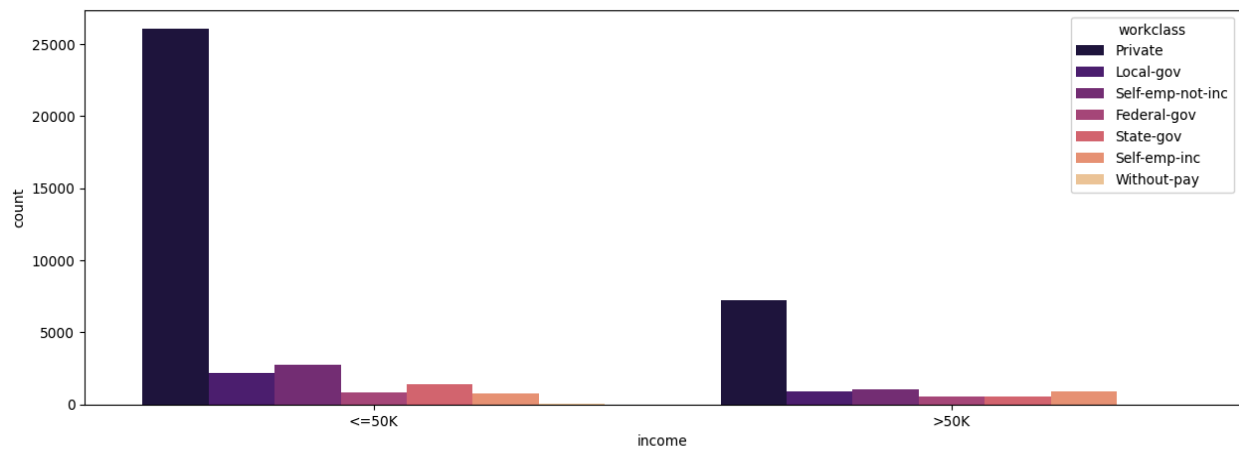
5. INCOME DISTRIBUTION BY MARITAL STATUS:

```
1   # @title Income Distribution by Marital Status
2   plt.figure(figsize=(18,5))
3   sns.countplot(data=df2, x="marital-status", hue="income")
4   plt.title("Income Distribution by Marital Status")
5   plt.xlabel("Marital Status")
6   plt.ylabel("Count")
7   plt.show()
8   print(df2['marital-status'].unique())
```

- A count plot was created using df2. The x-axis represents different marital statuses, with bars colored according to income levels to illustrate how income distribution varied across marital statuses. The plot was titled "Income Distribution by Marital Status," and labels were added to designate the x-axis as "Marital Status" and the y-axis as "Count." After displaying the plot, the unique values found in the marital-status column of df2 were printed, providing insight into the range of marital statuses covered in the dataset.
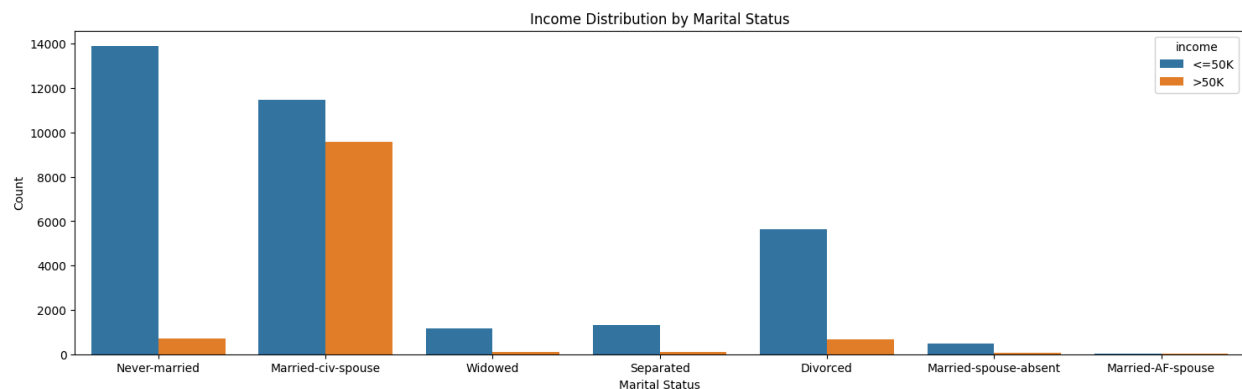
6. DISTRIBUTION OF INCOME:

```
1   # @title Income Graph
2   #df2 for income plots
3   df2['income'].value_counts().plot(kind='bar', color=sns.colo
    r_palette('Dark2'))
4   plt.title('Distribution of Income')
5   plt.xlabel('Income')
6   plt.ylabel('Count')
7   plt.xticks(rotation=0)
8   plt.show()
```

● Now the distribution of income from the df2 dataset was plotted using a bar chart. The
  `value_counts()` method was used on the `income` column to count occurrences of
  each income level, and then a bar plot was created with colors defined by Seaborn's
  'Dark2' palette. The plot was titled "Distribution of Income," with the x-axis labeled as
  "Income" and the y-axis as "Count." After setting up the plot, it was displayed to visualize
  the income distribution across different categories.
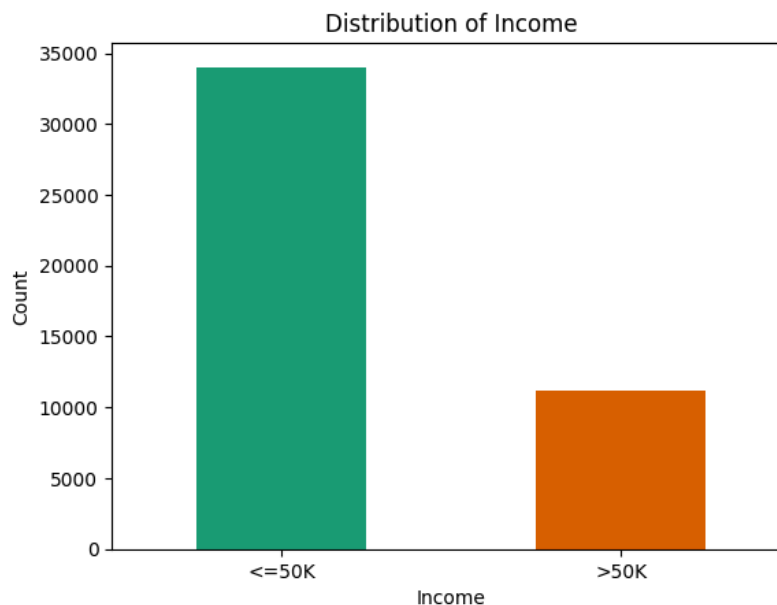
7. AGE DISTRIBUTION BY INCOME:

```python
1   # @title Age Distribution by Income
2
3   # Plot the histogram with a custom hue using Seaborn
4   plt.figure(figsize=(10, 3))
5   sns.histplot(data=df2, x="age", hue="income", kde=True, mul
    tiple="layer")
6   plt.title("Age Distribution by Income")
7   plt.xlabel("Age")
8   plt.ylabel("Count")
9   plt.show()
10
11  # Get the sorted unique ages
12  unique_age = df2['age'].unique()
13  sorted_age = np.sort(unique_age)
14  print(sorted_age)
```

Here we created a histogram with a kernel density estimate (KDE) using the age data from df2. The histogram bars were layered and colored based on different income levels, illustrating how the age distribution varied across income categories. A title "Age Distribution by Income" was added to the plot, with the x-axis labeled as "Age" and the y-axis as "Count" to provide clarity. Upon execution, the plot was displayed to visually represent these distributions. Additionally, the sorted unique ages from the age column of df2 were also printed, offering a clear view of the age ranges present in the dataset.
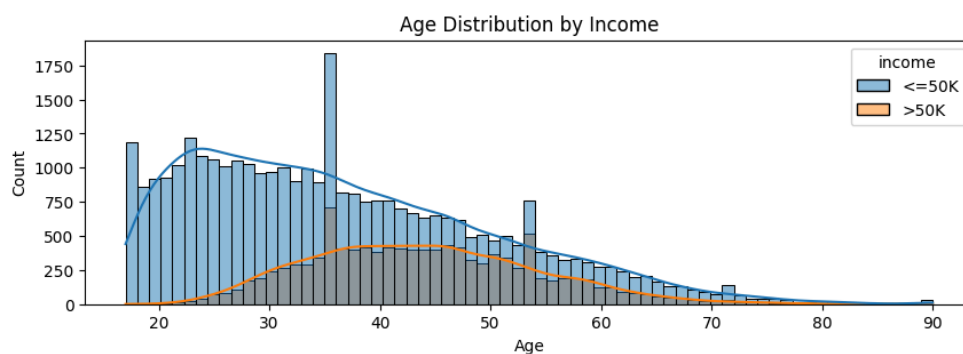


```
[17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64
 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88
 89 90]
```

8. DISTRIBUTION OF WORKCLASS CATEGORIES:

```python
# @title Distribution of Workclass Categories
workclass_counts = df2['workclass'].value_counts()
labels = workclass_counts.index
sizes = workclass_counts.values
plt.figure(figsize=(8,14))
plt.pie(sizes,labels=labels, autopct='%1.2f%%', startangle=1
40)
plt.title("Distribution of Workclass Categories")
plt.show()
print(df2['workclass'].unique())
```

- Here the frequency of each workclass category in the df2 dataset was computed using the value_counts() method on the workclass column. Then we prepared the labels and sizes for a pie chart: labels stored the unique workclass categories, while sizes held the corresponding counts of each category. A pie chart using plt.pie was created, where sizes represented the data values, labels were the category labels, %1.2f%% formatted the percentage display to two decimal places, and startangle=140 rotated the chart to begin from the 140-degree angle. Titled "Distribution of Workclass Categories," the pie chart was displayed to visually represent the distribution of workclass categories. Additionally, the unique values found in the workclass column of df2, presenting all distinct workclass categories available in the dataset were printed.

Distribution of Workclass Categories
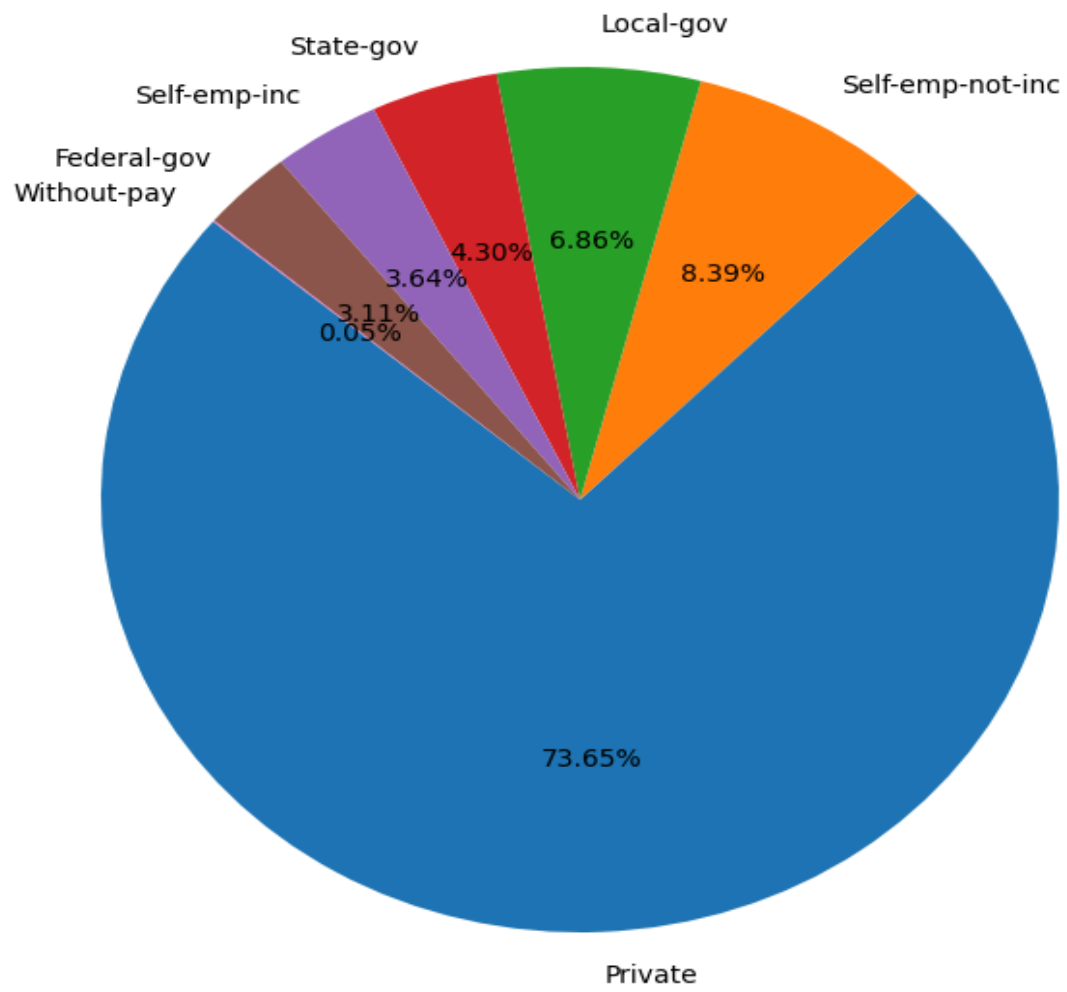
9. INCOME DISTRIBUTION BY GENDER:

```python
# @title Income Distribution by Gender
plt.figure(figsize=(4,3))
sns.countplot(data=df2,x="gender",hue="income")
plt.title("Income Distribution by Gender")
plt.xlabel("Gender")
plt.ylabel("Count")
plt.legend()
plt.show()
print(df2['gender'].unique())
```

Here the income distribution by gender was plotted using a count plot on df2, with bars colored by income levels. Titled "Income Distribution by Gender," the plot labels genders on the x-axis and counts on the y-axis. A legend clarified income level colors. It visually depicts income variations across genders, while printing unique gender categories from df2 for reference.

10. INCOME DISTRIBUTION BY EDUCATION:

```python
# @title Income Distribution by Education
education_order = df2.sort_values('educational-num')['educa
tion'].unique()
plt.figure(figsize=(7,3))
sns.countplot(data=df2,x="education",hue="income",order=edu
cation_order)
plt.title("Income Distribution by Education Level")
plt.xlabel("Education Level")
plt.ylabel("Count")
plt.xticks(rotation=60)
plt.legend(title="Income")
plt.show()
```

Here, the education levels were sorted by their `educational-num` values in `df2` to establish the x-axis order. We then created a count plot that illustrates income distribution across different education levels, where bars were color-coded based on income. Titled "Income Distribution by Education Level," our plot labeled education levels on the x-axis and displayed counts on the y-axis. To enhance readability, we rotated the x-axis labels by 60 degrees. Additionally, a legend titled "Income" was included to clarify the color representation for income levels, providing a visual insight into how income varies with educational attainment.
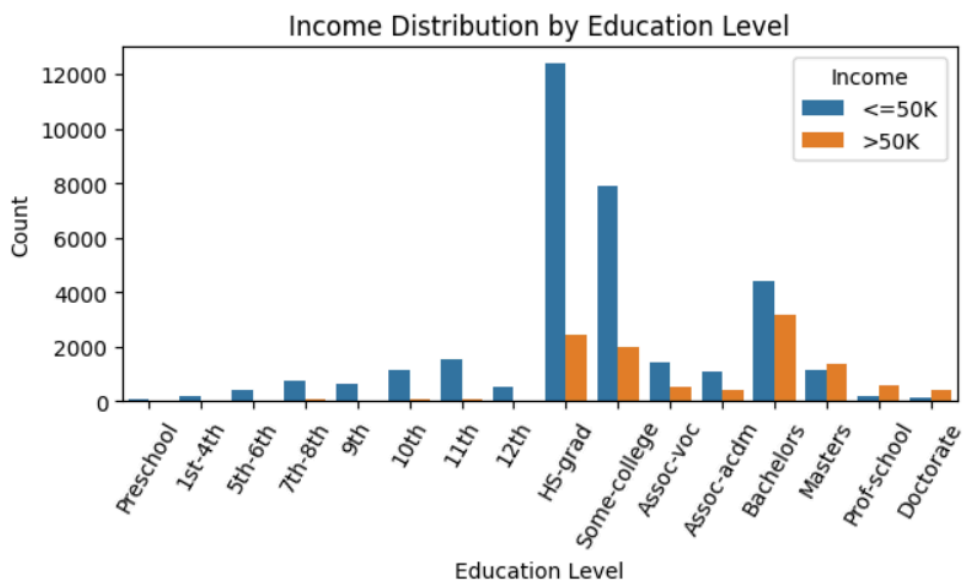
11. HISTOGRAMS:

```python
# @title Histogram
df.hist(figsize=(12,12), layout=(4,3), sharex=False);
# Noticeable that most of the data of capital gain and loss
is 0 so dropping them
df.drop(['capital-gain', 'capital-loss'], axis=1, inplace=True)
```

Histograms were initially generated for each numeric column in the DataFrame `df` to visualize their distributions. These histograms provided insights into the data's characteristics. Subsequently, the columns 'capital-gain' and 'capital-loss' were found to primarily contain zeros, indicating minimal variability. To streamline the dataset and focus on more informative features, these columns were removed from `df` using `drop(axis=1, inplace=True)`. This preprocessing step aimed to enhance the dataset's suitability for further analysis or modeling tasks.
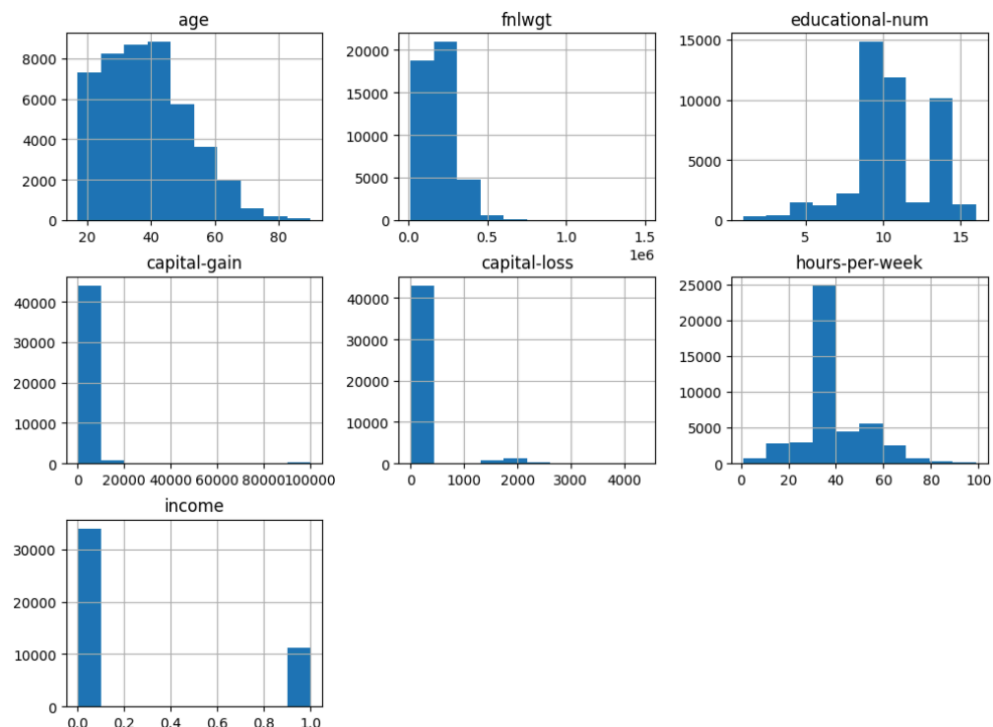
```
1    # @title After Dropping capital gain, Loss
2    df.hist(figsize=(12,12), Layout=(4,3), sharex=False);
```

- These histograms provided insights after the dropping of capital-gain and capital-loss.
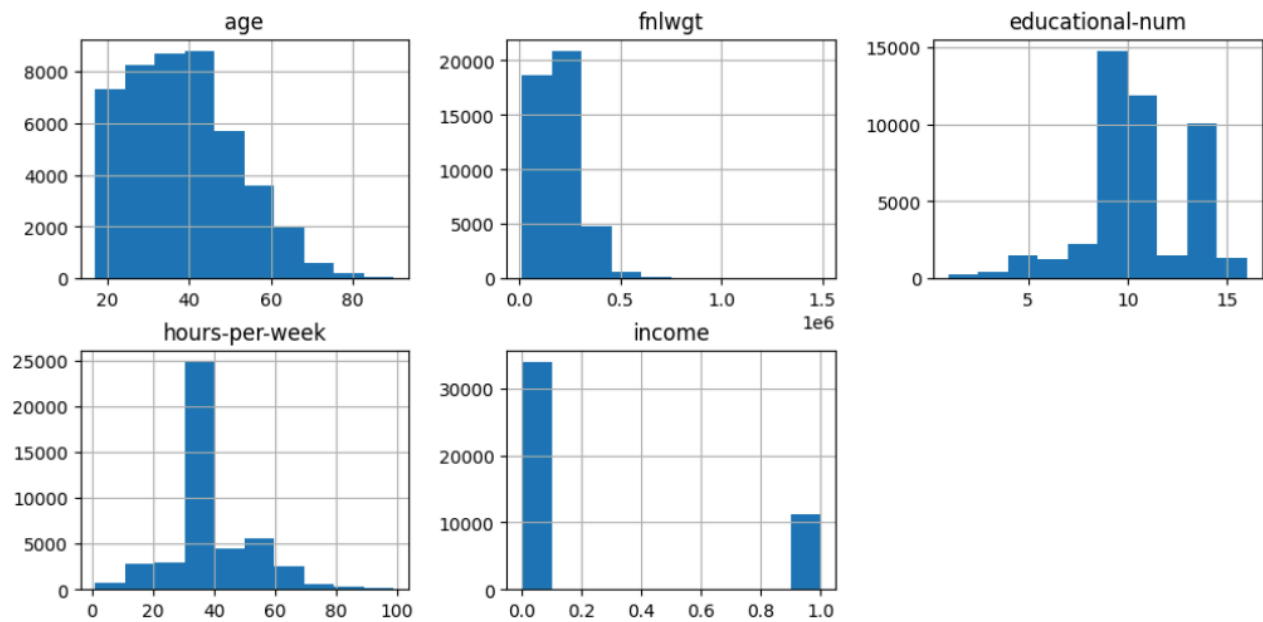
12. BOX PLOTS

```
1   # @title Boxplot for checking Outliers
2   df.plot(kind='box',figsize=(12,12),Layout=(3,3),sharex=Fals
    e,subplots=True);
```

We utilized boxplots to analyze potential outliers in each numeric column of the DataFrame df. Using .plot(kind='box'), we displayed these boxplots in a 3x3 grid with a figure size of 12x12 inches. Each subplot had its x-axis scale (sharex=False) and provided insights into the distribution, median, quartiles, and outliers within the dataset's numeric variables. This visualization was instrumental in identifying data anomalies and guiding subsequent exploratory analysis or preprocessing efforts to ensure data reliability and interpretability.
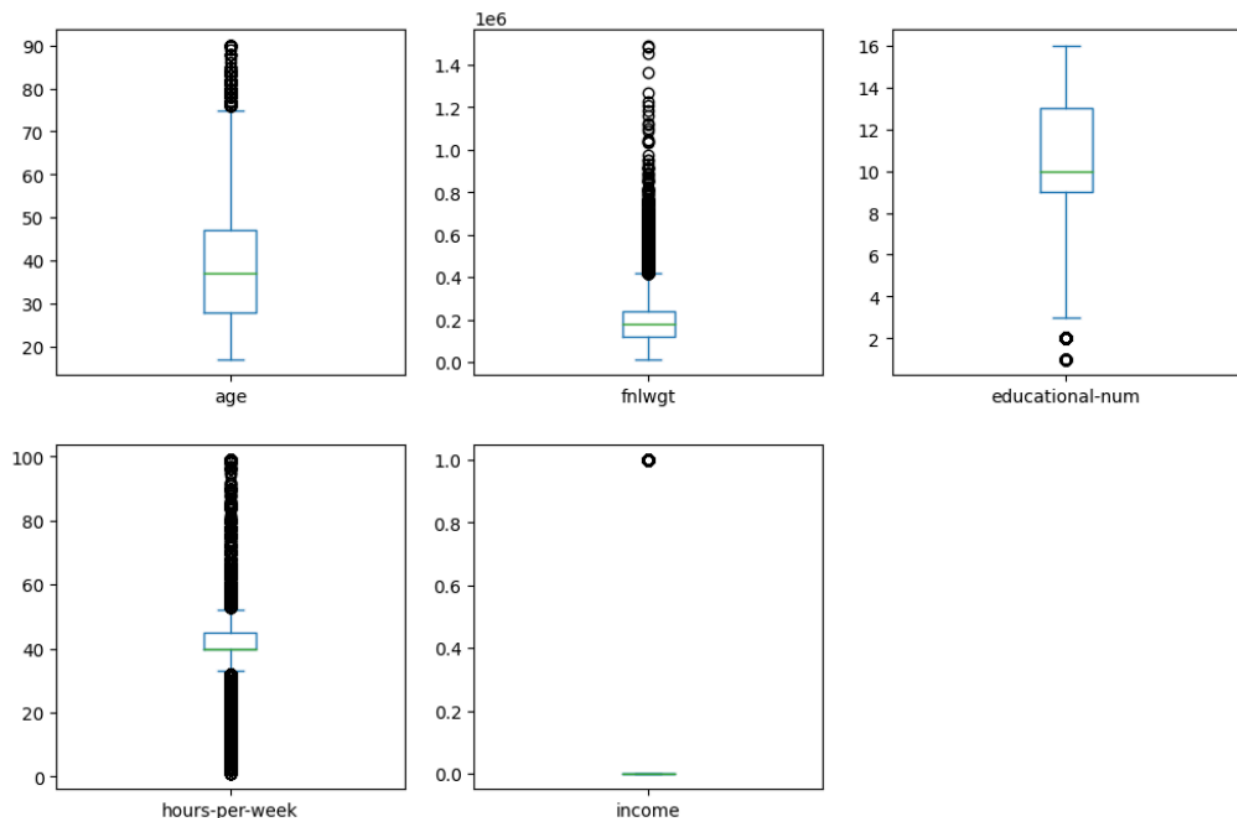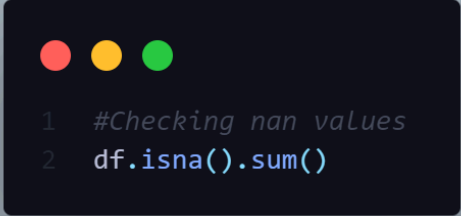
Finding out the Categorical Dataset

```
1   # To find out the Categorical dataset
2
3   categorical_df = df.select_dtypes(object)
4   categorical_df
```

This helped identify **categorical features** (columns with distinct categories) in the data. This was important because we needed to use different analysis techniques for these features compared to numerical ones.

Checking for Missing Values:

```
1   #Checking nan values
2   df.isna().sum()
```

df.isna().sum() was used to check if there were any remaining missing values (represented by NaN) in the DataFrame df. This provided a quick overview of the data quality after the initial cleaning steps.

Data Splitting and Preprocessing:

```python
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
# One-hot encoding categorical variables
df = pd.get_dummies(df, columns=['education', 'marital-status', 'race', 'gender', 'relationship', 'occupation', 'workclass', 'native-country'])

# Splitting the dataset into features and target variable
X = df.drop('income', axis=1)
y = df['income']

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Scaling the features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

GridSearchCV: This tool is used for efficiently searching through a hyperparameter space to find the optimal configuration for a machine learning model.

This section outlines the steps taken to prepare and split the Adult Income dataset for machine learning analysis. These steps ensured the data was suitable for training and evaluating models that could predict income levels.

1. **One-Hot Encoding:** Categorical features were converted into numerical representations using pd.get_dummies. This allowed machine learning models to handle these features effectively.

2. **Feature and Target Variable Separation:** The data was separated into features (independent variables) stored in `X` and the target variable (dependent variable) stored in `y`. `X` includes all columns except `"income"`, which represents the target variable we aimed to predict.

3. **Train-Test Split:** The data was divided into training and testing sets using `train_test_split`. The training set (80% by default) was used to train the models, and the testing set (20%) was used to evaluate their performance on unseen data. Specifying `random_state=42` ensured the reproducibility of the split if we ran the code again.

4. **Feature Scaling:** A `StandardScaler` was used to normalize the features in the training set (`X_train`). Scaling helps prevent features with larger scales from dominating the model during training and improves model performance. The scaler was then applied to the testing set (`X_test`) using the parameters learned from the training data to ensure consistency.

KNN Model Evaluation

```python
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix, accuracy_score


# Fitting the KNN model
knn = KNeighborsClassifier(n_neighbors=5)  # You can tune n_neighbors
knn.fit(X_train, y_train)

# Predicting the test set results
y_pred_knn = knn.predict(X_test)

# Making the Confusion Matrix
cm_knn = confusion_matrix(y_test, y_pred_knn)
print("KNN Confusion Matrix:")
print(cm_knn)

# Calculating the accuracy score
accuracy_knn = accuracy_score(y_test, y_pred_knn)
print('KNN Accuracy:', accuracy_knn)

# Calculating the accuracy score using the confusion matrix
accuracy_from_cm_knn = (cm_knn[0, 0] + cm_knn[1, 1]) / cm_knn.sum()
print('KNN Accuracy from Confusion Matrix:', accuracy_from_cm_knn)
```

This section describes the process of evaluating the K-Nearest Neighbors (KNN) model's performance in predicting income levels in the Adult Income dataset. This allowed us to assess how well the model generalized to unseen data and identify potential areas for improvement (e.g., by tuning hyperparameters like `n_neighbors`).

1. **Prediction on Test Set:** The trained KNN model (`knn`) was used to predict income labels for the unseen testing data (`X_test`). The predicted labels were stored in `y_pred_knn`.

2. **Confusion Matrix:** A confusion matrix (`cm_knn`) was generated using `confusion_matrix` from `sklearn.metrics`. This matrix visualized the number of correct and incorrect predictions for each income category (<= $50K or > $50K). It provided insights into how well the model classified the income levels.

3. **Accuracy Score:** Two methods were used to calculate the overall accuracy of the KNN model:

    - `accuracy_score` from `sklearn.metrics` directly compares the predicted labels (`y_pred_knn`) with the actual labels (`y_test`) in the testing set. This gives a percentage value, representing the proportion of correct predictions.

    - A manual calculation using the confusion matrix (`cm_knn`) was also done. It summed the number of correctly predicted values (diagonals) and divided by the total number of predictions. This approach demonstrates how accuracy can be derived from the confusion matrix.

Decision Tree Model Evaluation

```python
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, accuracy_score

# Fitting the Decision Tree model
dt = DecisionTreeClassifier(random_state=42)
dt.fit(X_train, y_train)

# Predicting the test set results
y_pred_dt = dt.predict(X_test)

# Making the Confusion Matrix
cm_dt = confusion_matrix(y_test, y_pred_dt)
print("Decision Tree Confusion Matrix:")
print(cm_dt)

# Calculating the accuracy score
accuracy_dt = accuracy_score(y_test, y_pred_dt)
print('Decision Tree Accuracy:', accuracy_dt)

# Calculating the accuracy score using the confusion matrix
accuracy_from_cm_dt = (cm_dt[0, 0] + cm_dt[1, 1]) / cm_dt.sum()
print('Decision Tree Accuracy from Confusion Matrix:', accuracy_from_cm_dt)
```

A Decision Tree Classifier model (dt) was created using DecisionTreeClassifier from sklearn.tree with a fixed random state (random_state=42) for reproducibility. The model was then trained on the training data (X_train, y_train) using the fit method.

Next, the trained model was used to predict income labels for the unseen testing data (X_test). The predicted labels were stored in y_pred_dt.

To evaluate the model's performance, a confusion matrix (cm_dt) was generated using confusion_matrix from sklearn.metrics. This matrix was printed to visualize the distribution of correct and incorrect predictions for each income category.

The previously used two methods were employed to calculate the overall accuracy of the Decision Tree model:

- accuracy_score from sklearn.metrics was used to compare the predicted labels (y_pred_dt) with the actual labels (y_test) in the testing set. The resulting value (accuracy_dt) represents the proportion of correct predictions.

- A manual calculation using the confusion matrix (cm_dt) was also performed to derive accuracy. It involved summing the number of correctly predicted values (diagonals) and dividing by the total number of predictions, storing the result in accuracy_from_cm_dt.

Random Forest Model Evaluation

```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, accuracy_score

# Fitting the Random Forest model
rf = RandomForestClassifier()
rf.fit(X_train, y_train)

# Predicting the test set results
y_pred_rf = rf.predict(X_test)

# Making the Confusion Matrix
cm_rf = confusion_matrix(y_test, y_pred_rf)
print("Random Forest Confusion Matrix:")
print(cm_rf)

# Calculating the accuracy score
accuracy_rf = accuracy_score(y_test, y_pred_rf)
print('Random Forest Accuracy:', accuracy_rf)

# Calculating the accuracy score using the confusion matrix
accuracy_from_cm_rf = (cm_rf[0, 0] + cm_rf[1, 1]) / cm_rf.sum()
print('Random Forest Accuracy from Confusion Matrix:', accuracy_from_cm_rf)
```

The Random Forest Classifier model (rf) was implemented using RandomForestClassifier from sklearn.ensemble. The model was trained on the training data (X_train, y_train) using the fit method.

Following training, the model was employed to predict income labels for the unseen testing data (`X_test`). The predicted labels were stored in `y_pred_rf`.

To assess the model's performance, a confusion matrix (`cm_rf`) was generated using `confusion_matrix` from `sklearn.metrics`. This matrix was printed to provide insights into the distribution of correct and incorrect predictions for each income category.

The `accuracy_score` function and manual calculation using the confusion matrix (`cm_rf`) were also used to calculate the overall accuracy of the Random Forest model.

Logistic Regression Model Evaluation

```python
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, accuracy_score

# Fitting the Logistic Regression model
logreg = LogisticRegression()
logreg.fit(X_train, y_train)

# Predicting the test set results
y_pred_logreg = logreg.predict(X_test)

# Making the Confusion Matrix
cm_logreg = confusion_matrix(y_test, y_pred_logreg)
print("Logistic Regression Confusion Matrix:")
print(cm_logreg)

# Calculating the accuracy score using the accuracy_score function
accuracy_logreg = accuracy_score(y_test, y_pred_logreg)
print('Logistic Regression Accuracy:', accuracy_logreg)

# Calculating the accuracy score using the confusion matrix
accuracy_from_cm_logreg = (cm_logreg[0, 0] + cm_logreg[1, 1]) / cm_logreg.sum()
print('Logistic Regression Accuracy from Confusion Matrix:', accuracy_from_cm_logreg)
```
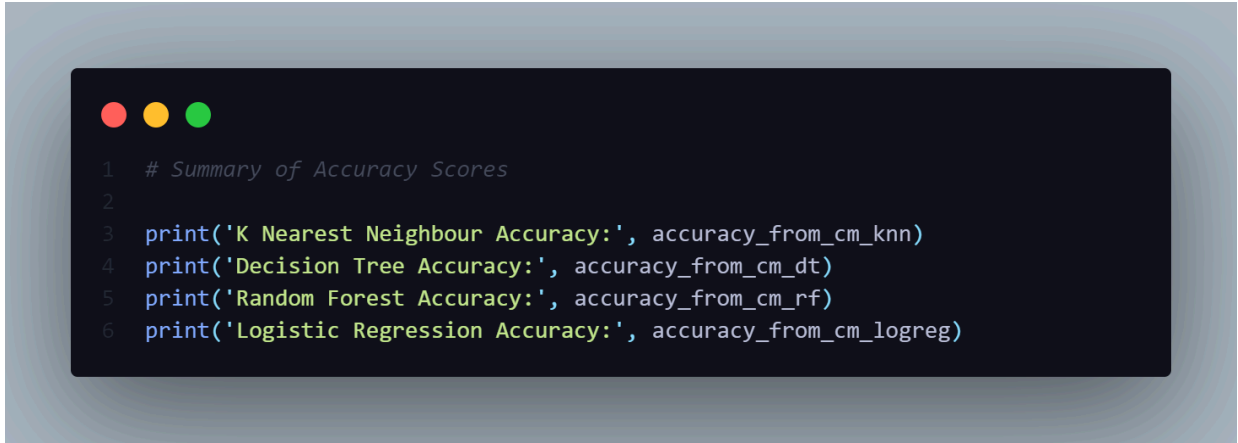
A Logistic Regression model (`logreg`) was created using `LogisticRegression` from `sklearn.linear_model` and subsequently trained on the training data (`X_train`, `y_train`) using the `fit` method.

Following training, the model was employed to predict income labels for the unseen testing data (`X_test`). The predicted labels were stored in `y_pred_logreg`.

To assess the model's performance, a confusion matrix (`cm_logreg`) was generated using `confusion_matrix` from `sklearn.metrics`. This matrix was printed to provide insights into the distribution of correct and incorrect predictions for each income category.

The `accuracy_score` function and manual calculation using the confusion matrix (`cm_rf`) were again used to calculate the overall accuracy of the Logistic Regression model.

Analysis of Model Performance

```
# Summary of Accuracy Scores

print('K Nearest Neighbour Accuracy:', accuracy_from_cm_knn)
print('Decision Tree Accuracy:', accuracy_from_cm_dt)
print('Random Forest Accuracy:', accuracy_from_cm_rf)
print('Logistic Regression Accuracy:', accuracy_from_cm_logreg)
```

This comparison of accuracy scores, alongside the previously presented confusion matrices, provided a comprehensive overview of how each model performed on the income prediction task. By analyzing these results, we could identify the model that achieved the best performance and explore potential reasons behind the variations in accuracy between the models.

Results and Discussion

Graphs and Plots

A. INCOME WITH EDUCATION:
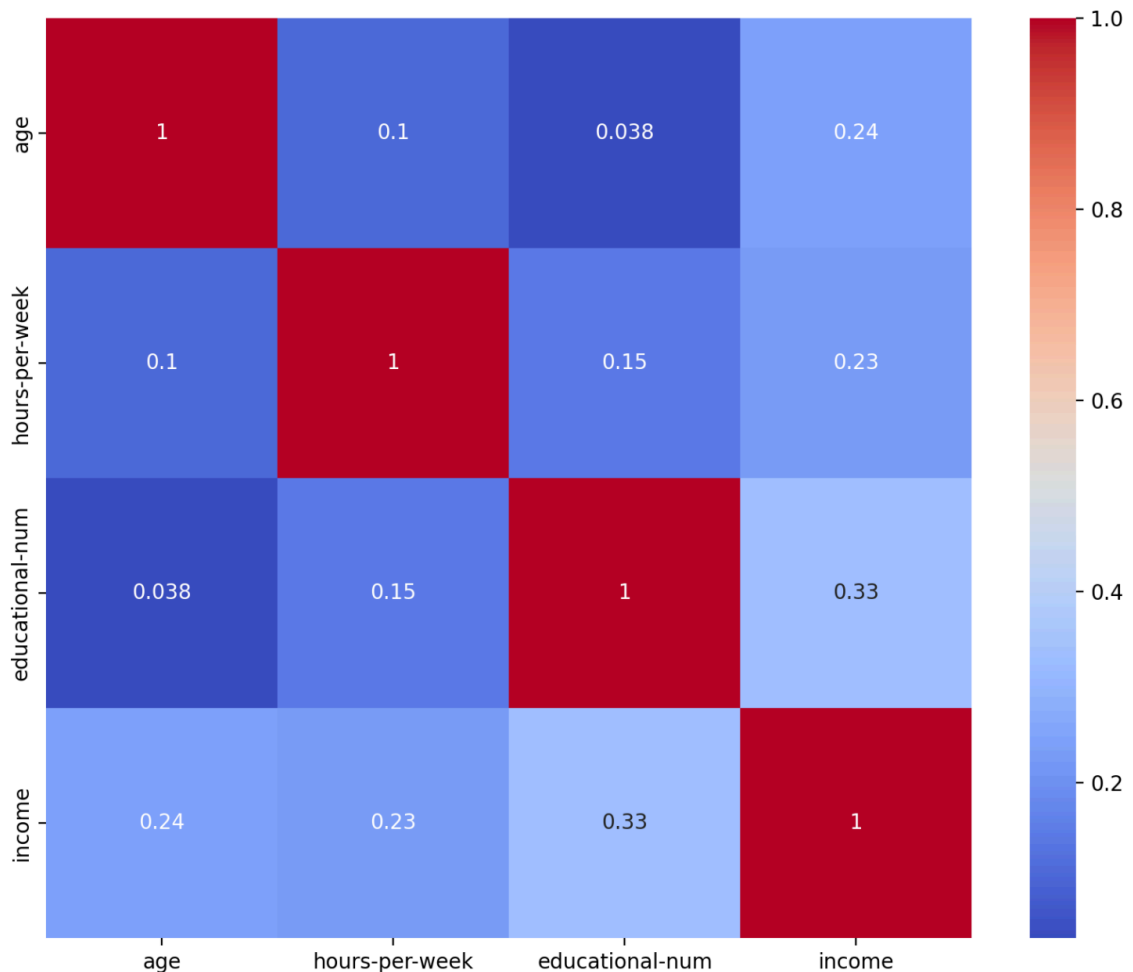
Here, it is quite evident from the heatmap that some factors have a more profound impact on income (i.e., they show a stronger positive correlation with income) than others. Here fnlwgt was excluded as it is a variable used to represent the number of people, i.e., it is essentially a sample weight, indicating how many individuals in the population share the same characteristics as the observed individual in the dataset, and did not have a direct impact on income. Capital-gain and capital-loss were also removed, as they are measures of the financial performance of an individual's investments(with capital-gain indicating profit and capital-loss indicating a loss) and do not necessarily highlight the difference in workplace based income disparity.

It can be seen from the heatmap that for income, the order of the factors on which the income depended was educational qualification, age, and then number of hours per week (with educational qualification having the most effect and number of hours per week the least)

B. INCOME WITH EDUCATION:
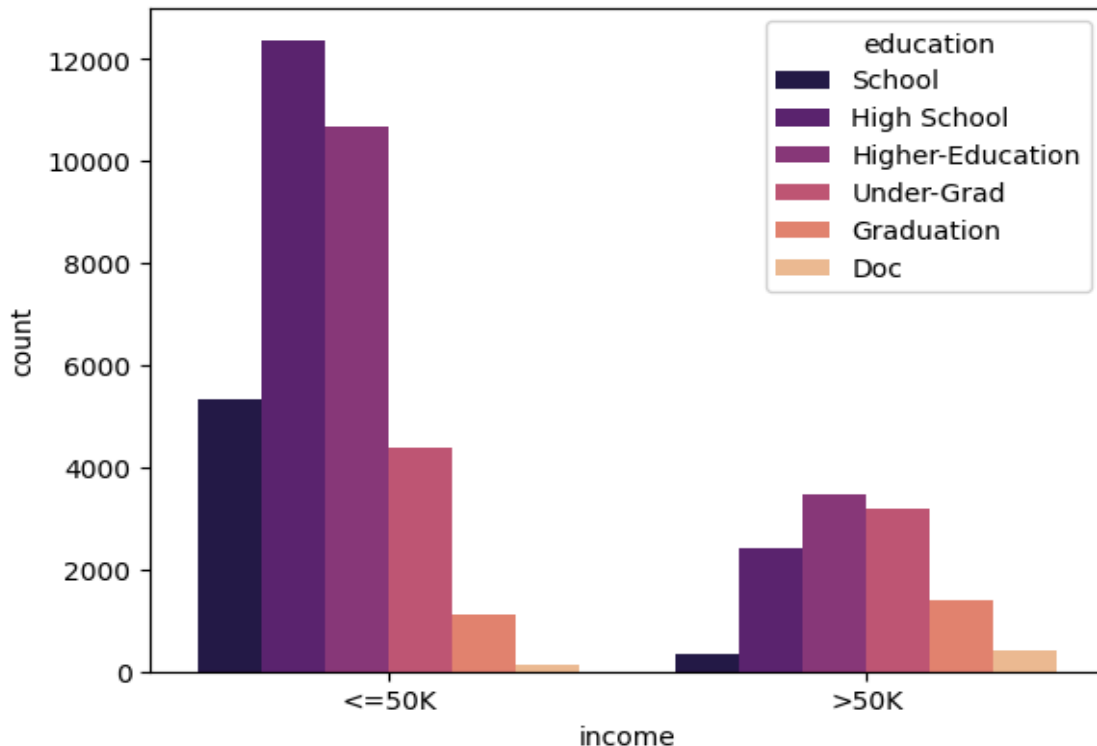
The graph displays the relationship between education level and income in the Adult Income dataset. Main points which can be observed from this graph are as follows -

1. Income Distribution by Education Level

- Individuals with lower education levels (School, High School) are more likely to have lesser income compared to those with higher education levels.

- The highest counts for income less than $50,000 per year are among individuals with High School and Higher-Education levels.

2. Higher Education and Higher Income

 - Individuals with higher education levels (Graduation, Doc) are more likely to have a higher income compared to those with lower education levels.

- The count of individuals with income more than $50,000 per year increases with increasing educational levels, indicating a positive correlation between higher education and higher income (hence confirming our earlier observation made using the heat map)

3. Income Disparity

 - There is a significant disparity in income for lower education levels, with more individuals earning `<=50K` than `>50K`.

 - For higher education levels, the distribution is more balanced, with a noticeable population earning more than $50,000 per year

In summary, the graph suggests that higher education levels are associated with a higher likelihood of earning more than $50,000, while lower education levels are predominantly associated with incomes of less than or equal to $50,000 per year.
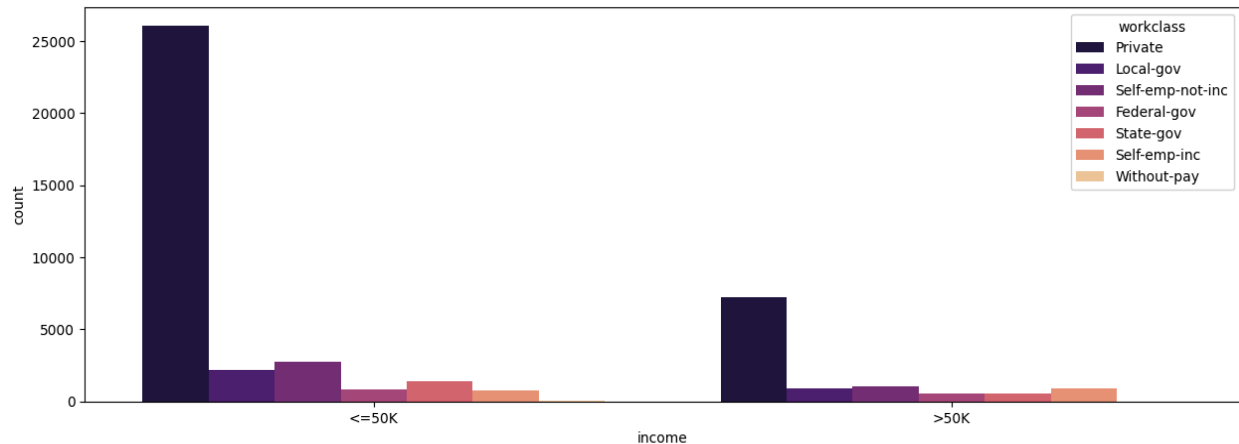
C. INCOME DISTRIBUTION BY WORKCLASS:
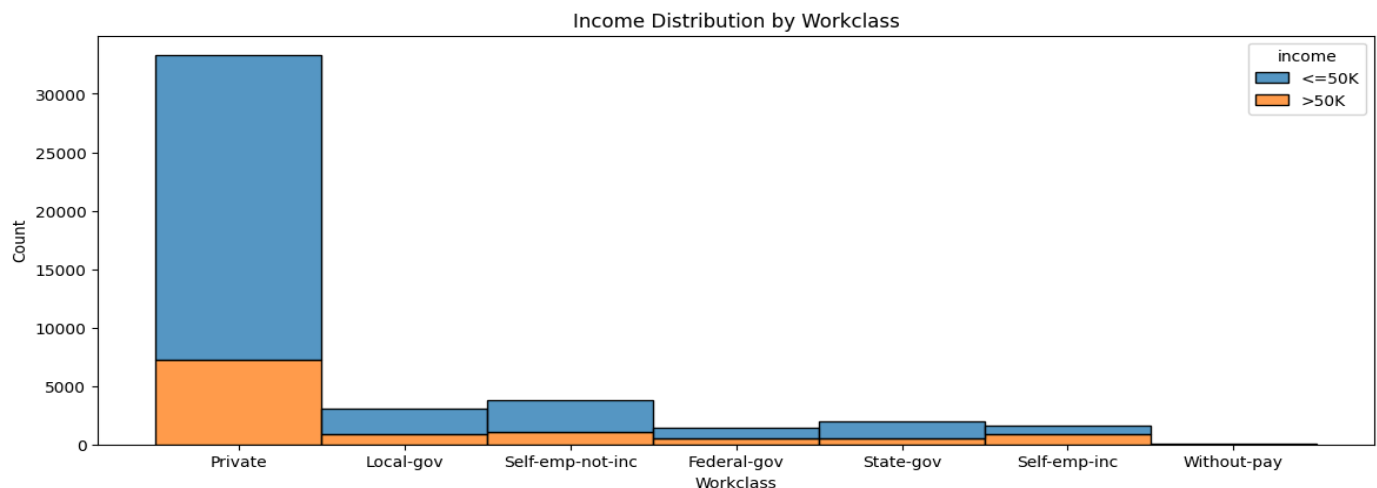
This is a stacked histogram that depicts the income distribution by workclass in the Adult Income Dataset. Key insights that can be drawn from this graph are:

- Most of the individuals in our dataset are employed in the private sector. This is evidenced by the tall bar for the Private workclass category. This shows the dominance of the Private Sector. Within the private sector, a significant proportion of individuals earn <=50K, but there is also a noticeable number of individuals earning >50K.

- Local, federal, and state government employees show a mix of income levels, but the counts are significantly lower than those in the private sector.

- Self-employed individuals (both incorporated and not incorporated) have a smaller overall representation. However, a substantial proportion of self-employed incorporated individuals earn >50K. Among all the various workclasses, self-employed incorporated individuals have a higher proportion of high-income earners compared to other workclasses.

- The without pay category has negligible representation and is almost invisible. The private sector also shows a considerable number of high-income earners, although it is outnumbered by those earning <=50K.

Similarly, the count plot for the same information provides a visual representation of the distribution of income across different workclasses by segregating them into 2 categories >50k and <=50K based on income.

The overall trend indicates that most individuals across different workclasses earn <=50K. The graph highlights the dominance of the private sector in terms of employment, with a significant proportion earning <=50K. Other workclasses have smaller representations, while certain workclasses, like self-employed incorporated, and private, to some extent, have a relatively higher proportion of higher income earners, i.e., >50k.

D. INCOME DISTRIBUTION BY MARITAL STATUS:
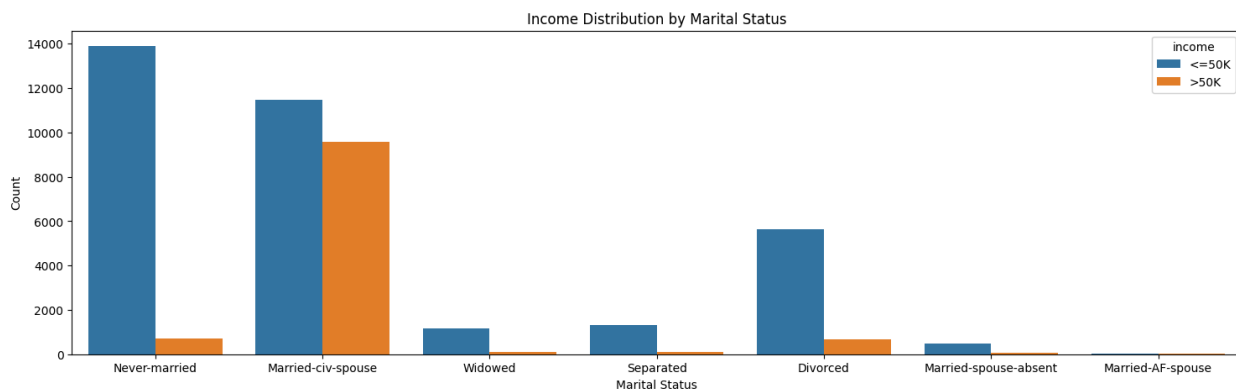
This count plot shows the distribution of income (`<=50K` and `>50K`) across different marital status categories. Here are the key insights that can be inferred from the graph across the various categories:

1. Never Married:

- The majority of individuals who have never married earn <=50K.

- Only a small proportion of never-married individuals earn >50K.

2. Married Civilian Spouse:

- A significant number of individuals who are married with a civilian spouse earn <=50K.

- However, a substantial proportion of individuals in this category earn `>50K`, indicating that being married to a civilian spouse is associated with higher income levels.

3. Widowed:

- The count of widowed individuals earning <=50K is relatively low.

- There are very few widowed individuals earning >50K.

4. Separated:

- Individuals who are separated predominantly earn <=50K.

- There is a very small proportion of separated individuals earning >50K.

5. Divorced:

- The majority of divorced individuals earn <=50K.

- A small proportion of divorced individuals earn >50K.

6. Married Spouse Absent:

  - Individuals with a married-spouse-absent status have a low count, with most earning <=50K and very few earning >50K.

7. Married AF spouse:

  - This category has a very low count overall, with individuals earning both `<=50K` and `>50K` being almost negligible.

It can be observed that 'Married Civilian Spouse' is the only category with a substantial proportion of individuals earning >50K, suggesting that being married is correlated with higher income. Married individuals, especially those married to civilian spouses, seem to have better financial stability, as indicated by the higher proportion. Single individuals (Never Married) predominantly earn <=50K, which indicates a correlation between marital status and income level. Divorced, separated, widowed, and married-spouse-absent categories have lower counts overall and predominantly feature individuals earning <=50K.

So from the plot, we can infer that marital status is a significant factor in income distribution. Being married to a civilian spouse is associated with higher incomes, while never-married, divorced, separated, widowed, and married-spouse-absent individuals are more likely to earn <=50K.

Also, from the plot, we can say that the data sample includes a diverse set of marital statuses, with a notable emphasis on individuals who are never married and those who are married to a civilian spouse. The high counts for Never Married and Married Civilian Spouse categories indicate these are common marital statuses in the population sampled.

E. DISTRIBUTION OF INCOME:
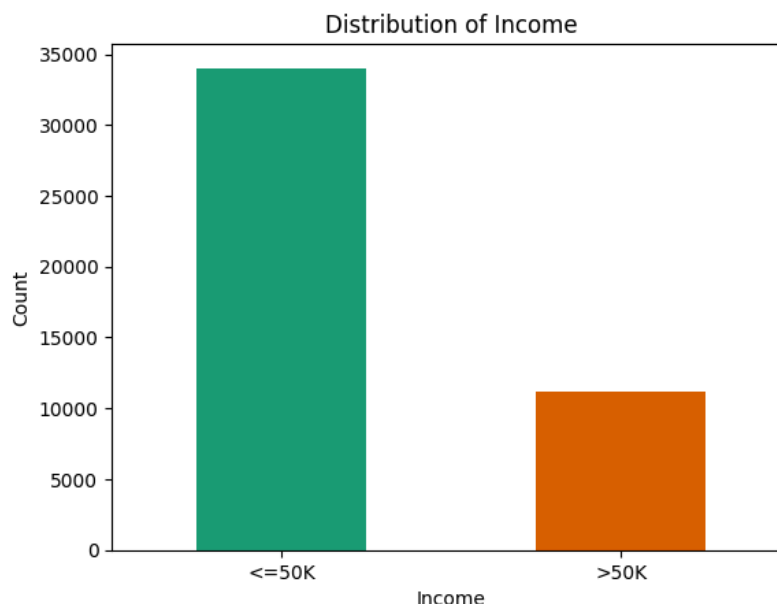
This graph shows the distribution of income across two categories: Income <= 50K (represented by the green bar) and Income >50K (represented by the orange bar). Key insights that can be inferred from this graph are:

- Uneven distribution: There's a significant disparity between the two income groups. The number of people earning $50K or less is much higher than those earning over $50K. This clearly shows that this is an imbalanced dataset, with the majority class being individuals earning $50,000 or less.

- Majority in lower income bracket: The green bar, representing those earning $50K or less, is approximately 3 times taller than the orange bar. This suggests that about 75% of the population in this dataset falls into the lower-income category.

It can be seen from the graph that the count for the lower income group is around 35,000, while the higher income group is about 11,000-12,000. The income data has been simplified into two categories <=50K and >50K, which provides a clear but broad overview of the income distribution.
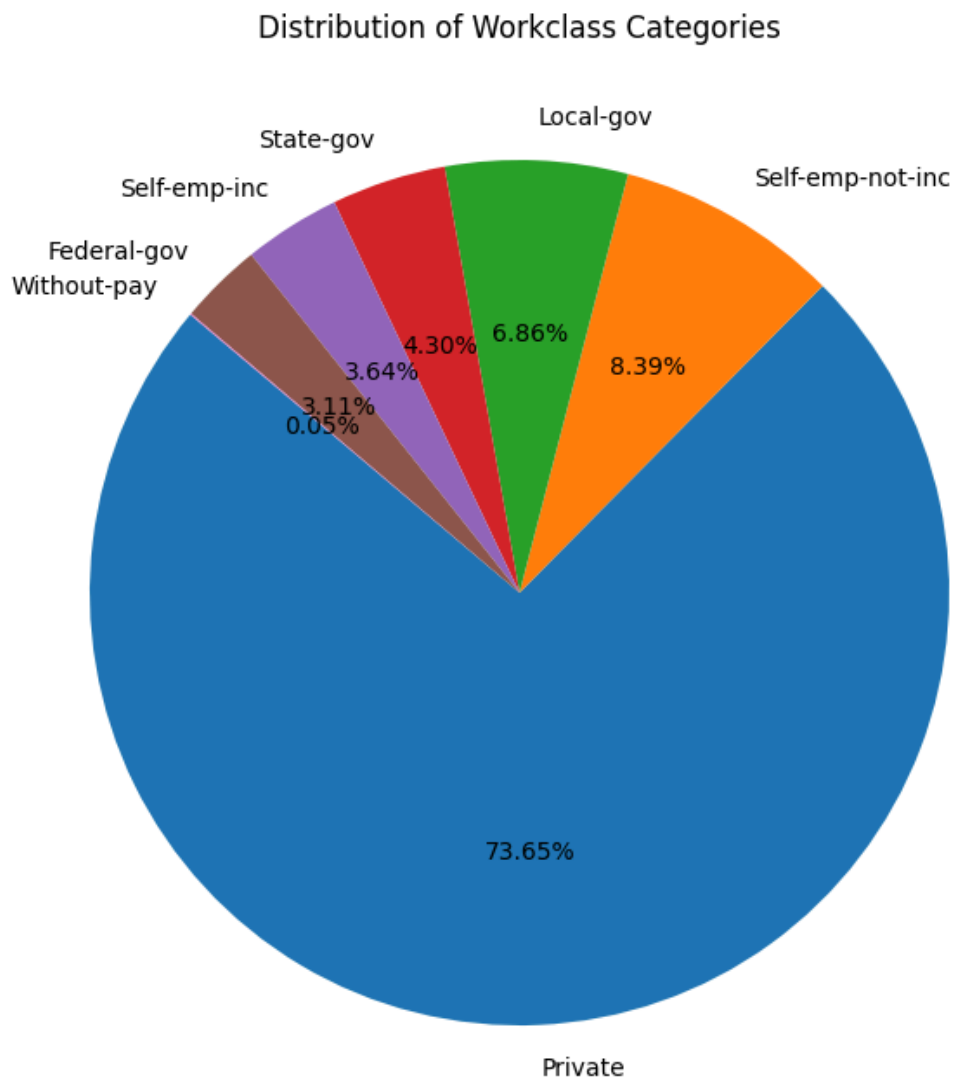
F. AGE DISTRIBUTION BY INCOME:
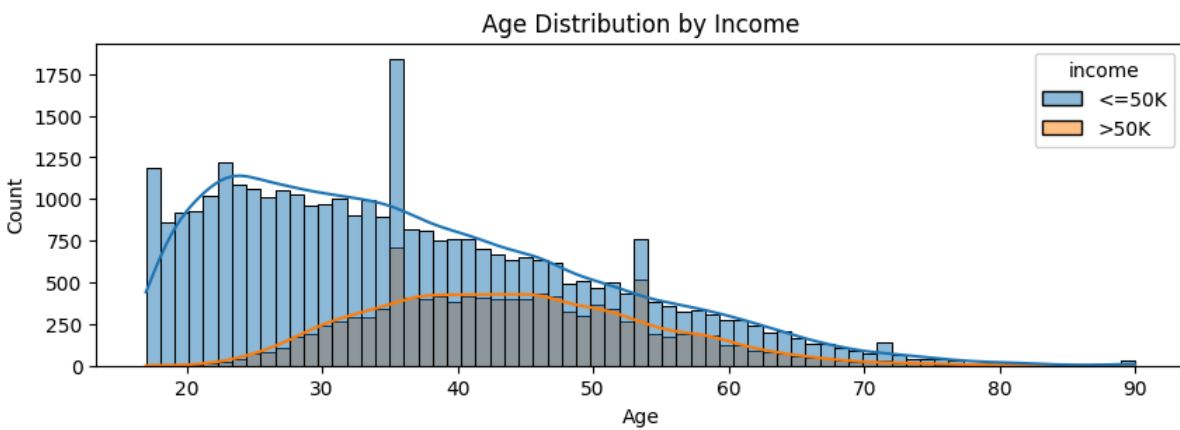
This graph provides a detailed view of the age distribution in relation to income levels for the adult income dataset. The dataset covers adults from roughly 18 to 90 years old. The graph maintains the two income categories from the previous chart: <=50K (blue) and >50K (orange). Here are the key insights:

- The proportion of higher income (>50K) individuals increases with age, peaking around 45-55 years. The age distribution follows a roughly normal curve with a peak in the 30s-40s range, which aligns with typical working-age populations.

- The 40-50 age range shows the highest concentration of individuals earning >50K, suggesting these are peak earning years.

- Younger adults (below 30) are predominantly in the lower income category. Also, there is a notable spike in the <=50K category for very young adults (around 20), likely representing entry-level workers or students.

- The proportion of higher earners gradually increases from the 20s through the 40s, illustrating career progression and income growth over time.

- Around age 60-65, there's a noticeable drop in both categories, more pronounced in the >50K group, likely indicating retirement. However, while the higher income group decreases sharply after 60, some individuals continue to earn >50K into their 70s and beyond, albeit in smaller numbers

- The Kernel Density Estimation (KDE) curves provide a smoothed view of the distribution, helping to identify overall trends beyond the bin-to-bin variations.

- Consistent with the previous graph, there are more individuals in the <=50K category across all age groups. This again confirms our earlier assumption of this dataset being an imbalanced one.

This visualization offers insights into the relationship between age and income, showing how earning potential typically evolves over a person's career. It highlights the challenges faced by younger workers, the peak earning years in middle age, and the economic impacts of retirement. These patterns could be valuable for various analyses, including career planning, economic policy, and targeted financial products or services.

G. DISTRIBUTION OF WORKCLASS CATEGORIES:

Distribution of Workclass Categories

This pie chart depicts the distribution of workclass categories in the adult income dataset. Here are some of the key insights that can be inferred from this graph:

1. Dominant private sector: The most striking feature is that the private sector accounts for 73.65% of the workforce in this dataset. This indicates a predominantly private sector-driven economy.

2. Government employment: Combined government employment (federal, state, and local) accounts for about 14.27% of the workforce, which is distributed as follows:

- Local government: 6.86%

- State government: 4.30%

- Federal government: 3.11%

3. Self-employment: There are two categories of self-employment:

 - Self-employed not incorporated: 8.39%

 - Self-employed incorporated: 3.64%

Together, they represent 12.03% of the workforce, indicating a significant portion of independent workers or small business owners.

4. Unpaid workers: There's a very small category (0.03%) labeled "Without-pay", likely representing volunteers, interns, or family workers in family-owned businesses.

6. Public vs. Private: The public sector (all government levels combined) is significantly smaller than the private sector. This indicates that the market is predominantly driven by the private sector.

7. Entrepreneurship: The presence of two self-employment categories suggests a nuanced view of entrepreneurship, distinguishing between incorporated and unincorporated businesses.

8. Economic structure: The distribution reflects the economic structure of the time when the data was collected, showing a strong private sector with a notable government presence and a significant self-employed segment.

This information is valuable for understanding the employment landscape represented in the dataset. It can be used to analyze income differences across sectors, study the impact of employment type on income levels, or investigate how different workclass categories correlate with other variables in the dataset, such as education, age, or income brackets.
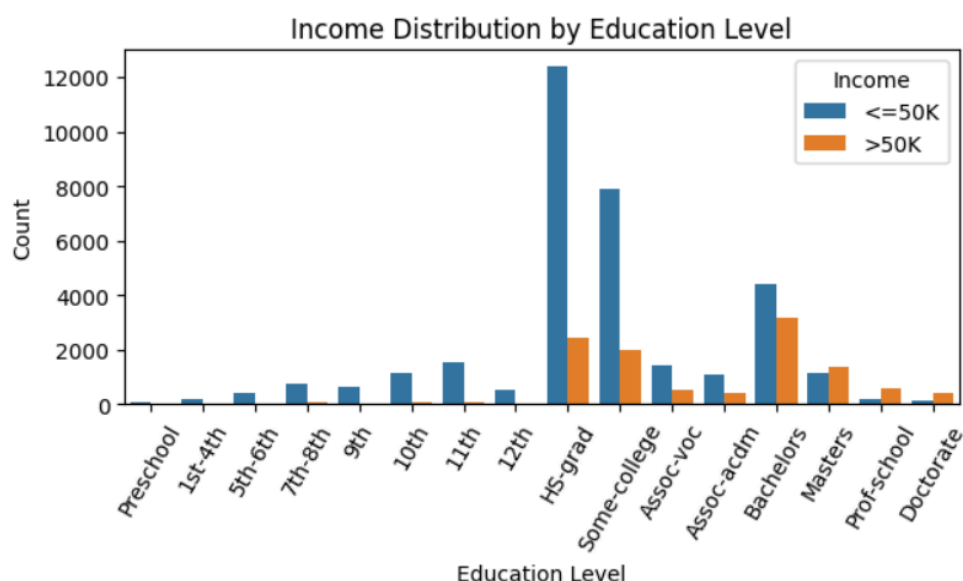
H. INCOME DISTRIBUTION BY GENDER:

This graph illustrates the income distribution by gender in the adult income dataset. Here are the key insights that can be gathered from this graph:

1. Gender representation: The dataset includes both male and female individuals, with males appearing to be more numerous overall.

2. Income categories: Consistent with previous graphs, income is divided into two categories: <=50K and >50K.

3. Gender income disparity: There's a clear disparity in income distribution between males and females:

- For males, the number earning <=50K is about twice that of those earning >50K.

- For females, the number earning <=50K is about five times that of those earning >50K.

4. Higher earners: Males are much more likely to be in the >50K category compared to females.

5. Lower income prevalence: For both genders, the majority fall into the <=50K category, but this is more pronounced for females.

6. Workforce composition: The graph suggests a larger male presence in the workforce, particularly in higher-paying positions.

This visualization highlights significant gender-based disparities in income distribution, reflecting broader societal and economic issues of the time. It underscores the importance of considering gender as a factor in economic and policy analyses based on this dataset. The binary gender categorization and broad income brackets limit the nuance of the analysis but still provide clear overall trends.

I. INCOME DISTRIBUTION BY EDUCATION:

This graph illustrates the income distribution across various education levels in the adult income dataset. Here are the key insights:

1. Education levels: The dataset includes a wide range of education levels, from preschool to doctorate degrees.

2. Income categories: Consistent with previous graphs, income is divided into two categories: <=50K (blue) and >50K (orange).

3. Correlation between education and income: There's a clear trend showing higher education levels generally correspond with a higher likelihood of earning >50K.

4. High school graduates: This is the largest group in the dataset, with a significant majority earning <=50K.

5. College impact: There's a noticeable increase in the proportion of >50K earners starting from the "Some-college" level.

6. Advanced degrees: Masters and Doctorate levels show a more balanced distribution between the two income categories, with a higher proportion in the >50K category compared to lower education levels.

7. Lower education levels: For education levels below high school, the vast majority fall into the <=50K category, with very few in the >50K category.

8. Bachelors degree: This level shows a significant jump in the proportion of >50K earners compared to lower education levels.

9. Associate degrees: Both academic and vocational associate degrees are represented, with academic associates showing a slightly higher proportion of >50K earners.

10. Outliers: There are some individuals with lower education levels (e.g., 9th grade, 11th grade) who still manage to earn >50K, though they are relatively few.

11. Dataset composition: The graph provides insight into the educational makeup of the dataset, which is skewed towards high school graduates and those with some college education.

12. Income ceiling effect: Even at the highest education levels, there's still a significant proportion earning <=50K, indicating that education alone doesn't guarantee higher income.

This visualization demonstrates the strong relationship between education and income potential in this dataset. It highlights the economic value of higher education, particularly college degrees and beyond. However, it also shows that income is not solely determined by education, as there's variation within each education level. This information would be crucial for analyzing factors influencing income and for building predictive models using this dataset.

J.INCOME DISTRIBUTION BY RACE

Income Distribution Based on Race
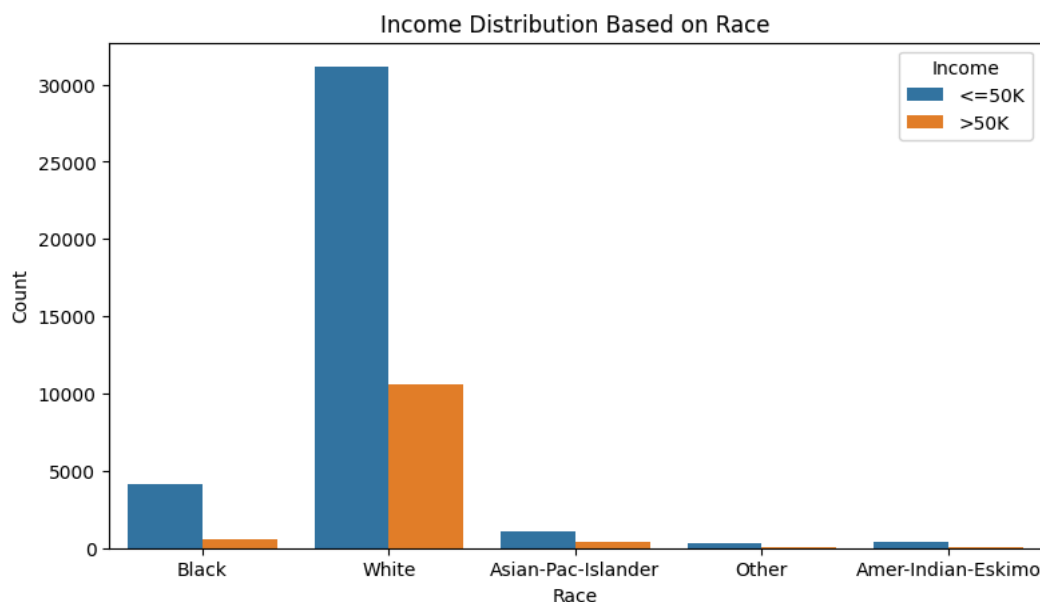
The graph shows income distribution across different racial groups, categorized into two income brackets: <=50K (less than or equal to $50,000) and >50K (greater than $50,000). Key insights that can be gathered from this graph are:

1. Income disparity: Across all racial groups, there are more people in the lower income bracket (<=50K) than in the higher income bracket (>50K).

2.Limited categories: The racial categories used (White, Black, Asian-Pac-Islander, Amer-Indian-Eskimo, Other) reflect the classification system of the time, which may not capture the full diversity of the population

-The White racial group has the largest number of individuals in both income brackets, with a higher proportion in the <=50K category. However, since the graph reflects the demographics of the US in the early 1990s(the time period during which the data was gathered), it explains the larger representation of the White population.

-The Black racial group shows a notable disparity, with significantly more individuals in the lower income bracket compared to the higher income bracket.

-The Asian-Pacific Islander group has a more balanced distribution between the two income brackets compared to other minority groups, though still more in the lower bracket.

-The "Other" and "American Indian-Eskimo" categories have the smallest populations, with very few individuals in the higher income bracket.
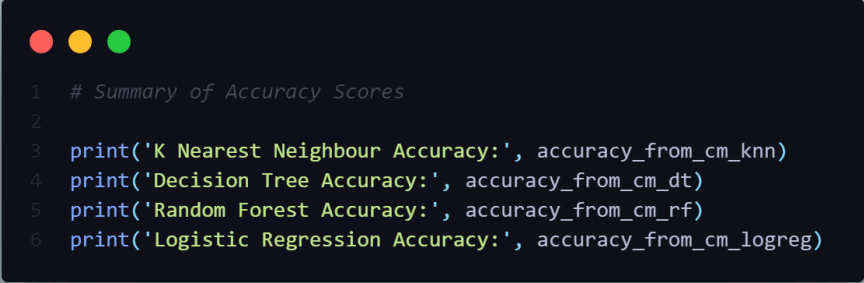
3. Economic inequality: The graph suggests economic disparities among racial groups, with minority groups generally having a smaller proportion of individuals in the higher income bracket. The disparity in income distribution across racial groups aligns with known socioeconomic inequalities in the US during that period when the data was collected.

While the graph shows racial and income intersections, it doesn't account for other factors in the dataset like education, occupation, or gender, which also play crucial roles in income determination. The insights highlight income disparities across racial groups, but it's crucial to consider additional factors and data for a comprehensive understanding of the socio-economic landscape.

Conclusion

In conclusion, our project "Knowledge Representation and Insight Generation from Structured Datasets" has been successful in developing an AI-based system that aims to extract valuable insights from the Adult Income Dataset. Our objective was to find helpful insights for decision-makers and predict income levels using demographic and socio-economic factors. For this task, logistic regression proved to be the most successful model, regularly outperforming other models such as random forests, decision trees, and K nearest neighbors (KNN). Unlike KNN, which relies on local similarity measures and can be sensitive to noise, Logistic Regression models the probability of an outcome based on a linear relationship between input features and the log-odds of the outcome. This approach performed well for us because it made it easy to understand how certain attributes such as age, occupation type, and level of education, influence income outcomes.

```
Logistic Regression Confusion Matrix:
[[6305  537]
 [ 972 1231]]
Logistic Regression Accuracy: 0.833167495854063
Logistic Regression Accuracy from Confusion Matrix: 0.833167495854063
```

```
1   # Summary of Accuracy Scores
2
3   print('K Nearest Neighbour Accuracy:', accuracy_from_cm_knn)
4   print('Decision Tree Accuracy:', accuracy_from_cm_dt)
5   print('Random Forest Accuracy:', accuracy_from_cm_rf)
6   print('Logistic Regression Accuracy:', accuracy_from_cm_logreg)
```

```
K Nearest Neighbour Accuracy: 0.8127142067440575
Decision Tree Accuracy: 0.770702045328911
Random Forest Accuracy: 0.824212271973466
Logistic Regression Accuracy: 0.833167495854063
```

For instance, our research showed a favorable correlation between income levels and specific occupations, and with higher levels of education. Clarifying these associations required the use of Logistic Regression, which can handle both continuous and categorical data and produce interpretable coefficients. Age also showed up as a crucial factor, demonstrating experience and career progression in determining salary levels. We examined our models' performance across various criteria and used confusion matrices to assess the accuracy of our models. In predicting income levels, Logistic Regression consistently outperformed K Nearest Neighbours (KNN), Decision Trees, and Random Forest using techniques like accuracy_score and manual calculations using confusion matrices. This demonstrated how well it predicted income levels based on socioeconomic and demographic variables.

Our Logistic Regression analysis yielded valuable insights that are essential for businesses and policymakers to take action. These results provide customized approaches to raise educational attainment, increase employment opportunities, and promote economic mobility. Organizations can help achieve more fair social results across a range of populations by tackling these variables.

In addition to model performance, our project highlighted the value of ongoing development. Adjusting model parameters for best performance, augmenting datasets to address imbalances and improve generalization, implementing ensemble techniques like Random Forests for improved robustness, and exploring additional features through advanced feature engineering techniques were some suggestions for improving model accuracy.

To further improve the accuracy and applicability of our models, we can also delve deeper into exploring these options:

- Parameter Tuning: Refine model parameters through techniques such as cross-validation to optimize regularization parameters or decision thresholds, ensuring the model better reflects real-world scenarios.
- Ensemble Techniques: Utilize ensemble techniques such as Random Forest, which combines predictions from diverse models to enhance accuracy and reliability while maintaining interpretability.
- Data Augmentation: Expand the dataset through techniques like synthetic data generation (For eg: SMOTE for imbalanced datasets) to improve model training and generalization.
- Feature Engineering: Investigate additional features or transformations that better capture intricate relationships, such as generating interaction terms between education and occupation to more accurately represent their combined effects.

Our AI-based solution can continue to offer trustworthy insights and assist well-informed decision-making across a variety of sectors by concentrating on these enhancements. By taking these actions, we hope to improve the tools that our models can use to use data to build successful organizations and have a beneficial social impact.

Based on the graphs generated, we can gather several key insights about the relationships based on the various factors present in our dataset:

1. Overall Income Distribution:

- The majority of the population earns ≤$50K, with a smaller portion earning >$50K.

2. Education and Income:

- There's a strong positive correlation between education level and income.

- Higher education (Bachelor's, Master's, Doctorate) significantly increases the likelihood of earning >$50K.

- Those with only school or high school education predominantly earn ≤$50K.

3. Age and Income:

- Income tends to increase with age, peaking around 40-50 years old, then declining.

- Younger individuals predominantly earn ≤$50K, reflecting new workers, interns, social workers etc belong predominantly to this age group.

4. Work Class and Income:

- The private sector employs the majority (73.65%), with most earning ≤$50K.

-Government jobs (federal, state, local) make up a smaller but significant portion of employment.

-Self-employment (both incorporated and not incorporated) accounts for about 12% of the workforce.

- Government jobs and self-employment,although smaller in comparison to the private sector, have higher proportions of >$50K earners.

5. Gender and Income:

- Males are more likely to earn >$50K compared to females.

- The gender disparity is more pronounced in the >$50K income bracket.

6. Hours Worked and Income:

- There's a positive correlation between hours worked per week and income i.e. higher the number of hours worked per week, higher will be the income of the individual(>50K).

7. Race and Income:

- Income distribution varies across different races, with some having higher proportions of >$50K earners.

8. Marital Status and Income:

- Married individuals, especially those with a civilian spouse, are more likely to be in the >$50K bracket.

- Never-married i.e. Single individuals are predominantly in the ≤$50K bracket.

9.Correlations (from heatmap):

-There's a strong positive correlation between age and income.

-Education level (educational-num) has a moderate positive correlation with income.

-Hours worked per week also show a positive correlation with income, though not as strong as education or age.

10.Racial Implications:

-All racial groups have more individuals in the lower income bracket (≤$50K) than in the higher bracket (>$50K). However the disparity is more pronounced for Black and American Indian-Eskimo groups.

-The White group has the largest number of individuals in both income brackets, reflecting the demographics of the US in the early 1990s when the data was collected. The racial categories used (White, Black, Asian-Pac-Islander, Amer-Indian-Eskimo, Other) reflect the classification system of the time, which may not capture the full diversity of the population.

-The Asian-Pacific Islander group shows a more balanced distribution between income brackets compared to other minority groups, while Black, American Indian-Eskimo, and "Other" categories have significantly fewer individuals in the higher income bracket.

-The data suggests economic disparities among racial groups, with minorities generally having a smaller proportion of individuals in the higher income bracket. These disparities align with known socioeconomic inequalities in the US during that period.

These insights highlight the complex interplay of demographic, educational, and occupational factors in determining income levels. Higher education, more work hours, certain work sectors, being middle-aged, and being married are associated with higher likelihood of earning over $50K annually. However, significant disparities exist based on gender and race, reflecting broader socioeconomic trends and potential systemic inequalities.

In conclusion, our work demonstrates the revolutionary potential of data-driven methodologies for producing useful insights from organised datasets. We open doors for innovation and fair growth by utilising advanced analytics and logistic regression, enabling organisations to make data-driven decisions that produce favourable results in various societal situations and industries.