Prediction of Salary Class of an Individual Using Logistic Regression

Machine Learning Deep Learning

Ruoming Jin

Vineeth Billakanti
811192961
Masters in Computer Science
Kent State University
vbillaka@kent.edu

Abstract:

Logistic regression is a statistical analysis approach that uses prior data set observations to predict a binary result, such as yes or no. By analyzing the relationship between one or more existing independent variables, a logistic regression model can predict a dependent data variable.

Problem:

The problem is to predict the salaried class of an individual is greater than $50,000 or less than $50,000 based on an individual's credentials like education level, age, gender, experience, occupation, etc. For example, the salary of an individual whose experience is above 15 years is most likely to be greater than $50,000. Prediction is not made by considering only one factor but all the factors that affect the income of an individual. I am going to explain how the problem is predicted by using Logistic Regression method.

Introduction:

A prediction is a guess about what will happen in the future. A forecast is based on information or experience in certain cases, but not always. Future occurrences are not always guaranteed, therefore specific data about the future is often hard to confirm. However, a forecast may be valuable in developing strategies for likely developments. In this study, the pay of an organization's employee is forecasted using prior compensation growth rates. The pay history of a person has been watched, and it may then be determined automatically based on that wage after a given length of time.

Now, the Logistic regression is a statistical analysis approach that uses prior data set observations to predict a binary result, such as yes or no. By analyzing the relationship between one or more existing independent variables, a logistic regression model can predict a dependent data variable.

Methodology:

Machine Learning (ML) is a branch of computer science that allows computers to understand data in a similar manner that humans do. To put it another way, machine learning (ML) is a sort of artificial intelligence that uses an algorithm or technique to extract patterns from raw data. The goal of machine learning is to enable computers to learn from their own experiences without the need for explicit programming or human involvement.

Logistic regression is a classification model rather than a regression model. For binary and linear classification issues, logistic regression is a simple and more efficient technique. It is a classification model that is simple to implement and delivers excellent results with linearly separable classes. It is a widely used categorization method in the business. Like the Adaline and perceptron, the logistic regression model is a statistical approach for binary classification that may be adapted to multiclass classification.

About Dataset:

The dataset is taken from Kaggle. The US Adult Census dataset is a repository of 32,561 entries with 15 variables. The dataset contains information about age, work class, education, occupation, relationship, country, income. Let’s look at dataset details.

	age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native.country	income
1	90	?	77053	HS-grad	9	Widowed	?	Not-in-family	White	Female	0	4356	40	United-States	<=50K
2	82	Private	132870	HS-grad	9	Widowed	Exec-managerial	Not-in-family	White	Female	0	4356	18	United-States	<=50K
3	66	?	186061	Some-college	10	Widowed	?	Unmarried	Black	Female	0	4356	40	United-States	<=50K
4	54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct	Unmarried	White	Female	0	3900	40	United-States	<=50K
5	41	Private	264663	Some-college	10	Separated	Prof-specialty	Own-child	White	Female	0	3900	40	United-States	<=50K
6	34	Private	216864	HS-grad	9	Divorced	Other-service	Unmarried	White	Female	0	3770	45	United-States	<=50K
7	38	Private	150601	10th	6	Separated	Adm-clerical	Unmarried	White	Male	0	3770	40	United-States	<=50K
8	74	State-gov	88638	Doctorate	16	Never-married	Prof-specialty	Other-relative	White	Female	0	3683	20	United-States	>50K
9	68	Federal-gov	422013	HS-grad	9	Divorced	Prof-specialty	Not-in-family	White	Female	0	3683	40	United-States	<=50K
10	41	Private	70037	Some-college	10	Never-married	Craft-repair	Unmarried	White	Male	0	3004	60	?	>50K
11	45	Private	172274	Doctorate	16	Divorced	Prof-specialty	Unmarried	Black	Female	0	3004	35	United-States	>50K
12	38	Self-emp-not-inc	164526	Prof-school	15	Never-married	Prof-specialty	Not-in-family	White	Male	0	2824	45	United-States	>50K
13	52	Private	129177	Bachelors	13	Widowed	Other-service	Not-in-family	White	Female	0	2824	20	United-States	>50K
14	32	Private	136204	Masters	14	Separated	Exec-managerial	Not-in-family	White	Male	0	2824	55	United-States	>50K
15	51	?	172175	Doctorate	16	Never-married	?	Not-in-family	White	Male	0	2824	40	United-States	>50K
16	46	Private	45363	Prof-school	15	Divorced	Prof-specialty	Not-in-family	White	Male	0	2824	40	United-States	>50K
17	45	Private	172822	11th	7	Divorced	Transport-moving	Not-in-family	White	Male	0	2824	76	United-States	>50K
18	57	Private	317847	Masters	14	Divorced	Exec-managerial	Not-in-family	White	Male	0	2824	50	United-States	>50K
19	22	Private	119592	Assoc-acdm	12	Never-married	Handlers-cleaners	Not-in-family	Black	Male	0	2824	40	?	>50K

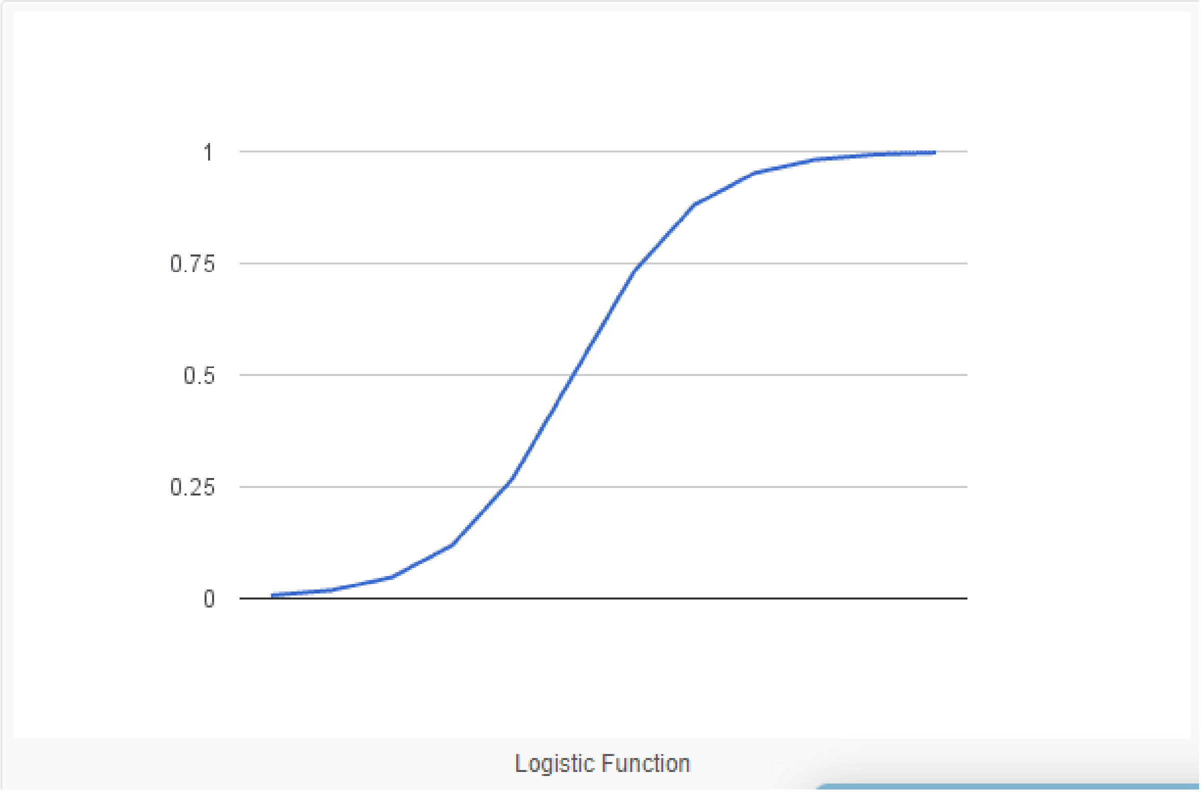
Here is the link of Dataset: Adult Census Income | Kaggle

Approach:

We may utilize the Naive Bayes approach, Linear Regression, and Logistic Regression to solve this prediction issue. Because the prediction variable (salary class) is dependent on a number of factors (both category and numerical), the Naive Bayes approach and Logistic Regression are superior than Linear Regression for this issue. Because our dataset contains more categories than numerical factors, Logistic Regression is the ideal approach for this issue.

The Supervised Learning approach includes one of the most prominent Machine Learning algorithms: logistic regression. It's used to predict a categorical dependent variable from a group of independent factors. As a result, the result must be a discrete or categorical value. It may be Yes or No, 0 or 1, true or false, and so on, but instead of providing precise values like 0 and 1, it delivers probabilistic values that are somewhere between 0 and 1. Furthermore, unlike linear regression, logistic regression does not need a linear connection between the input and output variables. This is due to the odds ratio being transformed via a nonlinear log transformation.

$$Logistic\ function = \frac{1}{1+e^{-x}}$$



Experimentation:

This prediction has been implemented through the following approaches:

- numpy:- numpy ('Numerical Python') is a Python library. It is a scientific computing core package that includes a strong n-dimensional array object as well as integration tools for C, C++, and other languages. It also has applications in linear algebra, random number generation, and other fields.
- matplotlib.pyplot:- matplotlib.pyplot is a set of command-style functions that allow matplotlib to behave similarly to MATLAB. Each pyplot function modifies a figure in some way, such as creating a figure, a plotting area in figure, Charting certain lines in a plotting area, decorating the plot with labels and so on.
- pandas:- For data processing and analysis, pandas is a software library created in Python. Its data structures and methods for manipulating numerical tables and time series are particularly useful. It's open-source software licensed under BSD license, which has three clauses.
- read_csv:- Python is a great language for doing data analysis, primarily because of the fantastic ecosystem of data-centric python packages. pandas is one of those packages and make importing and analyzing data much easier.
- Predict:-predict the values where find at the time period that how much there in that graph.

To begin, create an excel file dataset, which will then be opened in Jupyter notebook using ANACONDA Navigator. From there, we can use Anaconda Navigator to read the data set. First, in the Jupyter notebook, we'll import two variables : numpy and pandas. These functions are used as follows:

- NumPy is used for antialiasing a dynamic array or large set.

- matplotlib.pyplot is used for making graph
- pandas are mainly used like database where we store the dataset from the excel file. Through the data set we are forming graph and predicting.

Then import the dataset on the pandas through the help of "<any variable>. read_csv" and then showing that field that are importing from the excel file. Then the salary is predicted by using the logistic regression method.

Code and Output:

Importing the Libraries and reading the .CSV file(Data Set):

```
In [1]: #Importing the libraries
        import numpy as np
        import pandas as pd# data processing for input, for instance to import CSV file
```

```
In [2]: adult_data = pd.read_csv("C:/Users/Vineeth Billakanti/Desktop/ML & DL/Final project/adult.csv") #reading the CSV file
        adult_data.head() #displayng few results of the CSV file
```

Out[2]:

	age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native.cc
0	90	?	77053	HS-grad	9	Widowed	?	Not-in-family	White	Female	0	4356	40	United-
1	82	Private	132870	HS-grad	9	Widowed	Exec-managerial	Not-in-family	White	Female	0	4356	18	United-
2	66	?	186061	Some-college	10	Widowed	?	Unmarried	Black	Female	0	4356	40	United-
3	54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct	Unmarried	White	Female	0	3900	40	United-
4	41	Private	264663	Some-college	10	Separated	Prof-specialty	Own-child	White	Female	0	3900	40	United-

```
In [3]: adult_data.describe()
```

Out[3]:

	age	fnlwgt	education.num	capital.gain	capital.loss	hours.per.week
count	32561.000000	3.256100e+04	32561.000000	32561.000000	32561.000000	32561.000000
mean	38.581647	1.897784e+05	10.080679	1077.648844	87.303830	40.437456
std	13.640433	1.055500e+05	2.572720	7385.292085	402.960219	12.347429
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	1.000000
25%	28.000000	1.178270e+05	9.000000	0.000000	0.000000	40.000000
50%	37.000000	1.783560e+05	10.000000	0.000000	0.000000	40.000000
75%	48.000000	2.370510e+05	12.000000	0.000000	0.000000	45.000000
max	90.000000	1.484705e+06	16.000000	99999.000000	4356.000000	99.000000

```
In [4]: adult_data.income.value_counts() #Count of how many people have income <=50K and >50K
```

```
Out[4]: <=50K    24720
        >50K      7841
        Name: income, dtype: int64
```

```
In [5]: adult_data['income'] = adult_data['income'].map({'<=50K': 0, '>50K': 1})
        adult_data.head()
```

Out[5]:

	age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native.cour
0	90	?	77053	HS-grad	9	Widowed	?	Not-in-family	White	Female	0	4356	40	United-Sta
1	82	Private	132870	HS-grad	9	Widowed	Exec-managerial	Not-in-family	White	Female	0	4356	18	United-Sta
2	66	?	186061	Some-college	10	Widowed	?	Unmarried	Black	Female	0	4356	40	United-Sta
3	54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct	Unmarried	White	Female	0	3900	40	United-Sta
4	41	Private	264663	Some-college	10	Separated	Prof-specialty	Own-child	White	Female	0	3900	40	United-Sta

```
In [6]: print(adult_data.info()) #Displaying all the data information
        print(adult_data.shape) #Displays the shape of the arrays
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32561 entries, 0 to 32560
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   age             32561 non-null  int64 
 1   workclass       32561 non-null  object
 2   fnlwgt          32561 non-null  int64 
 3   education       32561 non-null  object
 4   education.num   32561 non-null  int64 
 5   marital.status  32561 non-null  object
 6   occupation      32561 non-null  object
 7   relationship    32561 non-null  object
 8   race            32561 non-null  object
 9   sex             32561 non-null  object
 10  capital.gain    32561 non-null  int64 
 11  capital.loss    32561 non-null  int64 
 12  hours.per.week  32561 non-null  int64 
 13  native.country  32561 non-null  object
 14  income          32561 non-null  int64 
dtypes: int64(7), object(8)
memory usage: 3.7+ MB
None
(32561, 15)
```

Data Cleaning:

```
In [7]: adult_data.isnull().sum() #Checking whether do we have null values or not in the given data set

Out[7]: 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
```

```
In [8]: adult_data.isin(['?']).sum() #We can see '?'in the dataset

Out[8]: age                  0
        workclass         1836
        fnlwgt               0
        education            0
        education.num        0
        marital.status       0
        occupation        1843
        relationship         0
        race                 0
```

```
In [9]: adult_data.replace('?', np.nan , inplace=True) #Replacing the '?' with null value.
```

```
In [10]: adult_data.isnull().sum()

Out[10]: 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     583
         income               0
         dtype: int64
```

```
In [11]: adult_data['workclass'] = adult_data['workclass'].fillna(adult_data['workclass'].mode()[0])
         adult_data['occupation'] = adult_data['occupation'].fillna(adult_data['occupation'].mode()[0])
         adult_data['native.country'] = adult_data['native.country'].fillna(adult_data['native.country'].mode()[0])
```

```
In [12]: adult_data.dtypes
```

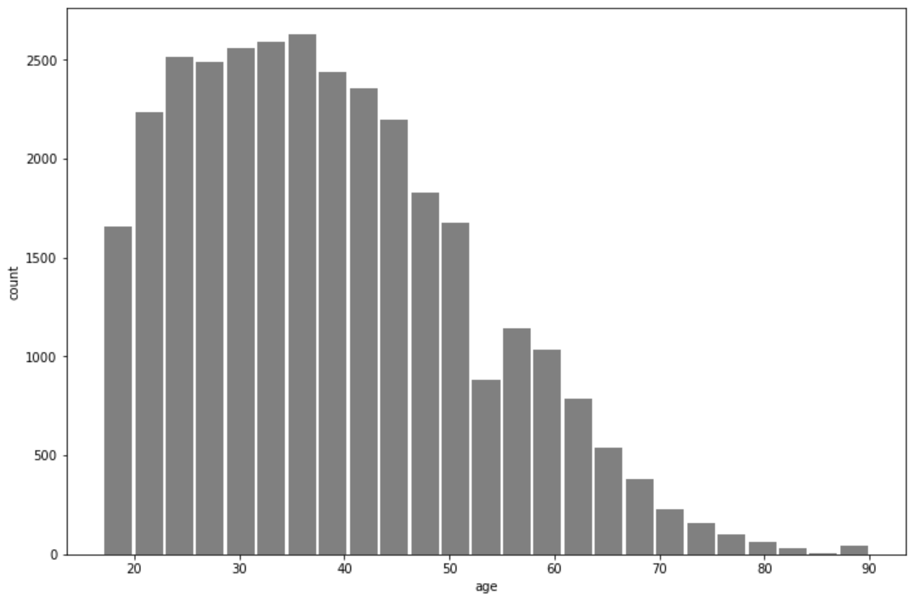
```
Out[12]: age                int64
         workclass         object
         fnlwgt             int64
         education         object
         education.num      int64
         marital.status    object
         occupation        object
         relationship      object
         race              object
         sex               object
         capital.gain       int64
         capital.loss       int64
         hours.per.week     int64
         native.country    object
         income             int64
         dtype: object
```

```
In [13]: from sklearn.preprocessing import LabelEncoder
         for col in adult_data.columns:
             if adult_data[col].dtypes == 'object':
                 le = LabelEncoder()
                 adult_data[col] = le.fit_transform(adult_data[col].astype(str))

         adult_data.dtypes #We can see some of the variables are not integers, we employ a label encoder to transform them to integers so
```

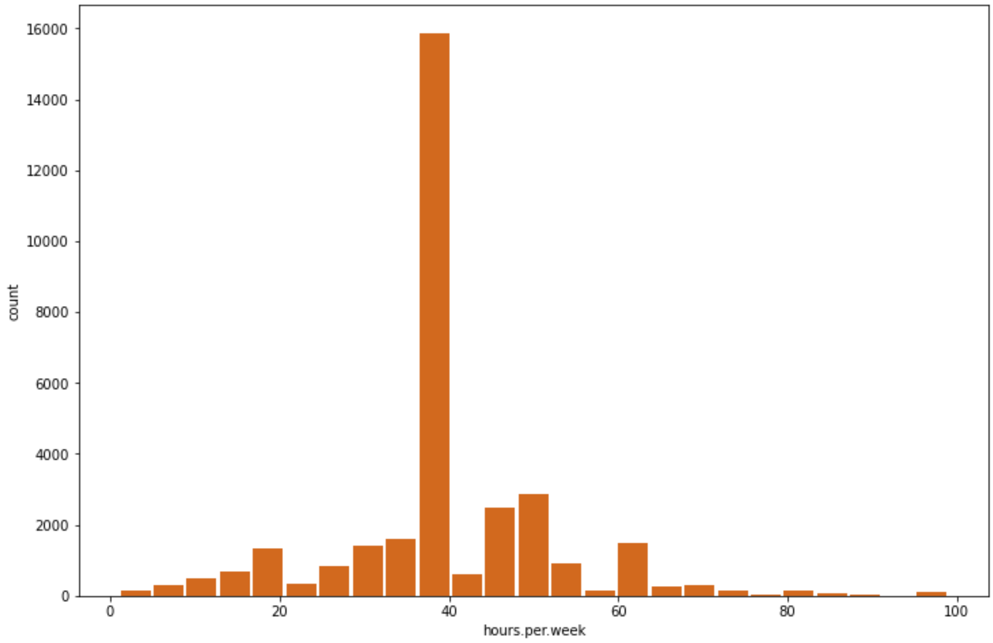
```
Out[13]: age               int64
         workclass         int32
         fnlwgt            int64
         education         int32
         education.num     int64
         marital.status    int32
         occupation        int32
         relationship      int32
         race              int32
         sex               int32
         capital.gain      int64
         capital.loss      int64
         hours.per.week    int64
         native.country    int32
         income            int64
         dtype: object
```

Data Exploring:

```
In [15]: import matplotlib.pyplot as plt  #Data Exploration
         adult_data.hist(column='age', bins=25, grid=False, figsize=(12,8), color='#808080', zorder=2, rwidth=0.9)
         ##plt.bar(adult_data,adult_data['income'])
         plt.xlabel('age')
         plt.ylabel('count')
         plt.title('')
         plt.show()
```



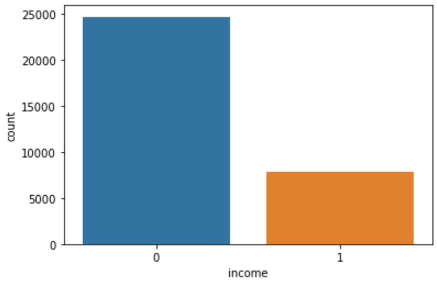
In [16]:
```
import matplotlib.pyplot as plt
adult_data.hist(column='hours.per.week', bins=25, grid=False, figsize=(12,8), color='#D2691E', zorder=2, rwidth=0.9)
##plt.bar(adult_data,adult_data['income'])
plt.xlabel('hours.per.week')
plt.ylabel('count')
plt.title('')
plt.show()
```



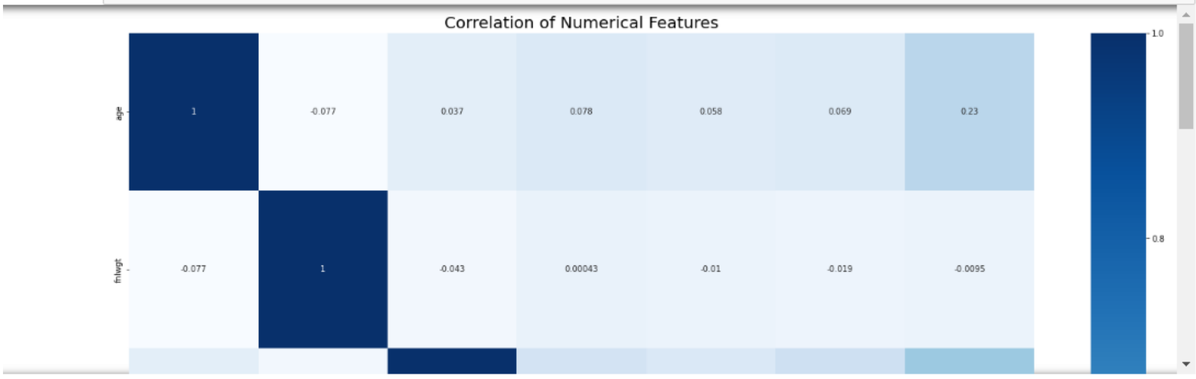
In [17]:
```
num_features = ['age', 'fnlwgt', 'education.num', 'capital.gain', 'capital.loss', 'hours.per.week', 'income']

cat_features = ['workclass', 'education', 'marital.status', 'occupation', 'relationship', 'race', 'sex', 'native.country']
```

In [18]:
```
import seaborn as sns  #Income counts
sns.countplot(adult_data['income'])
plt.show()
```

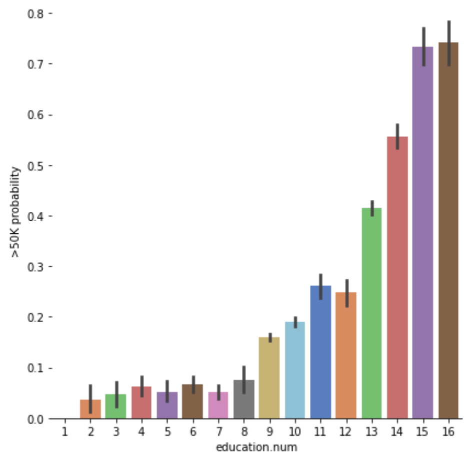
```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword a
rg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit ke
yword will result in an error or misinterpretation.
  warnings.warn(
```



```
In [19]: #Plotting for the numerical features
fig, ax = plt.subplots(figsize=(25, 25))
p = sns.heatmap(adult_data[num_features].corr(), annot=True, cmap="Blues")
plt.title("Correlation of Numerical Features", fontsize=20)
plt.show()
```




```
In [20]: #Education Number vs Income
g = sns.catplot(x="education.num",y="income",data=adult_data,kind="bar",height = 6,palette = "muted")
g.despine(left=True)
g = g.set_ylabels(">50K probability")
```



Data partitioning:

Here we split the data into 80% Training data and 20% testing data.

```
In [21]: from sklearn.model_selection import train_test_split #Partitioning the data
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2, random_state = 42)
```

Model Building:

```
In [22]: from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

```
In [23]: from sklearn.linear_model import LogisticRegression
lr = LogisticRegression(random_state = 0)
lr.fit(X_train, y_train)
lr_pred = lr.predict(X_test)
```

```
In [24]: from sklearn.metrics import confusion_matrix, accuracy_score
ac_lr = round(accuracy_score(y_test, lr_pred)*100,2)

cm_lr = confusion_matrix(y_test, lr_pred)
```

Confusion Matrix:

In [25]:
```
results = pd.DataFrame({
    'Model': ['Logistic Regression'],
    'Accuracy': [ac_lr],
    'Confusion_Matrix':[cm_lr]})
result_df = results.sort_values(by='Accuracy', ascending=False)
result_df = result_df.set_index('Model')
result_df.head(5) #confusion matrix
```

Out[25]:

	Accuracy	Confusion_Matrix
Model		
Logistic Regression	82.39	[[4686, 290], [857, 680]]

Results:

As this is a prediction problem Accuracy is the measure to find how our model is predicting the salaried class of an individual. The accuracy of the model is shown below.

Accuracy = (Correct predictions) / (Total predictions)
= (5627+722) / (5627+140+1190+722)
= 6349 / 7679
=0.8239

The accuracy of our model is 82.39% to predict the salary class of an individual by considering age, education, hours per week, occupation, and sex as categorical variables.

Conclusions:

From the results, we can conclude that the salary of an individual whose income is greater than $50,000 is male, whose education level is higher than a master's degree and who works more than 40 hours per week, and whose occupation is in the private sector.

References:

https://www.keboola.com/blog/logistic-regression-machine-learning
https://journalofbigdata.springeropen.com/articles/10.1186/s40537-022-00559-6