# Lab 1

**Code - Housing and Diabetes Dataset**

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import OrdinalEncoder, OneHotEncoder
from sklearn.preprocessing import StandardScaler, MinMaxScaler
from scipy import stats

from google.colab import drive
drive.mount('/content/drive')

import pandas as pd

data = pd.read_csv('/content/drive/MyDrive/ML_LAB/Lab-1/housing.csv')

print("Information of all columns:")
print(data.info())

print("\nStatistical Information of all numerical columns:")
print(data.describe())

print("\nCount of unique labels for 'Ocean Proximity' column:")
print(data['ocean_proximity'].value_counts())

print("\nColumns with missing values:")
print(data.isnull().sum()[data.isnull().sum() > 0])

df1 = pd.read_csv("/content/drive/MyDrive/ML_LAB/Lab-1/diabetes.csv")

print(df1.info())
print(df1.head())

# Check for missing values
# Check for missing values in each column
missing_values = df1.isnull().sum()

# Display columns with missing values
```

```
print(missing_values[missing_values > 0])

# Impute missing values for numerical columns with mean
num_columns = df1.select_dtypes(include=['float64', 'int64']).columns
imputer = SimpleImputer(strategy='mean')
df1[num_columns] = imputer.fit_transform(df1[num_columns])

# Impute missing values for categorical columns with the mode
cat_columns = df1.select_dtypes(include=['object']).columns
imputer_cat = SimpleImputer(strategy='most_frequent')
df1[cat_columns] = imputer_cat.fit_transform(df1[cat_columns])

#Handling Categorical Attributes
#Using Ordinal Encoding for gender COlumn and One-Hot Encoding for City Column
df_copy = df1.copy()
df_copy['Gender'] = df_copy['Gender'].str.upper()
# Remove leading/trailing spaces from the 'CLASS' column
df1['CLASS'] = df1['CLASS'].str.strip()

# Initialize OrdinalEncoder
ordinal_encoder = OrdinalEncoder(categories=[["M", "F"]])
# Fit and transform the data
df_copy["Gender_Encoded"] = ordinal_encoder.fit_transform(df_copy[["Gender"]])

# Initialize OneHotEncoder
onehot_encoder = OneHotEncoder()

# Fit and transform the "City" column
encoded_data = onehot_encoder.fit_transform(df1[["CLASS"]])

# Convert the sparse matrix to a dense array
encoded_array = encoded_data.toarray()

# Convert to DataFrame for better visualization
encoded_df = pd.DataFrame(encoded_array,
columns=onehot_encoder.get_feature_names_out(["CLASS"]))
df_encoded = pd.concat([df_copy, encoded_df], axis=1)

df_encoded.drop("Gender", axis=1, inplace=True)
df_encoded.drop("CLASS", axis=1, inplace=True)
```

```
print(df_encoded.head())

# Check the column names after OneHotEncoding
encoded_columns = onehot_encoder.get_feature_names_out(["CLASS"])
print(encoded_columns)

normalizer = MinMaxScaler()
df_encoded[['Urea']] = normalizer.fit_transform(df_encoded[['Urea']])
df_encoded.head()

scaler = StandardScaler()
df_encoded[['AGE']] = scaler.fit_transform(df_encoded[['AGE']])
df_encoded.head()

#Removing Outliers
# Outlier Detection and Treatment using IQR
#Pros: Simple and effective for mild outliers.
#Cons: May overly reduce variation if there are many extreme outliers.
df_encoded_copy1=df_encoded
df_encoded_copy2=df_encoded
df_encoded_copy3=df_encoded

Q1 = df_encoded_copy1['Urea'].quantile(0.25)
Q3 = df_encoded_copy1['Urea'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
df_encoded_copy1['Urea'] = np.where(df_encoded_copy1['Urea'] > upper_bound,
upper_bound,
                    np.where(df_encoded_copy1['Urea'] < lower_bound, lower_bound,
df_encoded_copy1['Urea']))

print(df_encoded_copy1.head())

#Removing Outliers
# Z-score method
#Pros: Good for normally distributed data.
#Cons: Not suitable for non-normal data; may miss outliers in skewed distributions.
```

```
df_encoded_copy2['Urea_zscore'] = stats.zscore(df_encoded_copy2['Urea'])
df_encoded_copy2['Urea'] = np.where(df_encoded_copy2['Urea_zscore'].abs() > 3,
np.nan, df_encoded_copy2['Urea'])  # Replace outliers with NaN
print(df_encoded_copy2.head())

#Removing Outliers
# Median replacement for outliers
#Pros: Keeps distribution shape intact, useful when capping isn’t feasible.
#Cons: May distort data if outliers represent real phenomena.
df_encoded_copy3['Urea_zscore'] = stats.zscore(df_encoded_copy3['Urea'])
median_Urea = df_encoded_copy3['Urea'].median()
df_encoded_copy3['Urea'] = np.where(df_encoded_copy3['Urea_zscore'].abs() > 3,
median_Urea, df_encoded_copy3['Urea'])
print(df_encoded_copy3.head())
```

## Output

```
Information of all columns:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   longitude           20640 non-null  float64
 1   latitude            20640 non-null  float64
 2   housing_median_age  20640 non-null  float64
 3   total_rooms         20640 non-null  float64
 4   total_bedrooms      20433 non-null  float64
 5   population          20640 non-null  float64
 6   households          20640 non-null  float64
 7   median_income       20640 non-null  float64
 8   median_house_value  20640 non-null  float64
 9   ocean_proximity     20640 non-null  object
dtypes: float64(9), object(1)
memory usage: 1.6+ MB
None

Statistical Information of all numerical columns:
          longitude      latitude  housing_median_age   total_rooms  \
count  20640.000000  20640.000000        20640.000000  20640.000000
mean    -119.569704     35.631861           28.639486   2635.763081
std        2.003532      2.135952           12.585558   2181.615252
min     -124.350000     32.540000            1.000000      2.000000
25%     -121.800000     33.930000           18.000000   1447.750000
50%     -118.490000     34.260000           29.000000   2127.000000
75%     -118.010000     37.710000           37.000000   3148.000000
max     -114.310000     41.950000           52.000000  39320.000000

       total_bedrooms    population    households  median_income  \
count    20433.000000  20640.000000  20640.000000   20640.000000
mean       537.870553   1425.476744    499.539680       3.870671
std        421.385070   1132.462122    382.329753       1.899822
min          1.000000      3.000000      1.000000       0.499900
25%        296.000000    787.000000    280.000000       2.563400
50%        435.000000   1166.000000    409.000000       3.534800
75%        647.000000   1725.000000    605.000000       4.743250
max       6445.000000  35682.000000   6082.000000      15.000100

       median_house_value
count        20640.000000
mean        206855.816909
std         115395.615874
min          14999.000000
25%         119600.000000
50%         179700.000000
75%         264725.000000
max         500001.000000

Count of unique labels for 'Ocean Proximity' column:
ocean_proximity
<1H OCEAN     9136
INLAND        6551
NEAR OCEAN    2658
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 14 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   ID         1000 non-null   int64  
 1   No_Pation  1000 non-null   int64  
 2   Gender     1000 non-null   object 
 3   AGE        1000 non-null   int64  
 4   Urea       1000 non-null   float64
 5   Cr         1000 non-null   int64  
 6   HbA1c      1000 non-null   float64
 7   Chol       1000 non-null   float64
 8   TG         1000 non-null   float64
 9   HDL        1000 non-null   float64
 10  LDL        1000 non-null   float64
 11  VLDL       1000 non-null   float64
 12  BMI        1000 non-null   float64
 13  CLASS      1000 non-null   object 
dtypes: float64(8), int64(4), object(2)
memory usage: 109.5+ KB
None
    ID  No_Pation Gender  AGE  Urea  Cr  HbA1c  Chol   TG  HDL  LDL  VLDL  \
0  502      17975      F   50   4.7  46    4.9   4.2  0.9  2.4  1.4   0.5   
1  735      34221      M   26   4.5  62    4.9   3.7  1.4  1.1  2.1   0.6   
2  420      47975      F   50   4.7  46    4.9   4.2  0.9  2.4  1.4   0.5   
3  680      87656      F   50   4.7  46    4.9   4.2  0.9  2.4  1.4   0.5   
4  504      34223      M   33   7.1  46    4.9   4.9  1.0  0.8  2.0   0.4   

    BMI CLASS  
0  24.0     N  
1  23.0     N  
2  24.0     N  
3  24.0     N  
4  21.0     N  
```

```
    ID  No_Pation  AGE  Urea  Cr  HbA1c  Chol   TG  HDL  LDL  VLDL   BMI  \
0  502      17975   50   4.7  46    4.9   4.2  0.9  2.4  1.4   0.5  24.0   
1  735      34221   26   4.5  62    4.9   3.7  1.4  1.1  2.1   0.6  23.0   
2  420      47975   50   4.7  46    4.9   4.2  0.9  2.4  1.4   0.5  24.0   
3  680      87656   50   4.7  46    4.9   4.2  0.9  2.4  1.4   0.5  24.0   
4  504      34223   33   7.1  46    4.9   4.9  1.0  0.8  2.0   0.4  21.0   

   Gender_Encoded  CLASS_N  CLASS_P  CLASS_Y  
0             1.0      1.0      0.0      0.0  
1             0.0      1.0      0.0      0.0  
2             1.0      1.0      0.0      0.0  
3             1.0      1.0      0.0      0.0  
4             0.0      1.0      0.0      0.0  
```

| | ID | No_Pation | AGE | Urea | Cr | HbA1c | Chol | TG | HDL | LDL | VLDL | BMI | Gender_Encoded | CLASS_N | CLASS_P | CLASS_Y |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 502 | 17975 | 50 | 0.109375 | 46 | 4.9 | 4.2 | 0.9 | 2.4 | 1.4 | 0.5 | 24.0 | 1.0 | 1.0 | 0.0 | 0.0 |
| 1 | 735 | 34221 | 26 | 0.104167 | 62 | 4.9 | 3.7 | 1.4 | 1.1 | 2.1 | 0.6 | 23.0 | 0.0 | 1.0 | 0.0 | 0.0 |
| 2 | 420 | 47975 | 50 | 0.109375 | 46 | 4.9 | 4.2 | 0.9 | 2.4 | 1.4 | 0.5 | 24.0 | 1.0 | 1.0 | 0.0 | 0.0 |
| 3 | 680 | 87656 | 50 | 0.109375 | 46 | 4.9 | 4.2 | 0.9 | 2.4 | 1.4 | 0.5 | 24.0 | 1.0 | 1.0 | 0.0 | 0.0 |
| 4 | 504 | 34223 | 33 | 0.171875 | 46 | 4.9 | 4.9 | 1.0 | 0.8 | 2.0 | 0.4 | 21.0 | 0.0 | 1.0 | 0.0 | 0.0 |

```python
scaler = StandardScaler()
df_encoded[['AGE']] = scaler.fit_transform(df_encoded[['AGE']])
df_encoded.head()
```

| | ID | No_Pation | AGE | Urea | Cr | HbA1c | Chol | TG | HDL | LDL | VLDL | BMI | Gender_Encoded | CLASS_N | CLASS_P | CLASS_Y |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 502 | 17975 | -0.401144 | 0.109375 | 46 | 4.9 | 4.2 | 0.9 | 2.4 | 1.4 | 0.5 | 24.0 | 1.0 | 1.0 | 0.0 | 0.0 |
| 1 | 735 | 34221 | -3.130017 | 0.104167 | 62 | 4.9 | 3.7 | 1.4 | 1.1 | 2.1 | 0.6 | 23.0 | 0.0 | 1.0 | 0.0 | 0.0 |
| 2 | 420 | 47975 | -0.401144 | 0.109375 | 46 | 4.9 | 4.2 | 0.9 | 2.4 | 1.4 | 0.5 | 24.0 | 1.0 | 1.0 | 0.0 | 0.0 |
| 3 | 680 | 87656 | -0.401144 | 0.109375 | 46 | 4.9 | 4.2 | 0.9 | 2.4 | 1.4 | 0.5 | 24.0 | 1.0 | 1.0 | 0.0 | 0.0 |
| 4 | 504 | 34223 | -2.334096 | 0.171875 | 46 | 4.9 | 4.9 | 1.0 | 0.8 | 2.0 | 0.4 | 21.0 | 0.0 | 1.0 | 0.0 | 0.0 |

```
    ID  No_Pation       AGE      Urea  Cr  HbA1c  Chol   TG  HDL  LDL  VLDL  \
0  502      17975 -0.401144  0.109375  46    4.9   4.2  0.9  2.4  1.4   0.5
1  735      34221 -3.130017  0.104167  62    4.9   3.7  1.4  1.1  2.1   0.6
2  420      47975 -0.401144  0.109375  46    4.9   4.2  0.9  2.4  1.4   0.5
3  680      87656 -0.401144  0.109375  46    4.9   4.2  0.9  2.4  1.4   0.5
4  504      34223 -2.334096  0.171875  46    4.9   4.9  1.0  0.8  2.0   0.4

    BMI  Gender_Encoded  CLASS_N  CLASS_P  CLASS_Y
0  24.0             1.0      1.0      0.0      0.0
1  23.0             0.0      1.0      0.0      0.0
2  24.0             1.0      1.0      0.0      0.0
3  24.0             1.0      1.0      0.0      0.0
4  21.0             0.0      1.0      0.0      0.0
```

```
    ID  No_Pation       AGE      Urea  Cr  HbA1c  Chol   TG  HDL  LDL  VLDL  \
0  502      17975 -0.401144  0.109375  46    4.9   4.2  0.9  2.4  1.4   0.5
1  735      34221 -3.130017  0.104167  62    4.9   3.7  1.4  1.1  2.1   0.6
2  420      47975 -0.401144  0.109375  46    4.9   4.2  0.9  2.4  1.4   0.5
3  680      87656 -0.401144  0.109375  46    4.9   4.2  0.9  2.4  1.4   0.5
4  504      34223 -2.334096  0.171875  46    4.9   4.9  1.0  0.8  2.0   0.4

    BMI  Gender_Encoded  CLASS_N  CLASS_P  CLASS_Y  Urea_zscore
0  24.0             1.0      1.0      0.0      0.0    -0.074031
1  23.0             0.0      1.0      0.0      0.0    -0.190760
2  24.0             1.0      1.0      0.0      0.0    -0.074031
3  24.0             1.0      1.0      0.0      0.0    -0.074031
4  21.0             0.0      1.0      0.0      0.0     1.326714
```

```
    ID  No_Pation       AGE      Urea  Cr  HbA1c  Chol   TG  HDL  LDL  VLDL  \
0  502      17975 -0.401144  0.109375  46    4.9   4.2  0.9  2.4  1.4   0.5
1  735      34221 -3.130017  0.104167  62    4.9   3.7  1.4  1.1  2.1   0.6
2  420      47975 -0.401144  0.109375  46    4.9   4.2  0.9  2.4  1.4   0.5
3  680      87656 -0.401144  0.109375  46    4.9   4.2  0.9  2.4  1.4   0.5
4  504      34223 -2.334096  0.171875  46    4.9   4.9  1.0  0.8  2.0   0.4

    BMI  Gender_Encoded  CLASS_N  CLASS_P  CLASS_Y  Urea_zscore
0  24.0             1.0      1.0      0.0      0.0    -0.074031
1  23.0             0.0      1.0      0.0      0.0    -0.190760
2  24.0             1.0      1.0      0.0      0.0    -0.074031
3  24.0             1.0      1.0      0.0      0.0    -0.074031
4  21.0             0.0      1.0      0.0      0.0     1.326714
```

**Code - Adult Income Dataset**

```
df1 = pd.read_csv("/content/drive/MyDrive/ML_LAB/Lab-1/adult.csv")

print(df1.info())
print(df1.head())

df1['workclass'].value_counts()
df1['occupation'].value_counts()

# Check for missing values
df1.replace('?', np.nan, inplace=True)

# Check for missing values in each column
missing_values = df1.isnull().sum()

# Display columns with missing values
print(missing_values[missing_values > 0])

# Impute missing values for numerical columns with mean
num_columns = df1.select_dtypes(include=['float64', 'int64']).columns
imputer = SimpleImputer(strategy='mean')
df1[num_columns] = imputer.fit_transform(df1[num_columns])

# Impute missing values for categorical columns with the mode
cat_columns = df1.select_dtypes(include=['object']).columns
imputer_cat = SimpleImputer(strategy='most_frequent')
df1[cat_columns] = imputer_cat.fit_transform(df1[cat_columns])
print("\nMissing values in each column:")
print(df1.isnull().sum())

#Handling Categorical Attributes
```

```
#Using Ordinal Encoding for gender COlumn and One-Hot Encoding for City Column
df_copy = df1.copy()
df_copy['gender'] = df_copy['gender'].str.upper()
# Remove leading/trailing spaces from the 'income' column
df1['income'] = df1['income'].str.strip()

# Initialize OrdinalEncoder
ordinal_encoder = OrdinalEncoder(categories=[["MALE", "FEMALE"]])
# Fit and transform the data
df_copy["Gender_Encoded"] = ordinal_encoder.fit_transform(df_copy[["gender"]])

# Initialize OneHotEncoder
onehot_encoder = OneHotEncoder()

# Fit and transform the "City" column
encoded_data = onehot_encoder.fit_transform(df1[["income"]])

# Convert the sparse matrix to a dense array
encoded_array = encoded_data.toarray()

# Convert to DataFrame for better visualization
encoded_df = pd.DataFrame(encoded_array,
columns=onehot_encoder.get_feature_names_out(["income"]))
df_encoded = pd.concat([df_copy, encoded_df], axis=1)

df_encoded.drop("gender", axis=1, inplace=True)
df_encoded.drop("income", axis=1, inplace=True)

print(df_encoded.head())

normalizer = MinMaxScaler()
df_encoded[['fnlwgt']] = normalizer.fit_transform(df_encoded[['fnlwgt']])
df_encoded.head()

scaler = StandardScaler()
df_encoded[['age']] = scaler.fit_transform(df_encoded[['age']])
df_encoded.head()

#Removing Outliers
# Outlier Detection and Treatment using IQR
```

```
#Pros: Simple and effective for mild outliers.
#Cons: May overly reduce variation if there are many extreme outliers.
df_encoded_copy1=df_encoded
df_encoded_copy2=df_encoded
df_encoded_copy3=df_encoded

Q1 = df_encoded_copy1['hours-per-week'].quantile(0.25)
Q3 = df_encoded_copy1['hours-per-week'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
df_encoded_copy1['hours-per-week'] = np.where(df_encoded_copy1['hours-per-week']
> upper_bound, upper_bound,
                    np.where(df_encoded_copy1['hours-per-week'] < lower_bound,
lower_bound, df_encoded_copy1['hours-per-week']))

print(df_encoded_copy1.head())

#Removing Outliers
# Z-score method
#Pros: Good for normally distributed data.
#Cons: Not suitable for non-normal data; may miss outliers in skewed distributions.

df_encoded_copy2['hours-per-week_zscore'] =
stats.zscore(df_encoded_copy2['hours-per-week'])
df_encoded_copy2['hours-per-week'] =
np.where(df_encoded_copy2['hours-per-week_zscore'].abs() > 3, np.nan,
df_encoded_copy2['hours-per-week'])  # Replace outliers with NaN
print(df_encoded_copy2.head())

#Removing Outliers
# Median replacement for outliers
#Pros: Keeps distribution shape intact, useful when capping isn’t feasible.
#Cons: May distort data if outliers represent real phenomena.
df_encoded_copy3['hours-per-week_zscore'] =
stats.zscore(df_encoded_copy3['hours-per-week'])
median_hoursperweek = df_encoded_copy3['hours-per-week'].median()
df_encoded_copy3['hours-per-week'] =
np.where(df_encoded_copy3['hours-per-week_zscore'].abs() > 3,
median_hoursperweek, df_encoded_copy3['hours-per-week'])
```

print(df_encoded_copy3.head())

**Output**

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48842 entries, 0 to 48841
Data columns (total 15 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   age              48842 non-null  int64
 1   workclass        48842 non-null  object
 2   fnlwgt           48842 non-null  int64
 3   education        48842 non-null  object
 4   educational-num  48842 non-null  int64
 5   marital-status   48842 non-null  object
 6   occupation       48842 non-null  object
 7   relationship     48842 non-null  object
 8   race             48842 non-null  object
 9   gender           48842 non-null  object
 10  capital-gain     48842 non-null  int64
 11  capital-loss     48842 non-null  int64
 12  hours-per-week   48842 non-null  int64
 13  native-country   48842 non-null  object
 14  income           48842 non-null  object
dtypes: int64(6), object(9)
memory usage: 5.6+ MB
None
   age  workclass  fnlwgt     education  educational-num      marital-status  \
0   25    Private  226802          11th                7       Never-married
1   38    Private   89814       HS-grad                9  Married-civ-spouse
2   28  Local-gov  336951    Assoc-acdm               12  Married-civ-spouse
3   44    Private  160323  Some-college               10  Married-civ-spouse
4   18          ?  103497  Some-college               10       Never-married

          occupation relationship   race  gender  capital-gain  capital-loss  \
0  Machine-op-inspct    Own-child  Black    Male             0             0
1    Farming-fishing      Husband  White    Male             0             0
2    Protective-serv      Husband  White    Male             0             0
3  Machine-op-inspct      Husband  Black    Male          7688             0
4                  ?    Own-child  White  Female             0             0

   hours-per-week native-country income
0              40  United-States  <=50K
1              50  United-States  <=50K
2              40  United-States   >50K
3              40  United-States   >50K
4              30  United-States  <=50K
```

| occupation | count |
|---|---|
| Prof-specialty | 6172 |
| Craft-repair | 6112 |
| Exec-managerial | 6086 |
| Adm-clerical | 5611 |
| Sales | 5504 |
| Other-service | 4923 |
| Machine-op-inspct | 3022 |
| ? | 2809 |
| Transport-moving | 2355 |
| Handlers-cleaners | 2072 |
| Farming-fishing | 1490 |
| Tech-support | 1446 |
| Protective-serv | 983 |
| Priv-house-serv | 242 |
| Armed-Forces | 15 |

dtype: int64

```
workclass         2799
occupation        2809
native-country     857
dtype: int64
```

```
Missing values in each column:
age                0
workclass          0
fnlwgt             0
education          0
educational-num    0
marital-status     0
occupation         0
relationship       0
race               0
gender             0
capital-gain       0
capital-loss       0
hours-per-week     0
native-country     0
income             0
dtype: int64
```

```
    age  workclass    fnlwgt     education  educational-num  \
0  25.0    Private  226802.0          11th              7.0
1  38.0    Private   89814.0       HS-grad              9.0
2  28.0  Local-gov  336951.0    Assoc-acdm             12.0
3  44.0    Private  160323.0  Some-college             10.0
4  18.0    Private  103497.0  Some-college             10.0

       marital-status         occupation relationship   race  capital-gain  \
0       Never-married  Machine-op-inspct    Own-child  Black           0.0
1  Married-civ-spouse    Farming-fishing      Husband  White           0.0
2  Married-civ-spouse    Protective-serv      Husband  White           0.0
3  Married-civ-spouse  Machine-op-inspct      Husband  Black        7688.0
4       Never-married     Prof-specialty    Own-child  White           0.0

   capital-loss  hours-per-week native-country  Gender_Encoded  income_<=50K  \
0           0.0            40.0  United-States             0.0           1.0
1           0.0            50.0  United-States             0.0           1.0
2           0.0            40.0  United-States             0.0           0.0
3           0.0            40.0  United-States             0.0           0.0
4           0.0            30.0  United-States             1.0           1.0

   income_>50K
0          0.0
1          0.0
2          1.0
3          1.0
4          0.0
```

Samraat Dabolay 1BM22CS236

| | age | workclass | fnlwgt | education | educational-num | marital-status | occupation | relationship | race | capital-gain | capital-loss | hours-per-week | native-country | Gender_Encoded | income_<=50K | income_>50K |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | -0.995129 | Private | 0.145129 | 11th | 7.0 | Never-married | Machine-op-inspct | Own-child | Black | 0.0 | 0.0 | 40.0 | United-States | 0.0 | 1.0 | 0.0 |
| 1 | -0.046942 | Private | 0.052451 | HS-grad | 9.0 | Married-civ-spouse | Farming-fishing | Husband | White | 0.0 | 0.0 | 50.0 | United-States | 0.0 | 1.0 | 0.0 |
| 2 | -0.776316 | Local-gov | 0.219649 | Assoc-acdm | 12.0 | Married-civ-spouse | Protective-serv | Husband | White | 0.0 | 0.0 | 40.0 | United-States | 0.0 | 0.0 | 1.0 |
| 3 | 0.390683 | Private | 0.100153 | Some-college | 10.0 | Married-civ-spouse | Machine-op-inspct | Husband | Black | 7688.0 | 0.0 | 40.0 | United-States | 0.0 | 0.0 | 1.0 |
| 4 | -1.505691 | Private | 0.061708 | Some-college | 10.0 | Never-married | Prof-specialty | Own-child | White | 0.0 | 0.0 | 30.0 | United-States | 1.0 | 1.0 | 0.0 |

```
        age  workclass    fnlwgt     education  educational-num  \
0 -0.995129    Private  0.145129          11th              7.0
1 -0.046942    Private  0.052451       HS-grad              9.0
2 -0.776316  Local-gov  0.219649    Assoc-acdm             12.0
3  0.390683    Private  0.100153  Some-college             10.0
4 -1.505691    Private  0.061708  Some-college             10.0

       marital-status         occupation relationship   race  capital-gain  \
0       Never-married  Machine-op-inspct    Own-child  Black           0.0
1  Married-civ-spouse    Farming-fishing      Husband  White           0.0
2  Married-civ-spouse    Protective-serv      Husband  White           0.0
3  Married-civ-spouse  Machine-op-inspct      Husband  Black        7688.0
4       Never-married     Prof-specialty    Own-child  White           0.0

   capital-loss  hours-per-week native-country  Gender_Encoded  income_<=50K  \
0           0.0            40.0  United-States             0.0           1.0
1           0.0            50.0  United-States             0.0           1.0
2           0.0            40.0  United-States             0.0           0.0
3           0.0            40.0  United-States             0.0           0.0
4           0.0            32.5  United-States             1.0           1.0

   income_>50K
0          0.0
1          0.0
2          1.0
3          1.0
4          0.0
        age  workclass    fnlwgt     education  educational-num  \
0 -0.995129    Private  0.145129          11th              7.0
1 -0.046942    Private  0.052451       HS-grad              9.0
2 -0.776316  Local-gov  0.219649    Assoc-acdm             12.0
3  0.390683    Private  0.100153  Some-college             10.0
4 -1.505691    Private  0.061708  Some-college             10.0

       marital-status         occupation relationship   race  capital-gain  \
0       Never-married  Machine-op-inspct    Own-child  Black           0.0
1  Married-civ-spouse    Farming-fishing      Husband  White           0.0
2  Married-civ-spouse    Protective-serv      Husband  White           0.0
3  Married-civ-spouse  Machine-op-inspct      Husband  Black        7688.0
4       Never-married     Prof-specialty    Own-child  White           0.0
```

```
   capital-loss  hours-per-week native-country  Gender_Encoded  income_<=50K  \
0           0.0            40.0  United-States             0.0           1.0
1           0.0            50.0  United-States             0.0           1.0
2           0.0            40.0  United-States             0.0           0.0
3           0.0            40.0  United-States             0.0           0.0
4           0.0            32.5  United-States             1.0           1.0

   income_>50K  hours-per-week_zscore
0          0.0              -0.192863
1          0.0               1.424021
2          1.0              -0.192863
3          1.0              -0.192863
4          0.0              -1.405526
        age  workclass    fnlwgt     education  educational-num  \
0 -0.995129    Private  0.145129          11th              7.0
1 -0.046942    Private  0.052451       HS-grad              9.0
2 -0.776316  Local-gov  0.219649    Assoc-acdm             12.0
3  0.390683    Private  0.100153  Some-college             10.0
4 -1.505691    Private  0.061708  Some-college             10.0

       marital-status         occupation relationship   race  capital-gain  \
0       Never-married  Machine-op-inspct    Own-child  Black           0.0
1  Married-civ-spouse    Farming-fishing      Husband  White           0.0
2  Married-civ-spouse    Protective-serv      Husband  White           0.0
3  Married-civ-spouse  Machine-op-inspct      Husband  Black        7688.0
4       Never-married     Prof-specialty    Own-child  White           0.0

   capital-loss  hours-per-week native-country  Gender_Encoded  income_<=50K  \
0           0.0            40.0  United-States             0.0           1.0
1           0.0            50.0  United-States             0.0           1.0
2           0.0            40.0  United-States             0.0           0.0
3           0.0            40.0  United-States             0.0           0.0
4           0.0            32.5  United-States             1.0           1.0

   income_>50K  hours-per-week_zscore
0          0.0              -0.192863
1          0.0               1.424021
2          1.0              -0.192863
3          1.0              -0.192863
4          0.0              -1.405526
```