dataAnalysis

April 2, 2024

[ ]:
```
%pip install pandas
```

```
Requirement already satisfied: pandas in
c:\users\utente\appdata\local\programs\python\python310\lib\site-packages
(2.1.1)
Requirement already satisfied: numpy>=1.22.4 in
c:\users\utente\appdata\local\programs\python\python310\lib\site-packages (from
pandas) (1.24.3)
Requirement already satisfied: python-dateutil>=2.8.2 in
c:\users\utente\appdata\local\programs\python\python310\lib\site-packages (from
pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in
c:\users\utente\appdata\local\programs\python\python310\lib\site-packages (from
pandas) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in
c:\users\utente\appdata\local\programs\python\python310\lib\site-packages (from
pandas) (2023.3)
Requirement already satisfied: six>=1.5 in
c:\users\utente\appdata\local\programs\python\python310\lib\site-packages (from
python-dateutil>=2.8.2->pandas) (1.16.0)
Note: you may need to restart the kernel to use updated packages.

WARNING: Ignoring invalid distribution -yping-extensions
(c:\users\utente\appdata\local\programs\python\python310\lib\site-packages)
WARNING: Ignoring invalid distribution -yping-extensions
(c:\users\utente\appdata\local\programs\python\python310\lib\site-packages)

[notice] A new release of pip is available: 23.3.2 -> 24.0
[notice] To update, run: python.exe -m pip install --upgrade pip
```

[ ]:
```
import pandas as pd

df = pd.read_csv("onlinefoods.csv")
df
```

[ ]:

	Age	Gender	Marital Status	Occupation	Monthly Income	\
0	20	Female	Single	Student	No Income	
1	24	Female	Single	Student	Below Rs.10000	
2	22	Male	Single	Student	Below Rs.10000	

```
3     22  Female         Single    Student       No Income
4     22    Male         Single    Student  Below Rs.10000
..   …     …             …        …              …
383   23  Female         Single    Student       No Income
384   23  Female         Single    Student       No Income
385   22  Female         Single    Student       No Income
386   23    Male         Single    Student  Below Rs.10000
387   23    Male         Single    Student       No Income

    Educational Qualifications  Family size  latitude  longitude  Pin code  \
0                Post Graduate            4   12.9766    77.5993    560001
1                     Graduate            3   12.9770    77.5773    560009
2                Post Graduate            3   12.9551    77.6593    560017
3                     Graduate            6   12.9473    77.5616    560019
4                Post Graduate            4   12.9850    77.5533    560010
..                         …          …       …         …        …
383              Post Graduate            2   12.9766    77.5993    560001
384              Post Graduate            4   12.9854    77.7081    560048
385              Post Graduate            5   12.9850    77.5533    560010
386              Post Graduate            2   12.9770    77.5773    560009
387              Post Graduate            5   12.8988    77.5764    560078

    Output   Feedback Unnamed: 12
0      Yes   Positive         Yes
1      Yes   Positive         Yes
2      Yes  Negative          Yes
3      Yes   Positive         Yes
4      Yes   Positive         Yes
..     …        …          …
383    Yes   Positive         Yes
384    Yes   Positive         Yes
385    Yes   Positive         Yes
386    Yes   Positive         Yes
387    Yes   Positive         Yes

[388 rows x 13 columns]
```

Cerchiamo quanti missin values ci sono in questo dataframe

[ ]:
```
percentage_missing = df.isnull().sum() * 100 / len(df)
print(percentage_missing)
```

```
Age                           0.0
Gender                        0.0
Marital Status                0.0
Occupation                    0.0
Monthly Income                0.0
Educational Qualifications    0.0
```

```
Family size                  0.0
latitude                     0.0
longitude                    0.0
Pin code                     0.0
Output                       0.0
Feedback                     0.0
Unnamed: 12                  0.0
dtype: float64
```

Nessuno! Fantastico! è un dataset fatto bene, a parte per la colonna Unnamed: 12

[ ]:
```
df = df.drop("Unnamed: 12", axis=1)
df
```

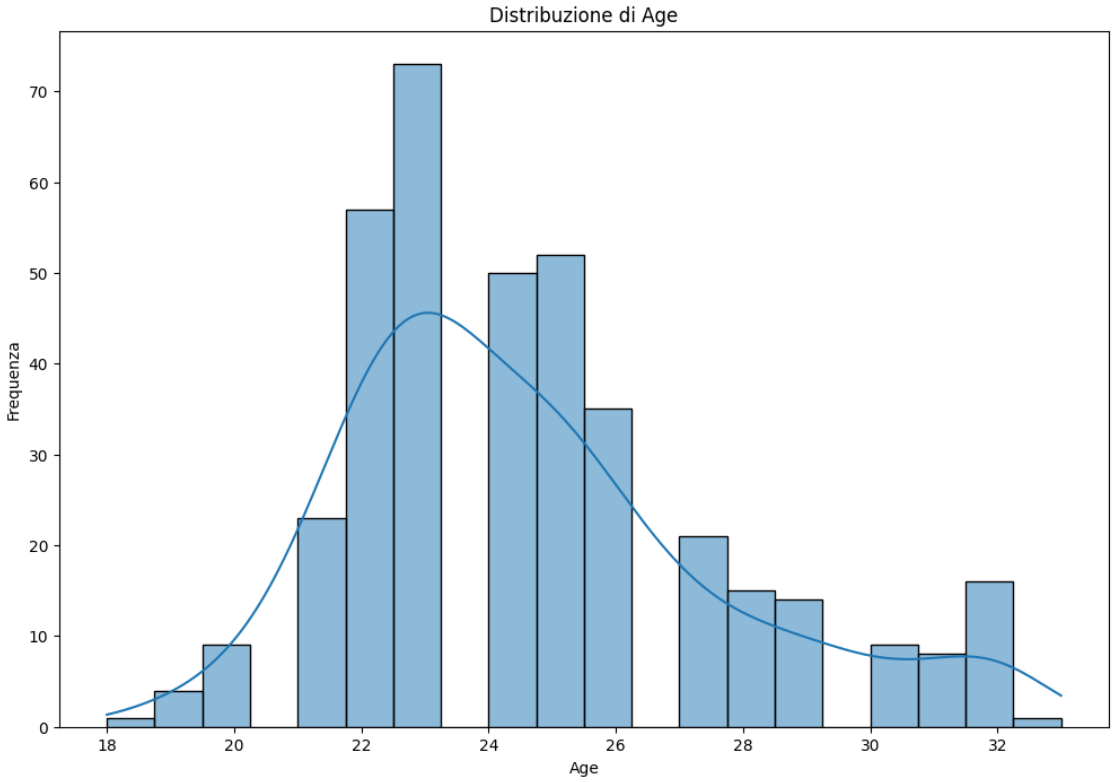
[ ]:
```
      Age  Gender Marital Status Occupation  Monthly Income  \
0      20  Female         Single    Student       No Income
1      24  Female         Single    Student  Below Rs.10000
2      22    Male         Single    Student  Below Rs.10000
3      22  Female         Single    Student       No Income
4      22    Male         Single    Student  Below Rs.10000
..    …     …             …         …              …
383   23  Female         Single    Student       No Income
384   23  Female         Single    Student       No Income
385   22  Female         Single    Student       No Income
386   23    Male         Single    Student  Below Rs.10000
387   23    Male         Single    Student       No Income

    Educational Qualifications  Family size  latitude  longitude  Pin code  \
0                Post Graduate            4   12.9766    77.5993    560001
1                     Graduate            3   12.9770    77.5773    560009
2                Post Graduate            3   12.9551    77.6593    560017
3                     Graduate            6   12.9473    77.5616    560019
4                Post Graduate            4   12.9850    77.5533    560010
..                         …          …       …        …       …
383              Post Graduate            2   12.9766    77.5993    560001
384              Post Graduate            4   12.9854    77.7081    560048
385              Post Graduate            5   12.9850    77.5533    560010
386              Post Graduate            2   12.9770    77.5773    560009
387              Post Graduate            5   12.8988    77.5764    560078

    Output   Feedback
0      Yes   Positive
1      Yes   Positive
2      Yes  Negative
3      Yes   Positive
4      Yes   Positive
..     …        …
383    Yes   Positive
```

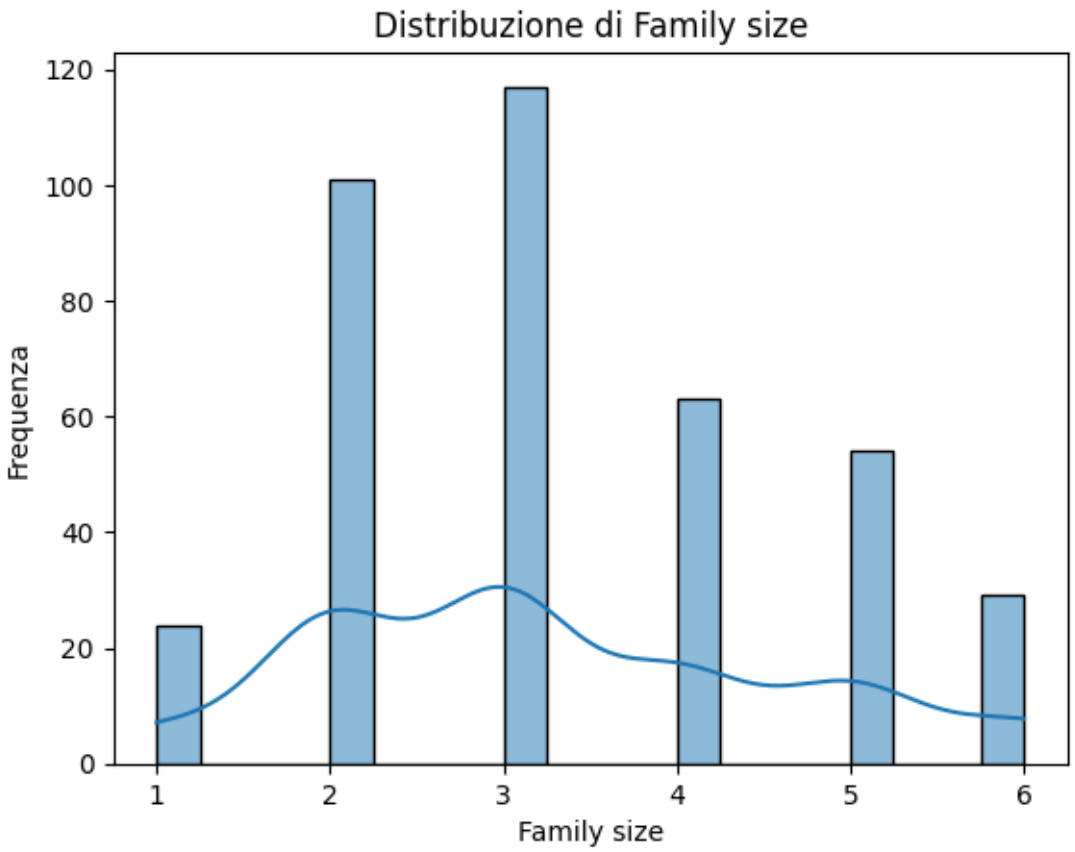
```
384    Yes   Positive
385    Yes   Positive
386    Yes   Positive
387    Yes   Positive

[388 rows x 12 columns]
```

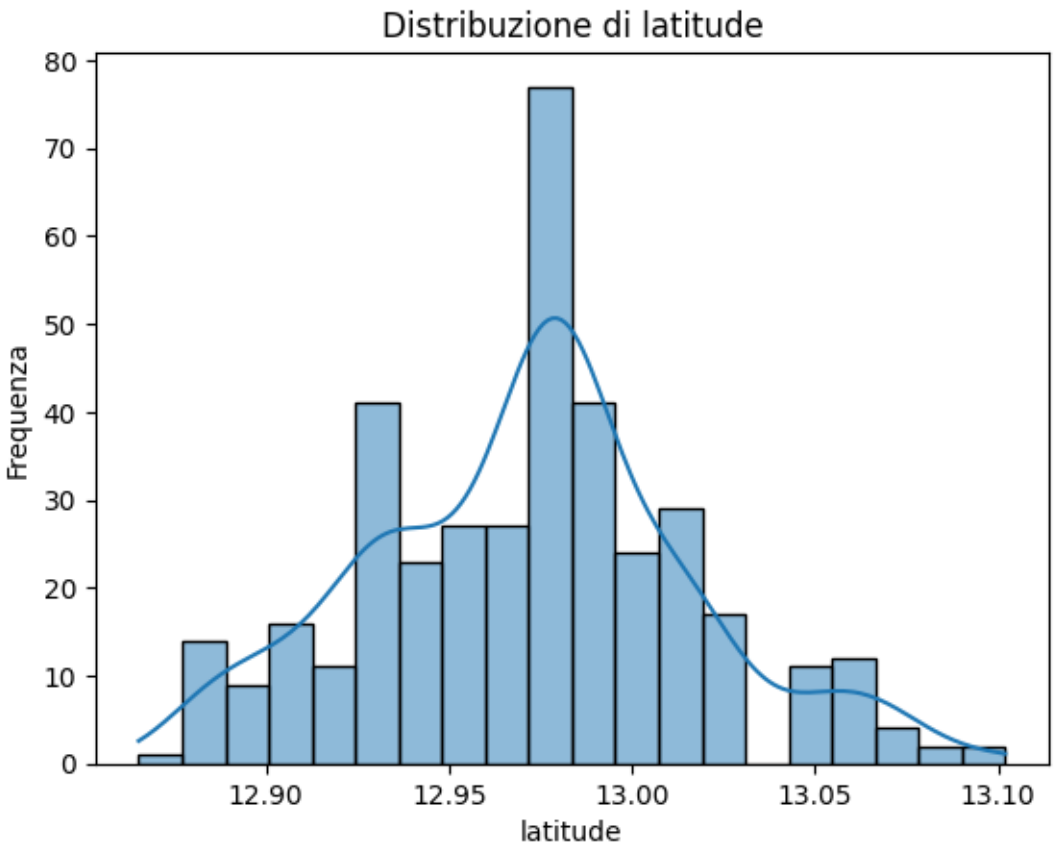
That's perfect!

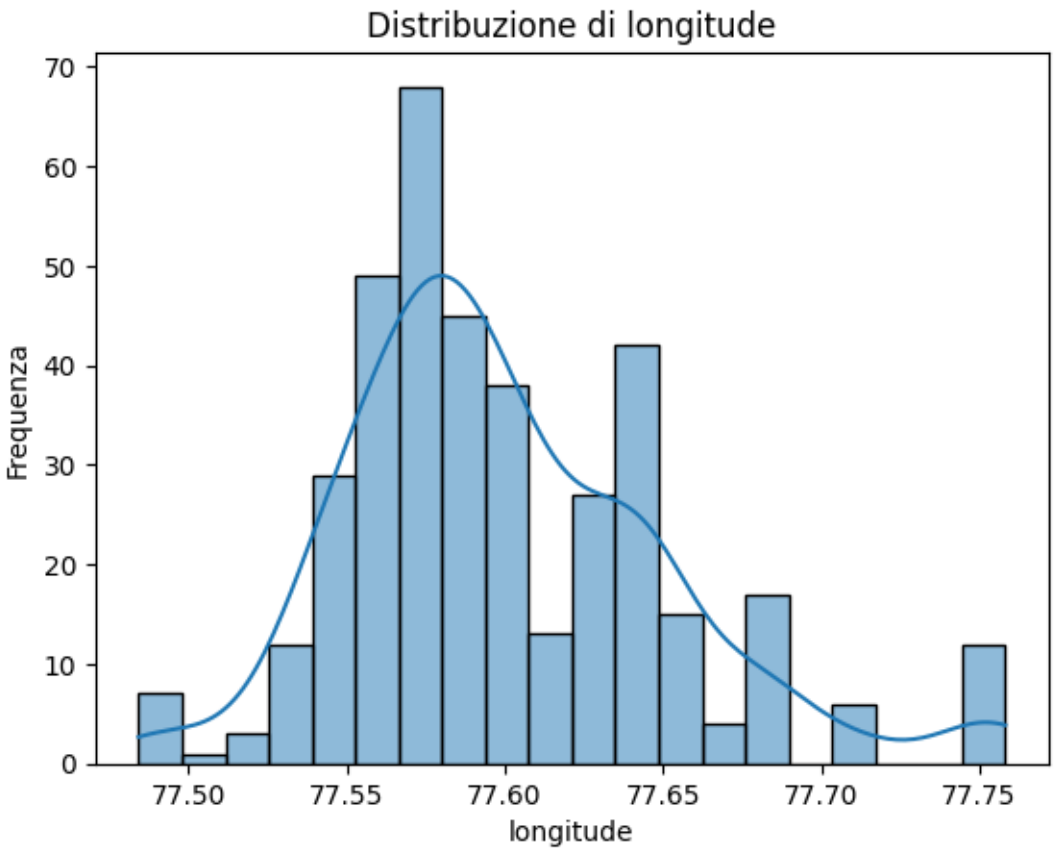
Adesso i grafici

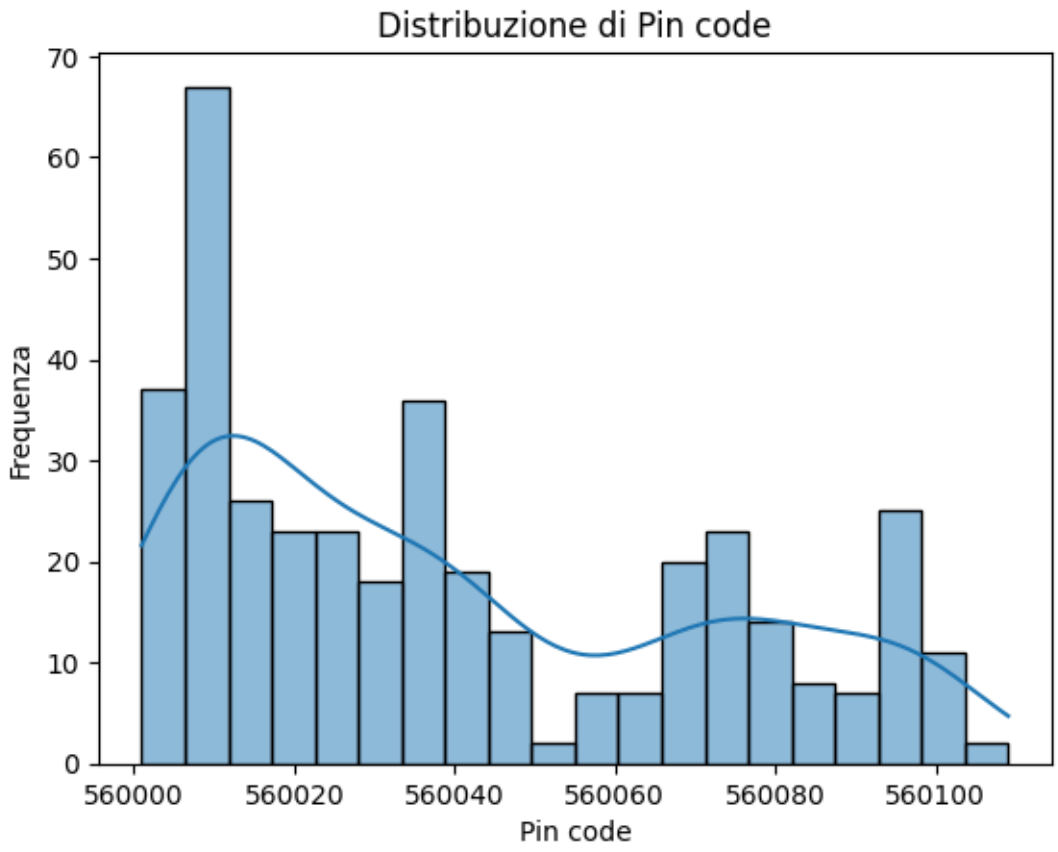
[ ]:
```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Assume che il tuo nuovo DataFrame sia già definito come df2_cleaned
df1 = df
# Seleziona solo le colonne numeriche
numeric_columns = df1.select_dtypes(include=['float64', 'int64'])

plt.figure(figsize=(12, 8))
for col in numeric_columns.columns:
    sns.histplot(data=df1, x=col, kde=True, bins=20)
    plt.title(f'Distribuzione di {col}')
    plt.xlabel(col)
    plt.ylabel('Frequenza')
    plt.show()
```


Distribuzione di Age
Frequenza
Age
0
10
20
30
40
50
60
70
18
20
22
24
26
28
30
32

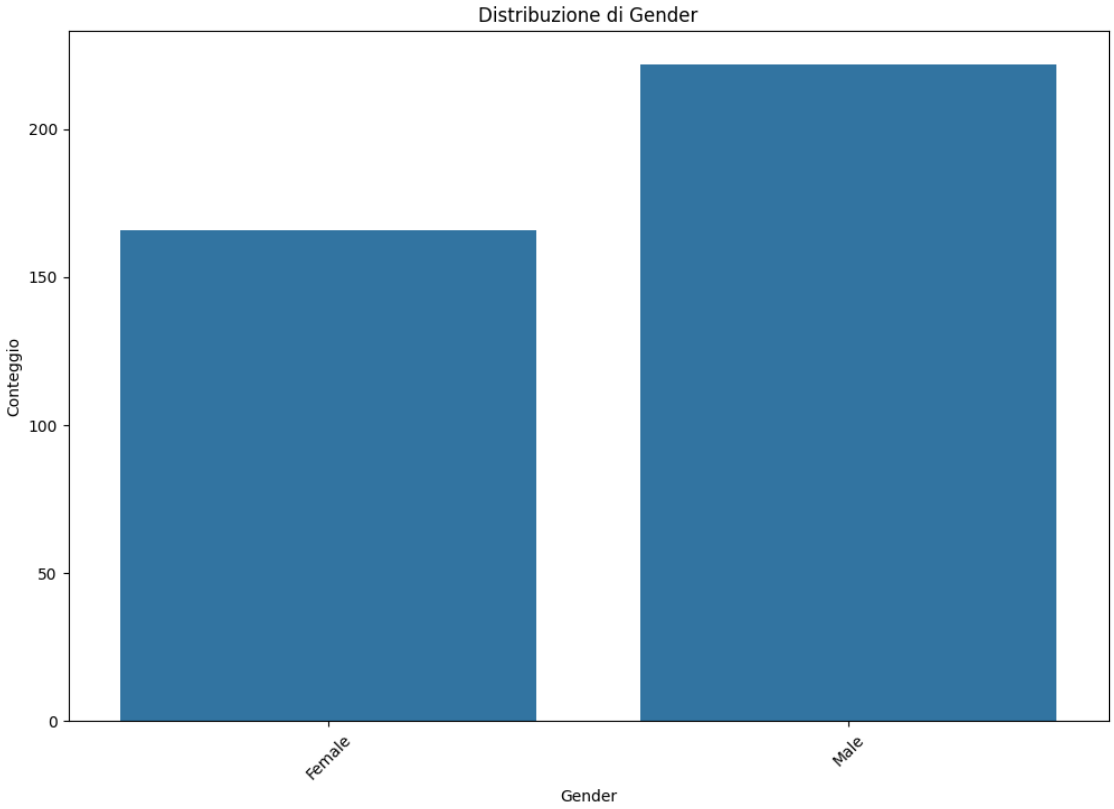

Distribuzione di Family size
Frequenza
Family size
0
20
40
60
80
100
120
1
2
3
4
5
6

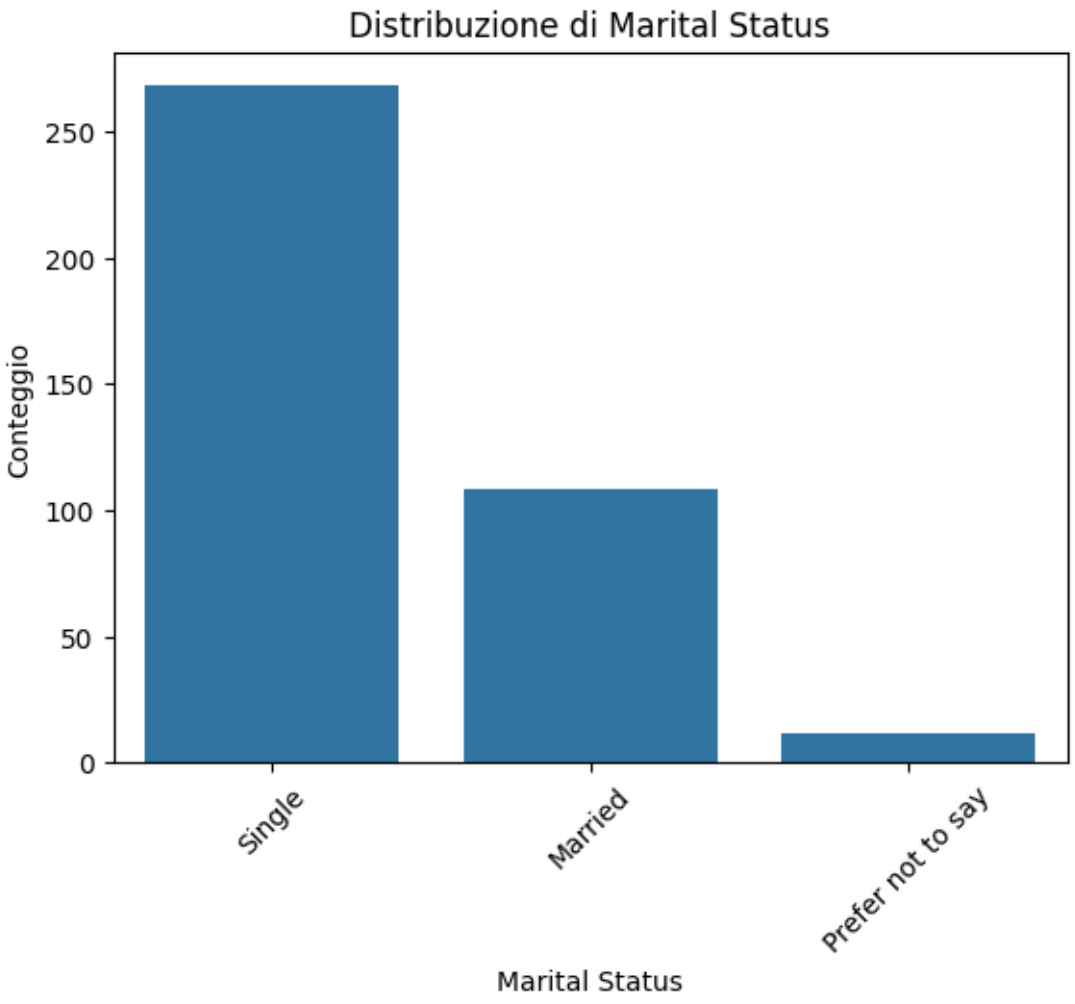

Distribuzione di latitude
Frequenza
latitude
0
10
20
30
40
50
60
70
80
12.90
12.95
13.00
13.05
13.10

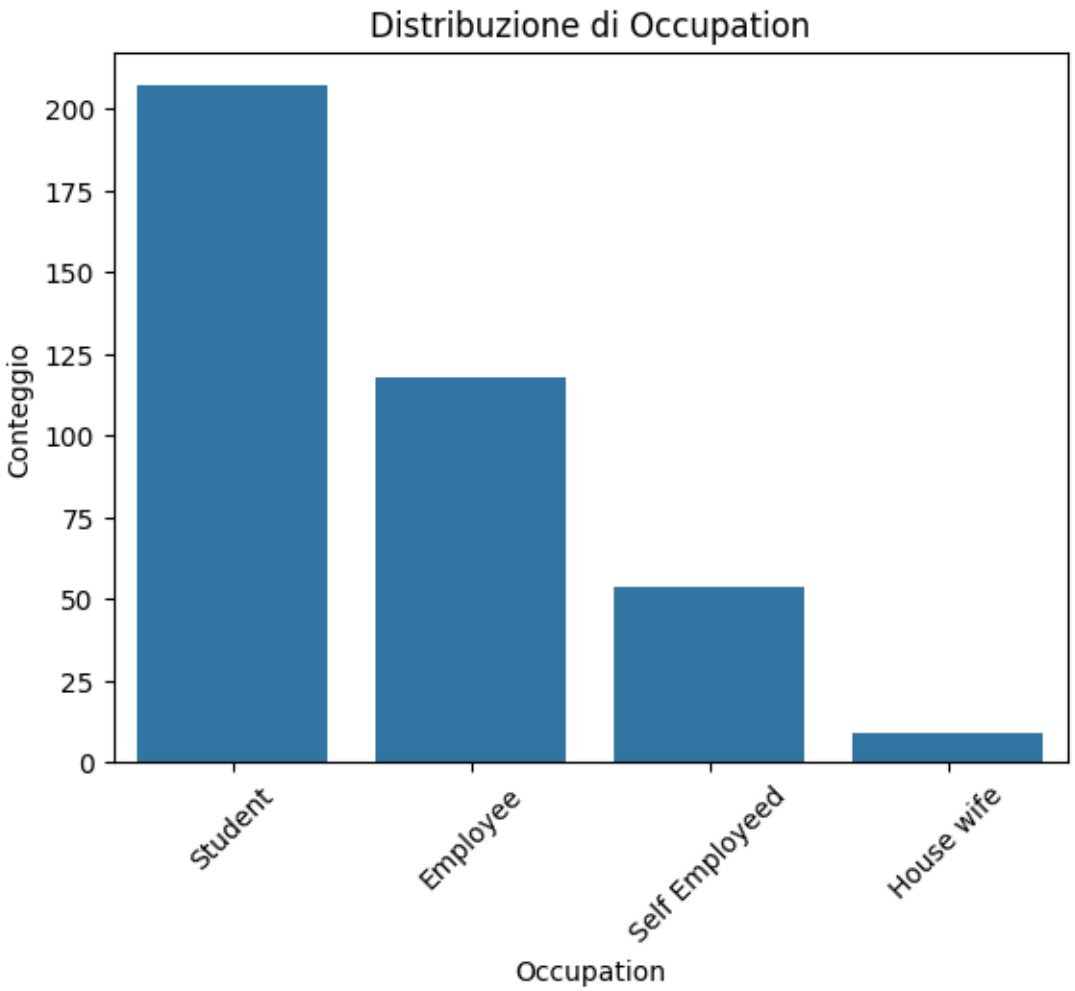

Distribuzione di longitude
Frequenza
longitude
0
10
20
30
40
50
60
70
77.50
77.55
77.60
77.65
77.70
77.75

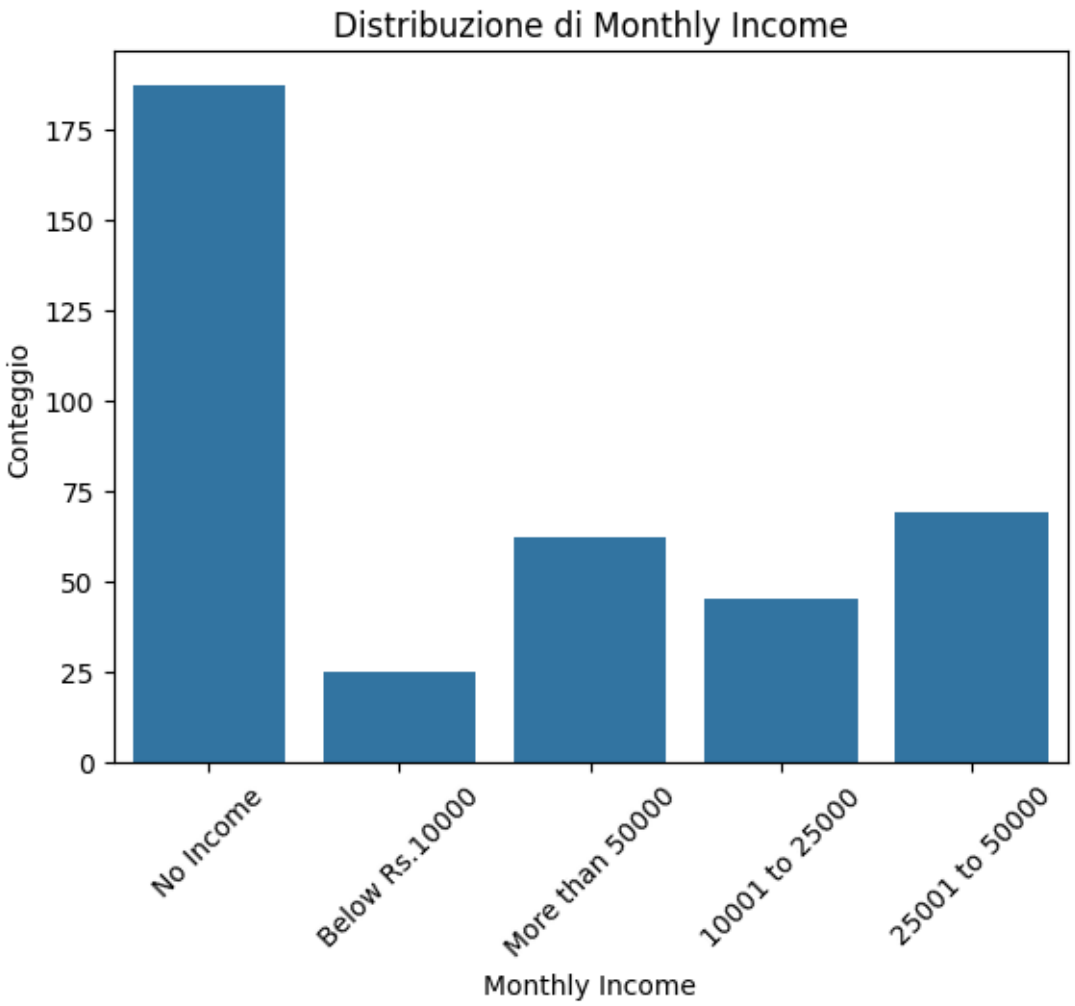


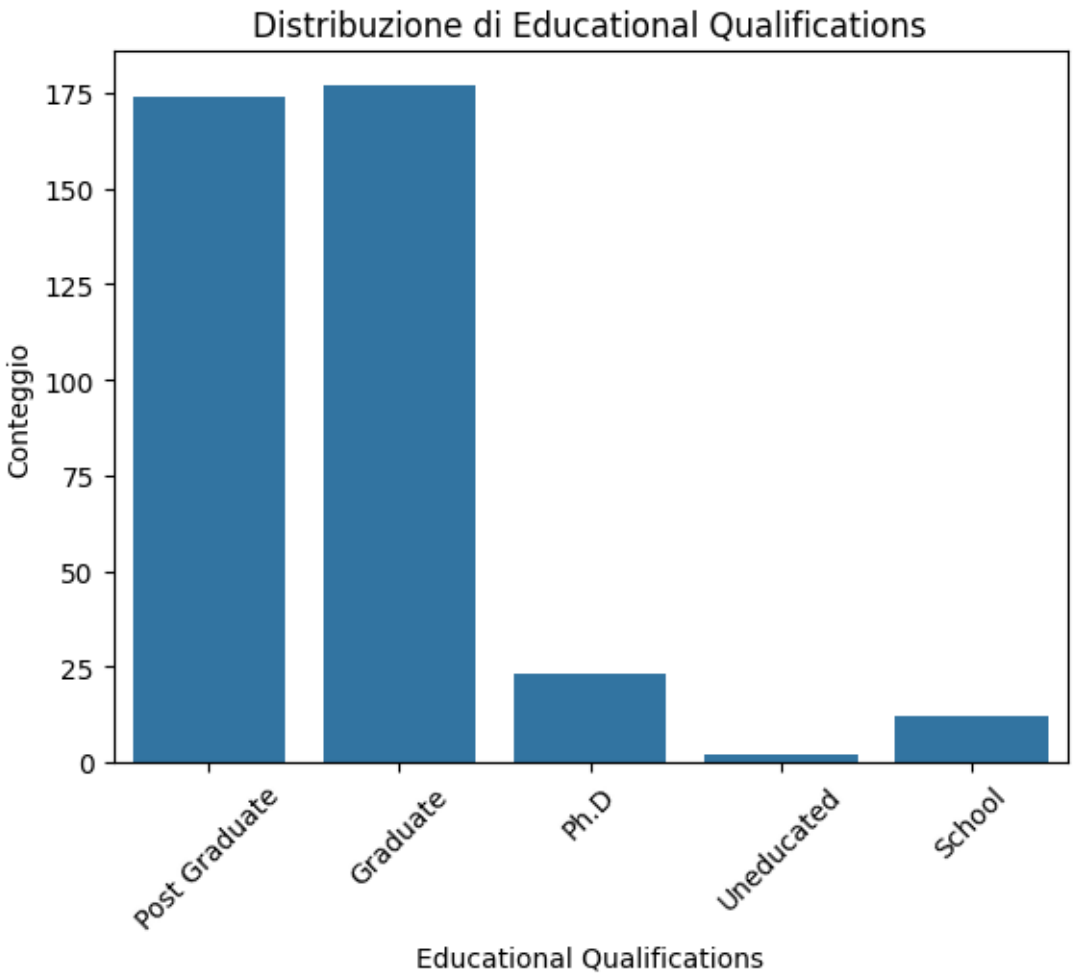
[ ]:
```
# Seleziona solo le colonne categoriche
categorical_columns = df1.select_dtypes(include=['object'])

plt.figure(figsize=(12, 8))
for col in categorical_columns.columns:
    sns.countplot(data=df1, x=col)
    plt.title(f'Distribuzione di {col}')
    plt.xlabel(col)
    plt.ylabel('Conteggio')
    plt.xticks(rotation=45)
    plt.show()
```

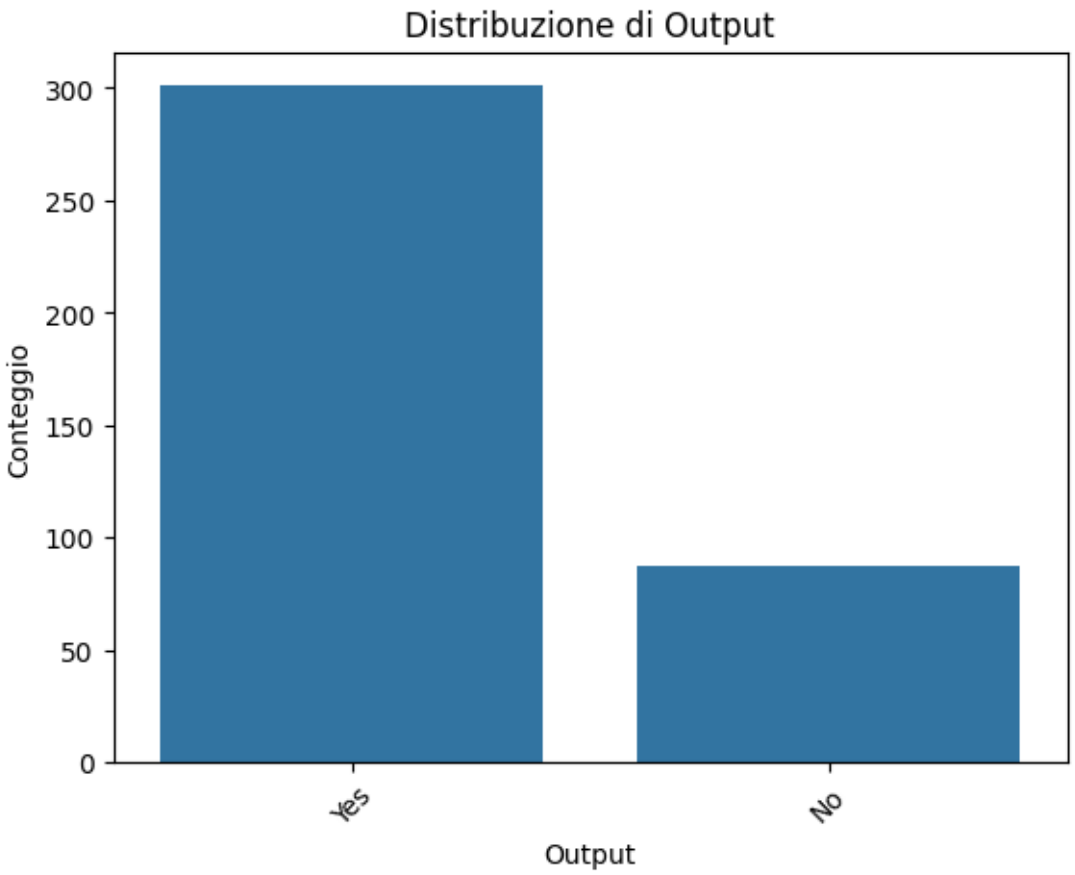

Distribuzione di Gender
Conteggio
0
50
100
150
200
Female
Male
Gender

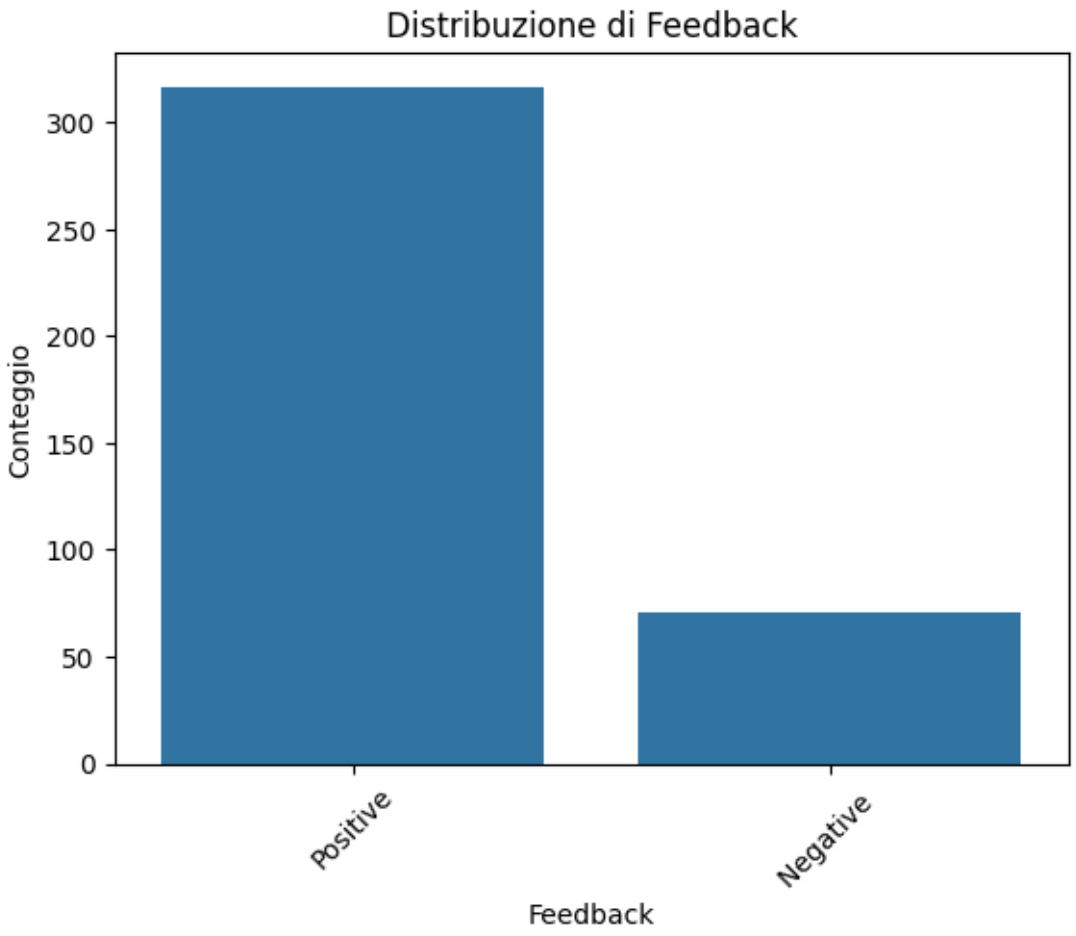

Distribuzione di Marital Status
Conteggio
0
50
100
150
200
250
Single
Married
Prefer not to say
Marital Status


Distribuzione di Occupation
Conteggio
0
25
50
75
100
125
150
175
200
Student
Employee
Self Employeed
House wife
Occupation


Distribuzione di Monthly Income
Conteggio
0
25
50
75
100
125
150
175
No Income
Below Rs.10000
More than 50000
10001 to 25000
25001 to 50000
Monthly Income


Distribuzione di Educational Qualifications
Conteggio
0
25
50
75
100
125
150
175
Post Graduate
Graduate
Ph.D
Uneducated
School
Educational Qualifications


Distribuzione di Output
Conteggio
300
250
200
150
100
50
0
Yes
No
Output



dobbiamo preparare i dati per darli ad un intelligenza artificiale

[ ]:
```
from sklearn.preprocessing import StandardScaler

# Seleziona solo le colonne numeriche nel DataFrame
numeric_columns = df1.select_dtypes(include=['float64', 'int64'])

# Inizializza lo StandardScaler
scaler = StandardScaler()

# Applica lo scaling alle colonne numeriche
scaled_numeric_columns = scaler.fit_transform(numeric_columns)

# Crea un nuovo DataFrame con le colonne numeriche scalate
df2_scaled = pd.DataFrame(scaled_numeric_columns, columns=numeric_columns.
 ↪columns)

# Visualizza il nuovo DataFrame scalato
```

```
df2_scaled
```

[ ]:
```
          Age  Family size  latitude  longitude  Pin code
0   -1.557620     0.532929  0.102224  -0.016759 -1.247274
1   -0.211614    -0.208205  0.111227  -0.445712 -0.992164
2   -0.884617    -0.208205 -0.381663   1.153112 -0.737055
3   -0.884617     2.015198 -0.557212  -0.751828 -0.673278
4   -0.884617     0.532929  0.291278  -0.913660 -0.960276
..        …          …       …         …        …
383 -0.548116    -0.949340  0.102224  -0.016759 -1.247274
384 -0.548116     0.532929  0.300280   2.104607  0.251493
385 -0.884617     1.274064  0.291278  -0.913660 -0.960276
386 -0.548116    -0.949340  0.111227  -0.445712 -0.992164
387 -0.548116     1.274064 -1.648771  -0.463260  1.208152

[388 rows x 5 columns]
```

[ ]:
```
from sklearn.preprocessing import OneHotEncoder

# Seleziona solo le colonne categoriche nel DataFrame
categorical_columns = df1.select_dtypes(include=['object'])

# Inizializza OneHotEncoder
encoder = OneHotEncoder()

# Applica l'encoding alle colonne categoriche e trasforma i dati
encoded_categorical_columns = encoder.fit_transform(categorical_columns)

# Ottieni i nomi delle categorie dall'encoder
encoded_categories = encoder.categories_

# Crea i nomi delle nuove colonne dopo l'encoding
encoded_column_names = []
for i, col in enumerate(categorical_columns.columns):
    encoded_column_names.extend([f"{col}_{category}" for category in␣
 ↪encoded_categories[i]])

df2_encoded = pd.DataFrame(encoded_categorical_columns.toarray(),␣
 ↪columns=encoded_column_names)

print(df2_encoded)
```

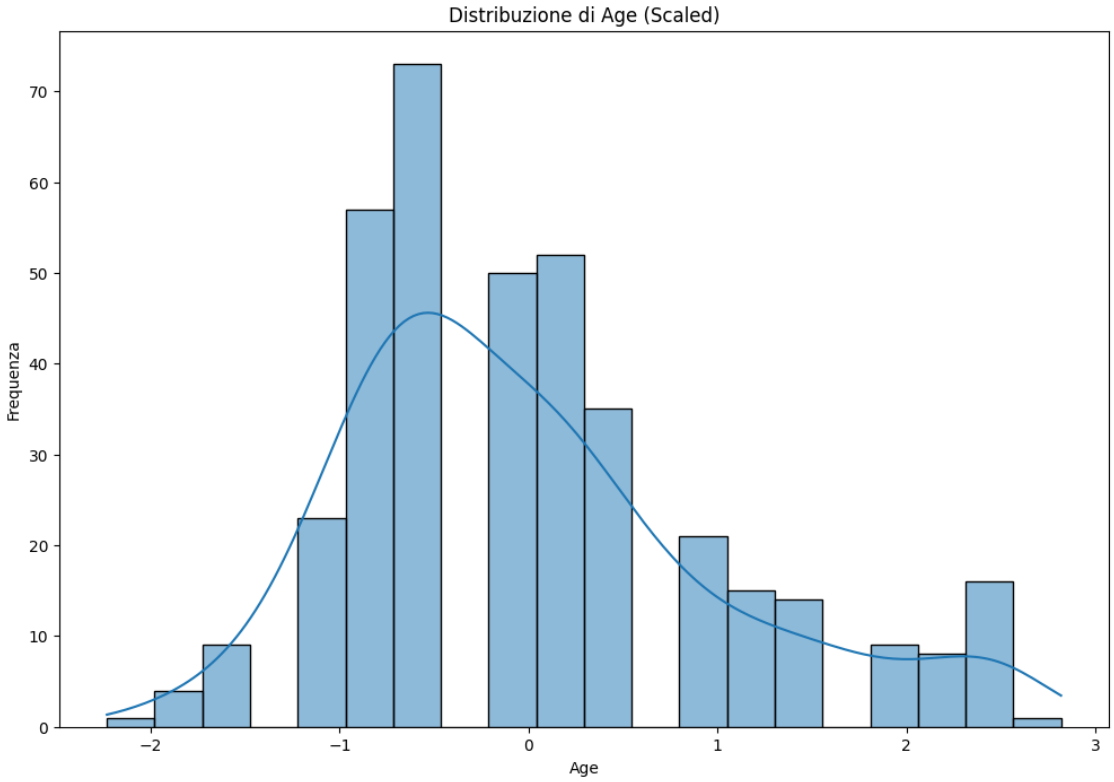
```
     Gender_Female  Gender_Male  Marital Status_Married  \
0              1.0          0.0                     0.0
1              1.0          0.0                     0.0
2              0.0          1.0                     0.0
3              1.0          0.0                     0.0
4              0.0          1.0                     0.0
```

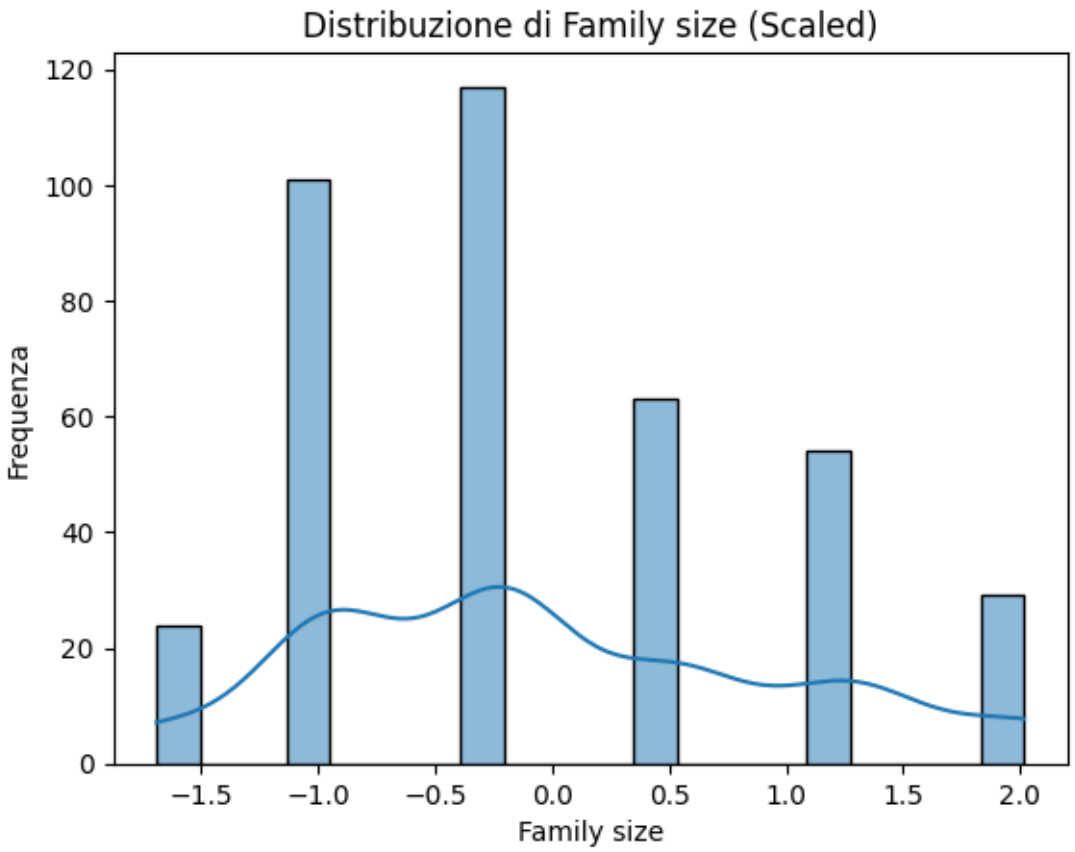
```
..              …          …                       …
383            1.0          0.0                     0.0
384            1.0          0.0                     0.0
385            1.0          0.0                     0.0
386            0.0          1.0                     0.0
387            0.0          1.0                     0.0

     Marital Status_Prefer not to say  Marital Status_Single  \
0                                 0.0                    1.0
1                                 0.0                    1.0
2                                 0.0                    1.0
3                                 0.0                    1.0
4                                 0.0                    1.0
..                                …                      …
383                               0.0                    1.0
384                               0.0                    1.0
385                               0.0                    1.0
386                               0.0                    1.0
387                               0.0                    1.0

     Occupation_Employee  Occupation_House wife  Occupation_Self Employeed  \
0                    0.0                    0.0                        0.0
1                    0.0                    0.0                        0.0
2                    0.0                    0.0                        0.0
3                    0.0                    0.0                        0.0
4                    0.0                    0.0                        0.0
..                   …                      …                          …
383                  0.0                    0.0                        0.0
384                  0.0                    0.0                        0.0
385                  0.0                    0.0                        0.0
386                  0.0                    0.0                        0.0
387                  0.0                    0.0                        0.0

     Occupation_Student  Monthly Income_10001 to 25000  …  \
0                   1.0                            0.0  …
1                   1.0                            0.0  …
2                   1.0                            0.0  …
3                   1.0                            0.0  …
4                   1.0                            0.0  …
..                  …                              …  …
383                 1.0                            0.0  …
384                 1.0                            0.0  …
385                 1.0                            0.0  …
386                 1.0                            0.0  …
387                 1.0                            0.0  …

     Monthly Income_No Income  Educational Qualifications_Graduate  \
0                         1.0                                  0.0
```

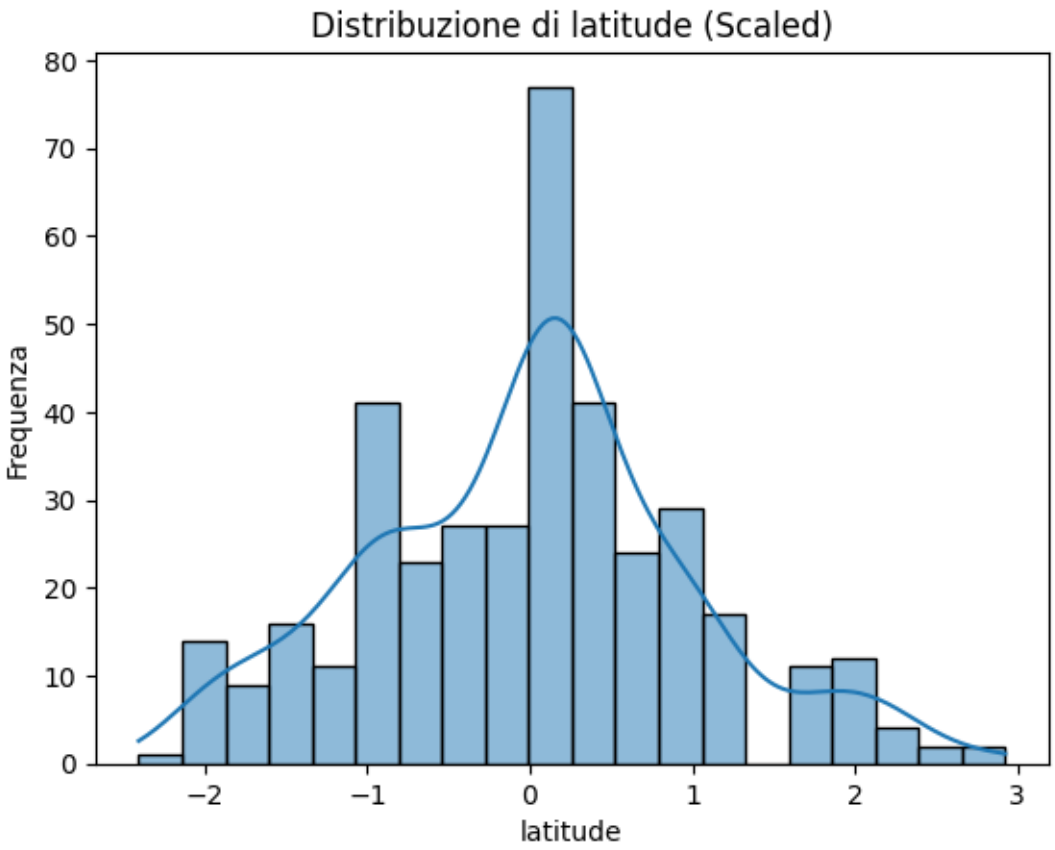
```
1                         0.0                                  1.0
2                         0.0                                  0.0
3                         1.0                                  1.0
4                         0.0                                  0.0
..                        …                                  …
383                       1.0                                  0.0
384                       1.0                                  0.0
385                       1.0                                  0.0
386                       0.0                                  0.0
387                       1.0                                  0.0

     Educational Qualifications_Ph.D  \
0                                0.0
1                                0.0
2                                0.0
3                                0.0
4                                0.0
..                               …
383                              0.0
384                              0.0
385                              0.0
386                              0.0
387                              0.0

     Educational Qualifications_Post Graduate  \
0                                         1.0
1                                         0.0
2                                         1.0
3                                         0.0
4                                         1.0
..                                        …
383                                       1.0
384                                       1.0
385                                       1.0
386                                       1.0
387                                       1.0

     Educational Qualifications_School  Educational Qualifications_Uneducated  \
0                                  0.0                                     0.0
1                                  0.0                                     0.0
2                                  0.0                                     0.0
3                                  0.0                                     0.0
4                                  0.0                                     0.0
..                                 …                                     …
383                                0.0                                     0.0
384                                0.0                                     0.0
385                                0.0                                     0.0
386                                0.0                                     0.0
```

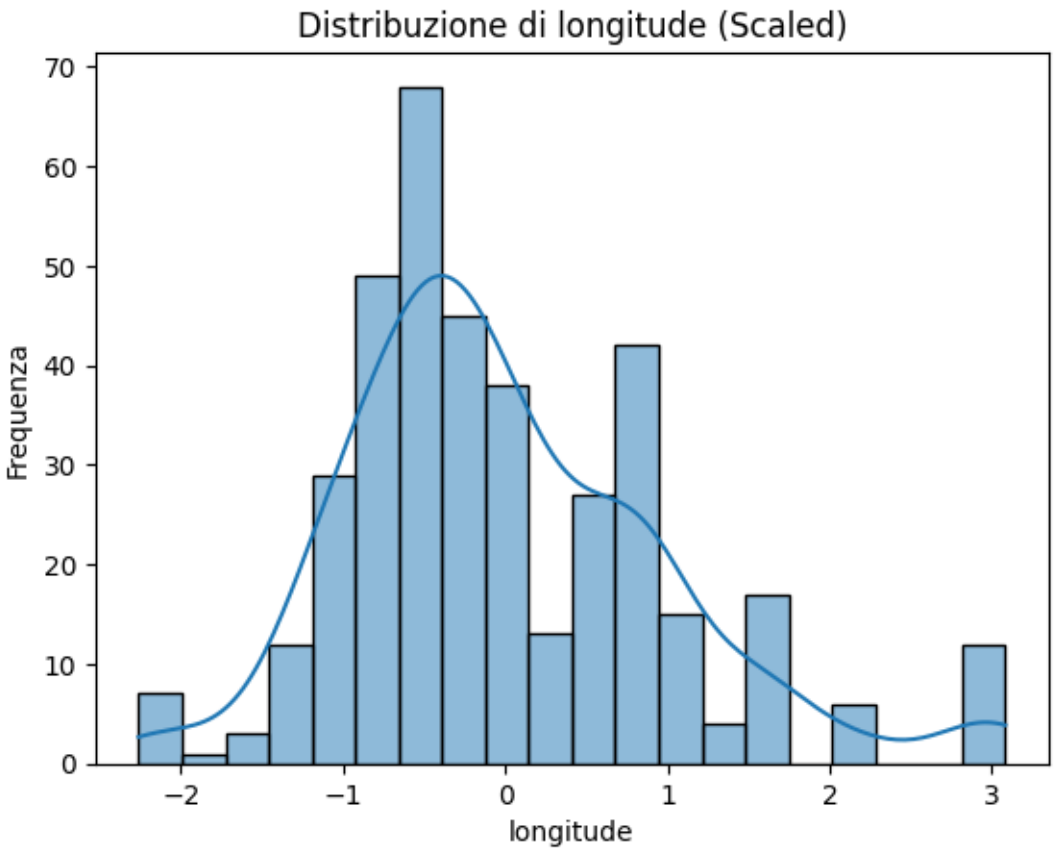
```
387                                0.0                                0.0

     Output_No  Output_Yes  Feedback_Negative   Feedback_Positive
0          0.0         1.0                0.0                 1.0
1          0.0         1.0                0.0                 1.0
2          0.0         1.0                1.0                 0.0
3          0.0         1.0                0.0                 1.0
4          0.0         1.0                0.0                 1.0
..         …           …                  …                   …
383        0.0         1.0                0.0                 1.0
384        0.0         1.0                0.0                 1.0
385        0.0         1.0                0.0                 1.0
386        0.0         1.0                0.0                 1.0
387        0.0         1.0                0.0                 1.0

[388 rows x 23 columns]
```

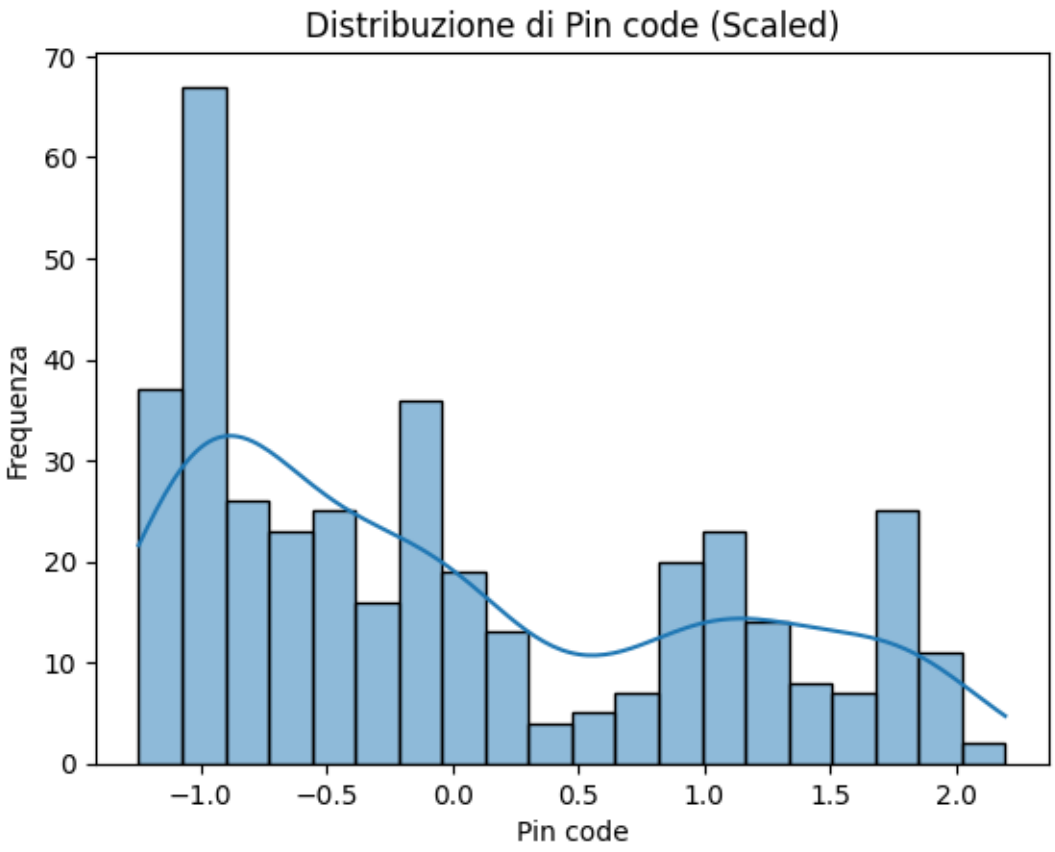
[ ]:
```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler, OneHotEncoder

# Assume che tu abbia già applicato lo scaling e l'encoding al DataFrame
# Quindi, abbiamo df2_scaled per le colonne numeriche scalate e df2_encoded per␣
 ↪le colonne categoriche codificate

# Visualizzazione delle distribuzioni delle colonne numeriche scalate␣
 ↪utilizzando un istogramma per ciascuna colonna
plt.figure(figsize=(12, 8))
for col in df2_scaled.columns:
    sns.histplot(data=df2_scaled, x=col, kde=True, bins=20)
    plt.title(f'Distribuzione di {col} (Scaled)')
    plt.xlabel(col)
    plt.ylabel('Frequenza')
    plt.show()

# Visualizzazione delle distribuzioni delle colonne categoriche codificate␣
 ↪utilizzando grafici a barre
plt.figure(figsize=(12, 8))
for col in df2_encoded.columns:
    sns.countplot(data=df2_encoded, x=col)
    plt.title(f'Distribuzione di {col} (Encoded)')
    plt.xlabel(col)
    plt.ylabel('Conteggio')
    plt.xticks(rotation=45)
    plt.show()
```

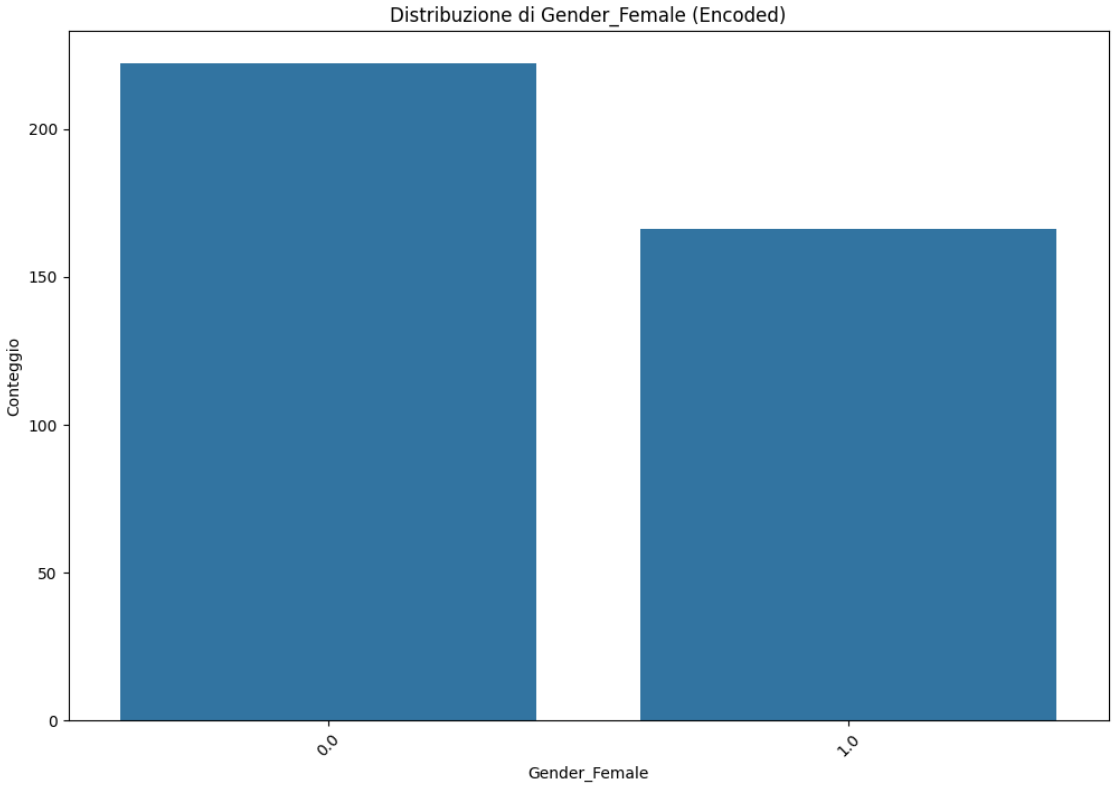

Distribuzione di Age (Scaled)
Frequenza
Age
0
10
20
30
40
50
60
70
−2
−1
0
1
2
3


Distribuzione di Family size (Scaled)
Frequenza
Family size
0
20
40
60
80
100
120
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

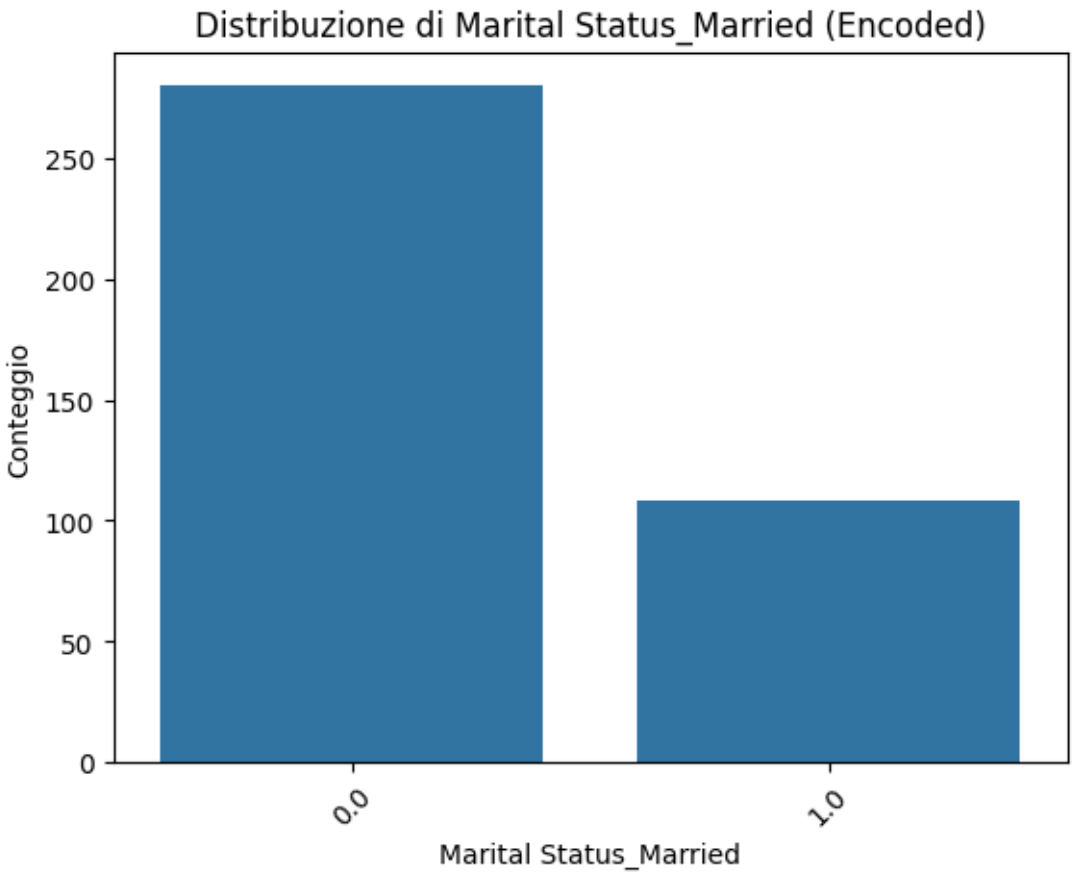

Distribuzione di latitude (Scaled)
Frequenza
latitude
0
10
20
30
40
50
60
70
80
−2
−1
0
1
2
3

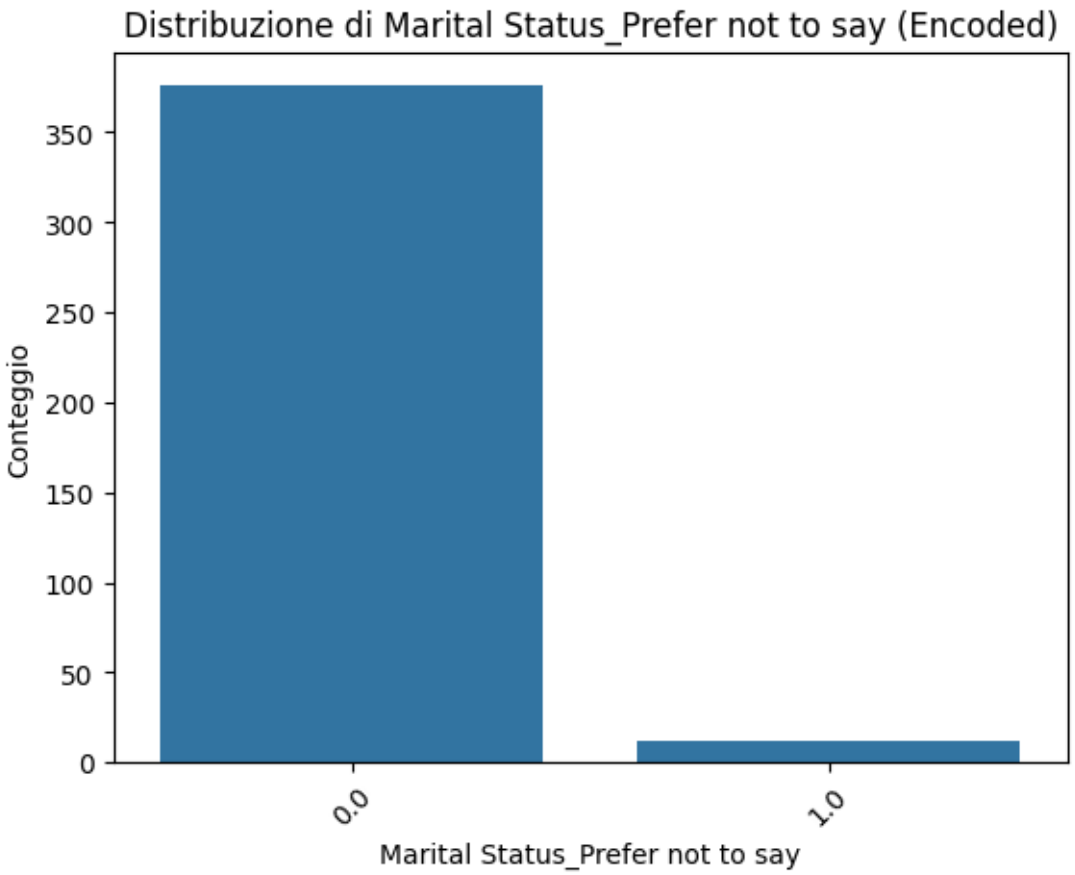

Distribuzione di longitude (Scaled)
Frequenza
longitude
0
10
20
30
40
50
60
70
−2
−1
0
1
2
3

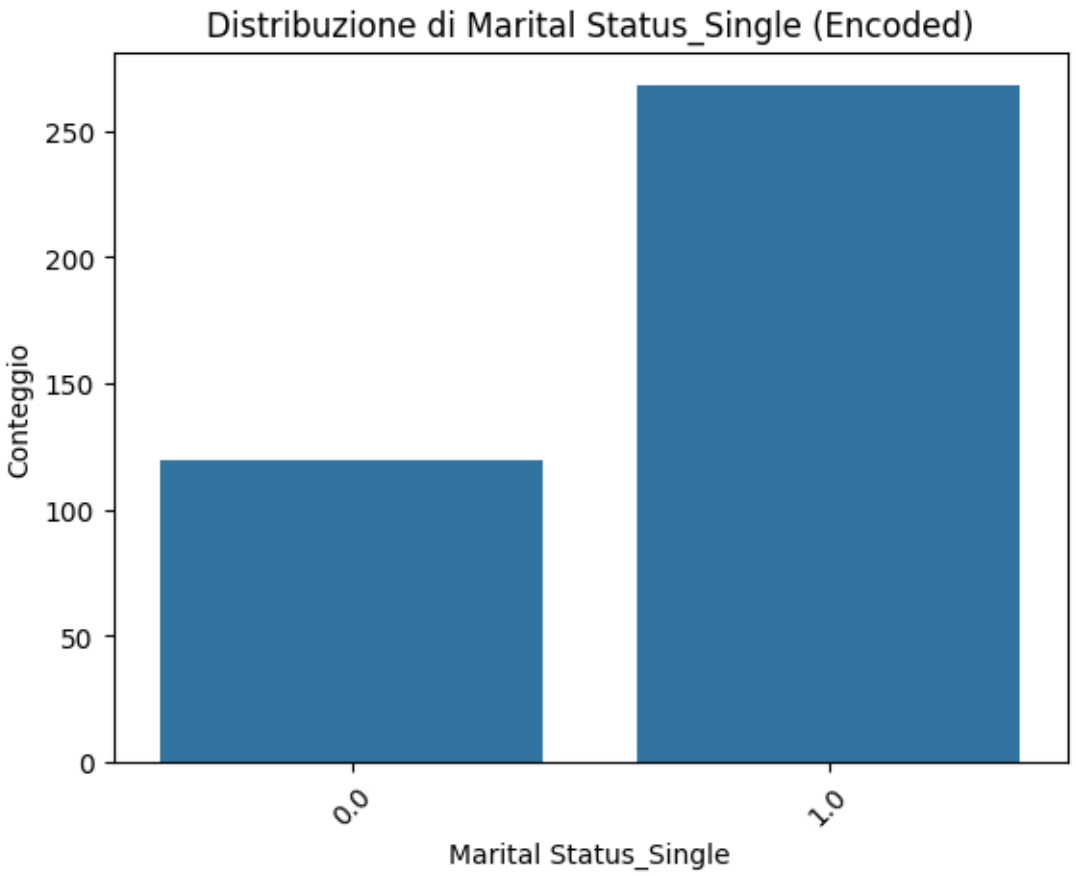

Distribuzione di Pin code (Scaled)
Frequenza
Pin code
0
10
20
30
40
50
60
70
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

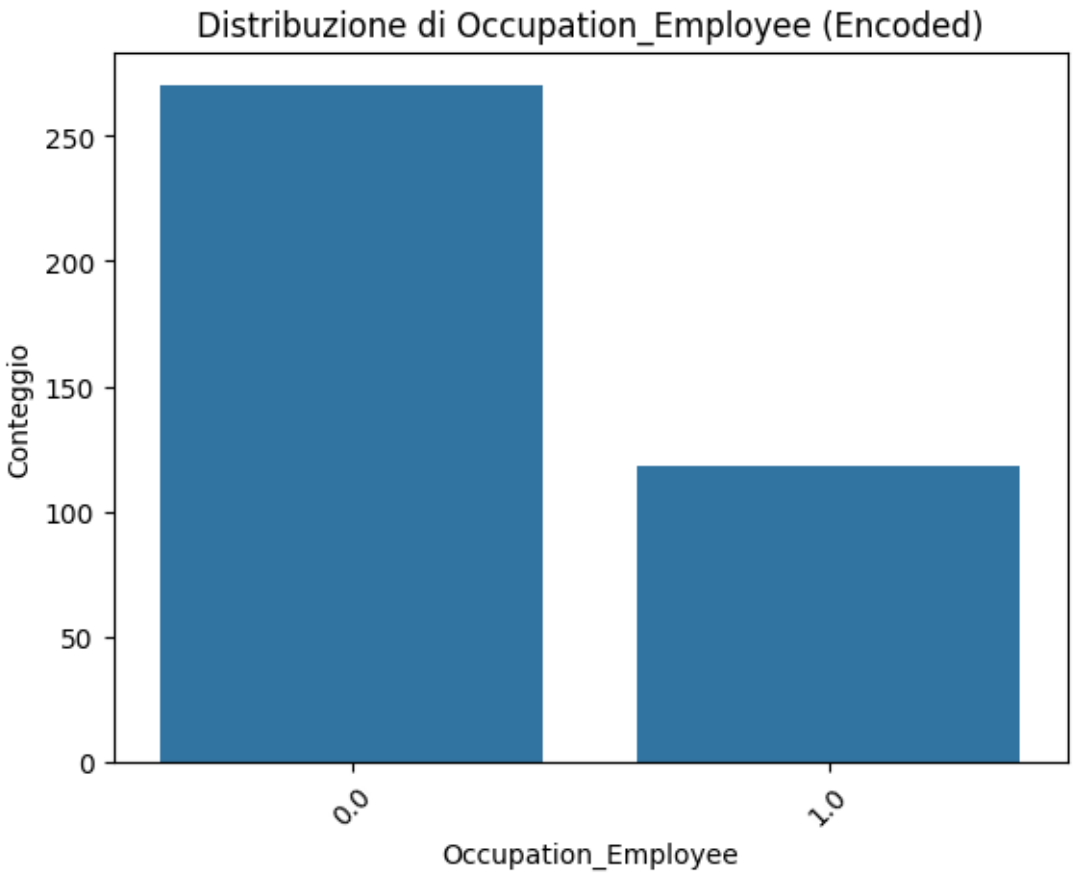

Distribuzione di Gender_Female (Encoded)
Conteggio
0
50
100
150
200
0.0
1.0
Gender_Female

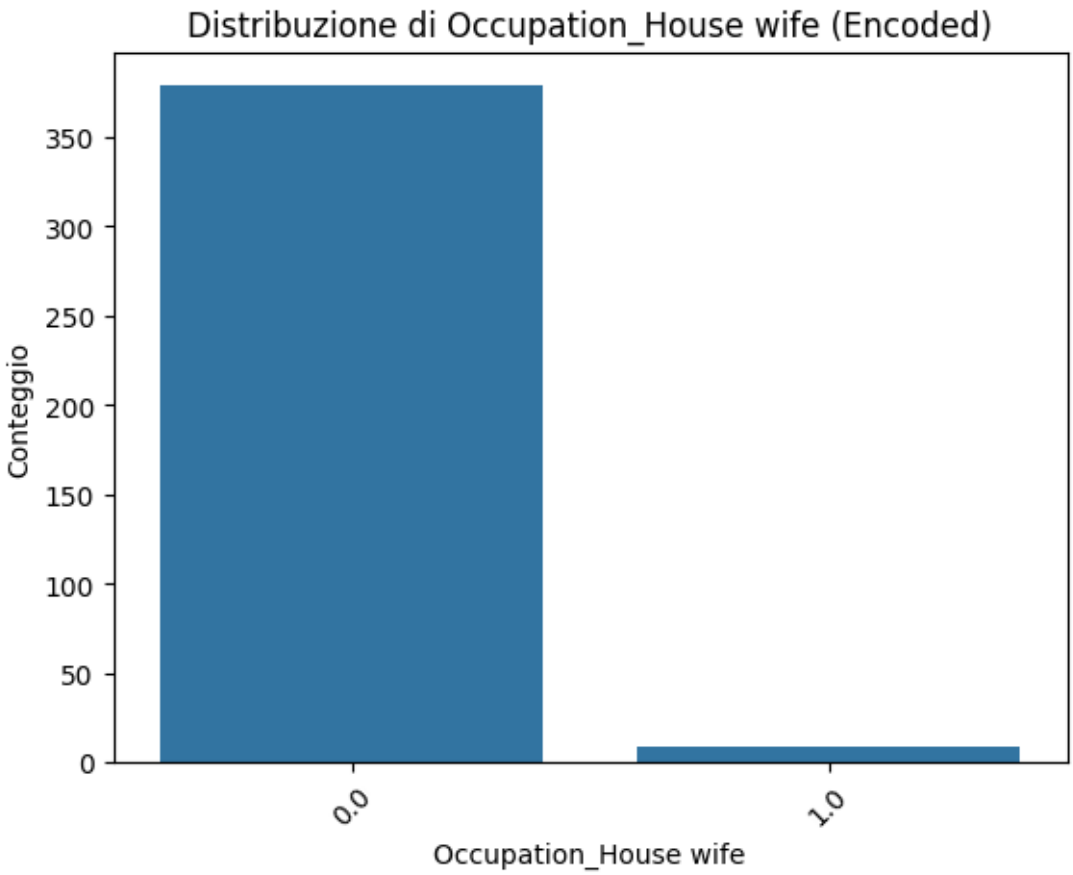

Distribuzione di Gender_Male (Encoded)
Conteggio
200
150
100
50
0
0.0
1.0
Gender_Male


Distribuzione di Marital Status_Married (Encoded)
Conteggio
250
200
150
100
50
0
0.0
1.0
Marital Status_Married

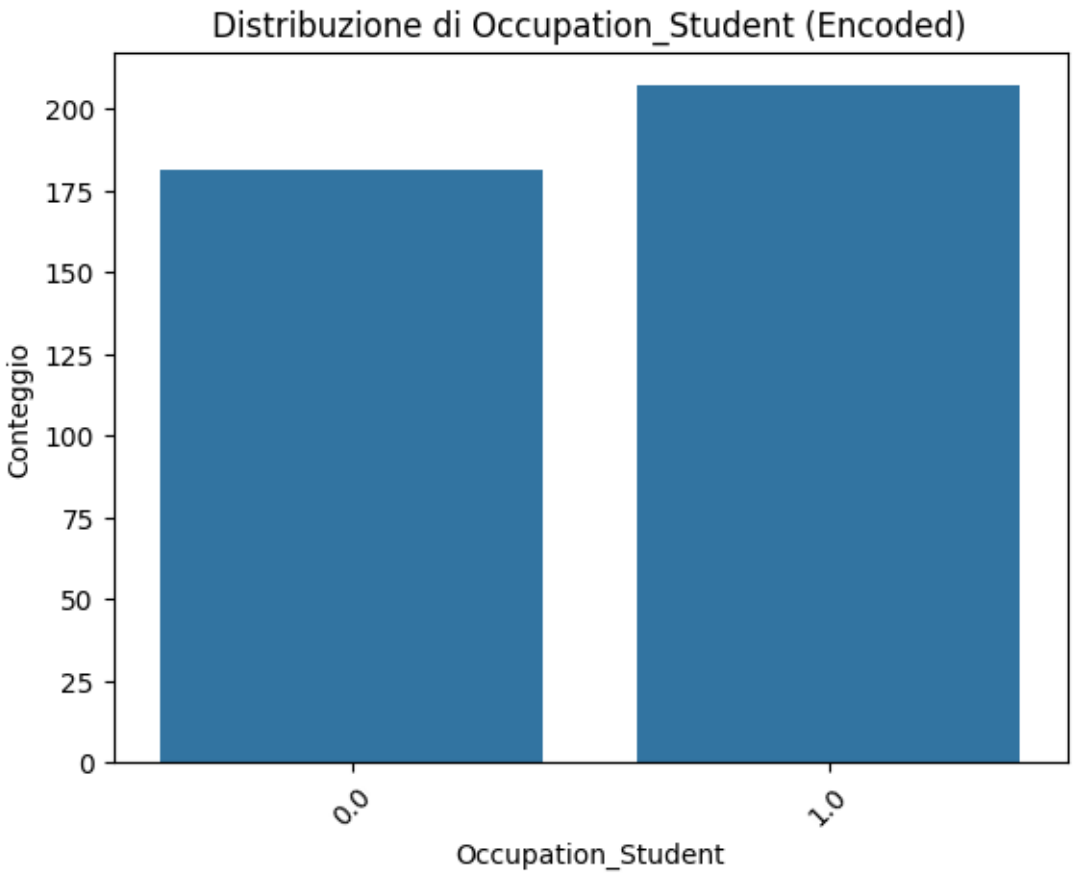

Distribuzione di Marital Status_Prefer not to say (Encoded)
Conteggio
350
300
250
200
150
100
50
0
0.0
1.0
Marital Status_Prefer not to say

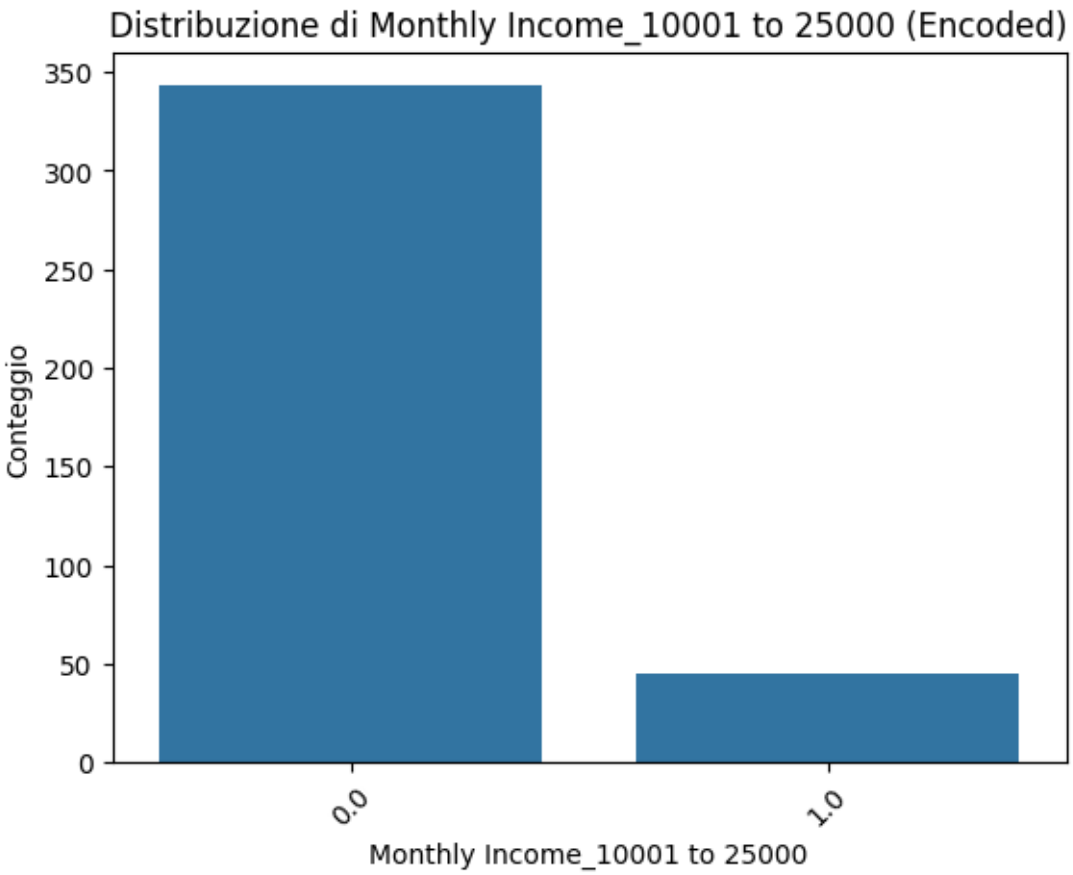

Distribuzione di Marital Status_Single (Encoded)
Conteggio
250
200
150
100
50
0
0.0
1.0
Marital Status_Single

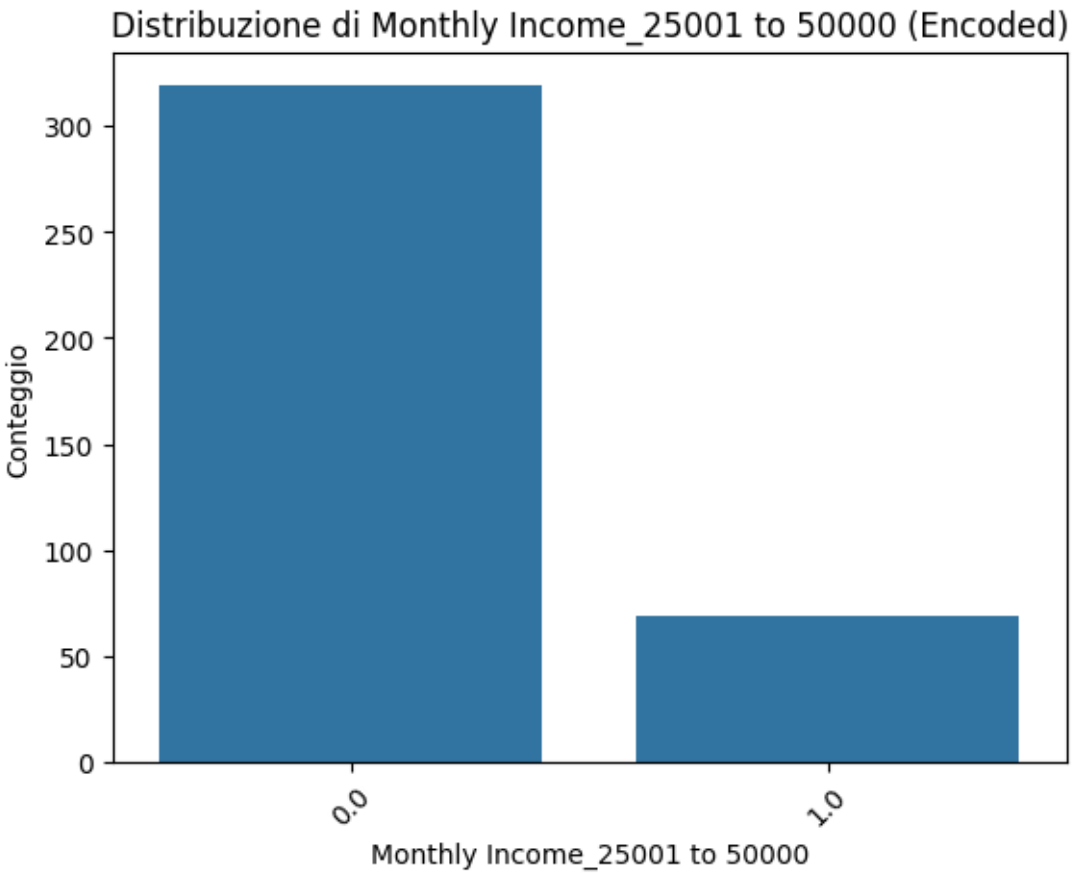

Distribuzione di Occupation_Employee (Encoded)
Conteggio
250
200
150
100
50
0
0.0
1.0
Occupation_Employee

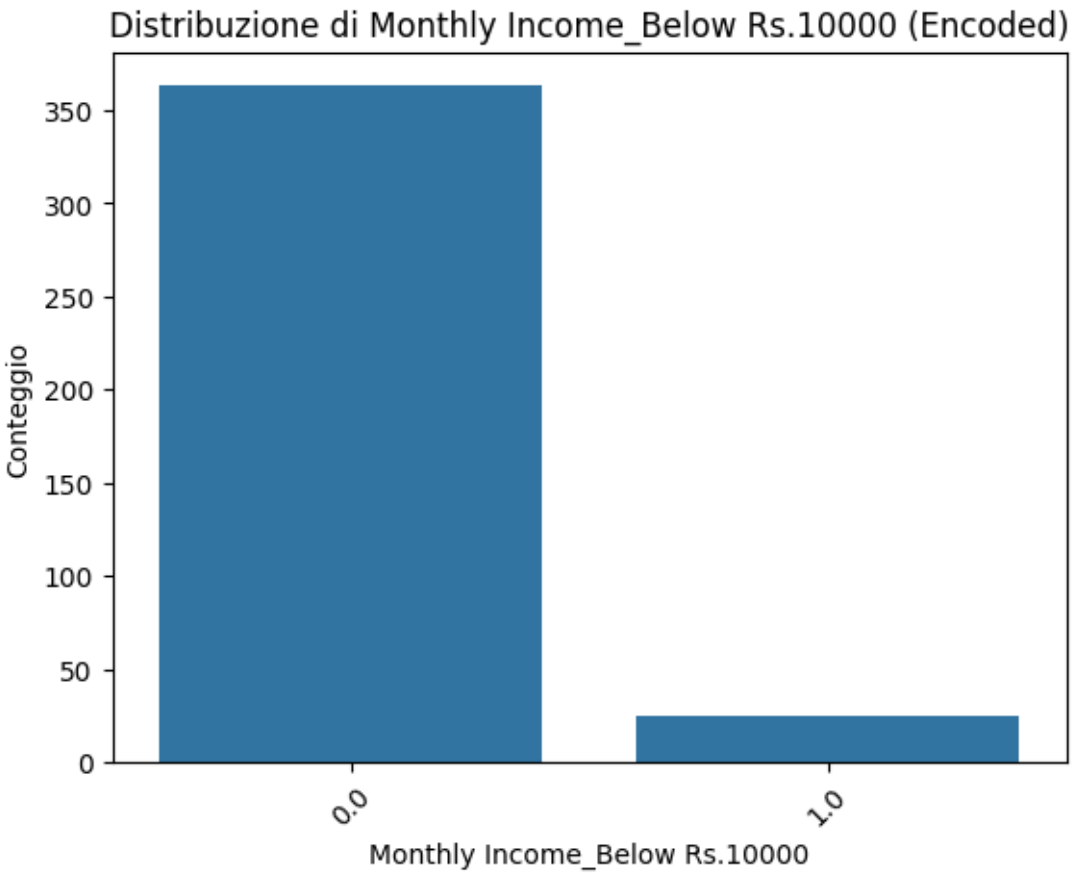

Distribuzione di Occupation_House wife (Encoded)
Conteggio
350
300
250
200
150
100
50
0
0.0
1.0
Occupation_House wife

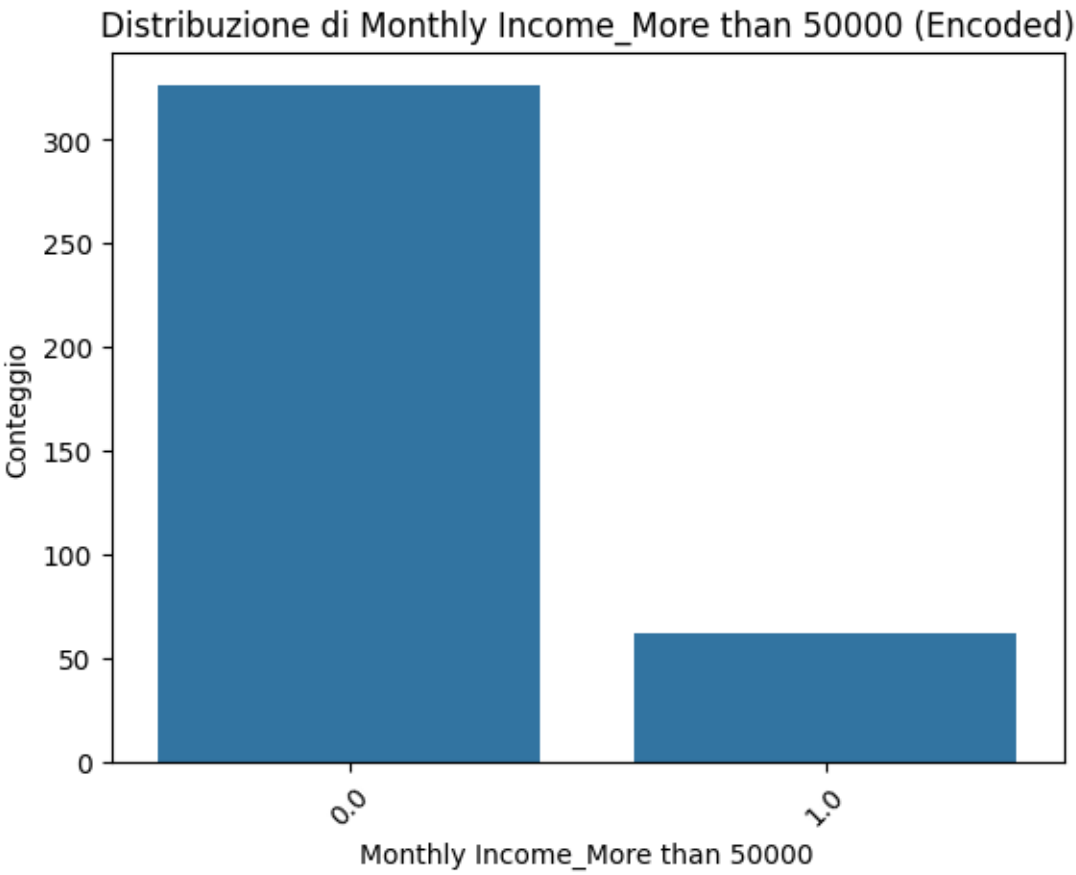

Distribuzione di Occupation_Self Employeed (Encoded)
Conteggio
350
300
250
200
150
100
50
0
0.0
1.0
Occupation_Self Employeed


Distribuzione di Occupation_Student (Encoded)
Conteggio
200
175
150
125
100
75
50
25
0
0.0
1.0
Occupation_Student

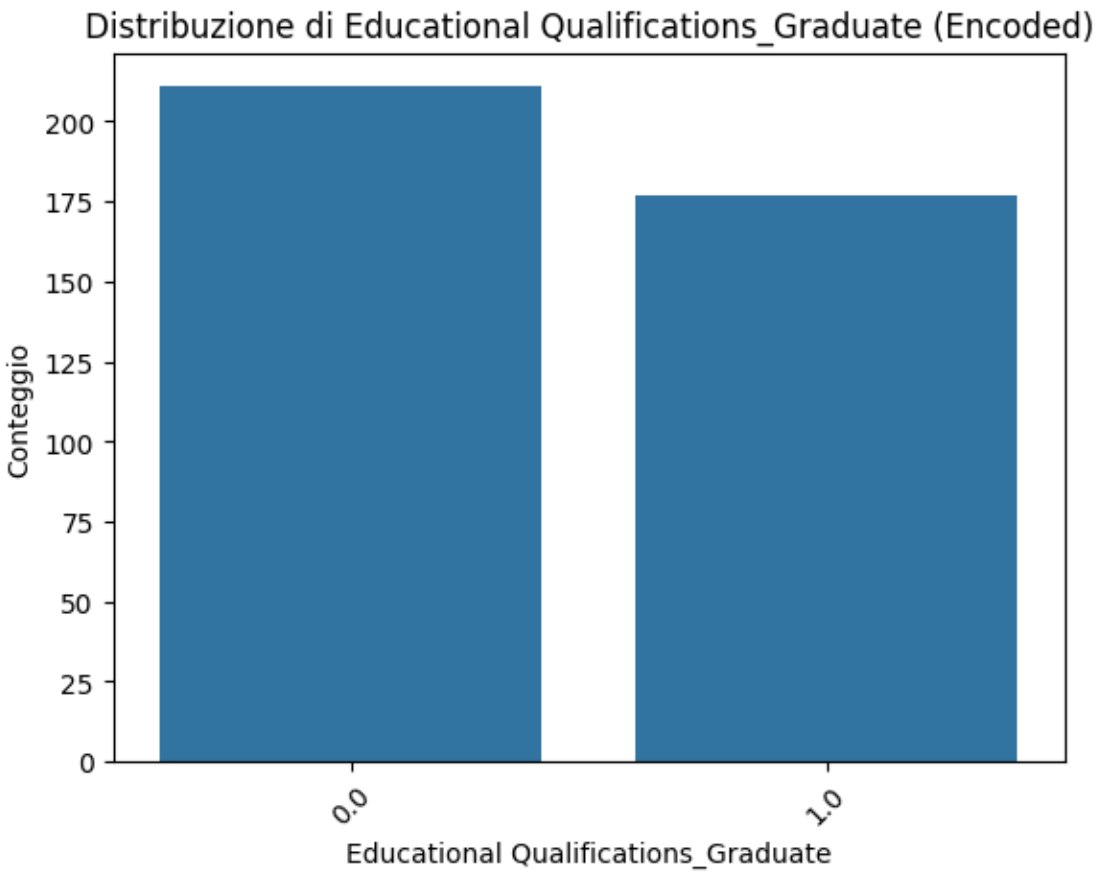

Distribuzione di Monthly Income_10001 to 25000 (Encoded)
Conteggio
350
300
250
200
150
100
50
0
0.0
1.0
Monthly Income_10001 to 25000

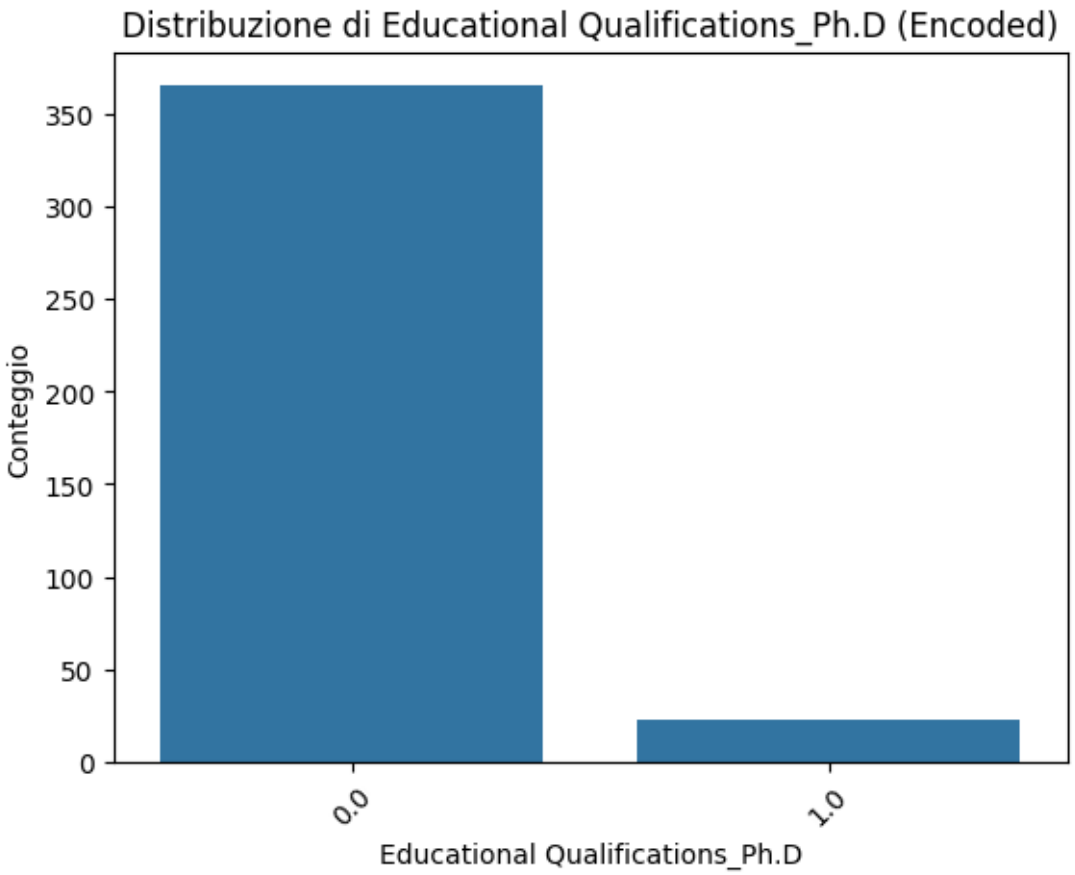

Distribuzione di Monthly Income_25001 to 50000 (Encoded)
Conteggio
300
250
200
150
100
50
0
0.0
1.0
Monthly Income_25001 to 50000


Distribuzione di Monthly Income_Below Rs.10000 (Encoded)
Conteggio
0
50
100
150
200
250
300
350
0.0
1.0
Monthly Income_Below Rs.10000

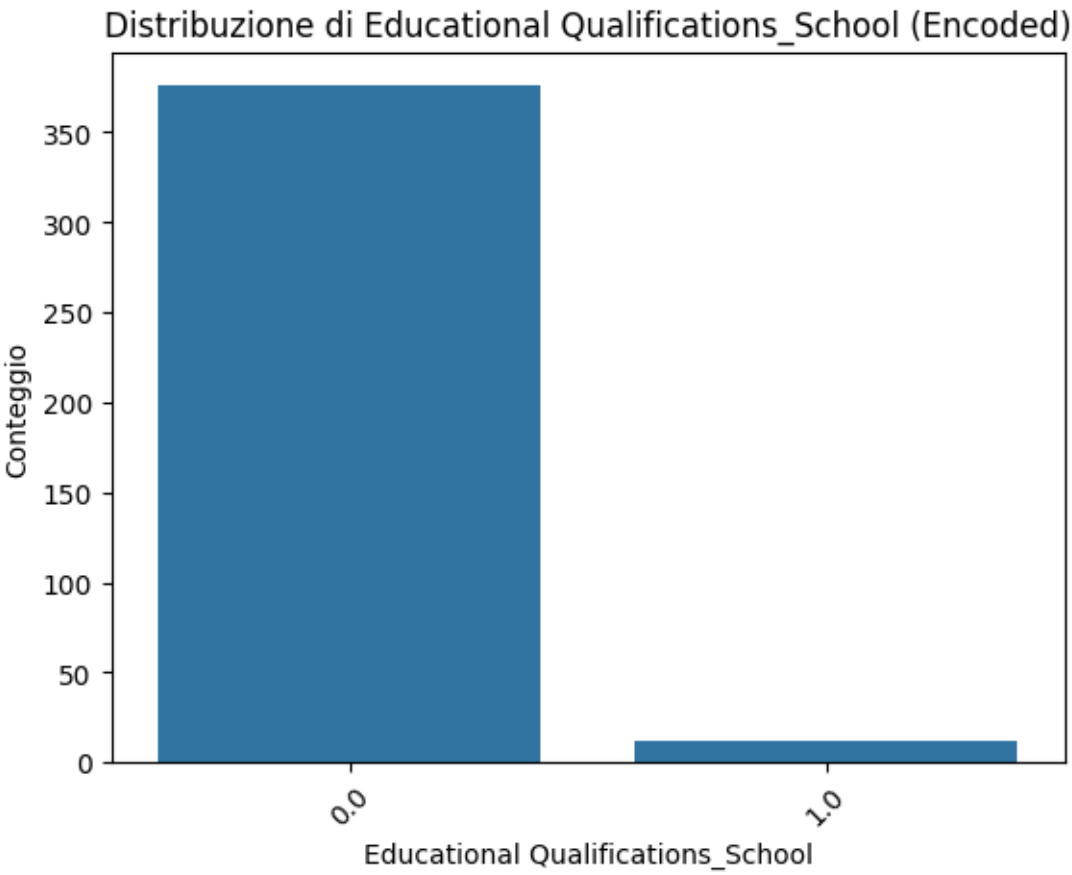

Distribuzione di Monthly Income_More than 50000 (Encoded)
Conteggio
300
250
200
150
100
50
0
0.0
1.0
Monthly Income_More than 50000

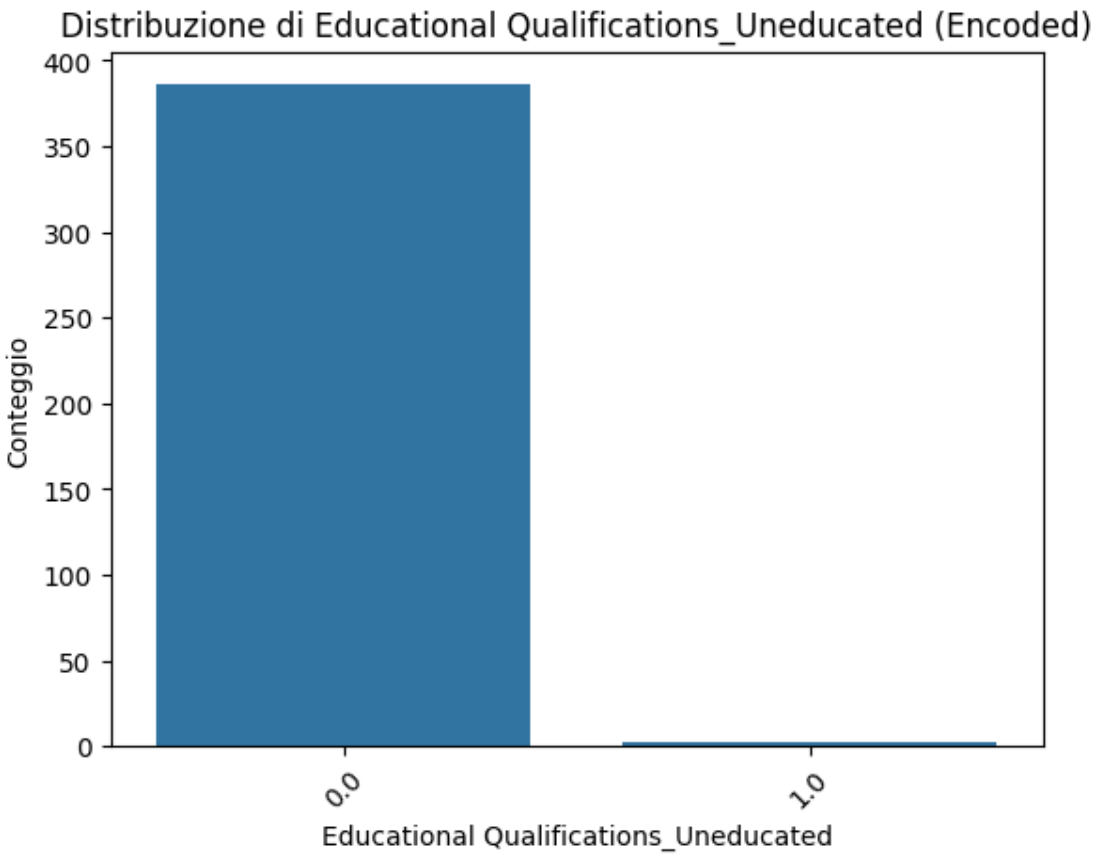

Distribuzione di Monthly Income_No Income (Encoded)
Conteggio
200
175
150
125
100
75
50
25
0
0.0
1.0
Monthly Income_No Income

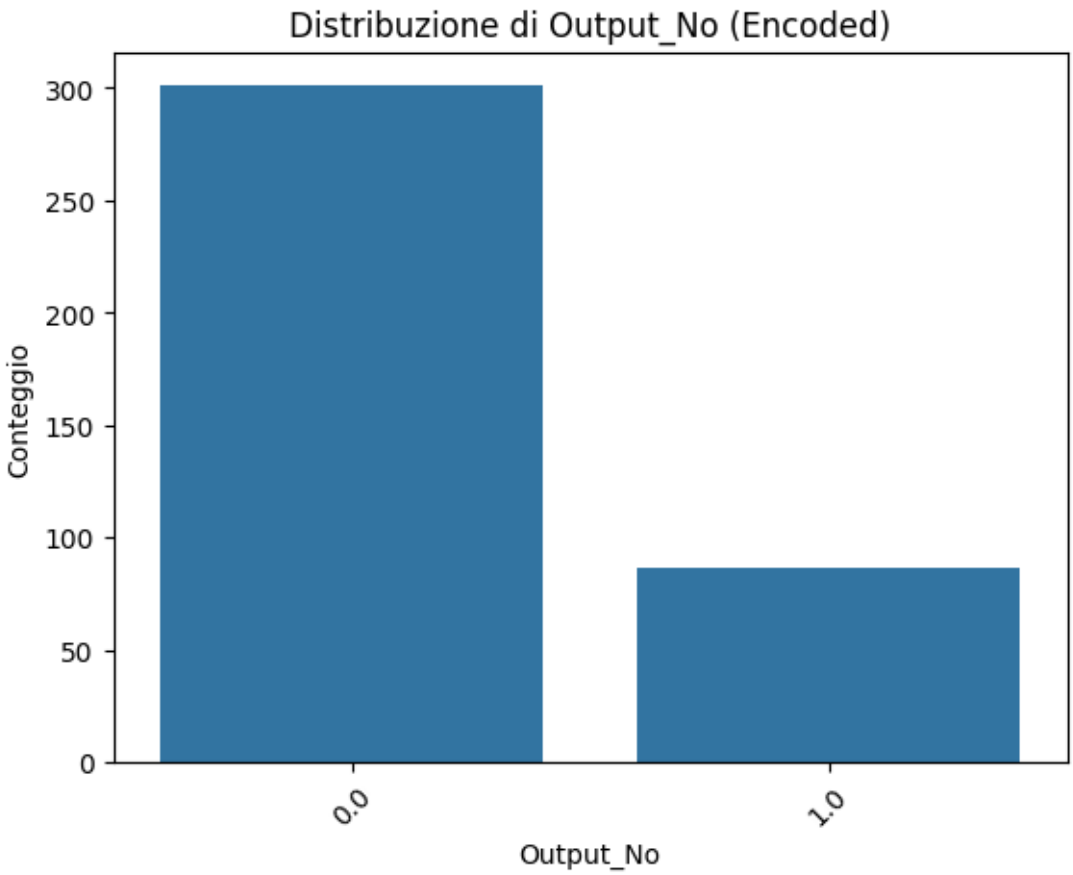

Distribuzione di Educational Qualifications_Graduate (Encoded)
Conteggio
200
175
150
125
100
75
50
25
0
0.0
1.0
Educational Qualifications_Graduate

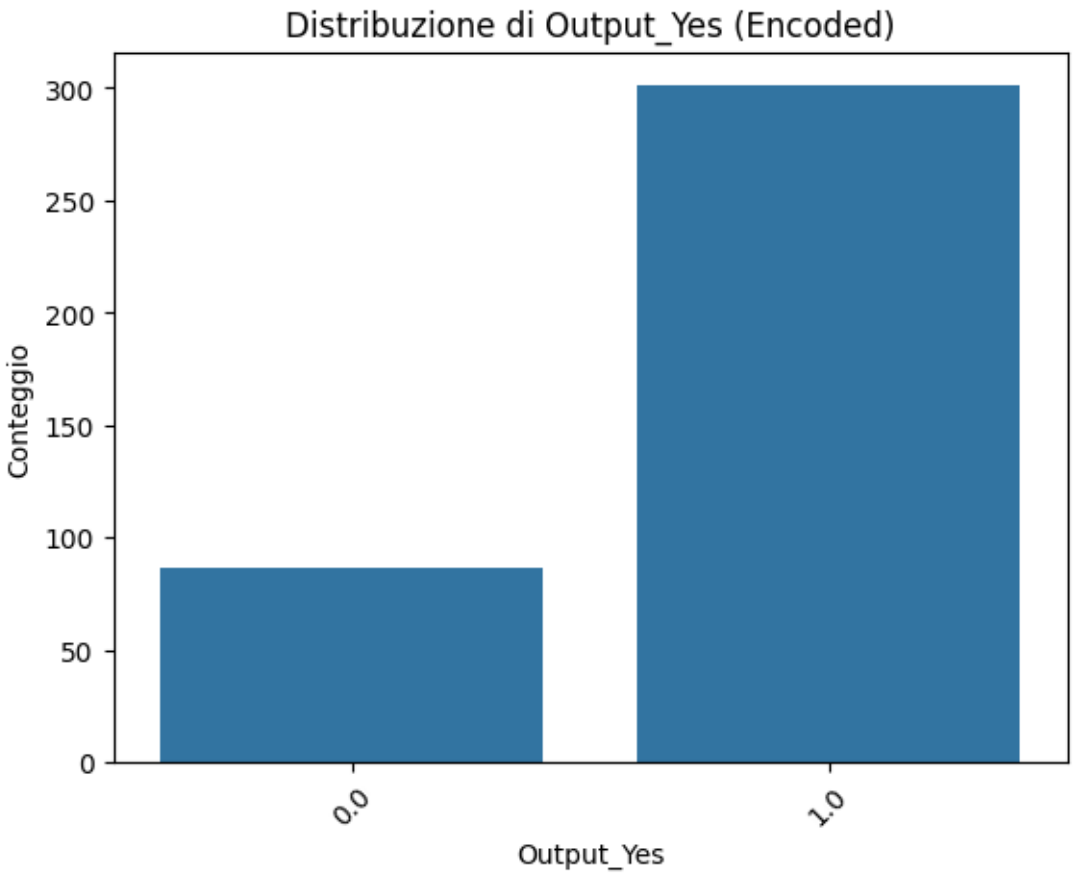

Distribuzione di Educational Qualifications_Ph.D (Encoded)
Conteggio
350
300
250
200
150
100
50
0
0.0
1.0
Educational Qualifications_Ph.D

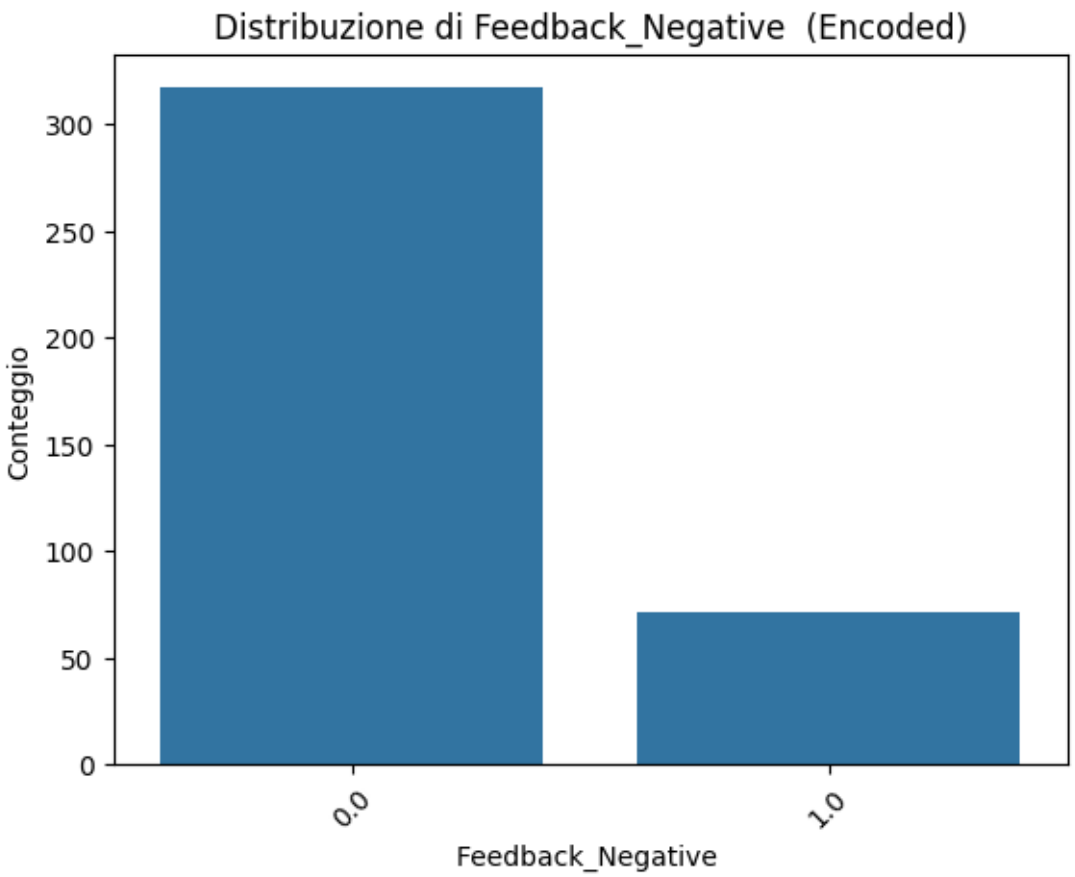

Distribuzione di Educational Qualifications_Post Graduate (Encoded)
Conteggio
0
25
50
75
100
125
150
175
200
0.0
1.0
Educational Qualifications_Post Graduate

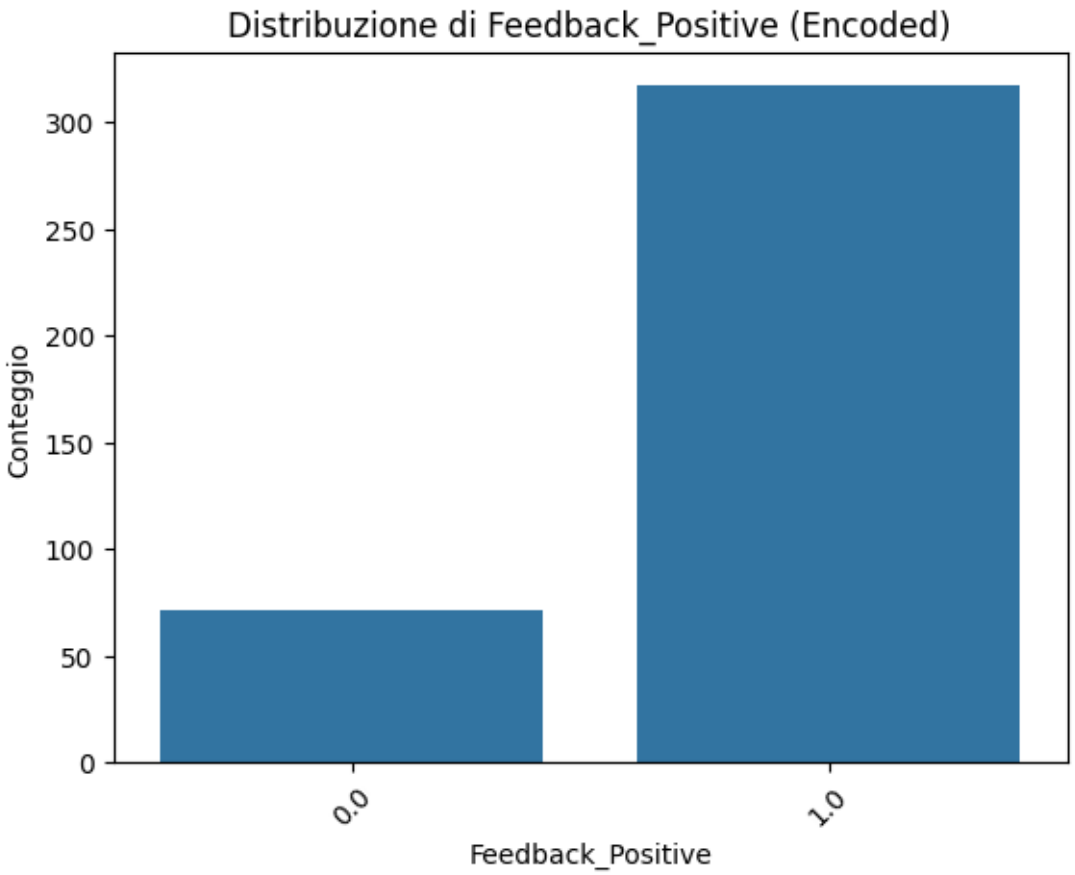

Distribuzione di Educational Qualifications_School (Encoded)
Conteggio
0
50
100
150
200
250
300
350
0.0
1.0
Educational Qualifications_School


Distribuzione di Educational Qualifications_Uneducated (Encoded)
Conteggio
400
350
300
250
200
150
100
50
0
0.0
1.0
Educational Qualifications_Uneducated


Distribuzione di Output_No (Encoded)
Conteggio
300
250
200
150
100
50
0
0.0
1.0
Output_No


Distribuzione di Output_Yes (Encoded)
Conteggio
300
250
200
150
100
50
0
0.0
1.0
Output_Yes


Distribuzione di Feedback_Negative (Encoded)
Conteggio
300
250
200
150
100
50
0
0.0
1.0
Feedback_Negative



[ ]:
```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report

# Definisci la variabile target
target_variable = df1['Feedback']

# Rimuovi la variabile target dal DataFrame codificato
X = df2_encoded
y = target_variable

# Suddivisione dei dati in training e testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# Inizializza il classificatore Random Forest
rf_classifier = RandomForestClassifier(random_state=42)

# Addestra il modello sul set di dati di addestramento
```

```
rf_classifier.fit(X_train, y_train)

# Effettua le predizioni sul set di dati di test
y_pred = rf_classifier.predict(X_test)

# Valuta le prestazioni del modello
accuracy = accuracy_score(y_test, y_pred)
classification_rep = classification_report(y_test, y_pred)

# Visualizza l'accuratezza e il report di classificazione
print("Accuratezza del modello:", accuracy)
print("\nReport di classificazione:\n", classification_rep)
```

```
Accuratezza del modello: 1.0

Report di classificazione:
               precision    recall  f1-score   support

   Negative        1.00      1.00      1.00        11
    Positive       1.00      1.00      1.00        67

    accuracy                           1.00        78
   macro avg       1.00      1.00      1.00        78
weighted avg       1.00      1.00      1.00        78
```

[ ]:
```
# Prendi una riga casuale dal dataset di test
sample_row = X_test.sample(n=1, random_state=42)

# Utilizza il modello addestrato per prevedere la variabile target per il dato␣
 ↪di input
predicted_class = rf_classifier.predict(sample_row)

# Visualizza la riga e la previsione del modello
print("Riga di input:")
print(sample_row)
print("\nPrevisto:", predicted_class)
```

```
Riga di input:
     Gender_Female  Gender_Male  Marital Status_Married  \
296            0.0          1.0                     0.0

     Marital Status_Prefer not to say  Marital Status_Single  \
296                               0.0                    1.0

     Occupation_Employee  Occupation_House wife  Occupation_Self Employeed  \
296                  0.0                    0.0                        0.0
```

```
     Occupation_Student  Monthly Income_10001 to 25000  …  \
296                 1.0                            0.0  …

     Monthly Income_No Income  Educational Qualifications_Graduate  \
296                       1.0                                  0.0

     Educational Qualifications_Ph.D  \
296                              0.0

     Educational Qualifications_Post Graduate  \
296                                       1.0

     Educational Qualifications_School  Educational Qualifications_Uneducated  \
296                                0.0                                     0.0

     Output_No  Output_Yes  Feedback_Negative   Feedback_Positive
296        0.0         1.0                 0.0                 1.0

[1 rows x 23 columns]

Previsto: ['Positive']
```