Target encoding

Target encoding (also called **mean encoding**) is a technique used to encode **categorical variables** by replacing each category with a **numeric value based on the target variable** — typically the **mean** of the target for each category.

Example: Suppose you're predicting customer churn (binary target: 1 if churned, 0 otherwise), and one of the categorical feature is City:

City	age	churn
New York	23	1
New York	33	0
Delhi	24	0
Delhi	44	1
Delhi	52	1
Chicago	37	0

Compute the mean churn rate for each city:

City	Mean Churn Rate
New York	(1+0)/2 = 0.5
Delhi	(0+1+1)/3 = 0.6667
Chicago	0/1 = 0

Replace city names with mean

City_encoded	age	churn
0.5	23	1
0.5	33	0
0.6667	24	0
0.6667	44	1
0.6667	52	1
0	37	0

Target encoding

It has risks of data leakage:
 If you compute target encoding using the **entire dataset**, information from the test set leaks into training: Solution is, use mean of training data.

City	age	churn
New York	23	1
New York	33	0
Delhi	24	0
Delhi	44	1
Delhi	52	1
Chicago	37	0

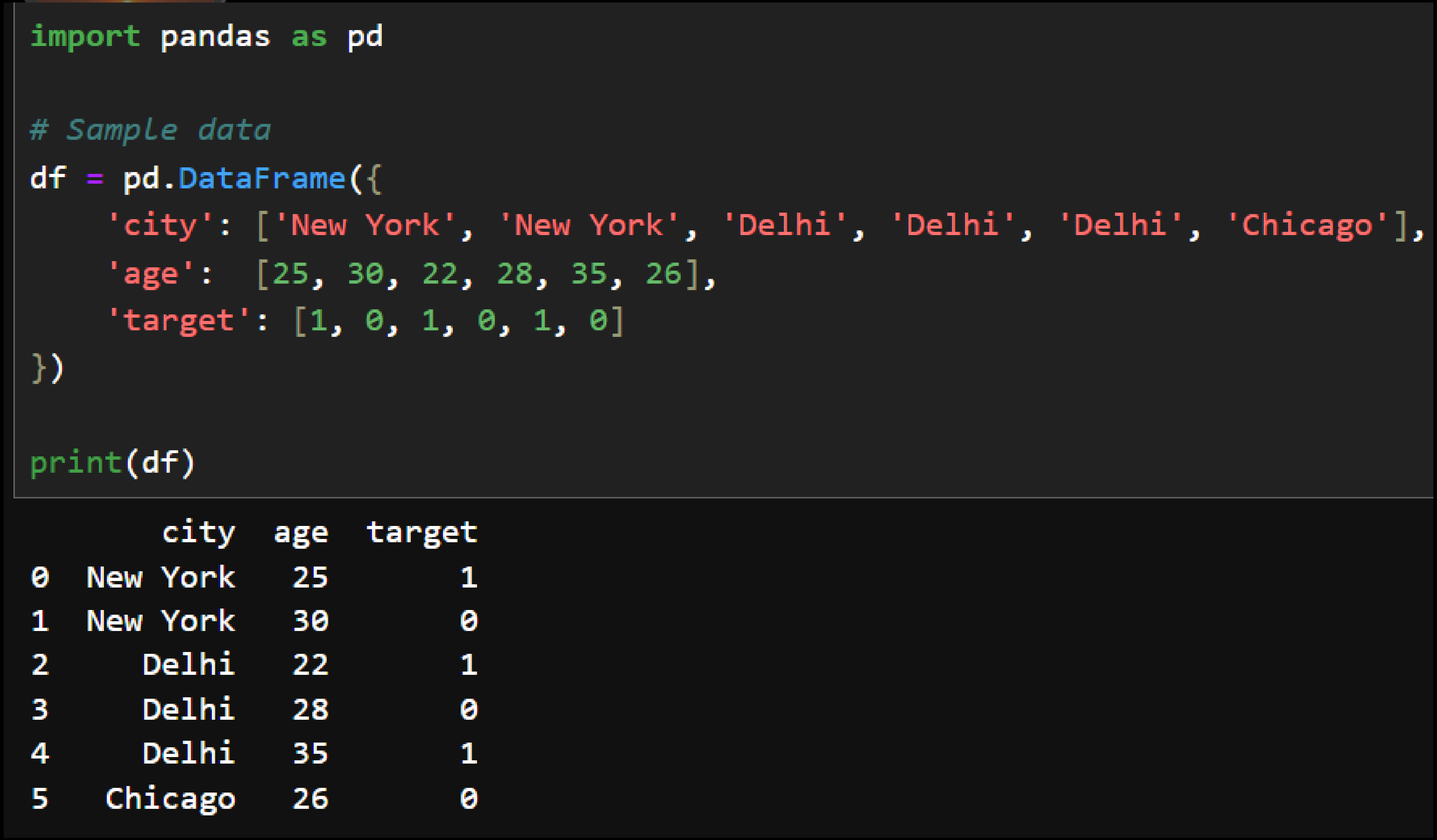
Compute the mean churn rate for each city:

City	Mean Churn Rate
New York	(1+0)/2 = 0.5
Delhi	(0+1+1)/3 = 0.6667
Chicago	0

Replace city names with mean

City_encoded	age	churn
0.5	23	1
0.5	33	0
0.6667	24	0
0.6667	44	1
0.6667	52	1
1	37	0

Target encoding: Python code



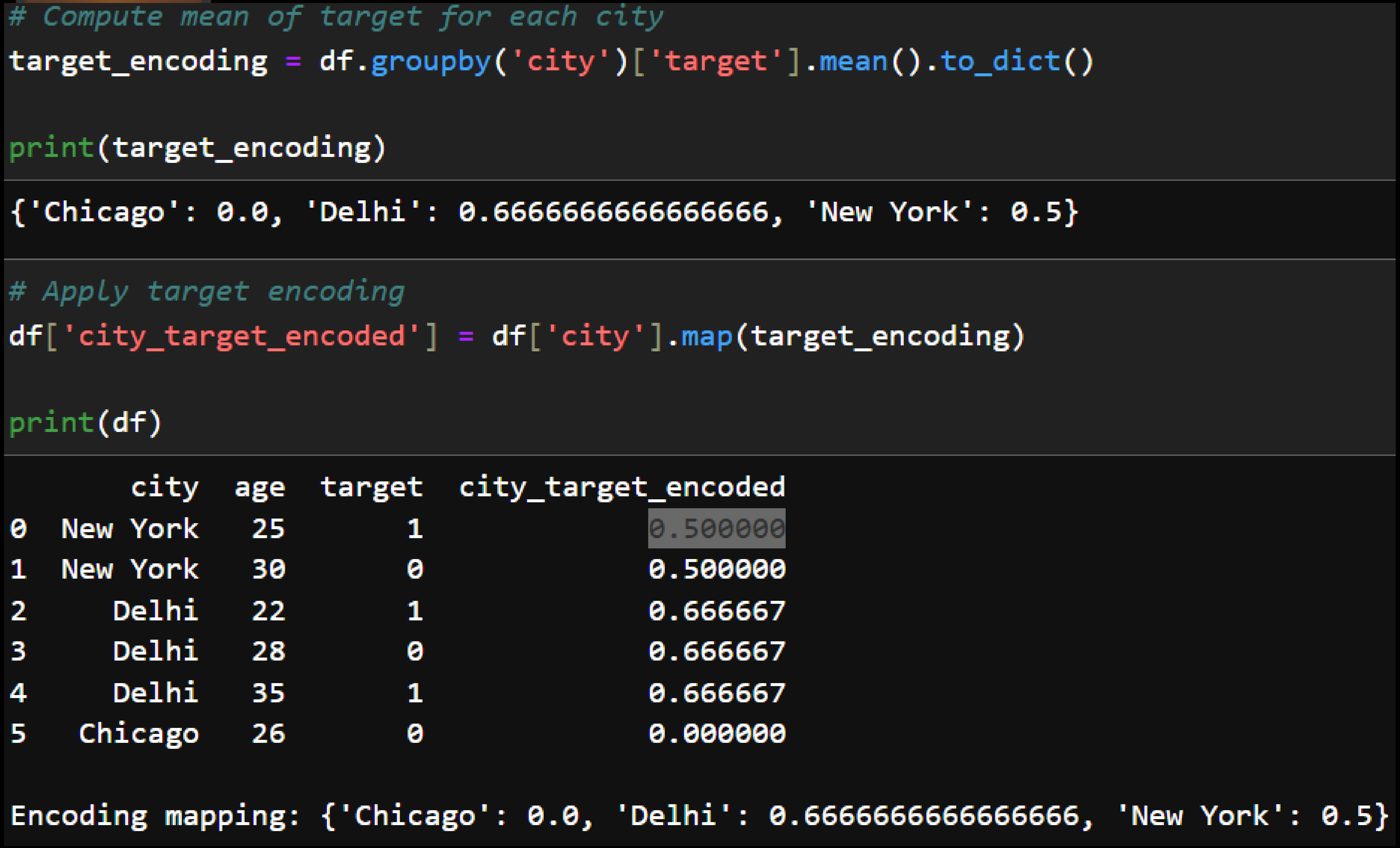
```python
import pandas as pd

# Sample data
df = pd.DataFrame({
    'city': ['New York', 'New York', 'Delhi', 'Delhi', 'Delhi', 'Chicago'],
    'age':  [25, 30, 22, 28, 35, 26],
    'target': [1, 0, 1, 0, 1, 0]
})

print(df)
```

```
       city  age  target
0  New York   25       1
1  New York   30       0
2     Delhi   22       1
3     Delhi   28       0
4     Delhi   35       1
5   Chicago   26       0
```

Target encoding: Python code



```python
# Compute mean of target for each city
target_encoding = df.groupby('city')['target'].mean().to_dict()

print(target_encoding)
```

```
{'Chicago': 0.0, 'Delhi': 0.6666666666666666, 'New York': 0.5}
```

```python
# Apply target encoding
df['city_target_encoded'] = df['city'].map(target_encoding)

print(df)
```

```
       city  age  target  city_target_encoded
0  New York   25       1             0.500000
1  New York   30       0             0.500000
2     Delhi   22       1             0.666667
3     Delhi   28       0             0.666667
4     Delhi   35       1             0.666667
5   Chicago   26       0             0.000000

Encoding mapping: {'Chicago': 0.0, 'Delhi': 0.6666666666666666, 'New York': 0.5}
```
]

Target encoding

When to use target encoding ?

- When the categorical variable is **high-cardinality** (many unique categories)
- Common with **tree-based models**

Frequency encoding

Uses the frequency of each category as its value.

city	age	churn
Miami	23	1
Miami	30	0
Delhi	45	0
Delhi	34	1
Delhi	43	1
Chicago	33	0

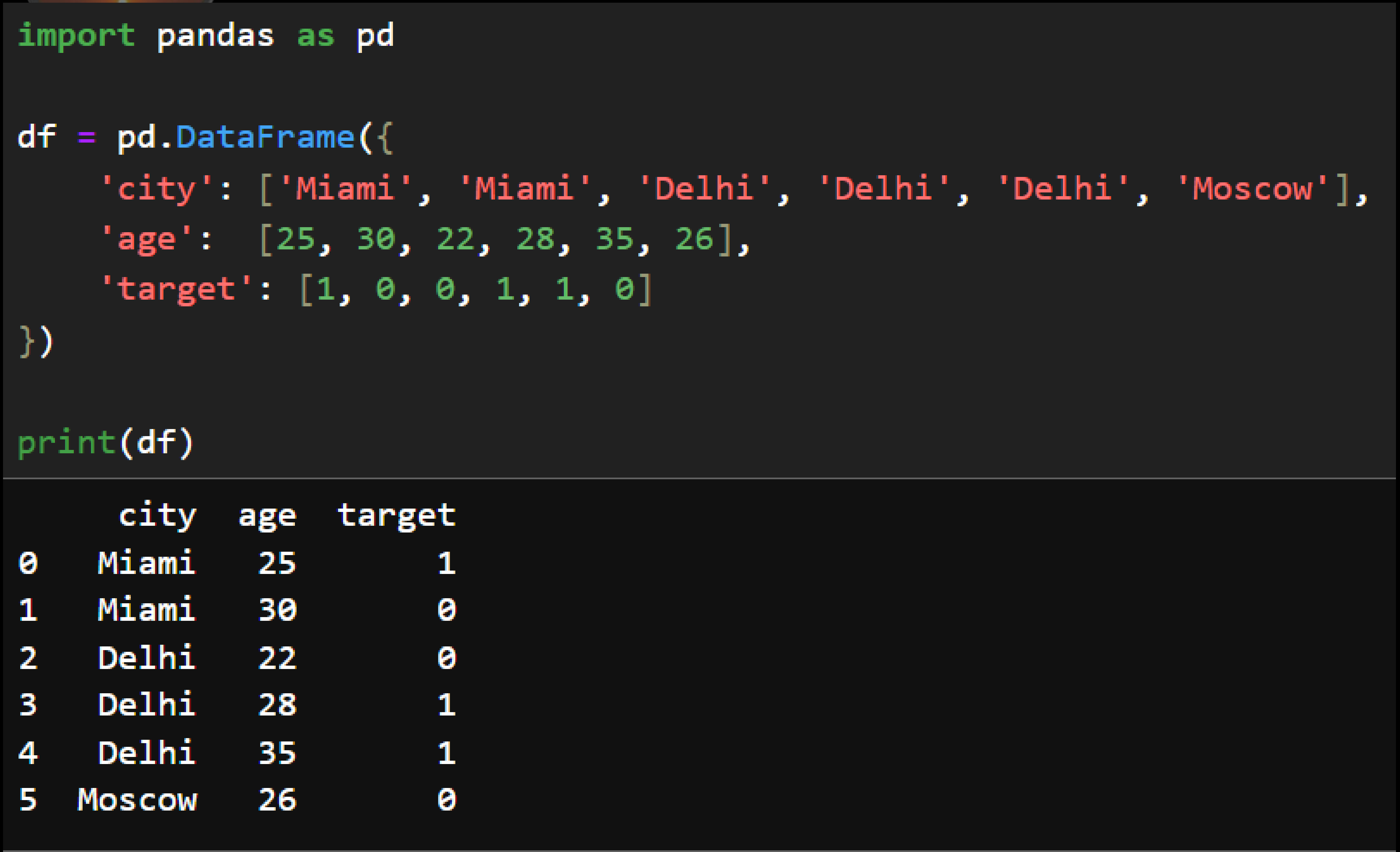
Compute the frequency for each city

city	frequency
Miami	2
Delhi	3
Chicago	1

Replace city with its frequency

city_encoded	age	churn
2	23	1
2	30	0
3	45	0
3	34	1
3	43	1
1	33	0

Frequency encoding: Python code



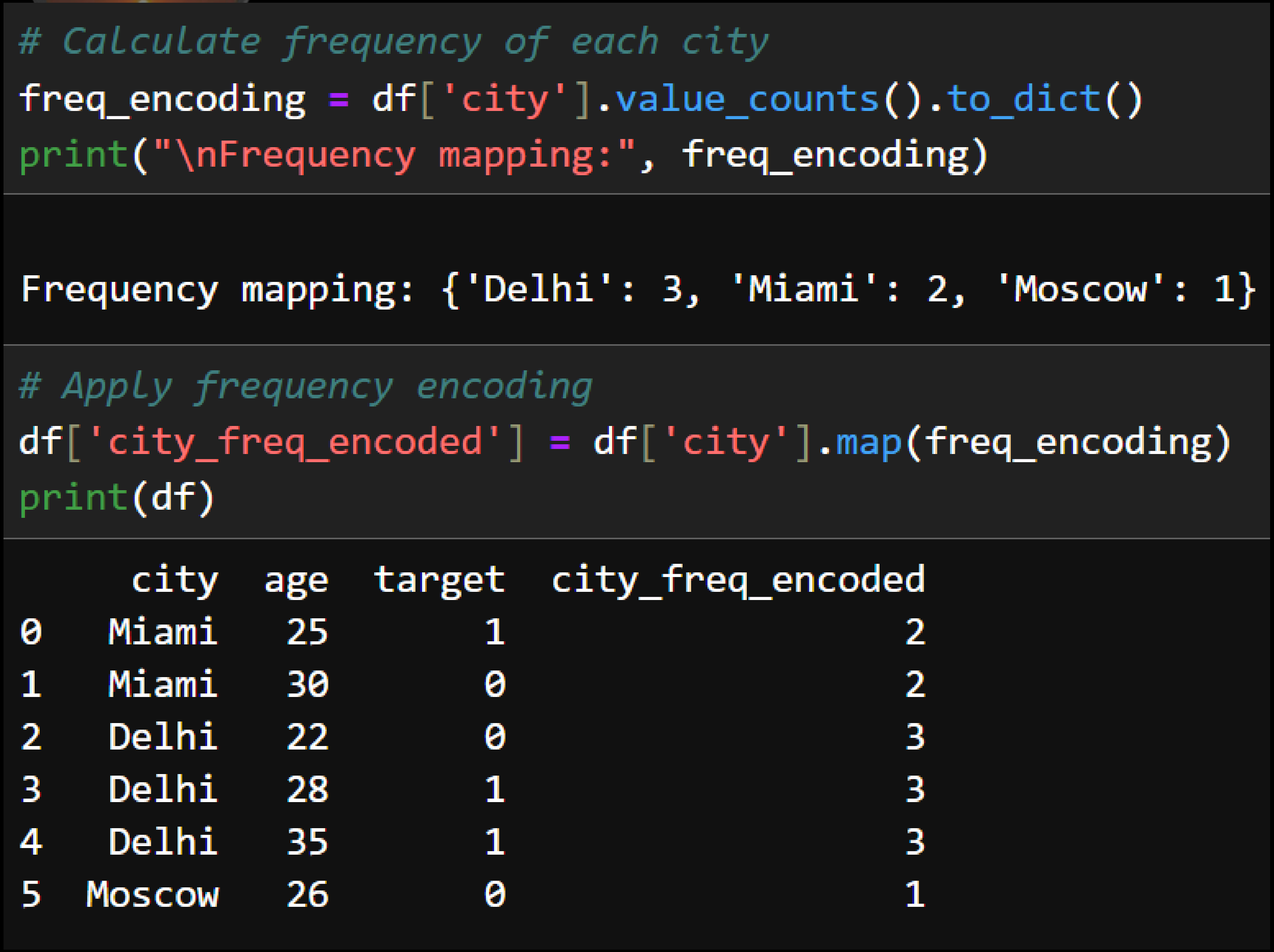
```python
import pandas as pd

df = pd.DataFrame({
    'city': ['Miami', 'Miami', 'Delhi', 'Delhi', 'Delhi', 'Moscow'],
    'age':  [25, 30, 22, 28, 35, 26],
    'target': [1, 0, 0, 1, 1, 0]
})

print(df)
```

```
     city  age  target
0   Miami   25       1
1   Miami   30       0
2   Delhi   22       0
3   Delhi   28       1
4   Delhi   35       1
5  Moscow   26       0
```

Frequency encoding: Python code



```python
# Calculate frequency of each city
freq_encoding = df['city'].value_counts().to_dict()
print("\nFrequency mapping:", freq_encoding)
```

```

Frequency mapping: {'Delhi': 3, 'Miami': 2, 'Moscow': 1}
```

```python
# Apply frequency encoding
df['city_freq_encoded'] = df['city'].map(freq_encoding)
print(df)
```

```
     city  age  target  city_freq_encoded
0   Miami   25       1                  2
1   Miami   30       0                  2
2   Delhi   22       0                  3
3   Delhi   28       1                  3
4   Delhi   35       1                  3
5  Moscow   26       0                  1
```

Heading Goes Here

Fhdsklf
Fjdsklf
Fjskldf