Instructor Do: Dealing with Categorical Data in ML

In [33]:
```
# initial imports
import pandas as pd
from path import Path
```

Dataset Information

The file `loans_data.csv` , contains simulated data about loans, there are a total of 500 records. Each row represents a loan application along an arbitrary year, where every column represents the following data about every loan application.

- `amount` : The loan amount in USD.
- `term` : The loan term in months.
- `month` : The month of the year when the loan was requested.
- `age` : Age of the loan applicant.
- `education` : Educational level of the loan applicant.
- `gender` : Gender of the loan applicant.
- `bad` : Stands for a bad or good loan applicant (`1` - bad, `0` - good).

In [34]:
```
# Load data
file_path = Path("../Resources/loans_data.csv")
loans_df = pd.read_csv(file_path)
loans_df.head(10)
```

Out[34]:

	amount	term	month	age	education	gender	bad
0	1000	30	June	45	High School or Below	male	0
1	1000	30	July	50	Bachelor	female	0
2	1000	30	August	33	Bachelor	female	0
3	1000	15	September	27	college	male	0
4	1000	30	October	28	college	female	0
5	300	7	July	35	Master or Above	male	0
6	1000	30	September	29	college	male	0
7	1000	30	May	36	college	male	0
8	1000	30	May	28	college	male	0
9	800	15	April	26	college	male	0

In [35]:
```
# Binary encoding using Pandas (single column)
loans_binary_encoded = pd.get_dummies(loans_df, columns=["gender"])
loans_binary_encoded.head()
```

Out[35]:

	amount	term	month	age	education	bad	gender_female	gender_male
0	1000	30	June	45	High School or Below	0	0	1
1	1000	30	July	50	Bachelor	0	1	0
2	1000	30	August	33	Bachelor	0	1	0
3	1000	15	September	27	college	0	0	1
4	1000	30	October	28	college	0	1	0

In [36]:
```
# Binary encoding using Pandas (multiple columns)
loans_binary_encoded = pd.get_dummies(loans_df, columns=["education", "gend
loans_binary_encoded.head()
```

Out[36]:

	amount	term	month	age	bad	education_Bachelor	education_High School or Below	education_Master or Above	edu
0	1000	30	June	45	0	0	1	0	
1	1000	30	July	50	0	1	0	0	
2	1000	30	August	33	0	1	0	0	
3	1000	15	September	27	0	0	0	0	
4	1000	30	October	28	0	0	0	0	

In [37]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df2 = loans_df.copy()
df2['education'] = le.fit_transform(df2['education'])
df2.head()
```

Out[37]:

	amount	term	month	age	education	gender	bad
0	1000	30	June	45	1	male	0
1	1000	30	July	50	0	female	0
2	1000	30	August	33	0	female	0
3	1000	15	September	27	3	male	0
4	1000	30	October	28	3	female	0

Integer Encoding

In [38]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df2 = loans_df.copy()
df2['gender'] = le.fit_transform(df2['gender'])
```

In [39]:
```
df2.head()
```

Out[39]:

	amount	term	month	age	education	gender	bad
0	1000	30	June	45	High School or Below	1	0
1	1000	30	July	50	Bachelor	0	0
2	1000	30	August	33	Bachelor	0	0
3	1000	15	September	27	college	1	0
4	1000	30	October	28	college	0	0

Custom Encoding

In [40]:
```
# Creating an instance of label encoder
label_encoder = LabelEncoder()
loans_df["month_le"] = label_encoder.fit_transform(loans_df["month"])
loans_df.head()
```

Out[40]:

	amount	term	month	age	education	gender	bad	month_le
0	1000	30	June	45	High School or Below	male	0	6
1	1000	30	July	50	Bachelor	female	0	5
2	1000	30	August	33	Bachelor	female	0	1
3	1000	15	September	27	college	male	0	11
4	1000	30	October	28	college	female	0	10

In [41]:
```
# Months dictionary
months_num = {
    "January": 1,
    "February": 2,
    "March": 3,
    "April": 4,
    "May": 5,
    "June": 6,
    "July": 7,
    "August": 8,
    "September": 9,
    "October": 10,
    "November": 11,
    "December": 12,
}
```

In [42]:
```
# Months' names encoded using the dictionary values
loans_df["month_num"] = loans_df["month"].apply(lambda x: months_num[x])
loans_df.head()
```

Out[42]:

	amount	term	month	age	education	gender	bad	month_le	month_num
0	1000	30	June	45	High School or Below	male	0	6	6
1	1000	30	July	50	Bachelor	female	0	5	7
2	1000	30	August	33	Bachelor	female	0	1	8
3	1000	15	September	27	college	male	0	11	9
4	1000	30	October	28	college	female	0	10	10

In [43]:
```
# Drop the month and month_le columns
loans_df = loans_df.drop(["month", "month_le"], axis=1)
loans_df.head()
```

Out[43]:

	amount	term	age	education	gender	bad	month_num
0	1000	30	45	High School or Below	male	0	6
1	1000	30	50	Bachelor	female	0	7
2	1000	30	33	Bachelor	female	0	8
3	1000	15	27	college	male	0	9
4	1000	30	28	college	female	0	10

In [ ]:

In [ ]:

In [ ]: