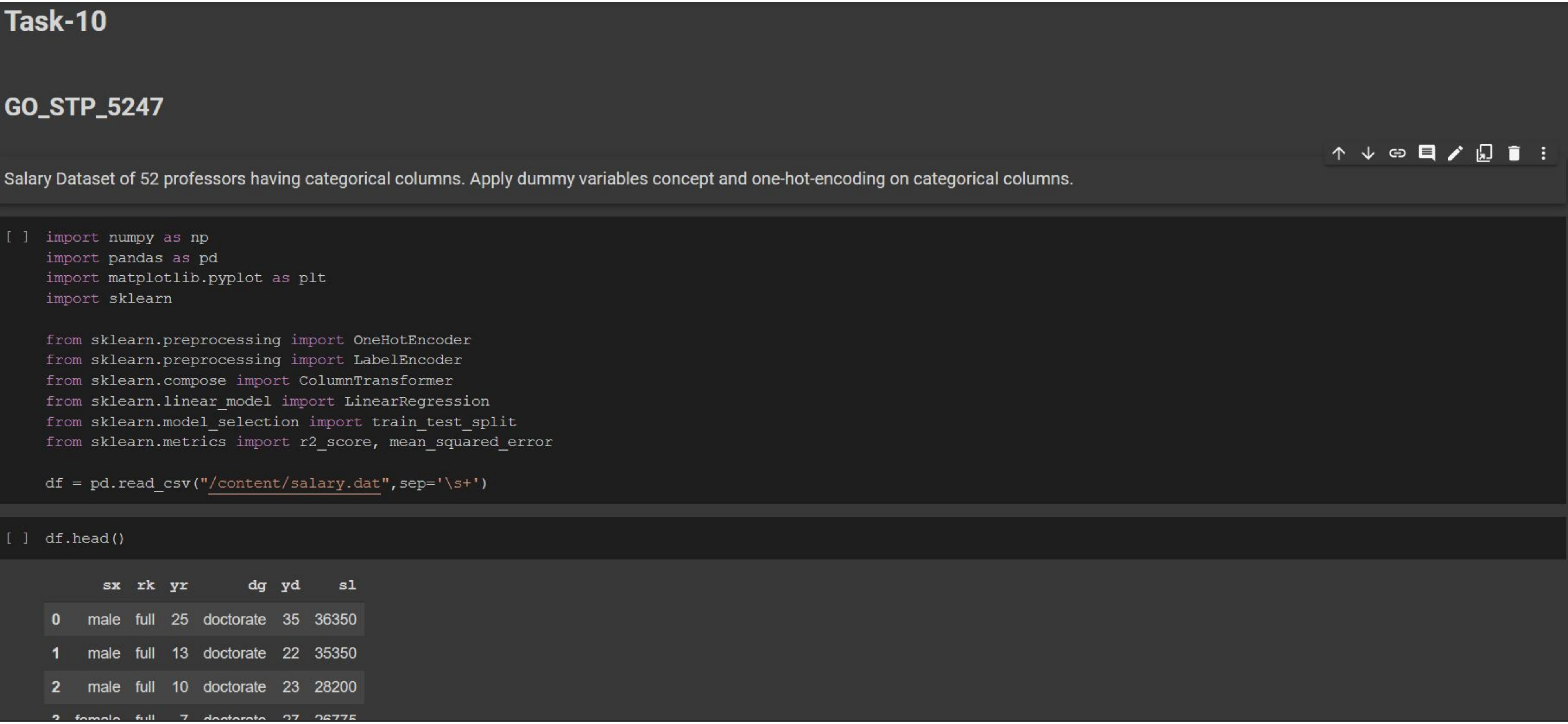


Task-10

GO_STP_5247

Salary Dataset of 52 professors having categorical columns. Apply dummy variables concept and one-hot-encoding on categorical columns.

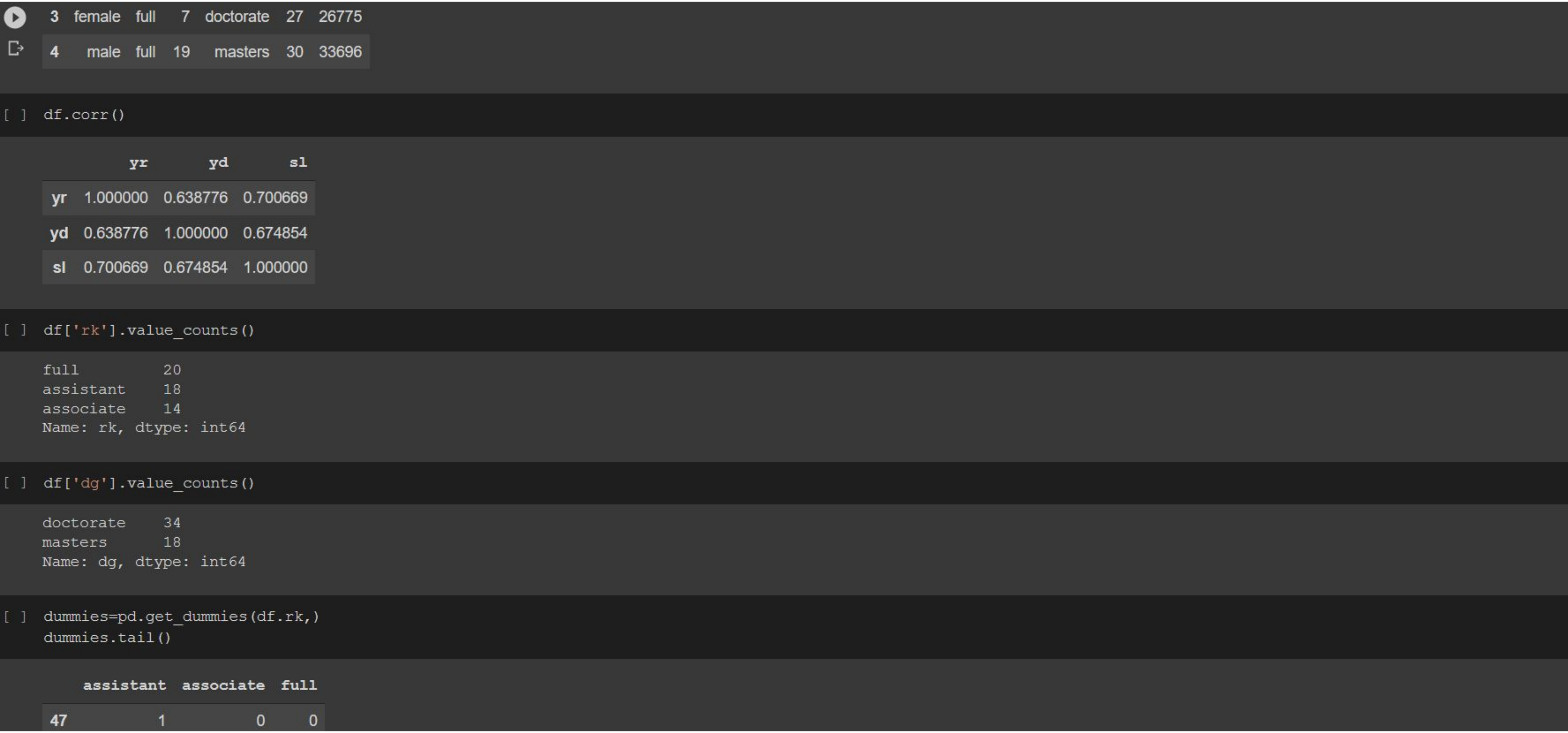
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import sklearn

from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import LabelEncoder
from sklearn.compose import ColumnTransformer
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score, mean_squared_error

df = pd.read_csv("/content/salary.dat",sep='\s+')
```

```
df.head()
```

	sx	rk	yr	dg	yd	sl
0	male	full	25	doctorate	35	36350
1	male	full	13	doctorate	22	35350
2	male	full	10	doctorate	23	28200
3	female	full	7	doctorate	27	26775



3	female	full	7	doctorate	27	26775
4	male	full	19	masters	30	33696

```
[ ] df.corr()
```

	yr	yd	sl
yr	1.000000	0.638776	0.700669
yd	0.638776	1.000000	0.674854
sl	0.700669	0.674854	1.000000

```
[ ] df['rk'].value_counts()
```

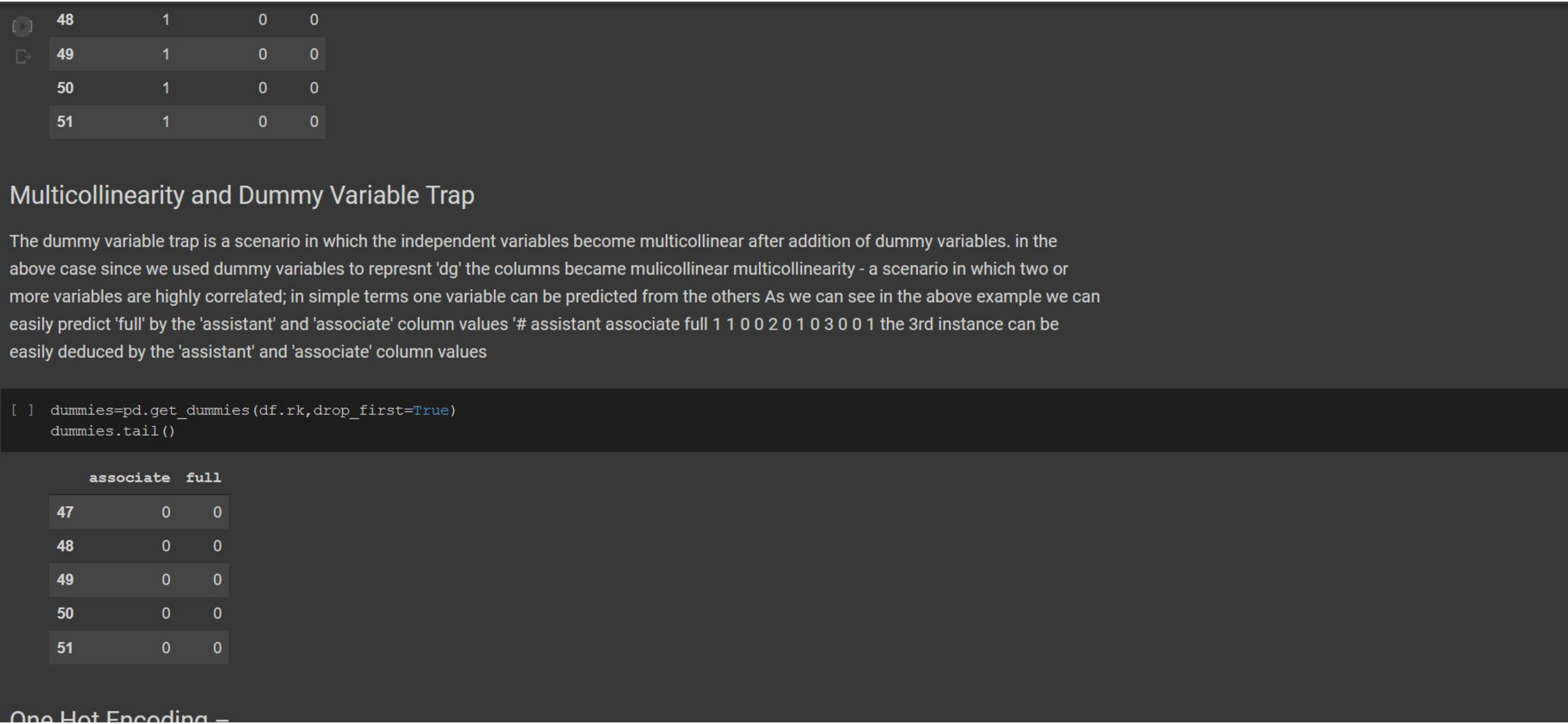
```
full         20
assistant    18
associate    14
Name: rk, dtype: int64
```

```
[ ] df['dg'].value_counts()
```

```
doctorate    34
masters      18
Name: dg, dtype: int64
```

```
[ ] dummies=pd.get_dummies(df.rk,)
    dummies.tail()
```

	assistant	associate	full
47	1	0	0



48	1	0	0
49	1	0	0
50	1	0	0
51	1	0	0

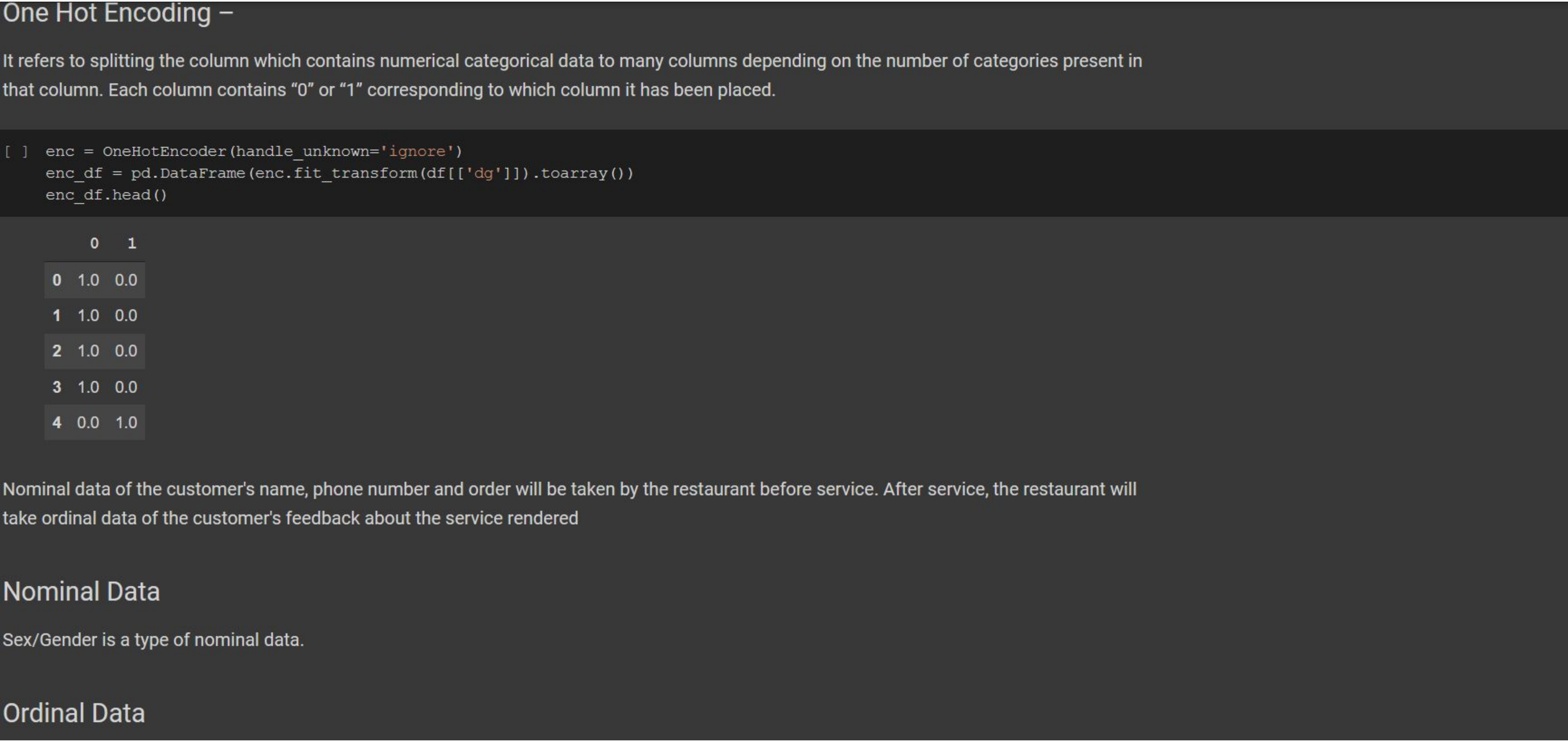
Multicollinearity and Dummy Variable Trap

The dummy variable trap is a scenario in which the independent variables become multicollinear after addition of dummy variables. in the above case since we used dummy variables to represnt 'dg' the columns became mulicollinear multicollinearity - a scenario in which two or more variables are highly correlated; in simple terms one variable can be predicted from the others As we can see in the above example we can easily predict 'full' by the 'assistant' and 'associate' column values '# assistant associate full 1 1 0 0 2 0 1 0 3 0 0 1 the 3rd instance can be easily deduced by the 'assistant' and 'associate' column values

```
[ ]  dummies=pd.get_dummies(df.rk,drop_first=True)
     dummies.tail()
```

	associate	full
47	0	0
48	0	0
49	0	0
50	0	0
51	0	0

One Hot Encoding –



One Hot Encoding –

It refers to splitting the column which contains numerical categorical data to many columns depending on the number of categories present in that column. Each column contains “0” or “1” corresponding to which column it has been placed.

```
[ ]  enc = OneHotEncoder(handle_unknown='ignore')
     enc_df = pd.DataFrame(enc.fit_transform(df[['dg']]).toarray())
     enc_df.head()
```

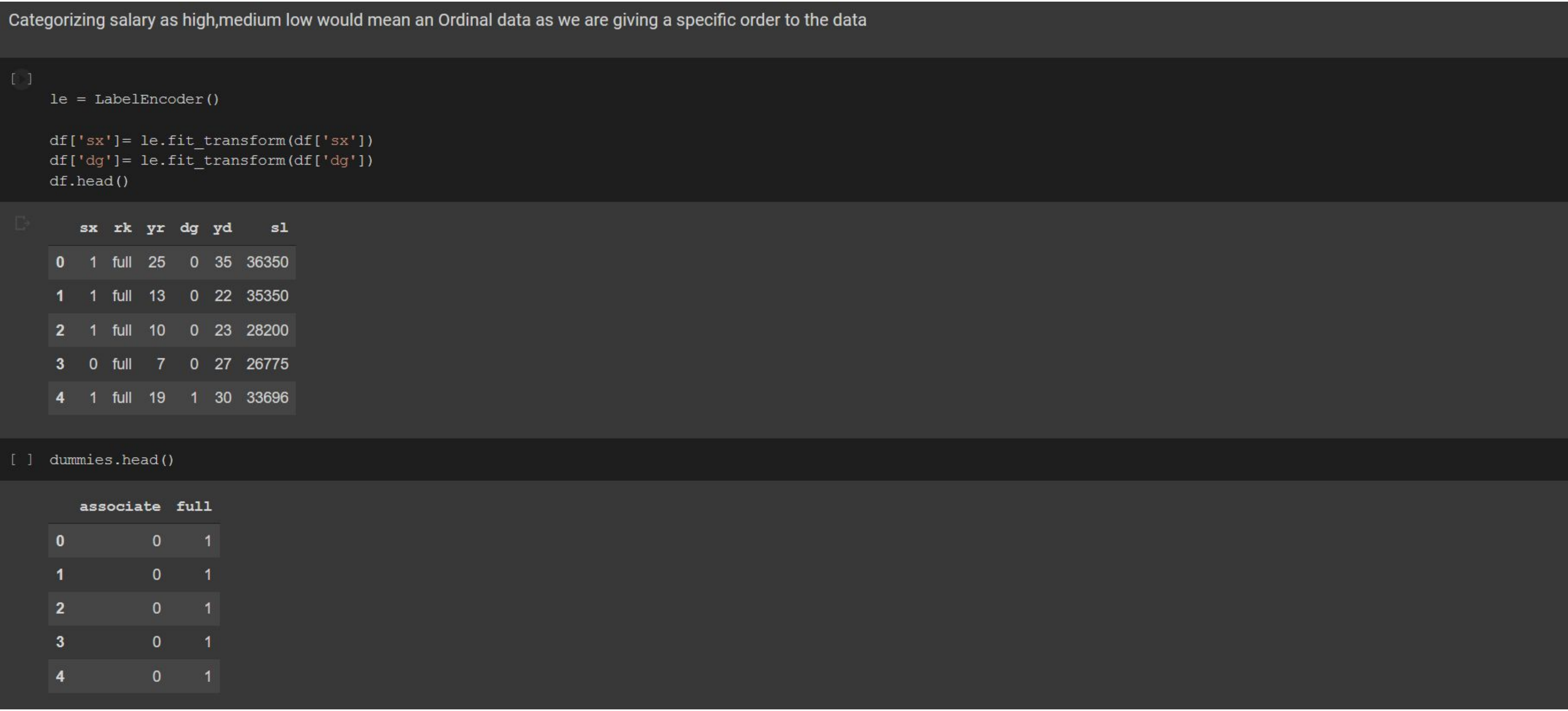
	0	1
0	1.0	0.0
1	1.0	0.0
2	1.0	0.0
3	1.0	0.0
4	0.0	1.0

Nominal data of the customer's name, phone number and order will be taken by the restaurant before service. After service, the restaurant will take ordinal data of the customer's feedback about the service rendered

Nominal Data

Sex/Gender is a type of nominal data.

Ordinal Data



Categorizing salary as high,medium low would mean an Ordinal data as we are giving a specific order to the data

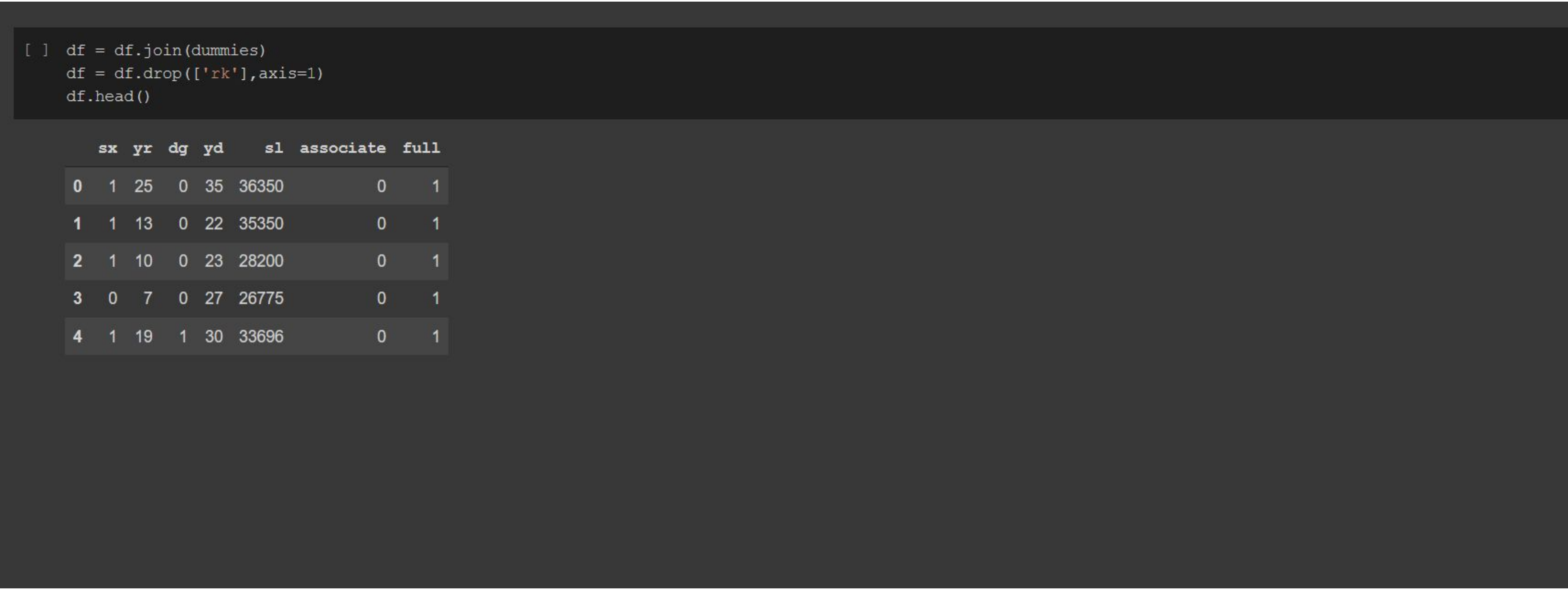
```
le = LabelEncoder()

df['sx']= le.fit_transform(df['sx'])
df['dg']= le.fit_transform(df['dg'])
df.head()
```

	sx	rk	yr	dg	yd	sl
0	1	full	25	0	35	36350
1	1	full	13	0	22	35350
2	1	full	10	0	23	28200
3	0	full	7	0	27	26775
4	1	full	19	1	30	33696

```
dummies.head()
```

	associate	full
0	0	1
1	0	1
2	0	1
3	0	1
4	0	1



```
[ ] df = df.join(dummies)
    df = df.drop(['rk'],axis=1)
    df.head()
```

```
   sx  yr  dg  yd     sl  associate  full
0   1  25   0  35  36350          0     1
1   1  13   0  22  35350          0     1
2   1  10   0  23  28200          0     1
3   0   7   0  27  26775          0     1
4   1  19   1  30  33696          0     1
```