INTRO TO PANDAS

LECTURE # 2

PANDAS

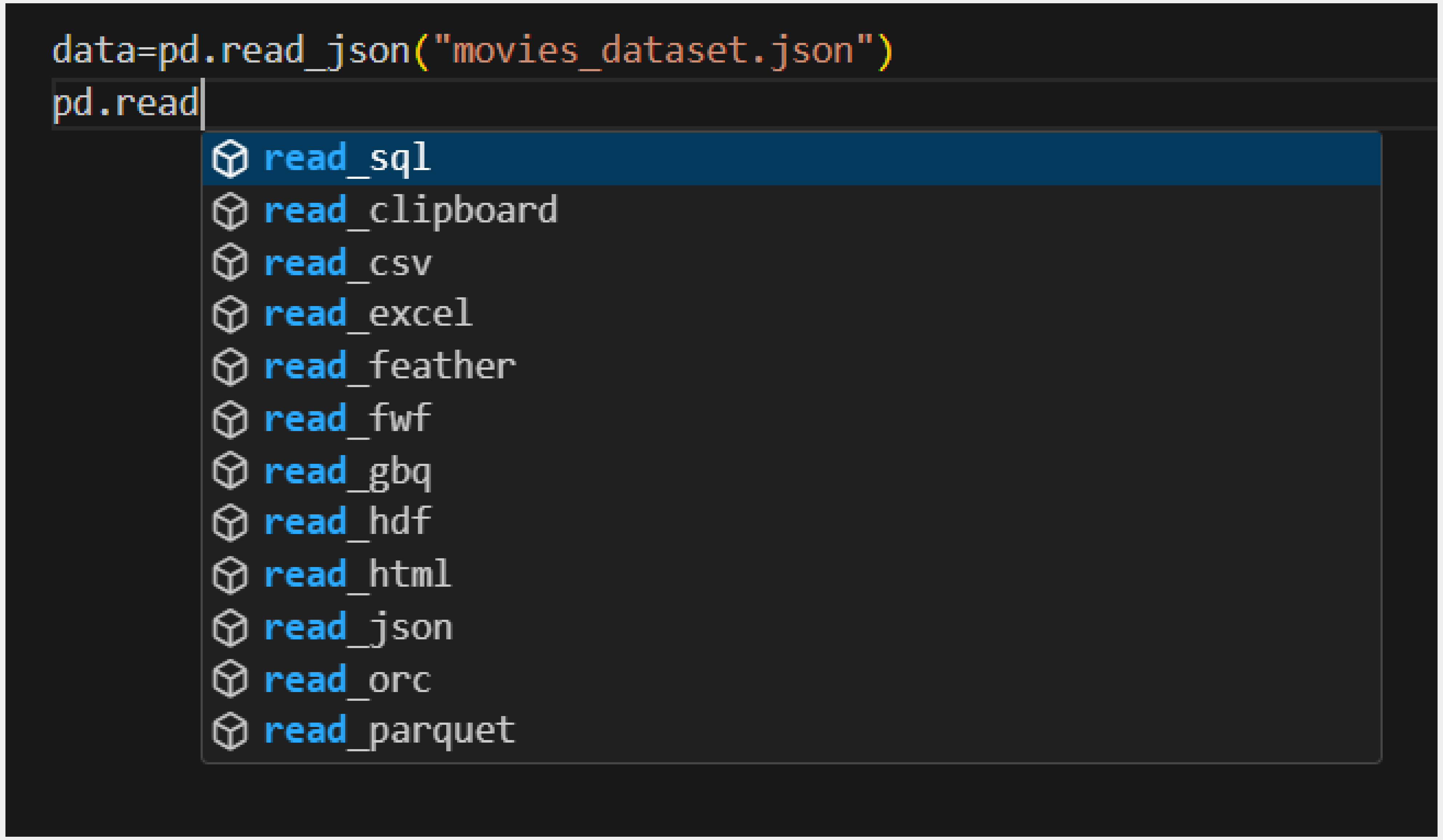
Library for data manipulation and analysis.

```
pip install pandas
```

```
import pandas as pd
```

PANDAS

For loading the Data



```
data=pd.read_json("movies_dataset.json")
pd.read
```

PANDAS

For displaying the Data loaded in the DATA FRAME

1

<Data Frame name>.head()

By default it will display **first** 5 rows

2

<Data Frame name>.tail()

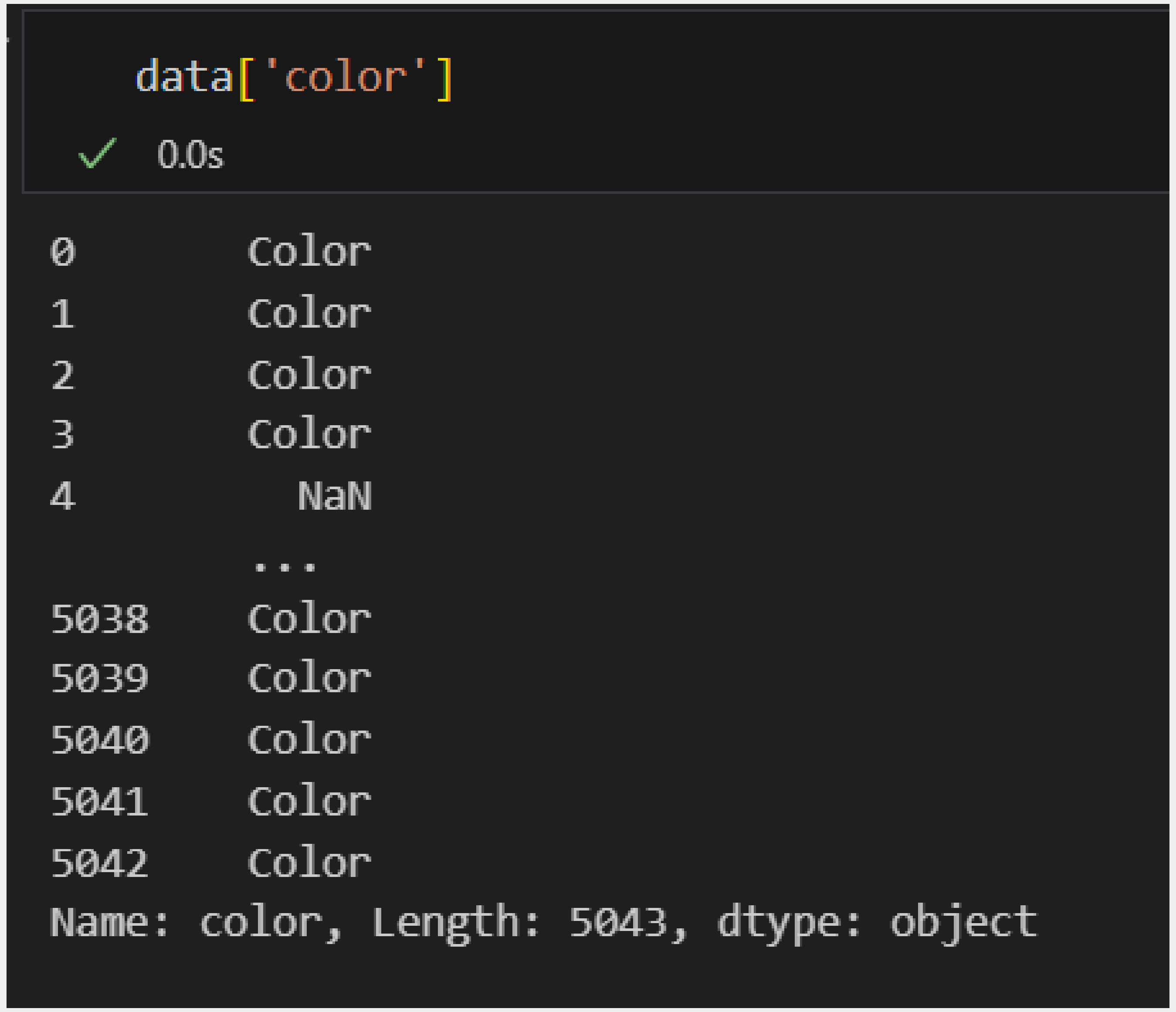
By default it will display **last** 5 rows

Inserting a number inside the head or tail function will display the specifies number of first/last rows

COLUMNS

1

To display the specific **column**



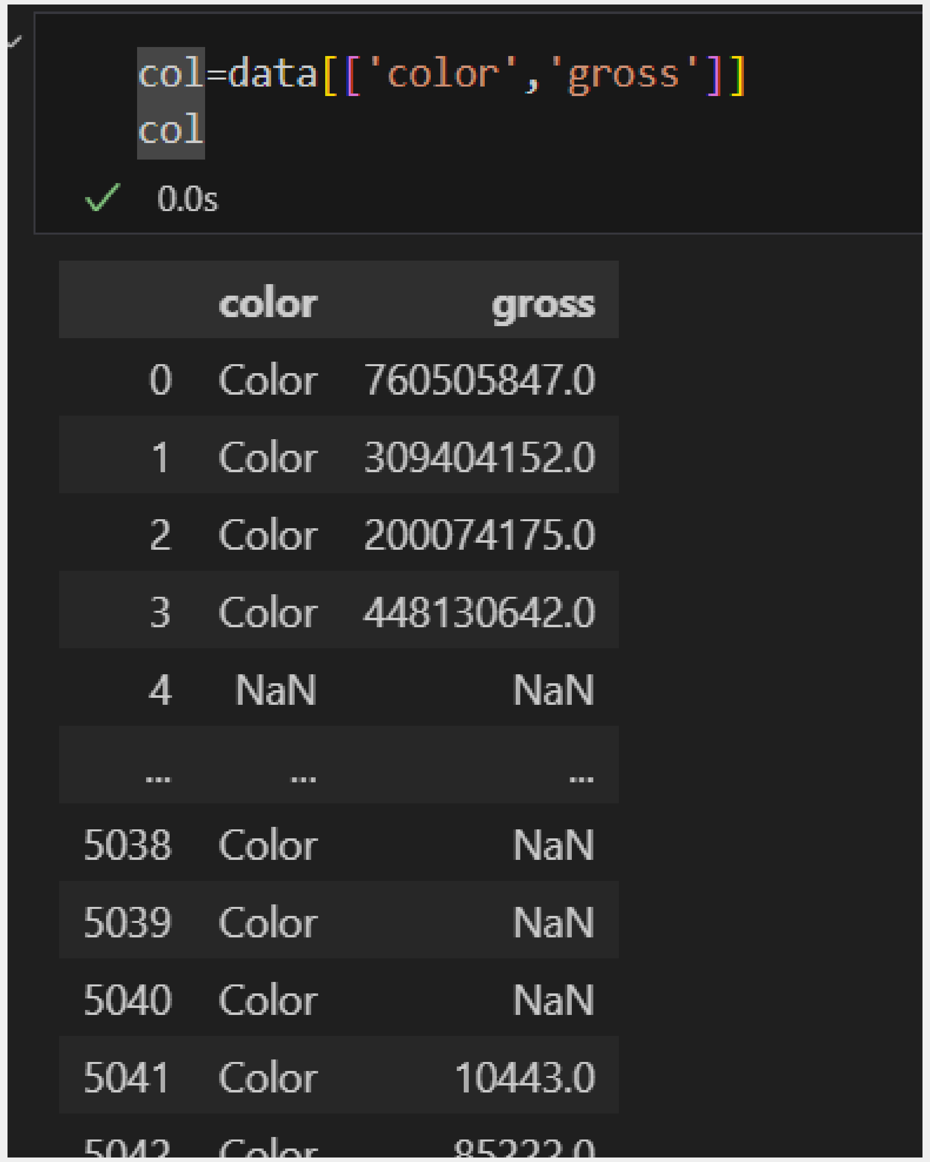
```
data['color']
```

✓ 0.0s

```
0       Color
1       Color
2       Color
3       Color
4         NaN
        ...
5038    Color
5039    Color
5040    Color
5041    Color
5042    Color
Name: color, Length: 5043, dtype: object
```

2

To display the **multiple columns**

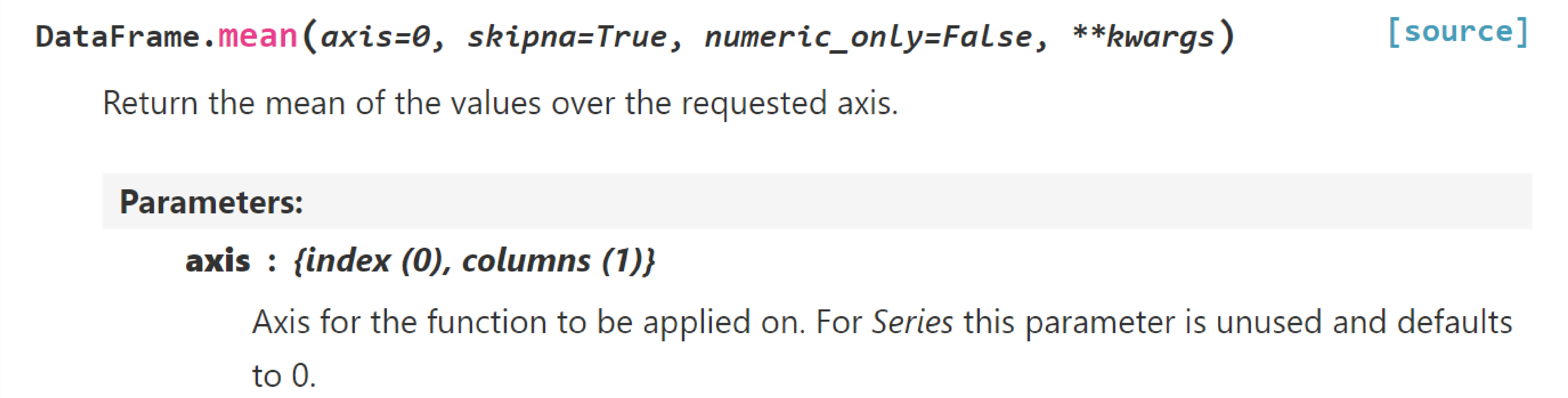


```
col=data[['color','gross']]
col
```

✓ 0.0s

	color	gross
0	Color	760505847.0
1	Color	309404152.0
2	Color	200074175.0
3	Color	448130642.0
4	NaN	NaN
...	...	...
5038	Color	NaN
5039	Color	NaN
5040	Color	NaN
5041	Color	10443.0
5042	Color	85222.0

AXIS



```
DataFrame.mean(axis=0, skipna=True, numeric_only=False, **kwargs)
```

[source]

Return the mean of the values over the requested axis.

Parameters:

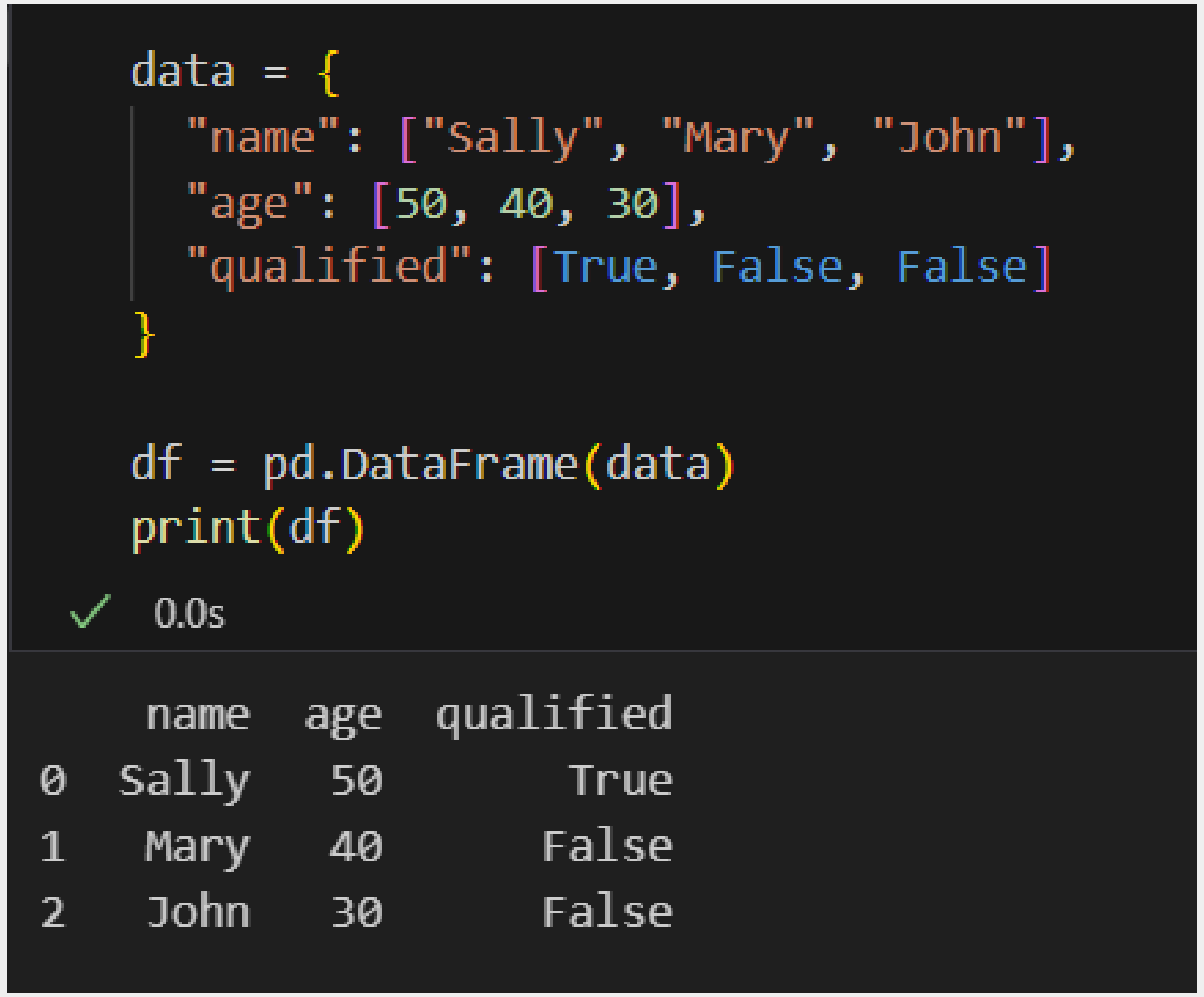
axis : *{index (0), columns (1)}*

Axis for the function to be applied on. For *Series* this parameter is unused and defaults to 0.

axis = 0, read every index that is across the column

axis = 1 reads every column that is across the rows

DROP

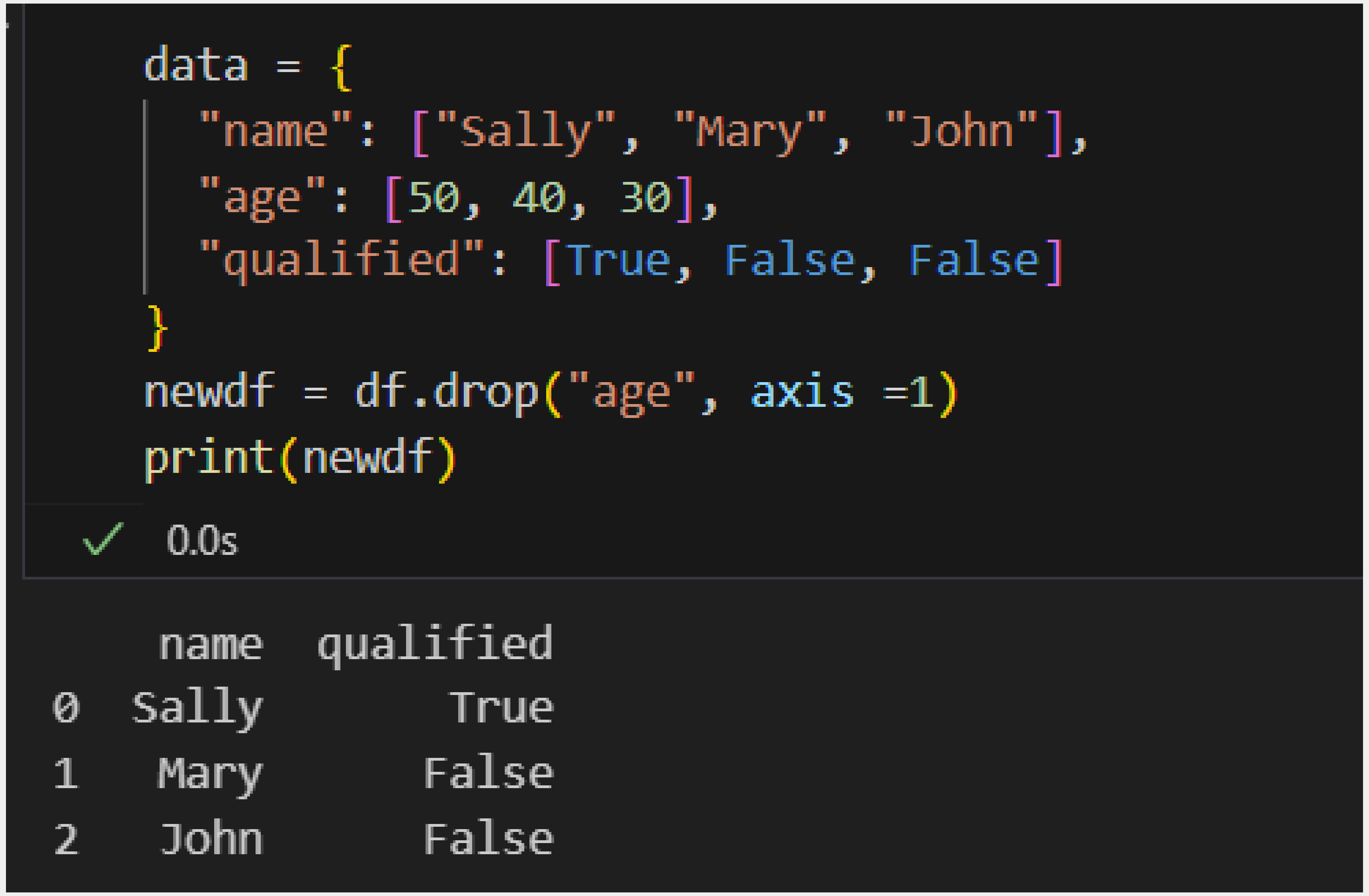


```
data = {
  "name": ["Sally", "Mary", "John"],
  "age": [50, 40, 30],
  "qualified": [True, False, False]
}

df = pd.DataFrame(data)
print(df)
```

✓ 0.0s

```
    name  age  qualified
0  Sally   50       True
1   Mary   40      False
2   John   30      False
```



```
data = {
  "name": ["Sally", "Mary", "John"],
  "age": [50, 40, 30],
  "qualified": [True, False, False]
}
newdf = df.drop("age", axis =1)
print(newdf)
```

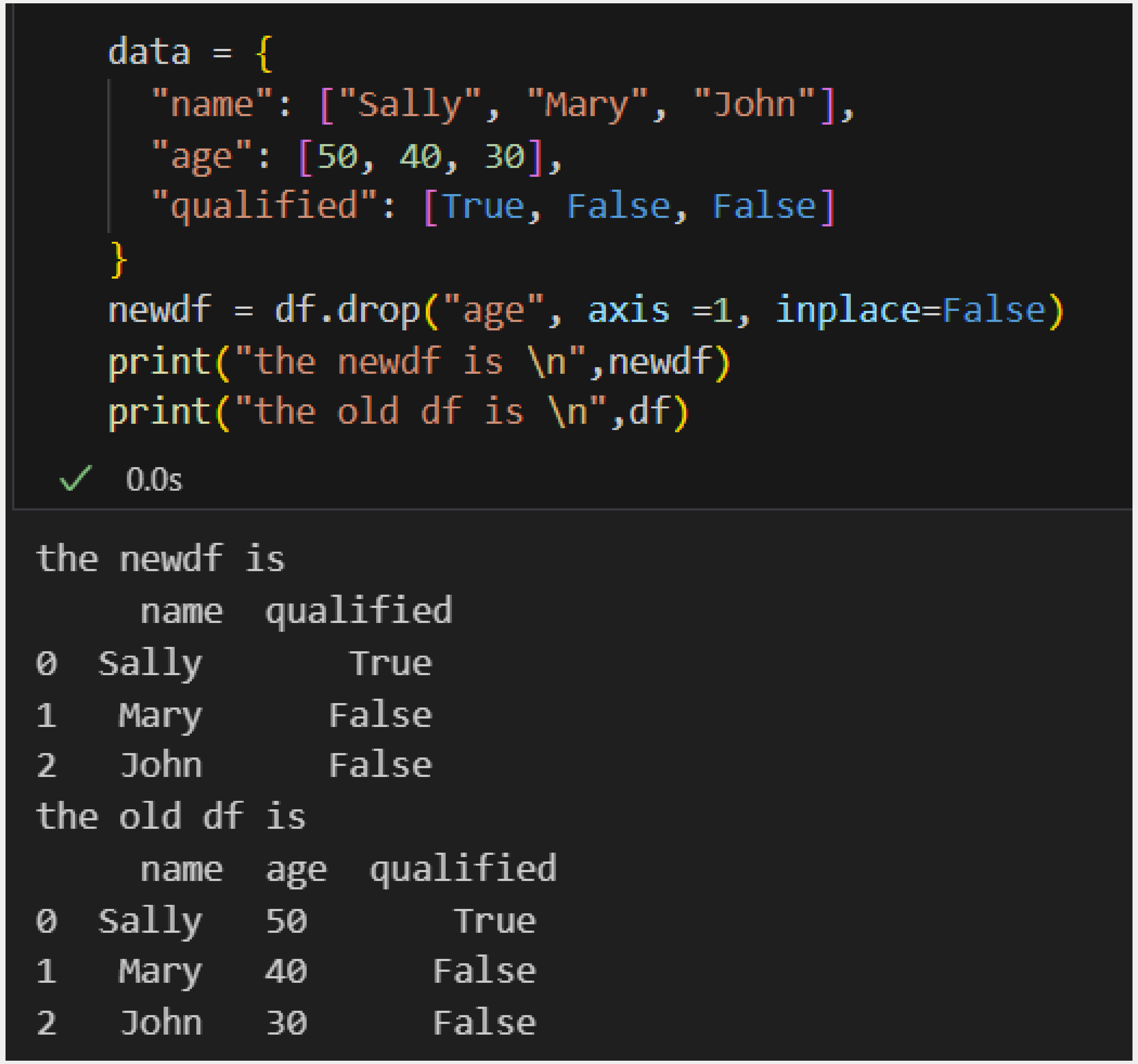
✓ 0.0s

```
    name  qualified
0  Sally       True
1   Mary      False
2   John      False
```

By default the axis is set to 0, set it to 1 to remove the column.

DROP WITH INPLACE

inplace = TRUE

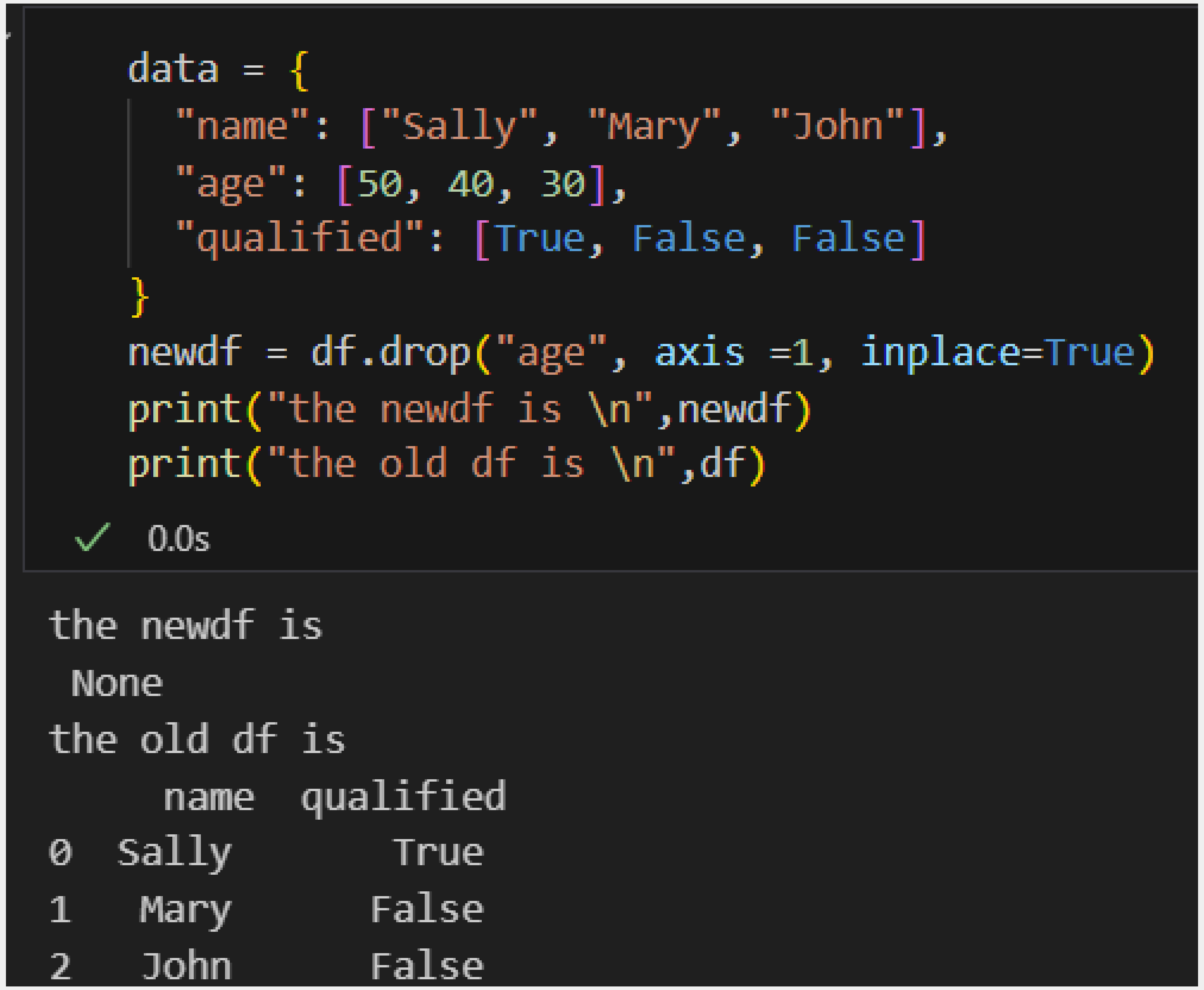


```python
data = {
  "name": ["Sally", "Mary", "John"],
  "age": [50, 40, 30],
  "qualified": [True, False, False]
}
newdf = df.drop("age", axis =1, inplace=False)
print("the newdf is \n",newdf)
print("the old df is \n",df)
```

✓ 0.0s

```
the newdf is
      name  qualified
0  Sally       True
1   Mary      False
2   John      False
the old df is
      name  age  qualified
0  Sally   50       True
1   Mary   40      False
2   John   30      False
```

inplace = TRUE

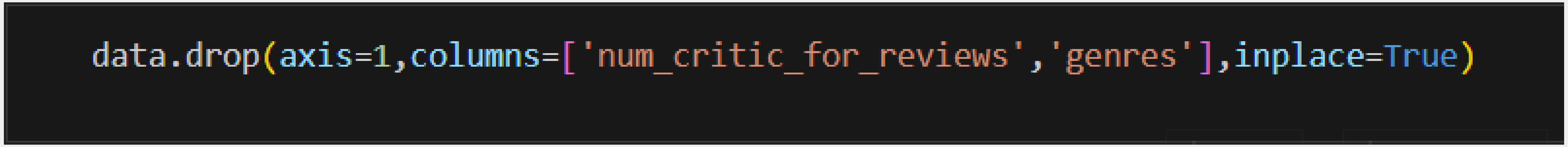


```python
data = {
  "name": ["Sally", "Mary", "John"],
  "age": [50, 40, 30],
  "qualified": [True, False, False]
}
newdf = df.drop("age", axis =1, inplace=True)
print("the newdf is \n",newdf)
print("the old df is \n",df)
```

✓ 0.0s

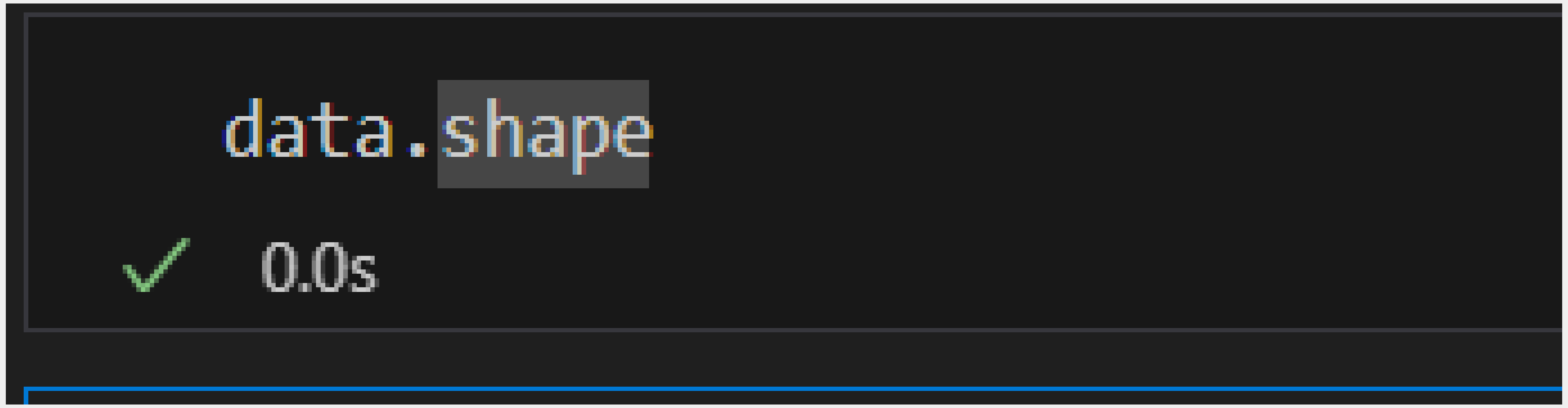
```
the newdf is
 None
the old df is
      name  qualified
0  Sally       True
1   Mary      False
2   John      False
```

DROPPING MORE THAN ONE COLS



```
data.drop(axis=1,columns=['num_critic_for_reviews','genres'],inplace=True)
```

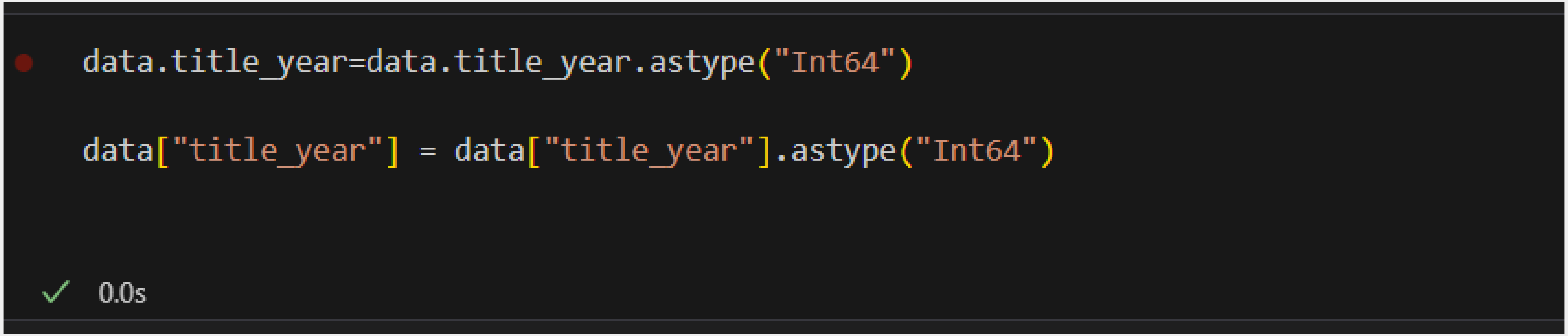
PRINTING THE ROW & COLS



```
data.shape
```

✓ 0.0s

CHANGING THE DATA TYPE

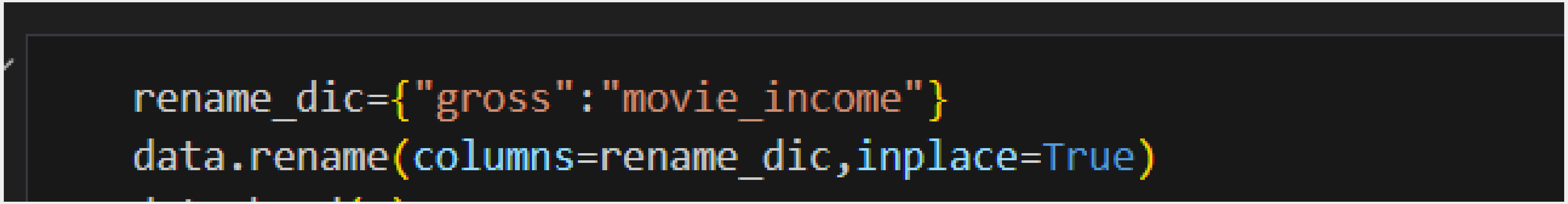


```
data.title_year=data.title_year.astype("Int64")

data["title_year"] = data["title_year"].astype("Int64")
```

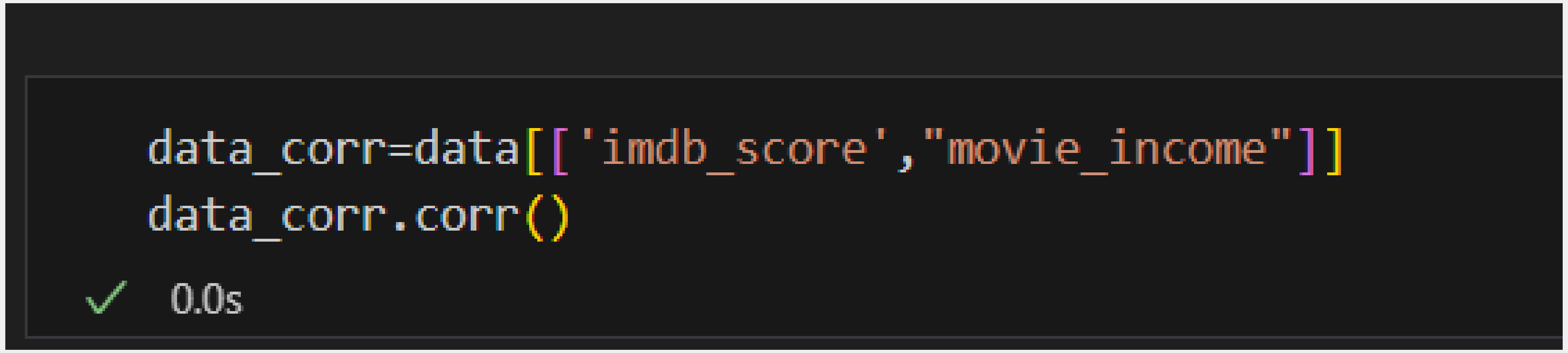
✓ 0.0s

RENAMING THE COLS



```
rename_dic={"gross":"movie_income"}
data.rename(columns=rename_dic,inplace=True)
```

CORRELATION MATRIX

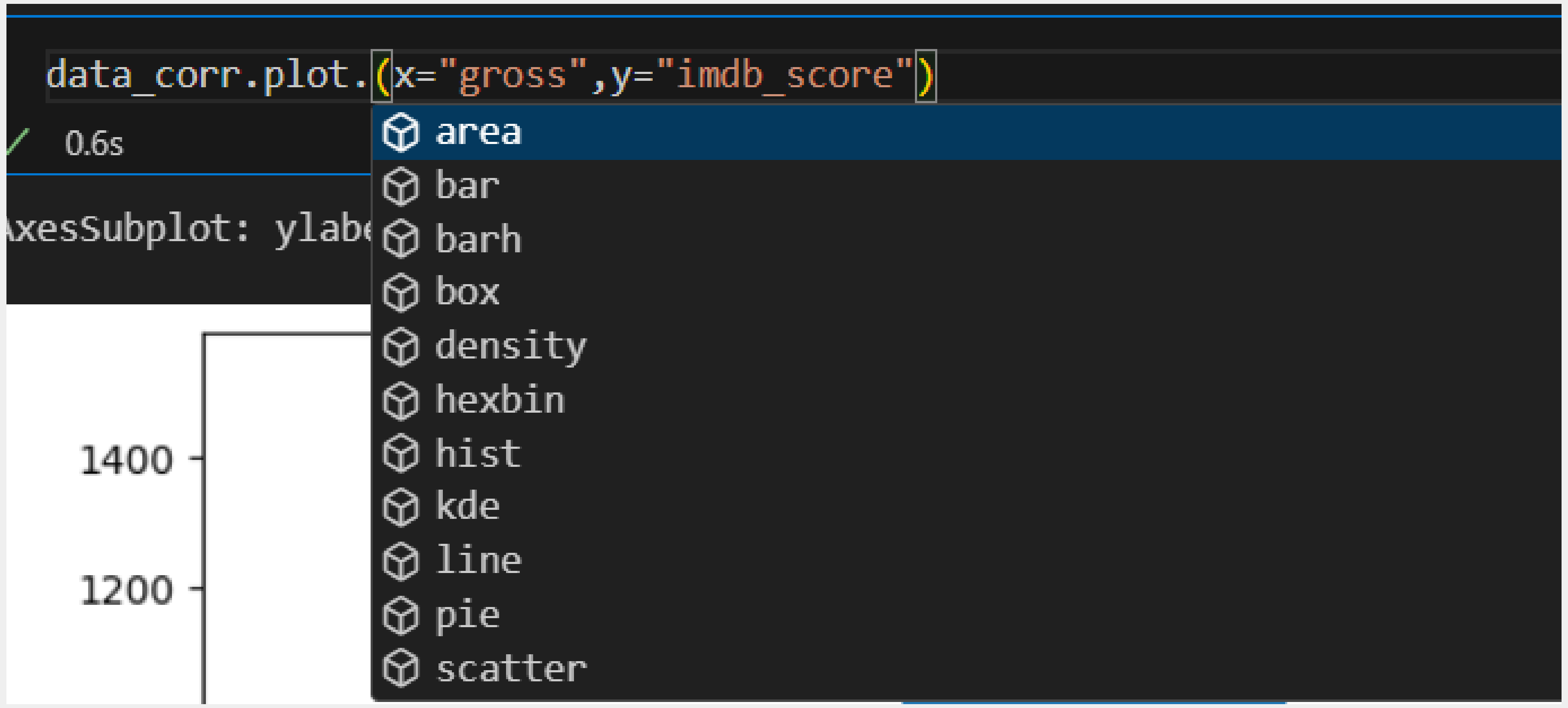


```
data_corr=data[['imdb_score',"movie_income"]]
data_corr.corr()
```

✓ 0.0s

SCATTER PLOT

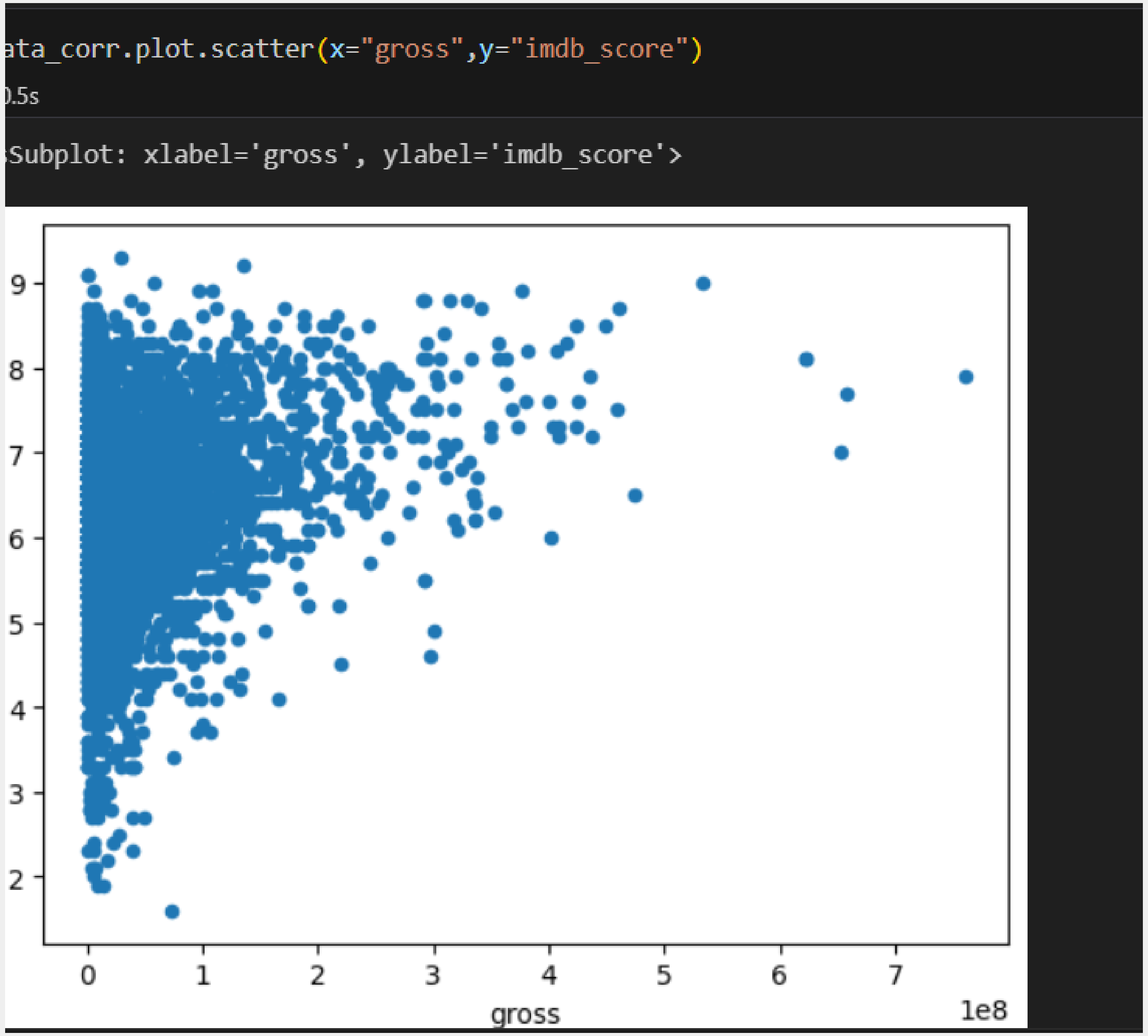
used to map two numeric values



```
data_corr.plot.(x="gross",y="imdb_score")
```

```
0.6s
```

```
xesSubplot: ylabe
```

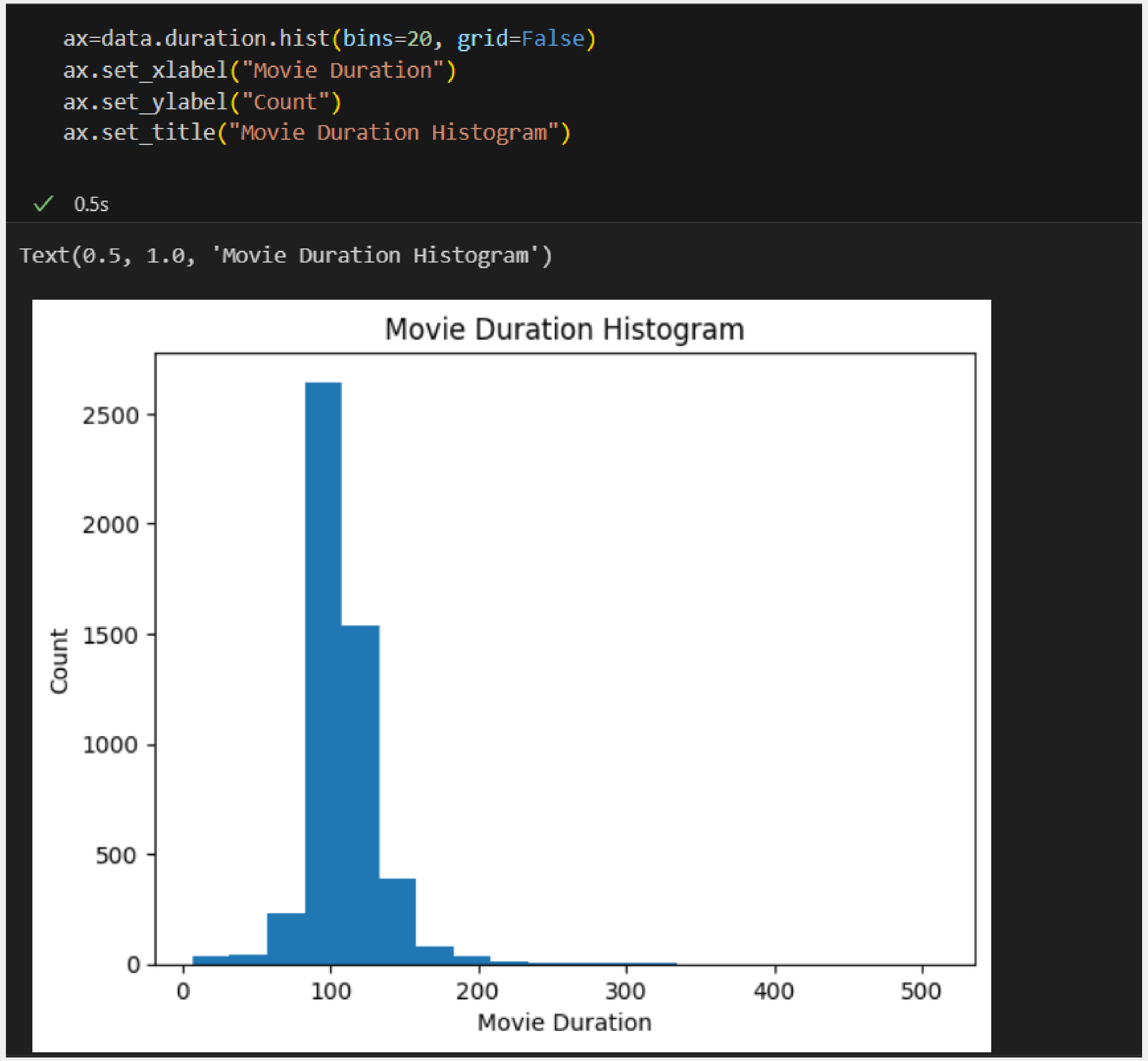


```
ata_corr.plot.scatter(x="gross",y="imdb_score")
```

```
0.5s
```

```
Subplot: xlabel='gross', ylabel='imdb_score'>
```

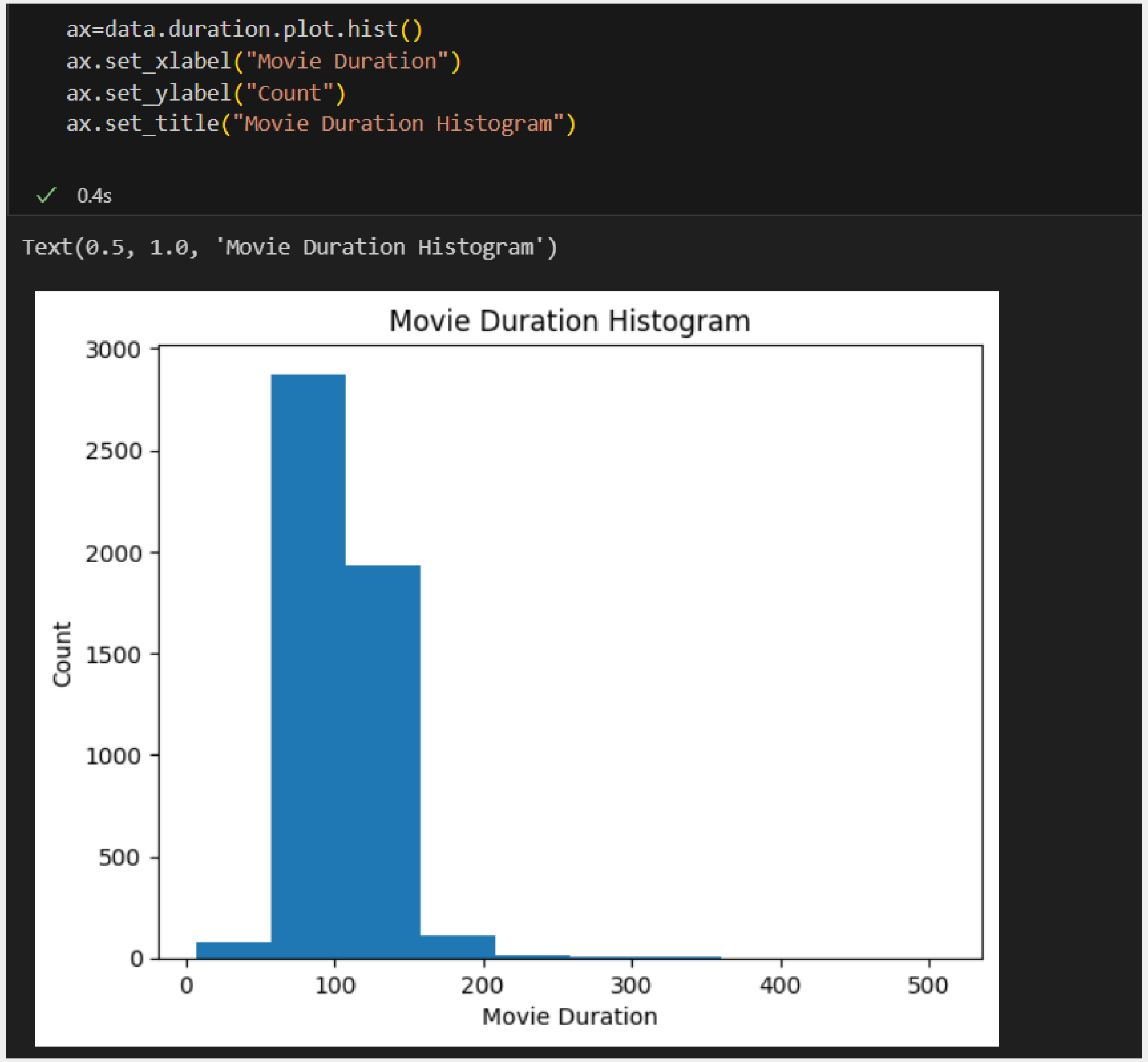
HISTOGRAM



```
ax=data.duration.hist(bins=20, grid=False)
ax.set_xlabel("Movie Duration")
ax.set_ylabel("Count")
ax.set_title("Movie Duration Histogram")
```

✓ 0.5s

```
Text(0.5, 1.0, 'Movie Duration Histogram')
```



```
ax=data.duration.plot.hist()
ax.set_xlabel("Movie Duration")
ax.set_ylabel("Count")
ax.set_title("Movie Duration Histogram")
```

✓ 0.4s

```
Text(0.5, 1.0, 'Movie Duration Histogram')
```