Working with data in python using pandas

Pandas

- Pandas is a Python library used for working with data sets

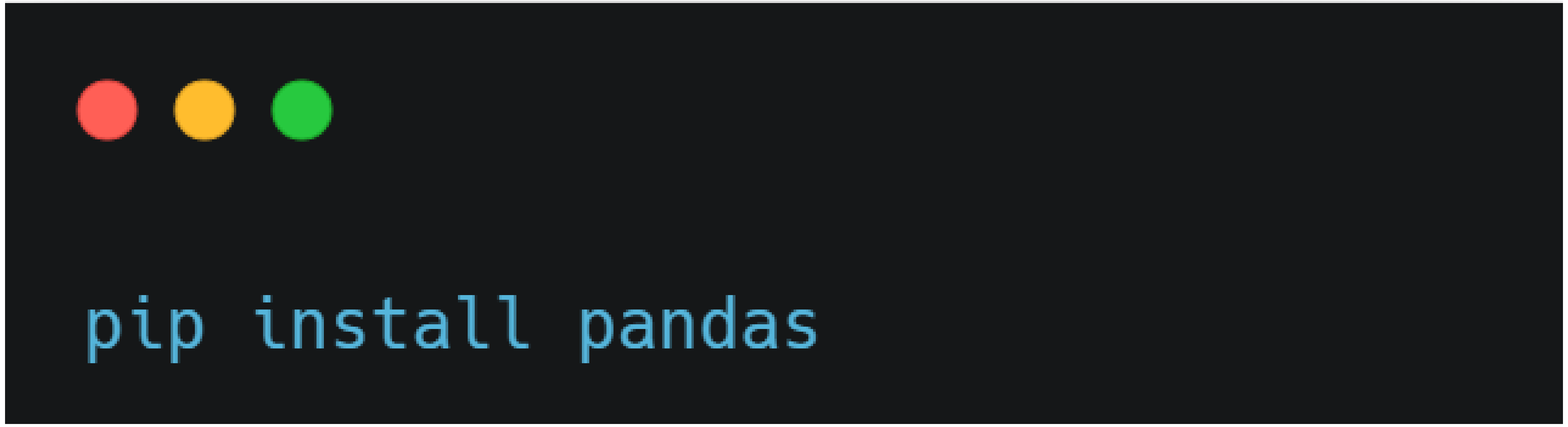
- It has functions for analyzing, cleaning, exploring, and manipulating data

- Pandas is fast and it has high performance & productivity for users

Why Pandas

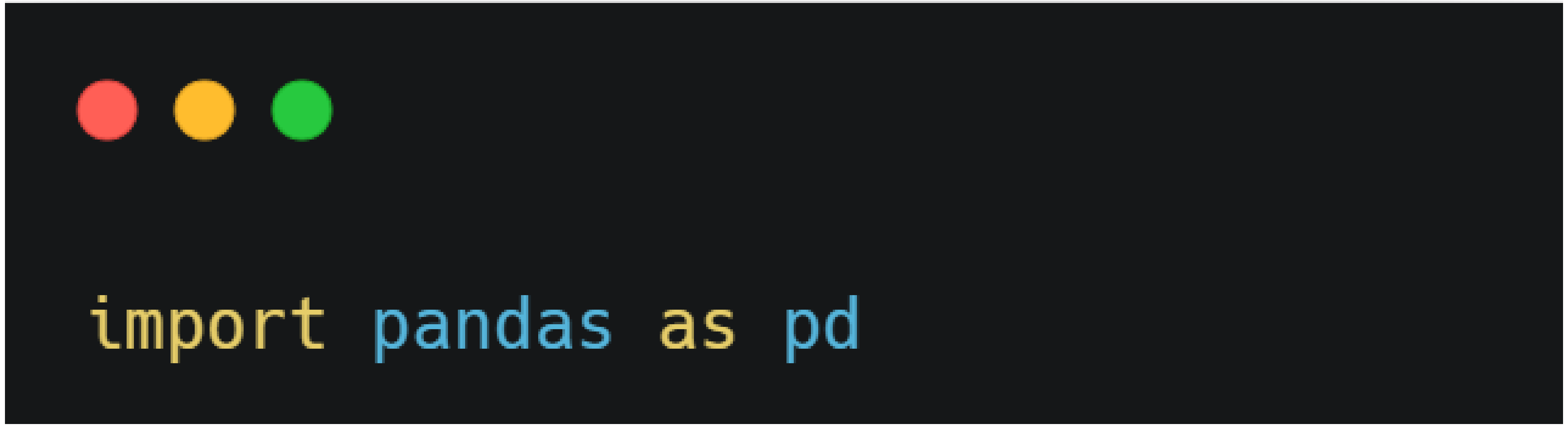
- Pandas allows us to analyze big data and make conclusions based on statistical theories
- Pandas can clean messy data sets, and make them readable and relevant
- Relevant data is very important in data science

Installation



```
pip install pandas
```

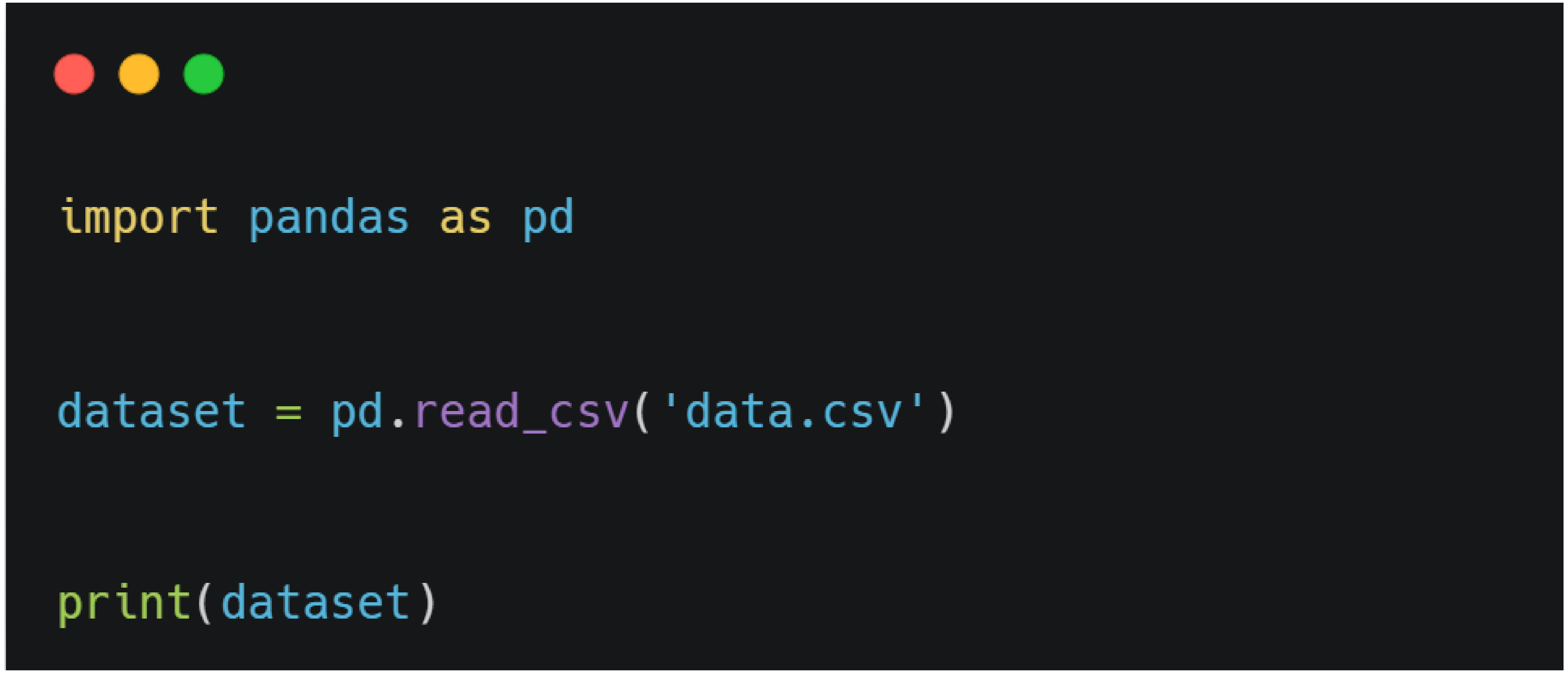
Import pandas



```python
import pandas as pd
```

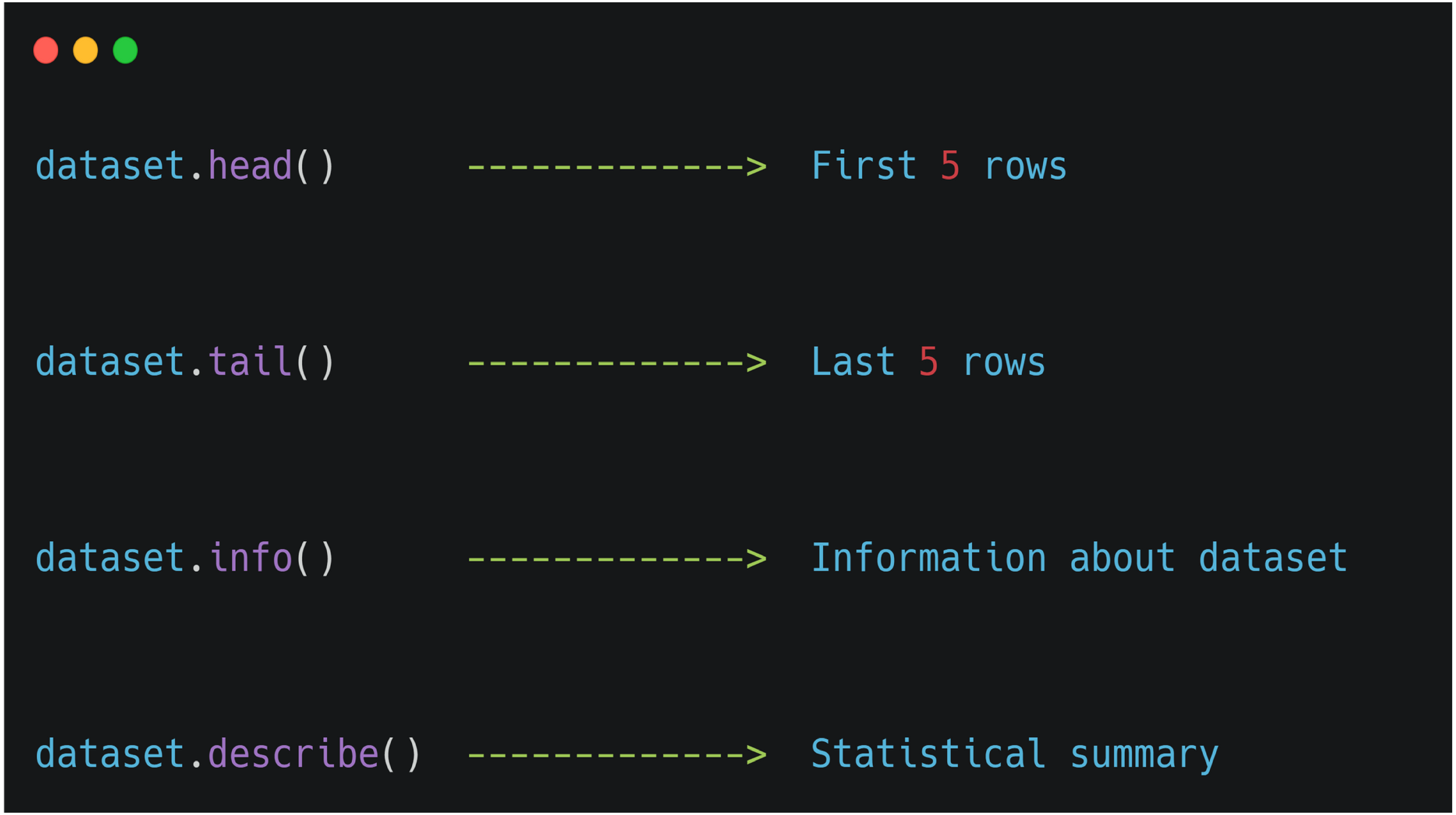
Data Analysis

Read CSV



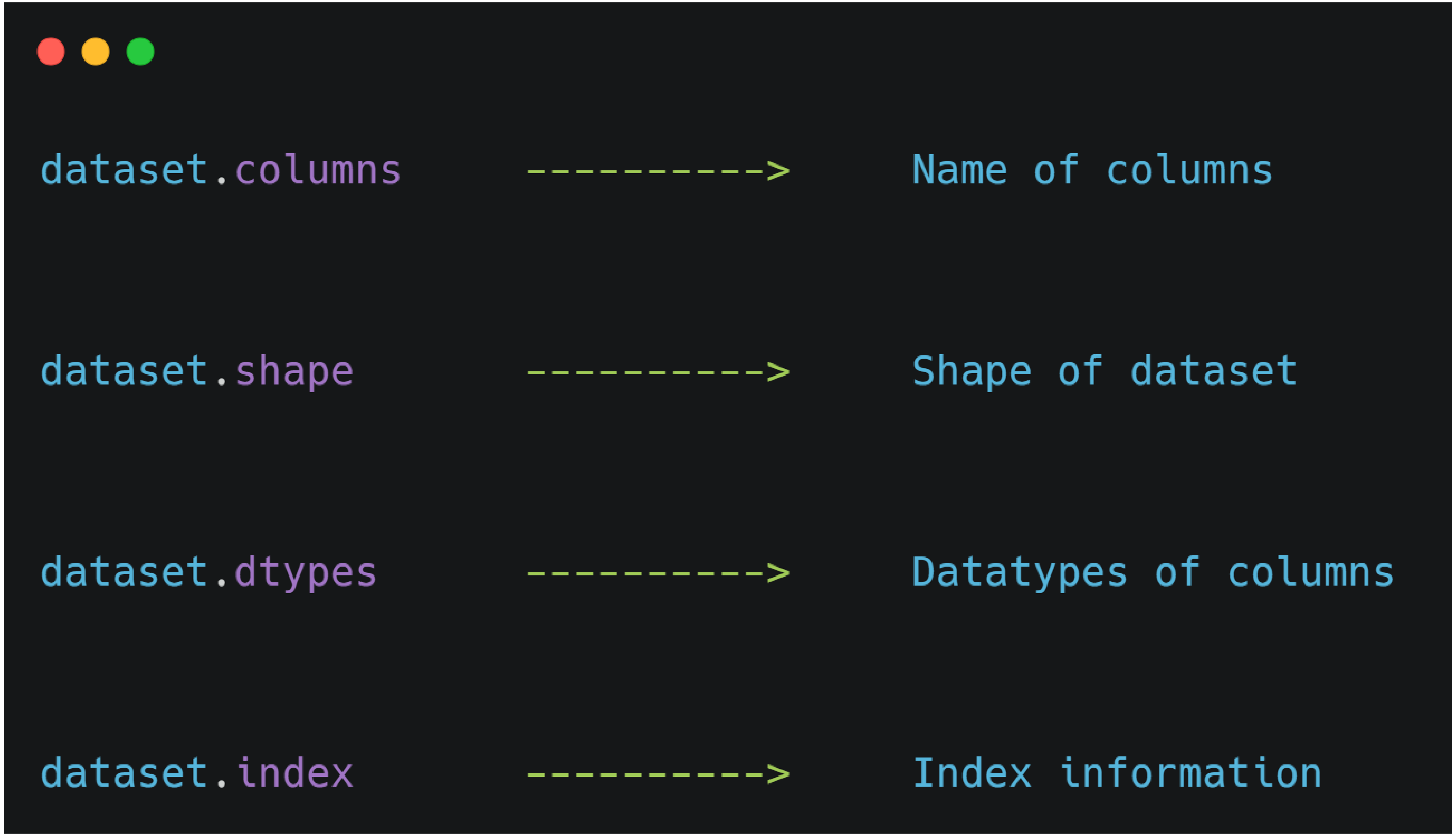
```python
import pandas as pd

dataset = pd.read_csv('data.csv')

print(dataset)
```

Analysing Data



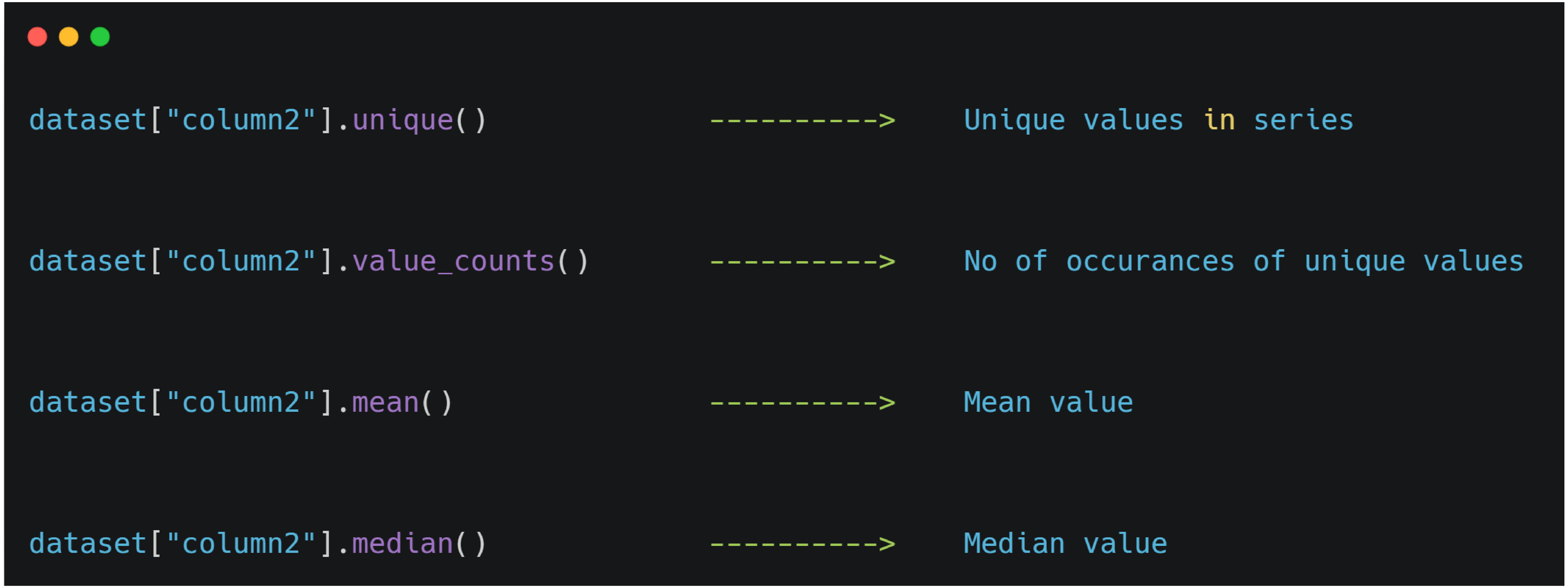
```
dataset.head()      ------------->  First 5 rows


dataset.tail()      ------------->  Last 5 rows


dataset.info()      ------------->  Information about dataset


dataset.describe()  ------------->  Statistical summary
```

Analysing Data



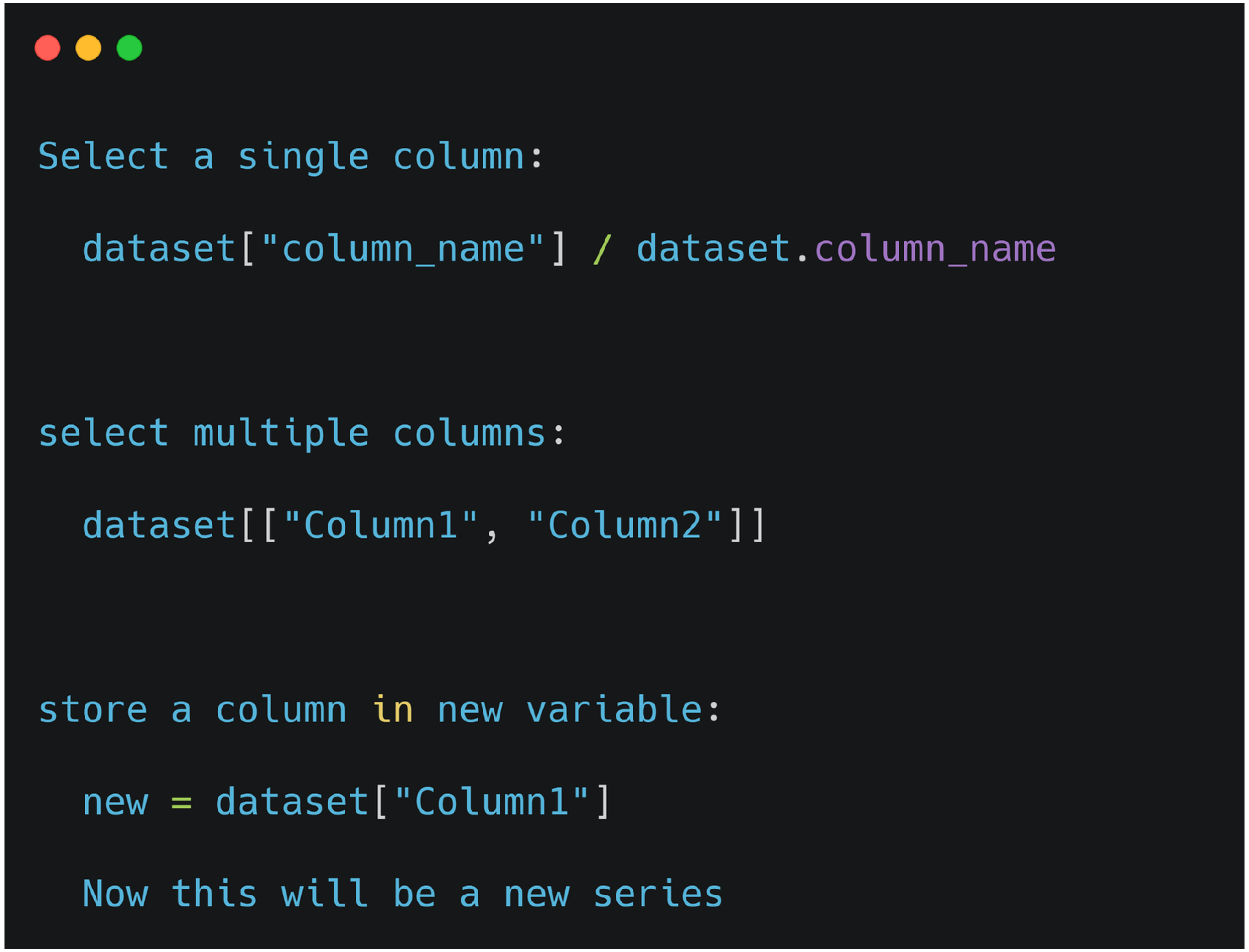
```
dataset.columns     ---------->     Name of columns


dataset.shape       ---------->     Shape of dataset


dataset.dtypes      ---------->     Datatypes of columns


dataset.index       ---------->     Index information
```

Analysing Data



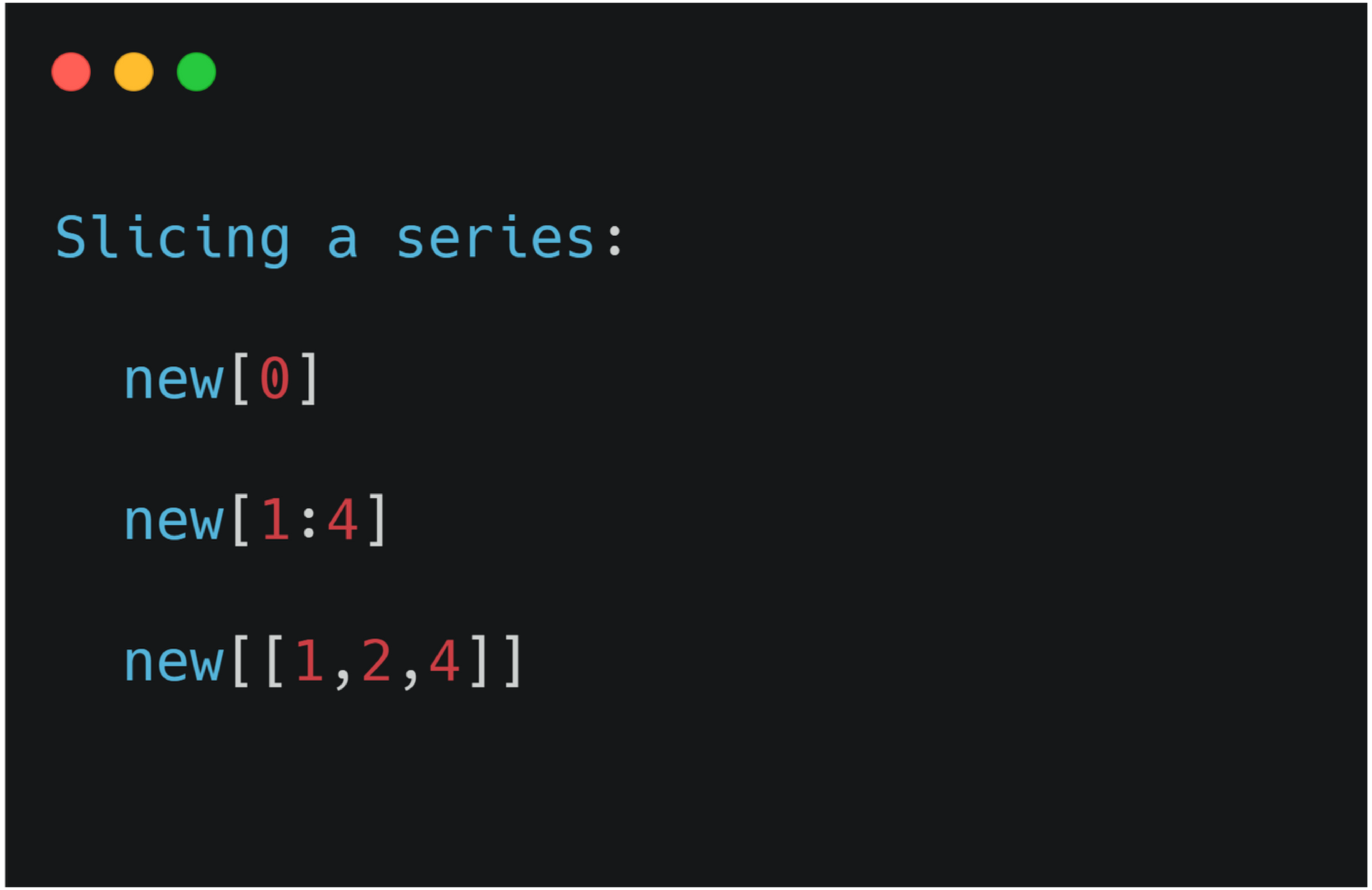
```
dataset["column2"].unique()             ---------->    Unique values in series


dataset["column2"].value_counts()       ---------->    No of occurances of unique values


dataset["column2"].mean()               ---------->    Mean value


dataset["column2"].median()             ---------->    Median value
```

Analysing Data



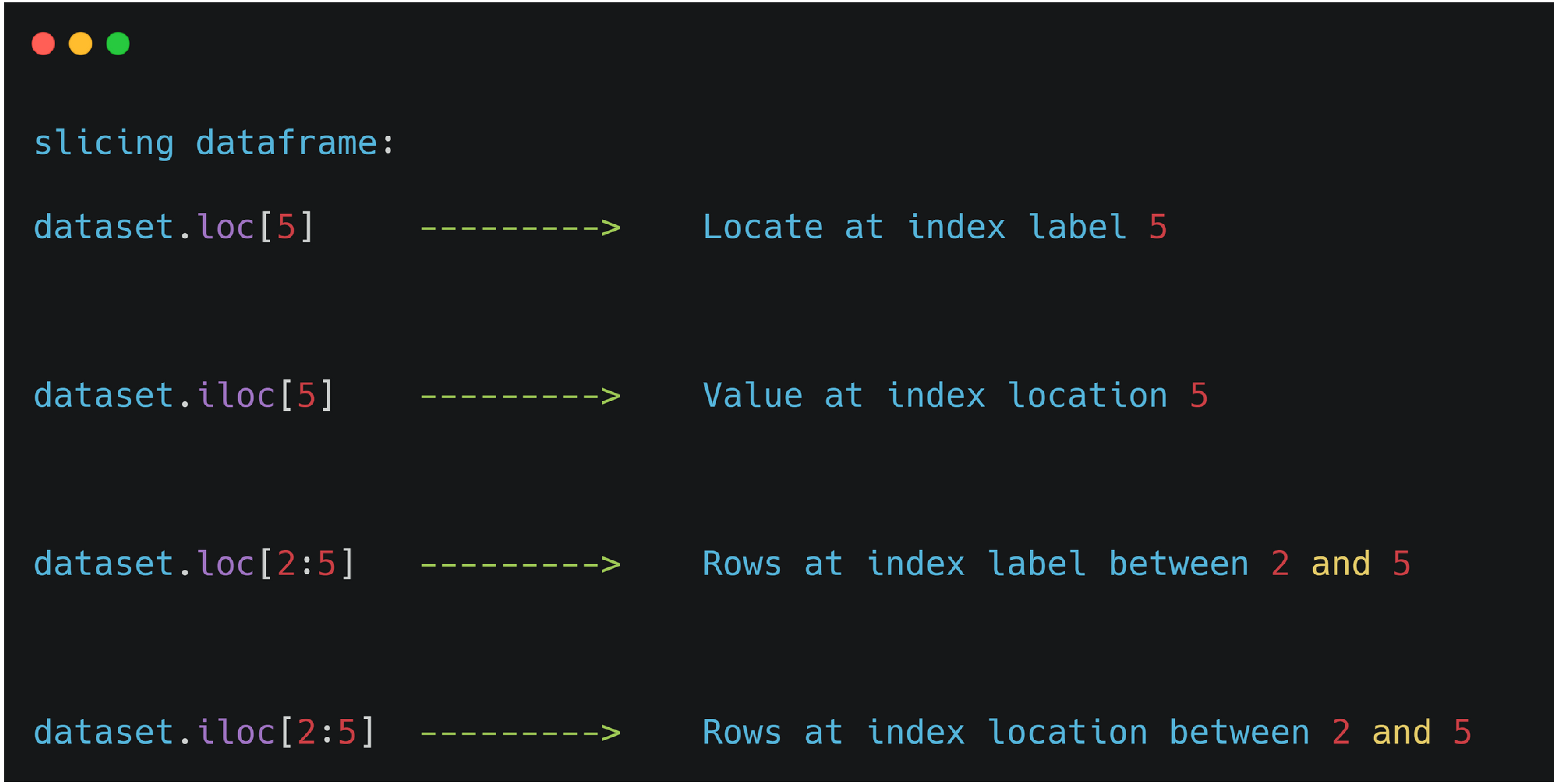
```
Select a single column:

  dataset["column_name"] / dataset.column_name


select multiple columns:

  dataset[["Column1", "Column2"]]


store a column in new variable:

  new = dataset["Column1"]

  Now this will be a new series
```

Analysing Data



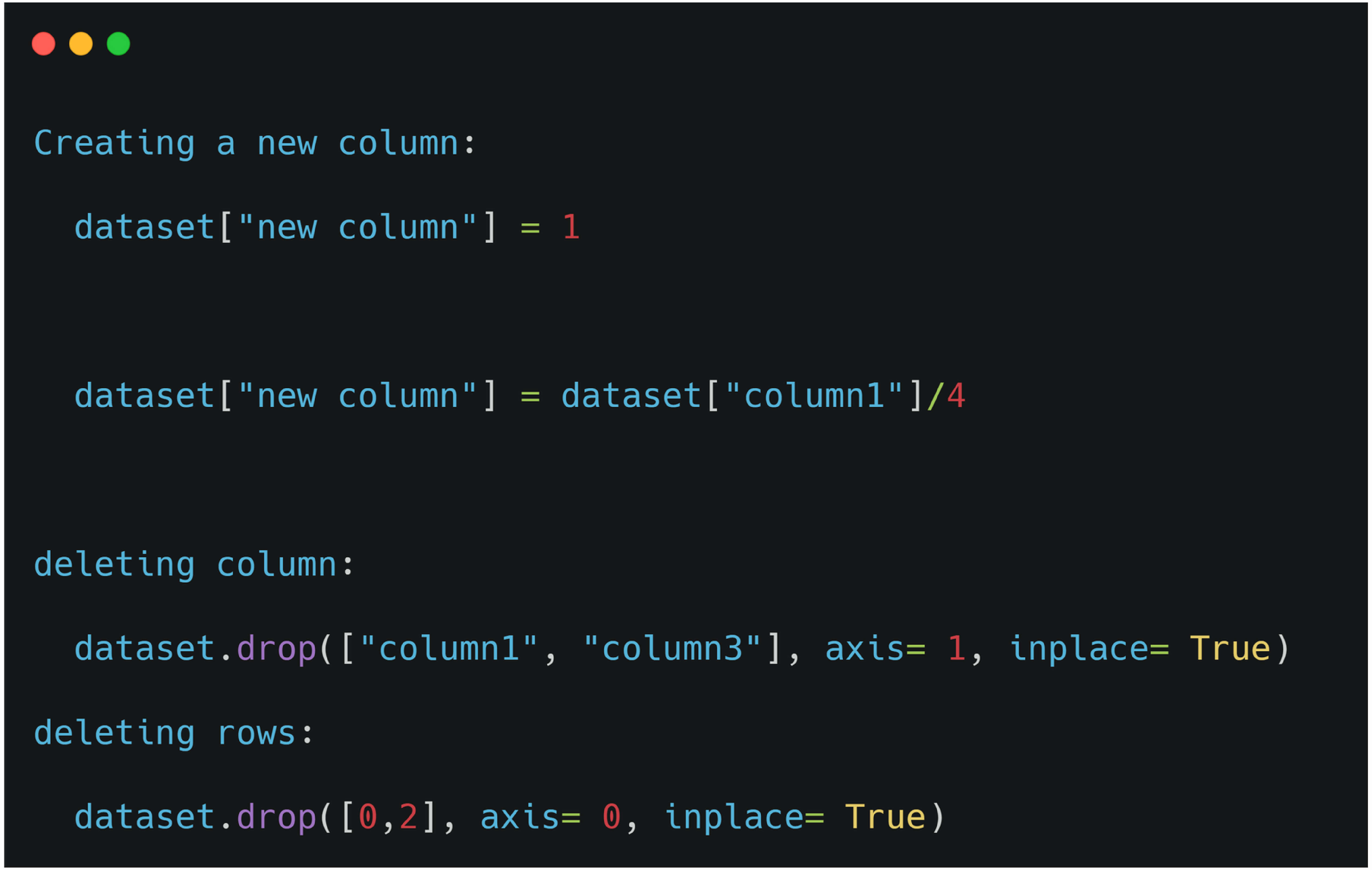
```
Slicing a series:

  new[0]

  new[1:4]

  new[[1,2,4]]
```

Analysing Data



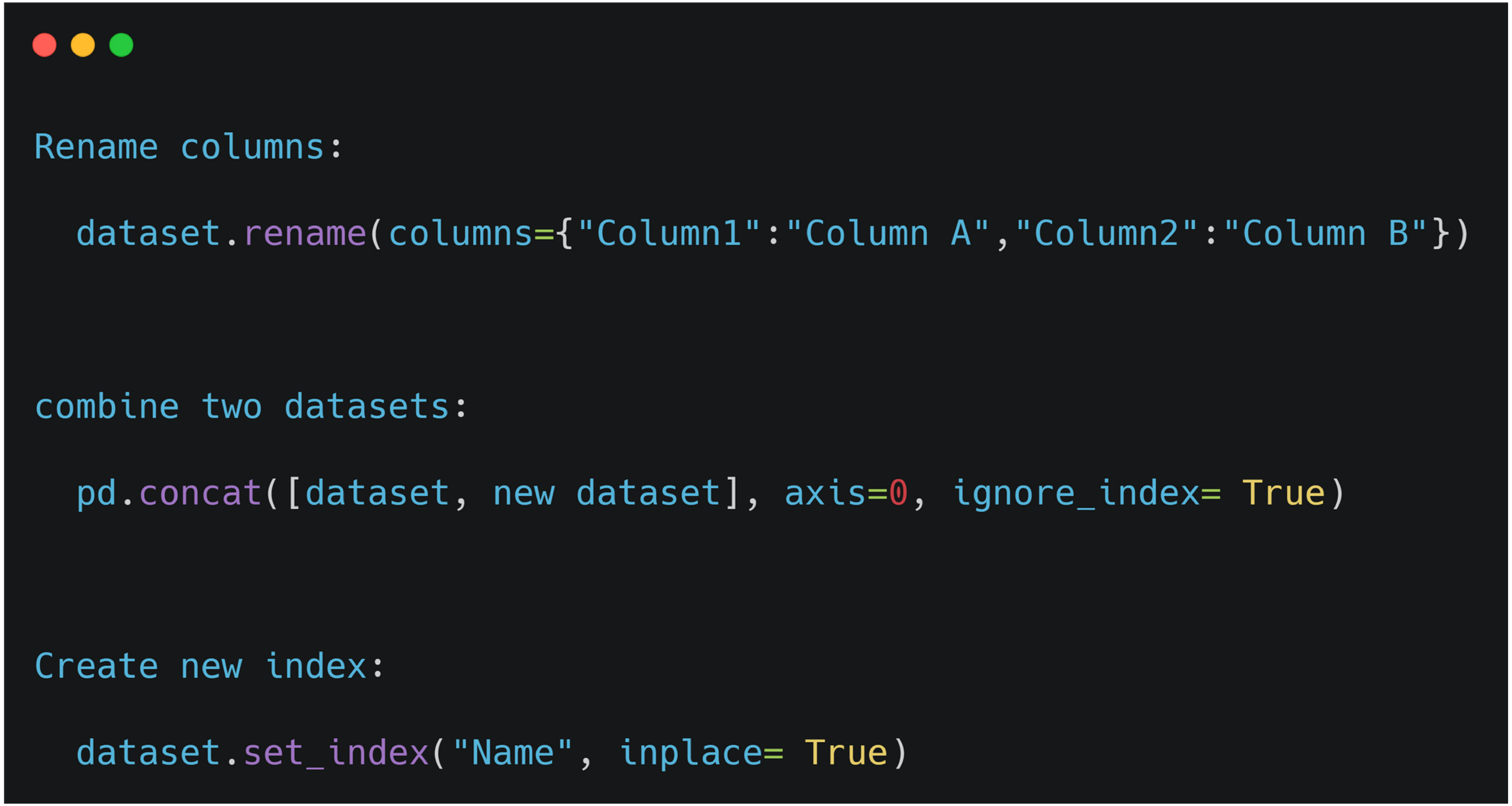
```
slicing dataframe:

dataset.loc[5]     --------->    Locate at index label 5


dataset.iloc[5]    --------->    Value at index location 5


dataset.loc[2:5]   --------->    Rows at index label between 2 and 5


dataset.iloc[2:5]  --------->    Rows at index location between 2 and 5
```

Analysing Data



```
Creating a new column:

  dataset["new column"] = 1


  dataset["new column"] = dataset["column1"]/4


deleting column:

  dataset.drop(["column1", "column3"], axis= 1, inplace= True)

deleting rows:

  dataset.drop([0,2], axis= 0, inplace= True)
```

Analysing Data



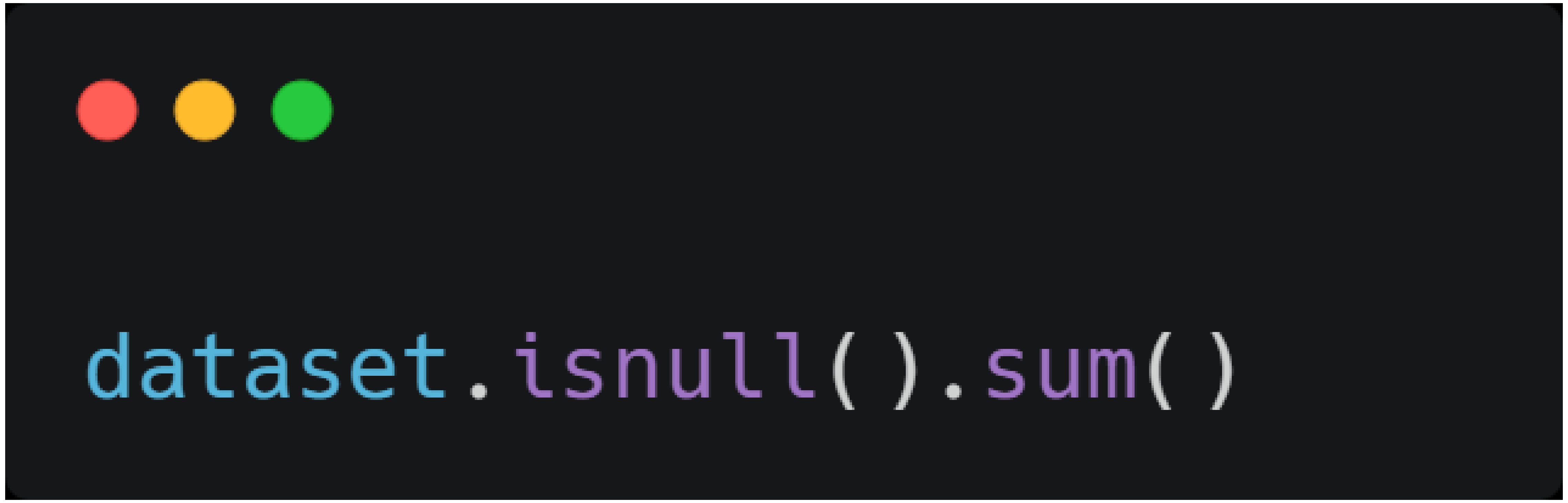
```
Rename columns:

  dataset.rename(columns={"Column1":"Column A","Column2":"Column B"})


combine two datasets:

  pd.concat([dataset, new dataset], axis=0, ignore_index= True)


Create new index:

  dataset.set_index("Name", inplace= True)
```

Data Preprocessing

Data Preprocessing

- Data cleaning means fixing unwanted or improper data in your dataset
- This will improve the accuracy of data by removing or correcting inaccuracies, missing values, duplicates, and irrelevant data
- It can make the data more consistent, reducing the risk of errors in downstream processes

Checking null values

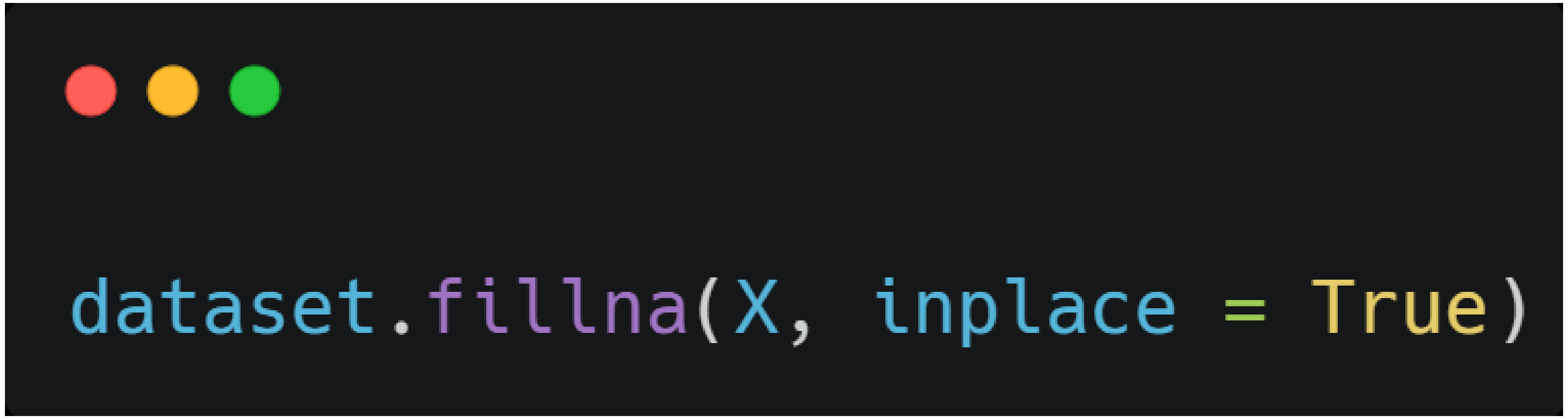


```
dataset.isnull().sum()
```

Handling Null values

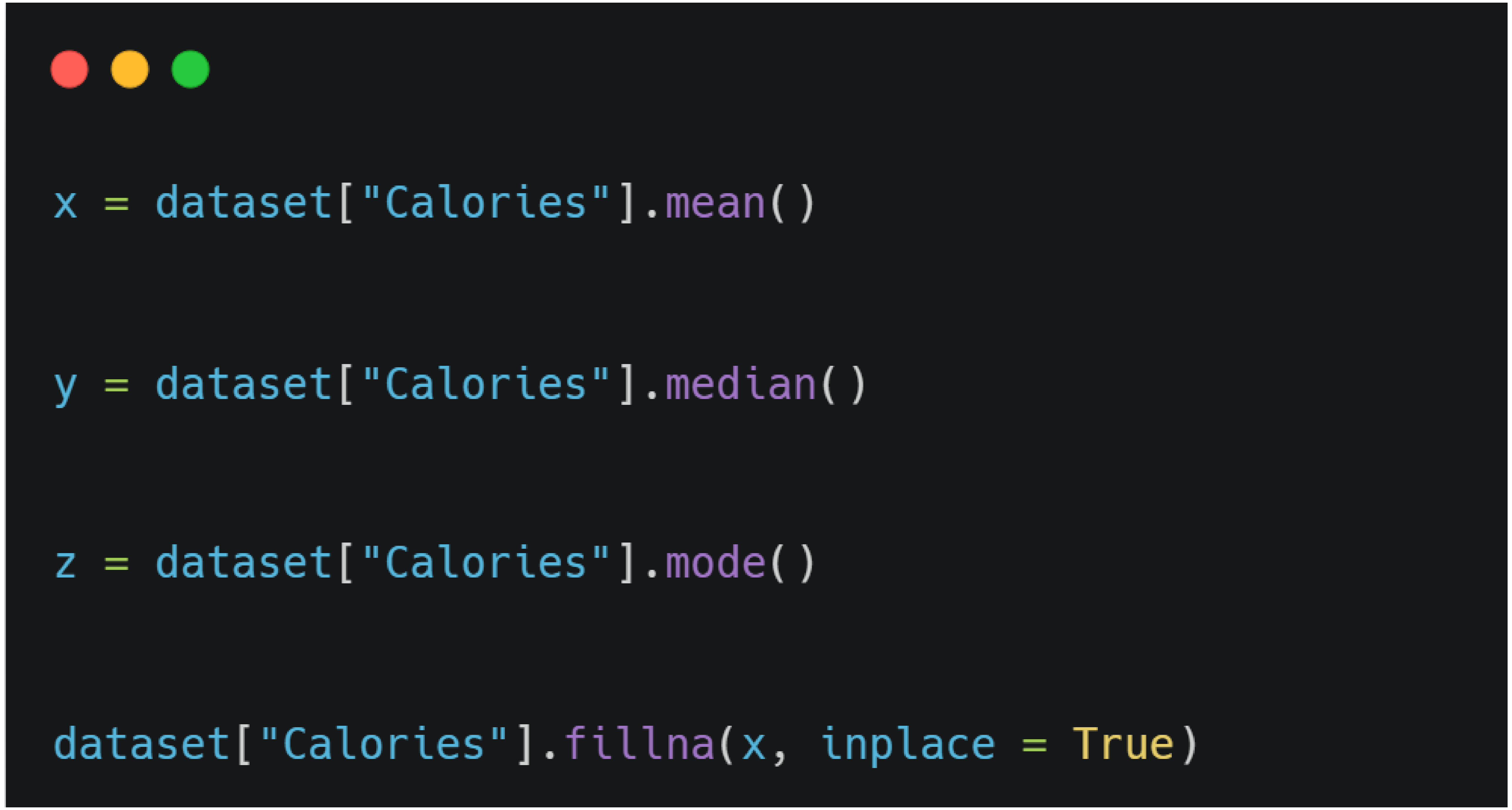
- Imputation
- Dropping

Handling Null values



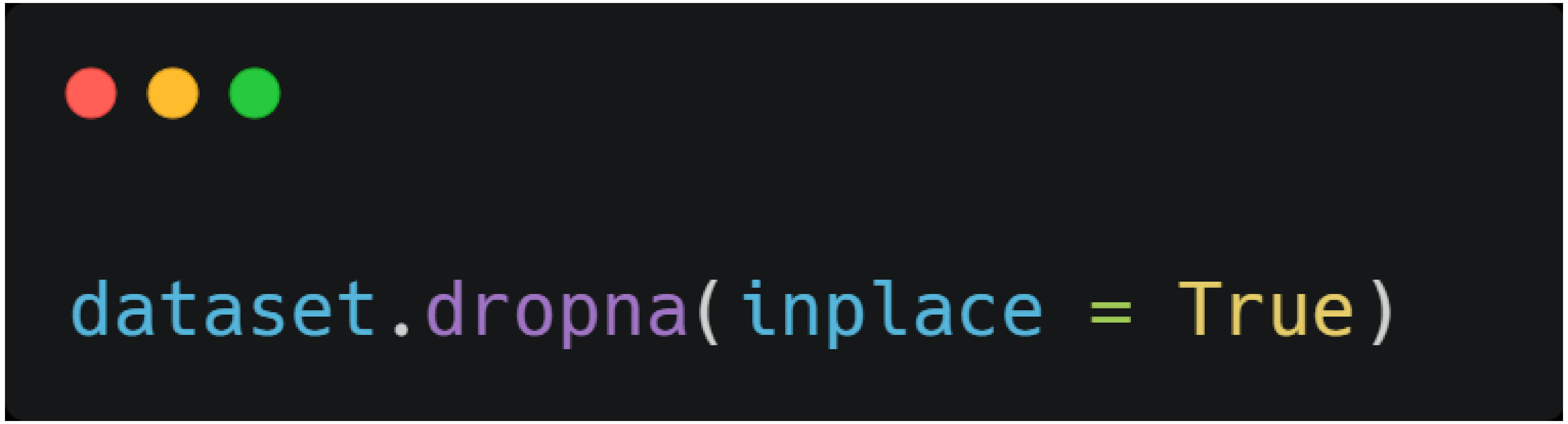
```
dataset.fillna(X, inplace = True)
```

mean, median, mode



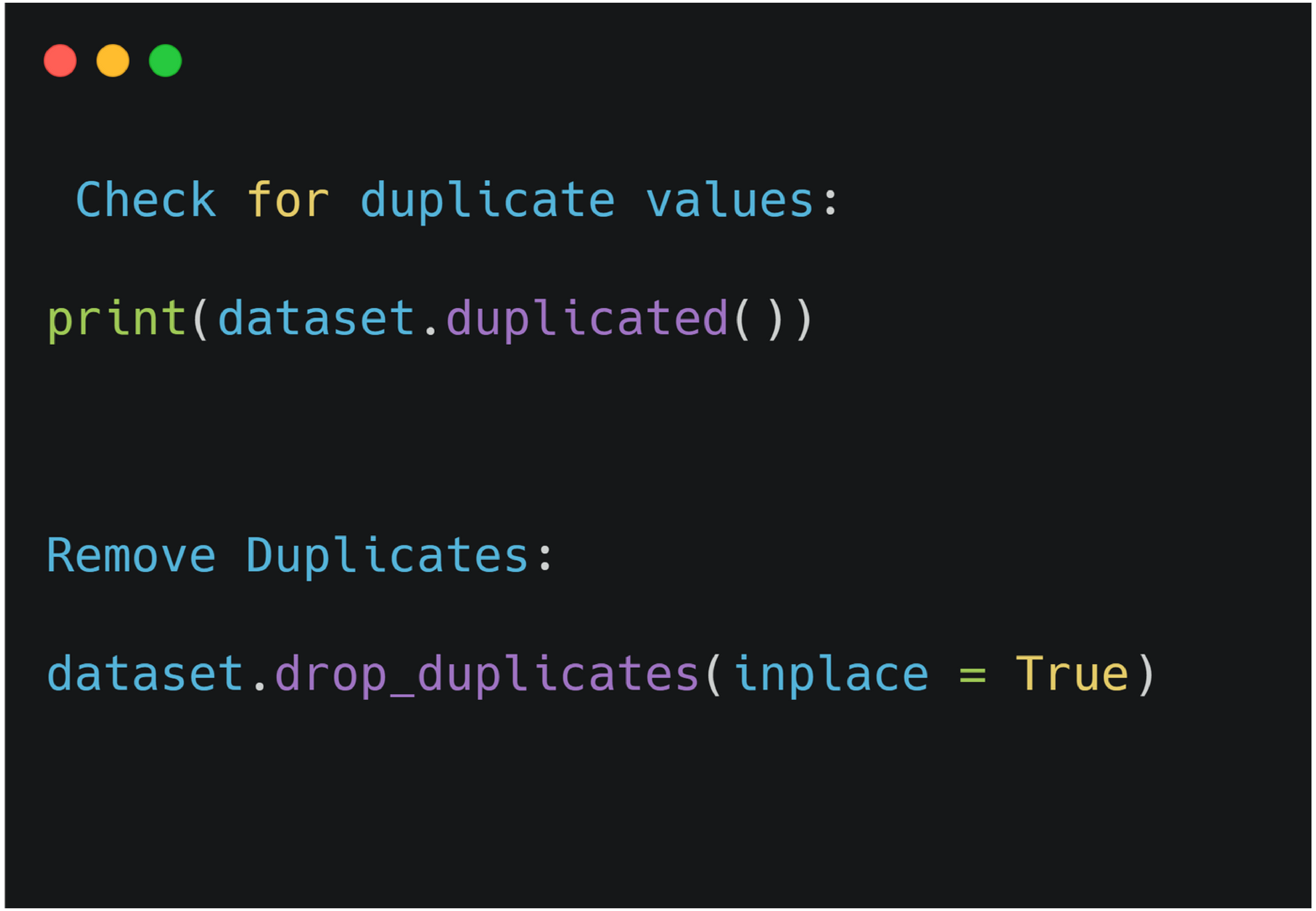
```
x = dataset["Calories"].mean()

y = dataset["Calories"].median()

z = dataset["Calories"].mode()

dataset["Calories"].fillna(x, inplace = True)
```

Dropping null values



```
dataset.dropna(inplace = True)
```

Removing duplicate values



```
 Check for duplicate values:

print(dataset.duplicated())



Remove Duplicates:

dataset.drop_duplicates(inplace = True)
```