Minor in AI

Pandas I

25 Feb 2025

Class Notes

Topics: Data Frames, Clean and prepare data frames

Pandas was created by Wes McKinney in 2008 while he was working at a financial firm called AQR Capital Management. He needed a powerful and flexible tool for analyzing large financial datasets, but existing tools like Excel and R were either too slow or lacked essential functionalities. The name "pandas" comes from the term "Panel Data", which is a statistical term referring to multidimensional structured datasets commonly used in econometrics and data analysis.

Before pandas, Python lacked a dedicated data analysis library. NumPy was useful for numerical computing, but it did not handle structured data (tables with labels) efficiently. pandas bridged the gap by introducing DataFrames, which made Python a strong alternative to R for data science.

Concepts discussed in class:

- Time series data, cross sectional data and panel data
- Data Frames
- In place
- Why we need query processing
- Applications from e-commerce sites
- Missing values and cleaning data frames

Program on what Pandas can do:

```
import pandas as pd
# Sample data for 5 e-commerce products
data = {
    'Product Domain': ['Electronics', 'Clothing', 'Electronics', 'Home & Kitchen', 'Books'],
    'Model': ['Smartphone X', 'T-Shirt', 'Laptop Pro', 'Coffee Maker', 'Python Programming'],
    'Color': ['Black', 'Blue', 'Silver', 'Red', 'N/A'],
```

```
    'Price': [899.99, 19.99, 1299.99, 49.99, 29.99],
    'Rating': [4.5, 4.0, 4.8, 3.5, 5.0]
}

# Create the DataFrame
df = pd.DataFrame(data)

# Display the entire DataFrame
print("Original DataFrame:\n", df)

# Accessing columns
print("\nProduct Models:\n", df['Model'])

# Accessing rows using index
print("\nSecond product:\n", df.iloc[1])

# Filtering data
print("\nProducts with rating >= 4.5:\n", df[df['Rating'] >= 4.5])

# Sorting data
print("\nProducts sorted by price:\n", df.sort_values(by='Price'))
```

Operation to understand the capability

```
import pandas as pd
df = pd.read_csv("sales_data-1.csv")
print(df)
df.fillna({'Price': df['Price'].mean()}, inplace=True)
print(df)
df['Product'] = df['Product'].str.replace('-', ' ').str.title()
print(df)
df.plot()
```

Understanding Basics in Pandas - Data Types

```
data = {'name': ['Alice', 'Bob', 'Charlie'], 'age': [25, 30, 35]}
df1 = pd.DataFrame(data)

data2 = {'id': [1, 2, 3], 'age': [25, 30, 35]}
df2 = pd.DataFrame(data2)

print(df1.dtypes, "\n")
print(df2.dtypes)
print("\n")

df1['name'] = df1['name'].astype('string')
print(df1.dtypes)
```

Working with Data Frames

```
import pandas as pd

# Sample DataFrame (replace with your actual data)
data = {'Name': ['Aryan', 'Aman', 'Akhil', 'Akash', 'Ayush'],
        'Age': [25, 30, 22, 28, 24],
        'City': ['Pune', 'Blore', 'Hyd', 'Pune', 'Delhi'],
        'Salary': [60000, 60000, 45000, 70000, 55000]}
df = pd.DataFrame(data)

# Displaying data
print("Head:\n", df.head(2))  # First 2 rows
print("\nTail:\n", df.tail(2))  # Last 2 rows

# Selecting columns
names = df['Name']
print("\nNames:\n", names)

# Selecting multiple columns
name_and_age = df[['Name', 'Age']]
```

```
print("\nName and Age:\n", name_and_age)

# Selecting rows and columns using loc[] (label-based indexing)
akhil_info = df.loc[2, ['Name', 'Age']] # Row with label 0, columns "Name" and "Age"
print("Akhil Info:\n", akhil_info)

# Selecting rows and columns using iloc[] (integer-based indexing)
aryan_info = df.iloc[0, [0, 1]] # Row with index 1, columns with indices 0 and 1
print("Aryan Info:\n", aryan_info)

print("\n")
young_people1 = df.loc[df['Age'] < 25]
print("Below 25:\n", young_people1)

print("\n")
# Filtering and conditional selection
young_people2 = df[df['Age'] < 25]
print("Below 25:\n", young_people2)

# Multiple conditions
high_earners_in_london = df[(df['Salary'] > 50000) & (df['City'] == 'Pune')]
print("High Earners in Pune:\n", high_earners_in_london)
```

Working with Missing Values

```
import pandas as pd

# Create a sample DataFrame with some missing values
data = {'Team': ['India', 'Australia', 'England', 'South Africa'],
        'Score': [250, None, 200, 220],
        'Overs': [50, 50, 45, None],
        'Result': ['Won', 'Lost', 'Won', 'Lost']}
df = pd.DataFrame(data)
```

```
# Check for missing values
print("Missing values:")
print(df.isnull())

# Drop rows with any missing values
df_dropped = df.dropna()
print("\nDataFrame after dropping rows with missing values:")
print(df_dropped)

# Fill missing values with a specific value (e.g., 0)
df_filled = df.fillna(0)
print("\nDataFrame after filling missing values with 0:")
print(df_filled)
```

```
import pandas as pd
import numpy as np
# Create a sample DataFrame with missing values
data = {'Student': ['Alice', 'Bob', 'Charlie'],
        'Test 1': [8, np.nan, 7],
        'Test 2': [9, 6, 8],
        'Test 3': [7, np.nan, 9],
        'Test 4': [6, 8, 7],
        'Test 5': [10, 7, 9]}
df = pd.DataFrame(data)

# Fill missing values with 5
df_filled_with_5 = df.fillna(5)

# Print the original, mean filled, and 5 filled DataFrames
print("Original DataFrame:\n", df)
print("\nDataFrame filled with 5:\n", df_filled_with_5)
```