Slip 18

18.1

import java.util.Observable;

import java.util.Observer;

// WeatherData is the concrete subject that extends Observable

class WeatherData extends Observable {

private float temperature;

private float humidity;

private float pressure;

public void measurementsChanged() {

setChanged();

notifyObservers();

}

public void setMeasurements(float temperature, float humidity, float pressure) {

this.temperature = temperature;

this.humidity = humidity;

this.pressure = pressure;

measurementsChanged();

}

public float getTemperature() {

return temperature;

}

public float getHumidity() {

return humidity;

}

public float getPressure() {

return pressure;

}

}

// DisplayElement is an interface implemented by concrete observers

interface DisplayElement {

void display();

}

// CurrentConditionsDisplay is a concrete observer

class CurrentConditionsDisplay implements Observer, DisplayElement {

private float temperature;

private float humidity;

private Observable observable;

public CurrentConditionsDisplay(Observable observable) {

this.observable = observable;

observable.addObserver(this);

}

@Override

public void update(Observable obs, Object arg) {

if (obs instanceof WeatherData) {

WeatherData weatherData = (WeatherData) obs;

this.temperature = weatherData.getTemperature();

this.humidity = weatherData.getHumidity();

display();

}

}

@Override

public void display() {

System.out.println("Current conditions: " + temperature + "F degrees and " + humidity + "% humidity");

}

}

// Client code to test the Observable pattern

public class WeatherStation {

public static void main(String[] args) {

// Create an observable (subject)

WeatherData weatherData = new WeatherData();

// Create observers (displays)

CurrentConditionsDisplay currentConditionsDisplay = new CurrentConditionsDisplay(weatherData);

// Simulate measurements change

weatherData.setMeasurements(80, 65, 30.4f);

}

}

18.2

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

dataset=pd.read\_csv('Position\_Salaries.csv')

x=dataset.iloc[:,1:-1].values

y=dataset.iloc[:,-1].values

print(dataset.head(5))

from sklearn.preprocessing import PolynomialFeatures

from sklearn.linear\_model import LinearRegression

p\_r=PolynomialFeatures(degree=4)

x\_poly=p\_r.fit\_transform(x)

lin\_reg=LinearRegression()

lin\_reg.fit(x\_poly,y)

LinearRegression()

y\_pred=lin\_reg.predict(x\_poly)

df=pd.DataFrame({'Real Values':y,'Predicted Values':y\_pred})

print(df)

x\_grid=np.arange(min(x),max(x),0.1)

x\_grid=x\_grid.reshape((len(x\_grid),1))

plt.scatter(x,y,color='yellow')

plt.scatter(x,y\_pred,color='red')

plt.plot(x\_grid,lin\_reg.predict(p\_r.fit\_transform(x\_grid)),color='black')

plt.title('Polynomial Regression')

plt.xlabel('position level')

plt.ylabel('Salary')

plt.show()