Slip 27

27.1

// Shape interface

interface Shape {

void draw();

}

// Concrete implementations of Shape

class Circle implements Shape {

@Override

public void draw() {

System.out.println("Drawing Circle");

}

}

class Rectangle implements Shape {

@Override

public void draw() {

System.out.println("Drawing Rectangle");

}

}

class Square implements Shape {

@Override

public void draw() {

System.out.println("Drawing Square");

}

}

// Abstract Factory interface

interface ShapeFactory {

Shape createShape();

}

// Concrete implementations of ShapeFactory

class CircleFactory implements ShapeFactory {

@Override

public Shape createShape() {

return new Circle();

}

}

class RectangleFactory implements ShapeFactory {

@Override

public Shape createShape() {

return new Rectangle();

}

}

class SquareFactory implements ShapeFactory {

@Override

public Shape createShape() {

return new Square();

}

}

// Client code using Abstract Factory

public class AbstractFactoryPatternExample {

public static void main(String[] args) {

// Creating a Circle using CircleFactory

ShapeFactory circleFactory = new CircleFactory();

Shape circle = circleFactory.createShape();

circle.draw();

// Creating a Rectangle using RectangleFactory

ShapeFactory rectangleFactory = new RectangleFactory();

Shape rectangle = rectangleFactory.createShape();

rectangle.draw();

// Creating a Square using SquareFactory

ShapeFactory squareFactory = new SquareFactory();

Shape square = squareFactory.createShape();

square.draw();

}

}

27.2

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()

27.3

cd myproject

python manage.py startapp myapp

python code

# myapp/views.py

from django.shortcuts import render

from django.http import HttpResponse

def index(request):

return HttpResponse("Hello! I am learning Django")

# myapp/urls.py

from django.urls import path

from .views import index

urlpatterns = [

path('', index, name='index'),

]

# myproject/urls.py

from django.contrib import admin

from django.urls import path, include

urlpatterns = [

path('admin/', admin.site.urls),

path('', include('myapp.urls')),

]

python manage.py runserver