گزارش سوال 4

ابتدا پکیج های لازم را ایمپورت میکنیم :

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from numpy.linalg import inv
```

سپس توابع مورد نیاز را پیاده سازی میکنیم :

```
def mean_squared_error(y_true, y_pred):
    return np.mean((y_true - y_pred) ** 2)
```

```
class Ridge:
    def __init__(self, alpha=1.0):
        self.alpha = alpha
        self.coef_ = None

    def fit(self, X, y):
        # Add a column of ones for the bias term
        X = np.c_[np.ones(X.shape[0]), X]

        # Calculate the coefficients using the Ridge formula
        n, m = X.shape
        identity_matrix = np.identity(m)
        self.coef_ = np.linalg.inv(X.T @ X + self.alpha * identity_matrix) @ X.T @ y

    def predict(self, X):
        # Add a column of ones for the bias term
        X = np.c_[np.ones(X.shape[0]), X]

        # Make predictions
        return X @ self.coef_
```

```python
class PolynomialFeatures:
    def __init__(self, degree=2):
        self.degree = degree

    def fit_transform(self, X):
        n, m = X.shape
        result = np.ones((n, 1))

        for d in range(1, self.degree + 1):
            result = np.concatenate([result, X ** d], axis=1)

        return result
```

اجرای رگرسیون خطی با توابع پایه چند جمله ای :

ابتدا دیتا را لود می کنیم سپس nan ها را حذف می کنیم. سپس خانه های خالی را با میانه پر می کنیم.

سپس با تابعی که پیاده سازی کردیم دیتا را به تست و ترین تقسیم می کنیم.

```python
def train_test_split(X, y, test_size=0.2, random_state=None):

    if random_state is not None:
        np.random.seed(random_state)

    num_samples = len(X)
    test_samples = int(test_size * num_samples)

    # Shuffle indices
    indices = np.random.permutation(num_samples)

    # Split the indices into training and test sets
    test_indices = indices[:test_samples]
    train_indices = indices[test_samples:]

    # Convert indices to integer values
    train_indices = train_indices.astype(int)
    test_indices = test_indices.astype(int)

    # Split the data based on the indices
    X_train, X_test = X.iloc[train_indices], X.iloc[test_indices]
    y_train, y_test = y.iloc[train_indices], y.iloc[test_indices]

    return X_train, X_test, y_train, y_test
```

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

در قسمت Polynomial basis functions 1 در ابتدا یک آرایه از درجه‌های چندجمله‌ای از 1 تا 10 تعیین می‌شود سپس پارامتر های ridge تعیین می شود.
ارزیابی درجه‌های مختلف برای هر درجه چندجمله‌ای:

ویژگی‌ها به صورت جداگانه برای هر متغیر ایجاد می‌شوند. به ازای هر متغیر، توان‌های مختلف (تا درجه تعیین شده) ایجاد می‌شوند.

ویژگی‌های ایجاد شده برای هر متغیر به یکدیگر اضافه می‌شوند

مدل روی داده‌های آموزش برای هر درجه چندجمله‌ای با استفاده از توابع چندجمله‌ای ایجاد شده و پارامتر های بهینه محاسبه می شود.
نمودارهای خطا بر حسب درجه چندجمله‌ای روی داده‌های آموزش و تست ترسیم می‌شوند.

در قسمت 2 هم همان مراحل لود و مرتب کردن دیتا را انجام می دهیم.

این کد برای آموزش یک مدل رگرسیون چندجمله‌ای بر روی داده‌های آموزش و سپس نمایش نتایج بر روی داده‌های تست استفاده می‌شود.
ابتدا داده به دو بخش آموزش و تست تقسیم می‌شود. این تقسیم بر اساس یک ماسک تصادفی انجام می‌شود که ٪80 از داده را برای آموزش و ٪20 را برای تست اختصاص می‌دهد.

تعیین درجه چندجمله‌ای: درجه چندجمله‌ای برای ساخت ویژگی‌ها تعیین می‌شود (در اینجا 3).

ایجاد ویژگی‌های چندجمله‌ای: بر اساس درجه مشخص شده

حل معادلات نرمال: پارامترهای مدل با استفاده از حل معادلات نرمال محاسبه می‌شوند.

پیش‌بینی نتایج: بر اساس پارامترهای محاسبه شده، مقدار پیش‌بینی بر روی داده‌های آموزش و نمونه‌های جدید برای تست انجام می‌شود.

نمایش نتایج: داده‌های آموزش و تست به همراه منحنی چندجمله‌ای یادگیری شده بر روی یک نمودار نمایش داده می‌شوند.

در قسمت 3 ارزیابی اثرات درجه‌های مختلف چندجمله‌ای بر روی یک مدل رگرسیون انجام می شود.
تعیین درجه‌های چندجمله‌ای: یک مجموعه از درجه‌های چندجمله‌ای (از 1 تا 10) برای ایجاد ویژگی‌های چندجمله‌ای انتخاب می‌شود.

تعیین پارامتر افزایشی برای جلوگیری از overfitting و سپس محاسبه خطا ها.
نمایش نتایج: خطاهای آموزش و تست بر حسب درجه چندجمله‌ای روی یک نمودار نمایش داده می‌شوند تا بتوانیم بهترین درجه را انتخاب کنیم.

این نمودار مفید است تا ببینیم چه تأثیری افزودن ویژگی‌های چندجمله‌ای به مدل دارد و چطور این تأثیر بر خطاهای آموزش و تست است.

در قسمت 1 Gaussian basis functions یک مدل رگرسیون با استفاده از توابع پایه گاوسی به عنوان ویژگی‌ها برای همگرایی داده‌ها به سمت یک تابع مقدار حقیقی ساخته می‌شود.

انتخاب داده‌های آموزش و تست: ابتدا از داده‌های اولیه، 100 نقطه به عنوان داده‌های آموزش و داده‌های تست انتخاب می‌شود.

تنظیم پارامترها که در اینجا پهنای باند مهم است
ساخت توابع پایه: توابع گاوسی بر اساس داده‌های آموزش و تست به صورت مجموعه‌ای از توابع گاوسی محاسبه می‌شوند. توابع گاوسی از فرمول گاوسی عادی استفاده می‌کنند.

افزودن ایجاد یک عبارت با ویژگی‌ها: یک عبارت متجه به توابع پایه اضافه می‌شود تا یک ترم بایاس اضافه شود.

حل معادلات نرمال: با حل معادلات نرمال، پارامترهای مدل (ضرایب) محاسبه می‌شوند.

پیش‌بینی و محاسبه خطاها و نمایش نتایج.

در قسمت 2 مراحل اصلی این طور است :

انتخاب داده‌های آموزش و تست: ابتدا از داده‌های اولیه، 100 نقطه به عنوان داده‌های آموزش و داده‌های تست انتخاب می‌شود.

تنظیم پارامترها: پارامترهای مهم در این مدل شامل تعداد توابع پایه هستند.

ساخت توابع پایه گاوسی: توابع گاوسی بر اساس داده‌های آموزش و تست به صورت مجموعه‌ای از توابع گاوسی محاسبه می‌شوند. توابع گاوسی از فرمول گاوسی عادی استفاده می‌کنند.

افزودن ایجاد یک عبارت با ویژگی‌ها: یک عبارت متجه به توابع پایه اضافه می‌شود تا یک ترم بایاس اضافه شود.

حل معادلات و پیش‌بینی و محاسبه خطاها و نمایش نتایج.