دانشگاه صنعتی امیر کبیر

(پلی تکنیک تهران)

دانشکده ریاضی و علوم کامپیوتر

تمرین اول درس یادگیری ماشین

عنوان: رگرسیون خطی

موعد تحویل:

۱. Linear Regression (رگرسیون خطی)

در این تمرین شما رگرسیون خطی چند متغیره[۱] را پیاده سازی خواهید کرد. این تمرین طولانی به نظر می رسد، اما در عمل شما تنها حدود ۱۰ خط کد را پیاده سازی می کنید. این تمرین برای آشنایی شما با مکانیزم کتابخانه scikit-learn و پیاده سازی الگوریتم های یادگیری ماشین در پایتون طراحی شده است.

Files and API (فایل‌ها و توابع)

در این تمرین یک سری فایل و توابع پایتون در اختیار شما قرار دارد:

`test_linreg_univariate.py`: اسکریپتی برای تست کردن رگرسیون خطی تک متغیره[۲]

- plotData۱D: رسم نمودار نقطه‌ای(پراکندگی)[۳] دیتای یک بعدی
- plotRegLine۱d: بوسیله تابع plotData۱D نمودار نقطه‌ای دیتا را رسم می‌کند و بوسیله پارامترهای یادگرفته شده توسط الگوریتم، خط رگرشن را در همان نمودار رسم می‌کند.
- visualizeObjective: رسم نموار surface و contour تابع هزینه[۴] (شما این تابع را تغییر نمی‌دهید)

`test_linreg_multivariate.py`: اسکریپتی برای تست کردن رگرسیون خطی متغیره

`linreg.py`: اسکریپی شامل کدهای رگرسیون خطی

LinearRegression: کلاس رگرسیون خطی چند متغیره

- __init__: کانستراکتور کلاس
- fit: تابعی برای یادگیری[۵] مدل رگرسیون خطی چند متغیره
- predict: تابعی برای پیش بینی ورودی جدید توسط مدل یاد گرفته شده
- computeCost: محاسبه مقدار تابع هزینه
- gradientDescent: بهینه سازی پارامترهای مدل توسط الگوریتم گرادیان کاهشی[۶]

Data Sets (مجموعه داده‌ها واقع در فولدر data)

- univariateData.dat: مجموعه داده‌ها برای مسئله رگرسیون تک متغیره
- multivariateData.dat: مجموعه داده‌ها برای مسئله رگرسیون چند متغیره

Visualizing the Data (بصرسازی داده‌ها)

بصرسازی داده‌ها بینش ارزشمندی از مسئله به ما ارائه می دهد، اما اغلب به عنوان بخشی از فرآیند یادگیری ماشین نادیده گرفته می‌شود. ما با رسم مجموعه داده‌های تک متغیره با استفاده از نمودار پراکندگی ۲ بعدی شروع خواهیم کرد. با این حال ما معمولاً با مجموعه داده‌های چند بعدی مواجه هستیم. هنگامی که از دو بعد فراتر می رویم، تجسم

---

۱ multivariate linear regression
۲ univariate linear regression
۳ scatter plot
۴ cost function (objective function)
۵ train
۶ gradient descent

بسیار دشوارتر می‌شود. در چنین مواردی یا باید هر بعد را به طور جداگانه رسم کنیم، یا از تکنیک های کاهش ابعاد (مانند PCA) برای کاهش تعداد ویژگی‌ها[۲] استفاده کنیم. بعداً در کلاس در مورد چنین تکنیک هایی صحبت خواهیم کرد.

با اجرای دستورات زیر در مفسر پایتون از داخل دایرکتوری hw۱ می توانید داده‌های تک متغیره را در متغیرهای X و y بصورت ماتریس بارگذاری کنید:

```
import numpy as np

filePath = "data/univariateData.dat"

file = open(filePath, 'r')

allData = np.loadtxt(file, delimiter=',')

X = np.matrix(allData[:,:-1])

y = np.matrix((allData[:,-1])).T

# get the number of instances (n) and number of features (d)

n, d = X.shape
```

سپس بوسیله‌ی تابع plotData۱D نمودار پراکندگی را رسم کنید، خروجی شما باید مانند Figure ۱ باشد:

```
from test_linreg_univariate import plotData1D

plotData1D(X, y)
```

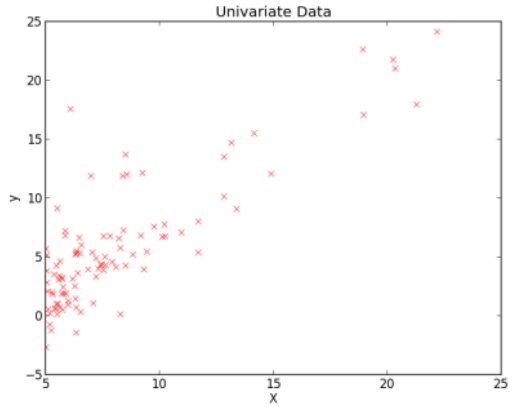

Figure 1: Scatter plot of the 1D data

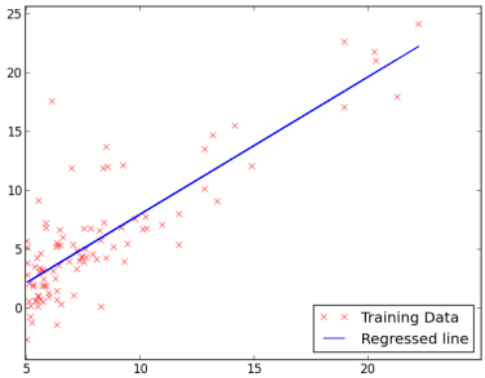

Figure 2: Regressed line of the 1D data

[۲] features

Implementation (پیاده سازی)

با تکمیل کلاس LinearRegression، رگرسیون خطی چند متغیره را از طریق گرادیان کاهشی اجرا کنید. اسم توابع (APIها) را ابدا تغییر ندهید. قسمت‌هایی از کد که باید تغییر دهید با کامنت "TODO" مشخص شده اند.

الگوریتم رگرسیون خطی بعد از یادگیری پارامترهای مدل به کمک دیتا، آن‌ها را بر روی بردار[۸] θ ذخیره می‌کند[۹]. در این تمرین از گرادیان کاهشی برای یافتن جواب بهینه استفاده می کنیم. دقت کنید که تابع هزینه رگرسیون خطی L۲، محدب است، بنابراین الگوریتم گرادیان کاهشی مینیمم مطلق را پیدا می کند.

$$(۱) \quad \hat{\theta} = min\, J(\theta)$$

$$(۲) \quad J(\theta) = \frac{1}{2n}\sum_{i=1}^{n}(h_\theta(x^{(i)}) - y^{(i)})^2$$

$$(۳) \quad h_\theta(x) = \theta^T x$$

تابع h_θ در رگرسیون خطی تک متغیره (ماتریس X تنها یک ستون دارد) به این شکل $h_\theta(x) = \theta^T x = \theta_0 + \theta_1 x$ است، که θ_0 همان عرض از مبدا[۱۰] است. برای هندل کردن عرض از مبدا در قالب معادله‌ی (۳) می توانیم یک ویژگی جدید به تمام سطرهای داده با مقدار ۱ اضافه کنیم. در واقع θ_0 را به عنوان ضریب x_0 در نظر می‌گیریم. برای اضافه کردن عرض از مبدا به کل مجموعه داده‌ها می توانیم ستونی از یک ها را به ماتریس X اضافه کنیم:

```
X = np.c_[np.ones((n, ۱)), X]
```

الگوریتم گرادیان کاهشی برای یافتن مقدار مینیمم تابع$J(\theta)$ فضای θهای ممکن را جست و جو می‌کند. حلقه for اولیه گرادیان کاهشی برای شما پیاده سازی شده است. شما فقط باید معادله را به روز رسانی کنید. در هر مرحله[۱۱] از گرادیان کاهشی باید به کمک معادله زیر **بصورت همزمان** همه پارامترها را به روز رسانی کنید.

$$(٤) \quad \theta_j \leftarrow \theta_j - \alpha\frac{1}{n}\sum_{i=1}^{n}(h_\theta(x^{(i)}) - y^{(i)})\, x_j^{(i)}$$

در هر مرحله از گرادیان کاهشی با به روز رسانی θ، به مقدار مینیمم تابع $J(\theta)$ نزدیک‌تر می‌شویم. متغیر α نرخ یادگیری[۱۲] است که آن را بسیار کوچک در نظر می‌گیریم (مثلا $\alpha = ۰.۰۱$). همیشه مقدار اولیه θ را یک مقدار کوچک تصادفی در حوالی صفر انتخاب می‌کنیم (می توان از توزیع نرمال با واریانسی کوچک و میانگین صفر استفاده کرد).

در نهایت تابع هزینه $J(\theta)$ (معادله‌ی ۲) را در قالب تابع computeCost در کلاس LinearRegression پیاده سازی می‌کنیم.

---

[۸] vector
[۹] fit
[۱۰] bias term
[۱۱] iteration
[۱۲] learning rate

مشکلات متداول:

- در هر مرحله از گرادیان کاهشی پارامترهای θ را بصورت همزمان آپدیت کنید. یعنی درهر مرحله بعد از به روز رسانی یکی از اعضای بردار θ، بردارθ را بصورت جزیی به روز رسانی نکنید و مقدار $h_\theta(x^i)$ را محاسبه کنید. در پایان حلقه بردار θ را کاملا به روز رسانی کنید.
- به یاد داشته باشید در هر مرحله از گرادیان کاهشی شما تنها فضای θهای ممکن را جست و جو می‌کنید و نباید X و y را تغییر دهید.

تست کردن کد نهایی:

یک راه ساده برای تست کردن کد چاپ کردن مقدار $J(\theta)$ در هر مرحله است. اگر مقدار آن با گذر زمان در هر مرحله بصورت یکنواخت در حال کاهش است یعنی کد شما درست کار می‌کند.

بعد از اتمام پیاده‌سازی کد، مدل خود را بر روی داده‌های univariateData آموزش دهید و سپس تابع plotRegLine۱D در فایل test_linreg_univariate.py را اجرا کنید.

```
from test_linreg_univariate import plotRegLine١D

from linreg import LinearRegression

X = np.c_[np.ones((n,١)), X]      # if you didn't this step already

lr_model = LinearRegression(alpha = ٠,٠١, n_iter = ١٥٠٠)

lr_model.fit(X, y)

plotRegLine١D(lr_model, X, y)
```

خروجی باید یک نمودار مانند Figure ۲ باشد.

Understanding Gradient Descent

در این قسمت نیازی به پیاده سازی کد ندارید. برای درک بهتر کد پیاده سازی شده در مرحله قبل، تابع هزینه و مسیر رسیدن به مینیمم آن که توسط الگوریتم گرادیان کاهشی انتخاب شده را رسم می‌کنیم.

برای مجموعه داده‌های تک متغیره می‌توانیم تغییر تابع هدف را در فضای θ_0 و θ_1 به‌عنوان یک نمودار سطحی[۱۳] و یک نمودار کانتور[۱۴] ترسیم کنیم تا شکل محدب و نزولی آن را نشان دهیم. خط آبی در شکل ۳ مسیر طی شده توسط نزول گرادیان را نشان می دهد و نقاط سرخابی نقاط را در هر تکرار نشان می دهد.

[۱۳] Surface plot
[۱۴] Contour plot

موعد تحویل:

معیار ارزیابی شما:

گزارش کار:

۱. نکته مهم در گزارش نویسی روشن بودن پاسخ ها می‌باشد، اگر فرضی برای حل سوال استفاده میکنید حتما آن را ذکر کنید، اگر جواب نهایی عددی است به صورت واضح آن را بیان کنید.
۲. هرگونه شباهت در گزارش به منزله تقلب میباشد و کل نمره تمرین صفر می‌باشد.
۳.