دانشگاه صنعتی امیرکبیر

(پلی‌تکنیک تهران)

دانشکده ریاضی و علوم کامپیوتر

تمرین اول درس یادگیری ماشین

عنوان تمرین : Regression

استاد درس

دکتر اکبری

تاریخ انتشار : 24 مهر

تاریخ برگزاری کلاس رفع اشکال : 29 مهر

مهلت تحویل : 5 آبان

تاریخ برگزاری کارگاه : 6 آبان

بخش اول : سؤالات تشریحی

۱.۱) باتوجه‌به الگوریتم گرادیان کاهشی، فرمول به‌روزرسانی وزن‌ها و bias را برای تابع هزینه زیر محاسبه کنید.

$$h_{a,b}(x) = \sum_{j=1}^{n} a_j \sin(b_j + x_j) \ , \ J(a,b) = \frac{1}{2}\sum_{i=1}^{n}\left(y_i - h_{a,b}(x_i)\right)^2$$

۱.۲) باتوجه‌به تابع هزینه بالا، آیا الگوریتم گرادیان کاهشی همواره به یک جواب یکسان و البته بهینه همگرا می‌شود ؟ تحلیل خود را توضیح دهید.

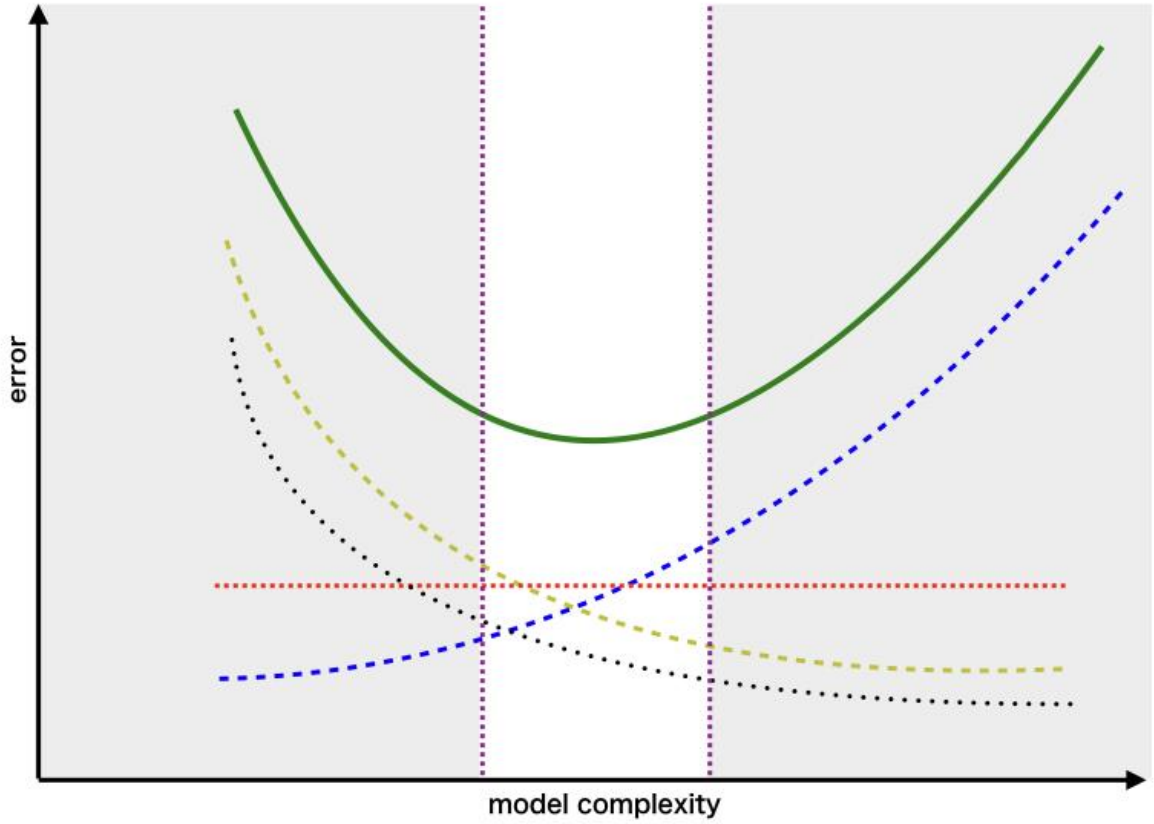
۲.۱) تابع توزیع احتمال زیر را در نظر بگیرید که در آن x یک عدد حقیقی مثبت است.

$$P_\theta(x) = 2\theta x e^{-\theta x^2}$$

فرض کنید m نمونه مستقل و با توزیع یکسان از این تابع تولید شده است. برای نمونه‌های تولید شده، مقدار θ بهینه را به کمک maximum likelihood estimator محاسبه کنید.

۳.۱) در نمودار زیر که نشان‌دهنده تأثیر تغییر پیچیدگی مدل بر میزان خطاست، هر یک از موارد خواسته شده را مشخص کنید.

- خطای آموزش
- واریانس
- مربع بایاس
- خطای ناشی از نویز
- خطای تست
- محدوده overfitting
- محدوده underfitting



بخش دوم : Reading Assignment

در این بخش باید درباره یکی از موضوعات زیر که در کلاس تدریس نشده، تحقیق و مطالعه کنید. هدف از این تمرین آشنایی با روند یادگیری مطالب جدید و همین‌طور ارائه این مطالب است. گزارش تحویلی شما باید بین دو الی سه صفحه باشد.

۱. **Active Linear Regression** (لینک منبع کمکی)

۲. **Empirical Risk Minimization**

(منبع کمکی : فصل سوم کتاب Bishop و The Elements of Statistical Learning)

۳. **LASSO Regression** (منبع کمکی : فصل سوم کتاب Bishop)

۴. **Beysian Linear Regression** (منبع کمکی : فصل سوم کتاب Bishop)

❖ نحوه تخصیص موضوعات به شکل تصادفی بوده و از طریق فرمول (n % 4 + 1) به دست می‌آید که n دو رقم آخر شماره دانشجویی شما است.

بخش سوم : پیاده‌سازی

در این بخش شما رگرسیون خطی چندمتغیره را پیاده‌سازی خواهید کرد و با پیاده‌سازی الگوریتم‌های یادگیری ماشین در پایتون آشنا می‌شوید.

فایل‌ها و توابع

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

test_linreg_univariate.py : ماژول تست کردن رگرسیون خطی تک‌متغیره

- plotData1D : رسم نمودار نقطه‌ای (پراکندگی) دیتای یک‌بعدی
- plotRegLine1D : به‌وسیله تابع plotData1D نمودار نقطه‌ای دیتا را رسم می‌کند و به‌وسیله پارامترهای یاد گرفته شده توسط الگوریتم، خط رگرسیون را در همان نمودار رسم می‌کند.
- visualizeObjective : رسم نمودار surface و contour تابع هزینه (کد این تابع را تغییر ندهید)

test_linreg_multivariative.py : ماژول تست کردن رگرسیون خطی چندمتغیره

linreg.py : ماژول کدهای رگرسیون خطی

LinearRegression : کلاس رگرسیون خطی چندمتغیره

- __init__ سازنده کلاس
- fit تابعی برای یادگیری مدل رگرسیون خطی چندمتغیره
- predict تابعی برای پیش‌بینی ورودی جدید توسط مدل آموزش‌دیده
- computeCost محاسبه مقدار تابع هزینه
- gradiantDescent بهینه‌سازی پارامترهای مدل توسط الگوریتم گرادیان کاهشی

مجموعه‌داده‌ها (واقع در پوشه data)

UnivariateData.dat : مجموعه‌داده برای مسئله رگرسیون تک‌متغیره

MultivariateData.dat : مجموعه‌داده برای مسئله رگرسیون چندمتغیره

holdout.npz : مجموعه تست برای ارزیابی رگرسیون چندمتغیره

مصورسازی داده‌ها

مصورسازی داده‌ها بینش ارزشمندی از مسئله به ما ارائه می‌دهد، اما اغلب به‌عنوان بخشی از فرایند یادگیری ماشین نادیده گرفته می‌شود. ما با رسم مجموعه‌داده‌های تک‌متغیره با استفاده از نمودار پراکندگی ۲ بعدی شروع

خواهیم کرد. اما معمولاً با مجموعه‌داده‌های چندبعدی مواجه هستیم. هنگامی که از دو بعد فراتر می‌رویم، تجسم بسیار دشوارتر می‌شود. در چنین مواردی یا باید هر بعد را به طور جداگانه رسم کنیم، یا از تکنیک‌های کاهش ابعاد (مانند PCA) برای کاهش تعداد ویژگی‌ها استفاده کنیم. به کمک دستورات زیر در فایل test_linreg_multivariate.py داده‌های تک‌متغیره در متغیرهای x و y به‌صورت ماتریس ذخیره می‌شوند.

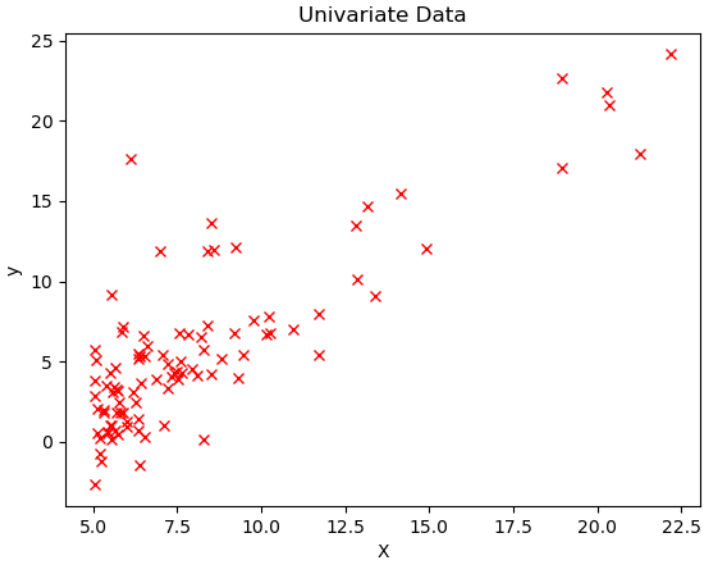
```
filePath = "data/univariateData.dat"
file = open(filePath,'r')
allData = np.loadtxt(file, delimiter=',')

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

n,d = X.shape
```

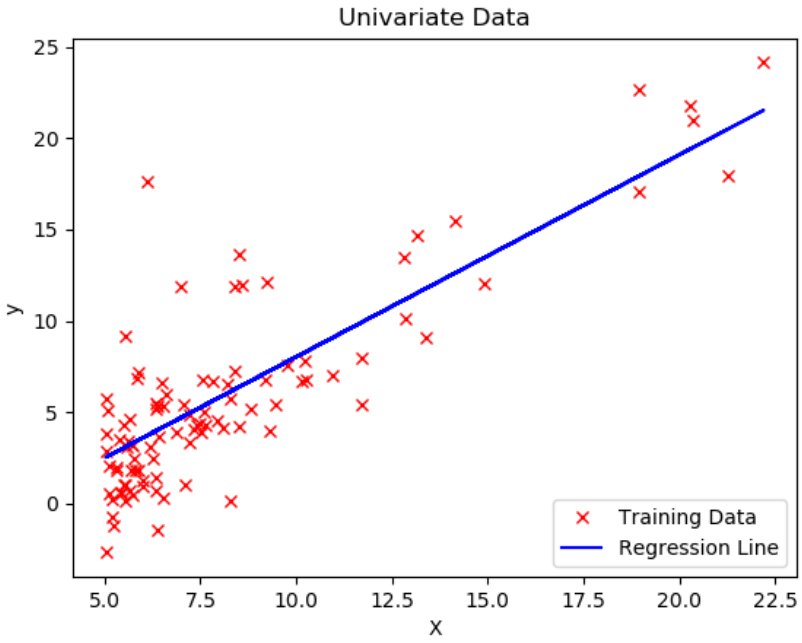
تصویر ۱ : نحوه ذخیره بردار ویژگی و بردار هدف

سپس به‌وسیلهٔ تابع plotData1D نمودار پراکندگی را رسم می‌شوند و خروجی شما باید مانند تصویر ۲ باشد.



تصویر ۲ : چگونگی توزیع داده‌های ورودی

درنهایت بعد از fit شدن مدل رگرسیون خطی رو داده‌ها خروجی مدل همانند تصویر ۳ به نمایش گذاشته می‌شود.



تصویر ۳ : خروجی مدل رگرسیون خطی

پیاده‌سازی

با تکمیل کلاس LinearRegression ، رگرسیون خطی چندمتغیره را از طریق گرادیان کاهشی اجرا کنید. اسم توابع را ابداً تغییر ندهید. قسمت‌هایی از کد که باید تغییر دهید با کامنت TODO مشخص شده‌اند.

الگوریتم رگرسیون خطی بعد از یادگیری پارامترهای مدل به کمک دیتا، آنها را بر روی بردار θ ذخیره می‌کند. در این تمرین از گرادیان کاهشی برای یافتن جواب بهینه استفاده می‌کنیم. دقت کنید که تابع هزینه رگرسیون خطی به‌صورت L2norm و محدب است، بنابراین الگوریتم گرادیان کاهشی مینیمم مطلق را پیدا می‌کند

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

$$J(\theta) = \frac{۱}{۲n}\sum_{i=۱}^{n}(h_\theta(x^{(i)}) - y^{(i)})^۲ \quad (۲)$$

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

تابع $h(\theta)$ در رگرسیون خطی تک‌متغیره (ماتریس X تنها یک ستون دارد) و به شکل $h_\theta(x) = \theta^T x = \theta_0 + \theta_1 x$ است، که θ_0 همان بایاس است. برای هندل کردن عرض از مبدا در قالب معادله (۳) می‌توانیم یک ویژگی جدید به تمام سطرهای داده با مقدار ۱ اضافه کنیم. در واقع θ_0 را به‌عنوان ضریب x_0 در نظر می‌گیریم. برای اضافه کردن بایاس به کل مجموعه‌داده‌ها می‌توانیم ستونی از یک ها را به ماتریس X اضافه کنیم.

```
X = np.c_[np.ones((n,1)), X]
```

تصویر ۴ : اضافه کردن X_0 به ماتریس X

الگوریتم گرادیان کاهشی برای یافتن مقدار مینیمم تابع $J(\theta)$ فضای θهای ممکن را جست‌وجو می‌کند. حلقه for اولیه گرادیان کاهشی برای شما پیاده‌سازی شده است. شما فقط باید معادله را به‌روزرسانی کنید. در هر مرحله از گرادیان کاهشی باید به کمک معادله زیر به‌صورت هم‌زمان همه پارامترها را به‌روزرسانی کنید.

$$\theta_j \leftarrow \theta_j - \alpha \frac{1}{n} \sum_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} \quad (۴)$$

در هر مرحله از گرادیان کاهشی با به‌روزرسانی θ، به مقدار مینیمم تابع $J(\theta)$ نزدیک‌تر می‌شویم. متغیر α نرخ یادگیری است که آن را بسیار کوچک در نظر می‌گیریم (مثلاً $\alpha = 0/01$). همیشه مقدار اولیه θرا یک مقدار کوچک تصادفی در حوالی صفر انتخاب می‌کنیم (می‌توان از توزیع نرمال با واریانسی کوچک و میانگین صفر استفاده کرد). در نهایت تابع هزینه $J(\theta)$ را در قالب تابع computeCost در کلاس LinearRegression پیاده‌سازی می‌کنیم.

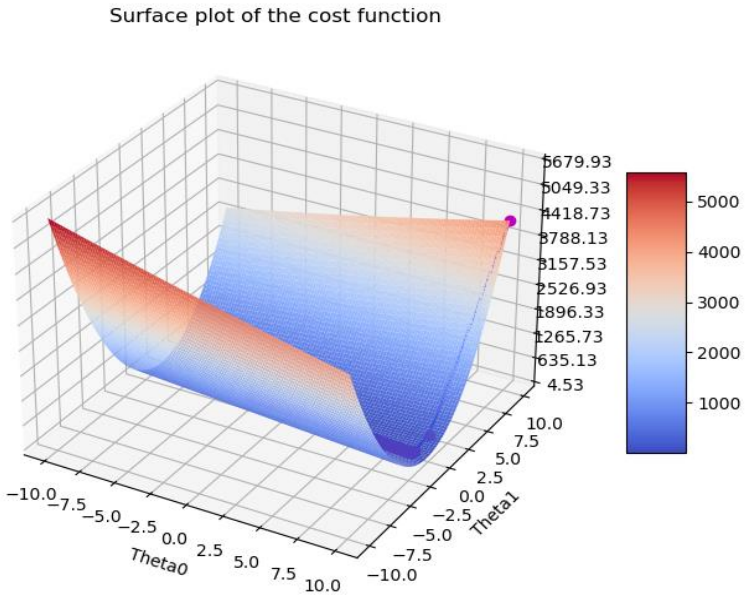
مشکلات متداول

- در هر مرحله از گرادیان کاهشی پارامترهای θ را به‌صورت هم‌زمان آپدیت کنید. یعنی در هر iteration بعد از به‌روزرسانی یکی از اعضای بردار θ، بردار θ را به‌صورت جزئی به‌روزرسانی نکنید و مقدار $h_\theta(x^i)$ را محاسبه کنید. باید در پایان حلقه بردار θ را یکجا به‌روزرسانی کنید.
- به یاد داشته باشید در هر مرحله از گرادیان کاهشی شما تنها فضای θ های ممکن را جست‌وجو می‌کنید و نباید x و y را تغییر دهید.

ارزیابی و تست کردن کدهای پیاده‌سازی شده

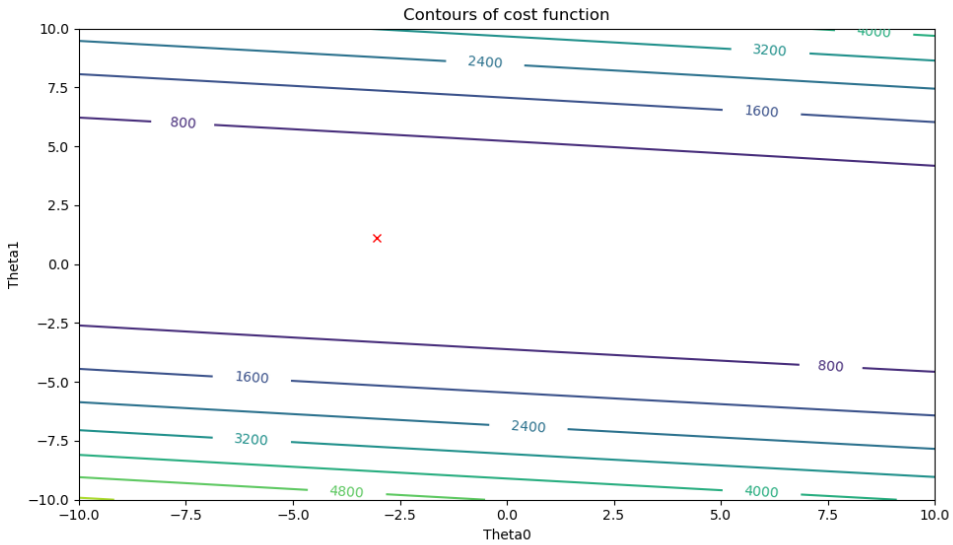
یک راه ساده برای تست کردن کد چاپ کردن مقدار $J(\theta)$ در هر مرحله است. اگر مقدار آن با گذر زمان در هر مرحله به‌صورت یکنواخت در حال کاهش است؛ یعنی کد شما به‌درستی کار می‌کند. بعد از اتمام پیاده‌سازی کد، مدل خود را بر روی داده‌های UnivariateData.dat آموزش دهید. بعد از اتمام آموزش مدل نمودار های مختلفی به شما نمایش داده می‌شود. برای این کار کافی است فایل test_linreg_univariate.py را اجرا کنید. ترتیب نمودار

های رسم شده به این شکل است که همانند تصویر ۲ ابتدا پراکندگی داده ها و سپس همانند تصویر ۳ نحوه fit شدن مدل روی داده ها نمایش داده می‌شود. سپس در نمودار بعدی که در تصویر ۴ مشاهده می کنید چگونگی یادگیری وزن ها و نحوه بروزرسانی وزن ها نمایش داده می‌شود.



تصویر ۴ : نمودار چگونگی بروزرسانی وزن ها

در نهایت در نمودار چگونگی تغییرات تابع هزینه همانند تصویر ۵ به شما نمایش داده می‌شود.



تصویر ۵ : نمودار تغییرات تابع هزینه

رگرسیون خطی چند متغیره

درصورتی‌که کد شما روی داده‌های تک‌متغیره درست کار می‌کند. اکنون برای پیاده‌سازی رگرسیون خطی چندمتغیره کافی است که فایل test_linreg_ multivariate.py را اجرا کنید. در این فایل روند پیاده‌سازی رگرسیون چندمتغیره همانند رگرسیون خطی تک‌متغیره است با این تفاوت که در این جا از دو ویژگی استفاده شده است. در نهایت شما با اجرای این فایل می‌توانید عملکرد مدل را روی مجموعه‌داده MultivariateData.dat مشاهده کنید.

سریع‌تر کردن کد با پیاده‌سازی به شکل بردار

اغلب می‌توانیم با برداری کردن کد، الگوریتم‌های یادگیری ماشین را بسیار سریع‌تر و مختصرتر کنیم (اگرچه با مجموعه‌داده‌های کوچکی که در این تمرین استفاده کردیم تفاوت محسوسی حس نمی‌شود.) پیاده‌سازی قبلی خود را برای محاسبه تابع هزینه و به‌روزرسانی گرادیان را با استفاده از عملیات ماتریسی (مثلاً بدون استفاده از حلقه for در معادله) تغییر دهید. به‌عنوان‌مثال فرم ماتریسی (برداری شده) تابع هزینه به شکل زیر است:

$$J(\theta)=\frac{1}{2n}(X\theta-y)^{T}(X\theta-y)$$

محاسبهٔ خطای مدل بر روی داده

در نهایت مجموعه دادهٔ holdout.npz در فولدر data را به‌وسیله کتابخانه numpy لود کنید، سپس کد خود را بر روی این دیتا اجرا کنید و میزان خطا (MSE , RMSE) را در گزارش کار ذکر کنید. برای پیاده‌سازی این بخش در فایل Model_Evaluation.ipynb توابع موردنیاز شما نوشته شده است. شما کافی است calculate_loss را تکمیل کنید. در این تابع بردار مقادیر واقعی و بردار مقادیر پیش‌بینی‌شده به‌عنوان ورودی گرفته می‌شود و کد شما باید مقدار (MSE , RMSE) را محاسبه کند و به‌عنوان خروجی برگرداند. بعد از پیاده‌سازی این تابع میزان خطا را به‌ازای مقادیر مختلف iteration و α به دست آورید و با تنظیم این دو هایپرپارامتر مقدار بهینه خطا را روی مجموعه‌داده holdout.npz گزارش دهید. برای تنظیم هایپرپارامتر می‌توانید از رویکرد های مختلفی مثل Grid search و ... استفاده کنید. برای انجام این کار مطالعه این document به شما کمک خواهد کرد. توجه کنید که در صورت نیاز می‌توانید نحوه پیاده‌سازی توابع فایل Model_Evaluation.ipynb را تغییر دهید و از سایر روش‌های تنظیم هایپرپارامترها استفاده کنید.

پیاده‌سازی به کمک Sklearn

تا این جای کار شما توانستید مدل‌های رگرسیون یک و چندمتغیره را از صفر پیاده‌سازی کنید و به کمک مجموعه‌داده holdout.npz عملکرد مدل رگرسیون چندمتغیره را ارزیابی کردید. در بخش آخر این تمرین از شما

می‌خواهیم که مدل رگرسیون چندمتغیره را به کمک کتابخانه Sklearn و مدل‌های آماده این کتابخانه پیاده‌سازی کنید. مراحل پیاده‌سازی به‌صورت زیر است:

۱. همانند بخش قبلی به کمک کتابخانه numpy مجموعه‌داده MultivariateData.dat را load کنید و بردارهای ویژگی و برچسب را مشخص کنید. دقت کنید که در این بخش به‌جای numpy matrix از numpy array استفاده کنید.
۲. در مرحله بعدی بردار ویژگی‌ها را به کمک StandardScaler نرمال‌سازی کنید. این دو مرحله برای داده‌های تست یعنی holdout.npz نیز انجام دهید.
۳. سپس در این مرحله یک شی از کلاس LinearRegression بسازید و مدل ایجاد شده را روی داده‌های آموزش fit کنید.
۴. درنهایت در مرحله آخر، داده‌های تست را به متد predict از کلاس LinearRegression پیش‌بینی کنید و میزان خطا (MSE , RMSE) را گزارش دهید.
۵. درنهایت نتایج به‌دست‌آمده را با نتایج قسمت قبلی تمرین مقایسه کنید و تحلیل خود را از نتایج به‌دست‌آمده ارائه دهید.

هنگام پیاده‌سازی این بخش به موارد زیر توجه کنید.

- ✓ دقت کنید به کمک attribute های _coef و _intercept می‌توانید مقدار وزن‌ها و Bias را به دست آورید و با مقادیر به‌دست‌آمده در قسمت قبلی تمرین مقایسه کنید.
- ✓ معمولاً هنگام پیاده‌سازی مسائل و پروژه‌های یادگیری ماشین فرض می‌شود که به مجموعه‌داده تست دسترسی نداریم و تنها مجموعه در دسترس ما مجموعه‌داده آموزش است. هنگام نرمال‌سازی مجموعه‌داده تست به این موضوع دقت کنید.
- ✓ درصورتی‌که با کتابخانه scikit-learn و کلاس های LinearRegression و StandardScaler آشنایی ندارید مطالعه لینک های مشخص شده به شما برای پیاده‌سازی این بخش کمک بسیاری خواهد کرد.

معیار ارزیابی شما

بخش اول سؤالات تشریحی (۲۰ نمره) : سؤال اول (۱۰ نمره) / سؤال دوم و سوم (هرکدام ۵ نمره)

بخش دوم Reading Assignment (۲۰ نمره)

بخش سوم پیاده‌سازی (۱۰۰ نمره) :

- تکمیل پیاده‌سازی رگرسیون تک‌متغیره و چندمتغیره (در کد با "TODO" مشخص شد) (۴۰ نمره)
- سریع‌تر کردن کد پیاده‌سازی شده به کمک بردارها (۲۰ نمره)

- تنظیم هایپرپارامترها برای رسیدن به کمترین خطای مدل بر روی holdout.npz (۱۵ نمره)
- پیاده‌سازی مسئله به کمک کتابخانه Scikit-learn و تحلیل و مقایسه نتایج به‌دست‌آمده (۲۵ نمره)
- این تمرین حدود ۱ نمره از ۲۰ نمره نهایی شما را شامل می‌شود.

نکات تکمیلی

- ✓ انجام این تمرین بسته به تسلط شما به مطالب درس و زبان پایتون حداقل بین ۳ الی ۵ روز از وقت مفید شما را خواهد گرفت. به همین علت انجام این تمرین را به‌روزهای پایانی موکول نکنید. همین‌طور باتوجه‌به برنامه فشرده کلاس و حجم زیاد مطالب، مهلت تحویل این تمرین تمدید نخواهد شد.
- ✓ لطفاً مکان فایل‌های داده شده را تغییر ندهید. با انجام این کار با error مواجه خواهید شد.
- ✓ مشاهدات و استنباط خود از قسمت‌های مصورسازی داده‌ها، تست کردن کد، تحلیل نتایج به‌دست‌آمده و محاسبه مقدار خطای مدل بر روی holdout.npz را در قالب یک گزارش بنویسید. آپلود این قسمت اجباری است، در غیر این صورت نمره تمرین را نخواهید گرفت. (اگر کد شما به توضیحات بیشتر نیاز داشت می‌توانید به گزارش ضمیمه کنید.)
- ✓ نکته مهم در گزارش‌نویسی و سؤال تشریحی روشن بودن پاسخ است نه حجم زیاد، اگر فرضی برای حل سؤال استفاده می‌کنید حتماً آن را ذکر کنید، و پاسخ نهایی را به‌صورت واضح بیان کنید. گزارش کد و پاسخ سؤال تشریحی به‌صورت فایل pdf باشد.
- ✓ هرگونه شباهت در گزارش و پاسخ تشریحی به منزله تقلب است و کل نمره تمرین را نخواهید گرفت.
- ✓ فایل pdf مربوط به بخش اول و دوم تمرین و همین‌طور گزارش مربوط به بخش پیاده‌سازی را به همراه کدها را به‌صورت یکجا در قالب یک فایل zip در سامانه کورسز آپلود کنید (نام فایل = شماره دانشجویی.)
- ✓ در صورت هرگونه ابهام درباره این تمرین می‌توانید در کلاس رفع اشکال سؤالات خودتان رو بپرسید و یا از طریق ایمیل‌های زیر با ما در ارتباط باشید.

محمدعلی سفیدی اصفهانی : mohammadali.esfahani@aut.ac.ir

ملیکا سپیدبند : melikasepidband@aut.ac.ir

با آرزوی سلامتی و موفقیت برای شما عزیزان