

دانشگاه صنعتی امیر کبیر
(پلی تکنیک تهران)

دانشکده مهندسی کامپیوتر

تمرین اول درس یادگیری ماشین

دکتر ناظر فرد

غلامرضا دار ۴۰۰۱۳۱۰۱۸

پاییز ۱۴۰۰

فهرست مطالب

بخش اول: پرسشهای تشریحی	۳
سوال (۱)	۳
سوال (۲)	۴
سوال (۳)	۵
سوال (۴)	۸
سوال (۵)	۹
سوال (۶)	۱۰
بخش دوم: پیاده سازی	۱۱
سوال (۱)	۱۱
سوال (۲)	۲۴
سوال امتیازی)	۳۰
منابع	۳۵

بخش اول: پرسشهای تشریحی

سوال (۱)

درستی یا نادرستی عبارات زیر را با ذکر دلیل مشخص کنید.

الف) در یک الگوریتم یادگیری، کاهش خطای آموزش، منجر به کاهش خطای آزمون میشود.

نادرست. زیرا ممکن است مدل ما داده ها را حفظ کرده باشد و اصطلاحاً **Overfit** شده باشد. همچنین مدلی قابلیت **Generalization** خوبی ندارد و با دیدن داده جدید (**Test data**) نمی تواند حدس درستی بزند.

ب) با کاهش تعداد داده های آموزش، الگوریتم بیشتر مستعد بیش برازش میشود.

درست. هرچه تعداد داده ها کمتر باشد احتمال اینکه بتوانیم پترن های کلی داده ها را ببینیم بیشتر می شود این نویز ها با بقیه داده ها میانگین گرفته می شوند و تاثیر کمتری بر تصمیم نهایی می گذارند.

ج) افزایش پیچیدگی مدل در رگرسیون همیشه سبب کاهش خطای آموزش و افزایش خطای آزمون میشود.

نادرست. افزایش پیچیدگی همواره باعث کاهش خطای آموزش می شود اما همیشه باعث افزایش خطای آزمون نمی شود. اتفاقاً برای مدلی که پیچیدگی بسیار کمی دارد، افزایش پیچیدگی می تواند خطای آزمون را به طرز قابل توجهی کاهش می دهد.

د) از میان معیارهای **MSE**، **RMSE** و **MAE** معیار **MAE** در برابر داده های پرت و نویزی عملکرد بهتری دارد.

نادرست. معیار **MSE** به دلیل اینکه خطاها را به توان ۲ می رساند عملکرد مناسب تری در مقابل نویز دارد. **MAE** با مقادیر خطای کم و زیاد به طور یکسان برخورد میکند اما **MSE** خطاهای زیاد را بیشتر **Punish** می کند و در نتیجه در حضور نویز و داده پرت زیاد عملکرد بهتری دارد.

سوال ۲)

فرض کنید که از رگرسیون چندجمله‌ای استفاده میکنید و با رسم منحنی‌ها متوجه میشوید که اختلاف زیادی بین خطای آموزشی و خطای اعتبارسنجی وجود دارد. توضیح دهید که چه اتفاقی رخ داده است و سه راه حل برای حل این مشکل ارائه دهید.

بیش‌برازش یا Overfitting رخ داده و مدل، داده‌ی آموزش را به نوعی حفظ کرده است.

- راه حل اول کم کردن پیچیدگی مدل است. در مورد رگرسیون چندجمله‌ای می‌توان درجه چندجمله‌ای را کم کرد.
- راه حل دوم افزودن دیتاست تا پترن‌هایی که در سمپل داده آموزش ما وجود نداشته را هم به مدل نشان دهیم. همچنین اگر داده‌های Train, Valid به طور همگن تقسیم نشده‌اند بهتر است کل داده‌ها را شافل کنیم و بعد به دو دسته Train, Valid تقسیم کنیم.
- راه حل سوم استفاده از Regularization است که در این صورت مقادیر مربوط به وزن‌ها کوچکتر میشوند که به Generalization مدل کمک میکند.

سوال ۳

مجموعه داده آموزشی شامل n داده به فرم (x_i, x_j) در اختیار داریم که x_i دارای d بعد است. تابع هزینه SSE به صورت زیر محاسبه میشود.

$$J(w) = \sum_{i=1}^n (y_i - w^T x_i)^2$$

الف) نشان دهید که در رگرسیون خطی با تابع هزینه SSE ، w بهینه به این صورت است:

$$\hat{w} = (X^T X)^{-1} X^T y$$

ابتدا باید رابطه J را به شکل ماتریسی در بیاوریم. برای این کار یک Design matrix به شکل مقابل تهیه میکنیم:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$$J(w) = \frac{1}{2m} (Xw - y)^T (Xw - y)$$

$$J(w) = \frac{1}{2m} ((Xw)^T - y^T) (Xw - y)$$

$$J(w) = \frac{1}{2m} (Xw)^T Xw - (Xw)^T y - y^T (Xw) + y^T y$$

$$J(w) = \frac{1}{2m} (Xw)^T Xw - 2(Xw)^T y + y^T y$$

$$J(w) = \frac{1}{2m} w^T X^T Xw - 2w^T X^T y + y^T y$$

مشتق جزئی میگیریم (با کمک یک سری فرمول های مشتق ماتریس مثل $\frac{dX^T X}{dx} = 2X$)

$$\frac{\partial J(w)}{\partial w} = 2X^T Xw - 2X^T y = 0$$

$$X^T Xw - X^T y = 0$$

حالا طرفین را در اینورس $X^T X$ ضرب میکنیم.

$$w = (X^T X)^{-1} X^T y$$

ب) دو مشکلی که استفاده از این رابطه دارد را ذکر کنید و برای هر کدام راه حلی ارائه دهید.

مشکل اول معادله نرمال پیچیدگی زمانی محاسبه آن برای تعداد فیچر بالاست. وقتی تعداد فیچر ها بسیار زیاد باشد محاسبه اینورس ماتریس $n \times n$ کار زمان بری خواهد شد. راه حل این مشکل میتواند کاهش ابعاد به کمک روش هایی مثل PCA و روش های دیگر باشد که در نهایت تعداد فیچر ها کاهش می یابد.

مشکل دوم زمانی است که فیچر های وابسته داشته باشیم و اصطلاحاً فیچر ها redundant باشند. این امر باعث میشود اینورس ماتریس $X^T X$ وجود نداشته باشد. راه حل این مشکل میتواند feature selection با redundancy کمتر یا استفاده از pseudo inverse برای محاسبه اینورس ماتریس باشد. همچنین اگر تعداد فیچر ها بسیار زیاد باشد به طوری که $m \leq n$ نیاز است تعداد فیچر ها را کم کنیم یا از regularization استفاده کنیم.

ج) اگر یک جمله منظم ساز نرْم ۲ به صورت $\|w\|^2$ به رابطه $loss$ اضافه کنیم، فرم بسته w بهینه را بدست آورید.

$$J(w) = \sum_{i=1}^n (y_i - w^T x_i)^2 + \lambda w^T w$$

مانند قسمت الف سوال اما این بار یک جمله منظم ساز هم داریم

$$J(w) = \frac{1}{2m} (Xw - y)^T (Xw - y) + \lambda w^T w$$

$$J(w) = \frac{1}{2m} ((Xw)^T - y^T)(Xw - y + \lambda w^T w)$$

$$J(w) = \frac{1}{2m} (Xw)^T Xw - (Xw)^T y - y^T (Xw) + y^T y + \lambda w^T w$$

$$J(w) = \frac{1}{2m} (Xw)^T Xw - 2(Xw)^T y + y^T y + \lambda w^T w$$

$$J(w) = \frac{1}{2m} w^T X^T Xw - 2w^T X^T y + y^T y + \lambda w^T w$$

مشتق جزئی میگیریم

$$\frac{\partial J(w)}{\partial w} = 2X^T Xw - 2X^T y + 0 + 2\lambda w = 0$$

$$X^T Xw - X^T y + \lambda w = 0$$

$$X^T Xw + \lambda w = X^T y$$

$$(X^T X + \lambda I)w = X^T y$$

حالا طرفین را در اینورس $(X^T X + \lambda I)$ ضرب میکنیم.

$$w = (X^T X + \lambda I)^{-1} X^T y$$

د) توضیح دهید اضافه کردن جمله منظم‌ساز چه مزیت‌هایی نسبت به حالت عادی دارد.

اضافه کردن جمله منظم‌ساز باعث می‌شود مقدار زیاد وزن‌ها جریمه شود و در نهایت مقدار همه وزن‌ها عددی کوچک و متعادل باشد. با این کار هیچ ویژگی (Feature) خاصی بسیار پراهمیت‌تر از ویژگی‌های دیگر نخواهد بود. همچنین منظم‌سازی مدل را تشویق می‌کند ساختار ساده‌تری داشته باشد که این کار باعث می‌شود مدل از Overfitting دوری کند.

سوال ۴)

با در نظر گرفتن الگوریتم‌های زیر برای رگرسیون خطی، به سوالات پاسخ دهید.

۱- معادله نرمال

۲- روشهای مبتنی بر گرادیان نزولی شامل:

- a. Batch GD : گرادیان‌ها بر اساس کل داده آموزش محاسبه میشوند.
- b. Stochastic GD : گرادیان‌ها بر اساس یک نمونه‌گیری از داده محاسبه میشوند.
- c. Mini-Batch GD : گرادیان‌ها با استفاده از مجموعه‌های تصادفی کوچکی از نمونه‌ها به دست می‌آیند.

الف) اگر یک مجموعه داده آموزش با میلیون‌ها ویژگی در اختیار داشته باشید، از کدام الگوریتم رگرسیون خطی می‌توان استفاده کرد؟ دلایل انتخاب خود را شرح دهید.

انتخاب بهتر در این مورد الگوریتم‌های بر پایه **Gradient Descent** هستند. زیرا محاسبه ترم $(X^T X)^{-1}$ که در روش نرمال مورد نیاز است دارای پیچیدگی بالاتری از اجرای الگوریتم Gradient Descent حتی به تعداد Iteration های متعدد است.

$$O(n^3) > O(k \cdot n^2)$$

ب) فرض کنید که ویژگی‌های موجود در مجموعه داده آموزش در مقیاس‌های بسیار متفاوتی هستند. این ویژگی روی هر کدام از الگوریتم‌های مطرح شده، چه تاثیری میگذارد و راه حل شما چیست؟

این ویژگی در مورد الگوریتم Normal Equation تاثیری ندارد. زیرا اساس کار معادله نرمال ضرب و اینورس ماتریس هاست و مقیاس اعداد تاثیری در این محاسبات ندارد.

اما در مورد الگوریتم Gradient Descent اینکه مقادیر ویژگی‌ها در مقیاس‌های تقریباً یکسان (ترجیحاً -1 تا +1) باشد تاثیرگذار است زیرا در بازه‌های بزرگ سرعت Descend بیشتر از بازه‌های کوچک است و این ناهماهنگی ممکن است دچار غیربهبود شدن الگوریتم شود. راه حلی که برای این مشکل وجود دارد **Feature Scaling** نام دارد. پس از اعمال Feature Scaling تمام مقادیر ویژگی‌ها در بازه تقریباً یکسان و یکنواختی قرار میگیرند که این کار سرعت الگوریتم Gradient Descent را افزایش می‌دهد. یکی از معروف‌ترین راه‌ها برای Feature Scaling در ادامه ذکر شده :

$$x_i = \frac{x_i - \mu_i}{\sigma_i}$$

(که μ_i میانگین مقادیر ویژگی i ام است و σ_i انحراف از معیار مقادیر ویژگی i ام است.)

سوال ۵)

فرض کنید یک مدل با واریانس بالا داریم و برای بهبود عملکرد مدل تصمیم گرفته‌ایم که تعداد داده‌های آموزشی را افزایش دهیم. به نظر شما چه نتیجه‌ای حاصل خواهد شد؟ سوال قبل را در مورد مدلی که بایاس بالایی دارد نیز پاسخ دهید.

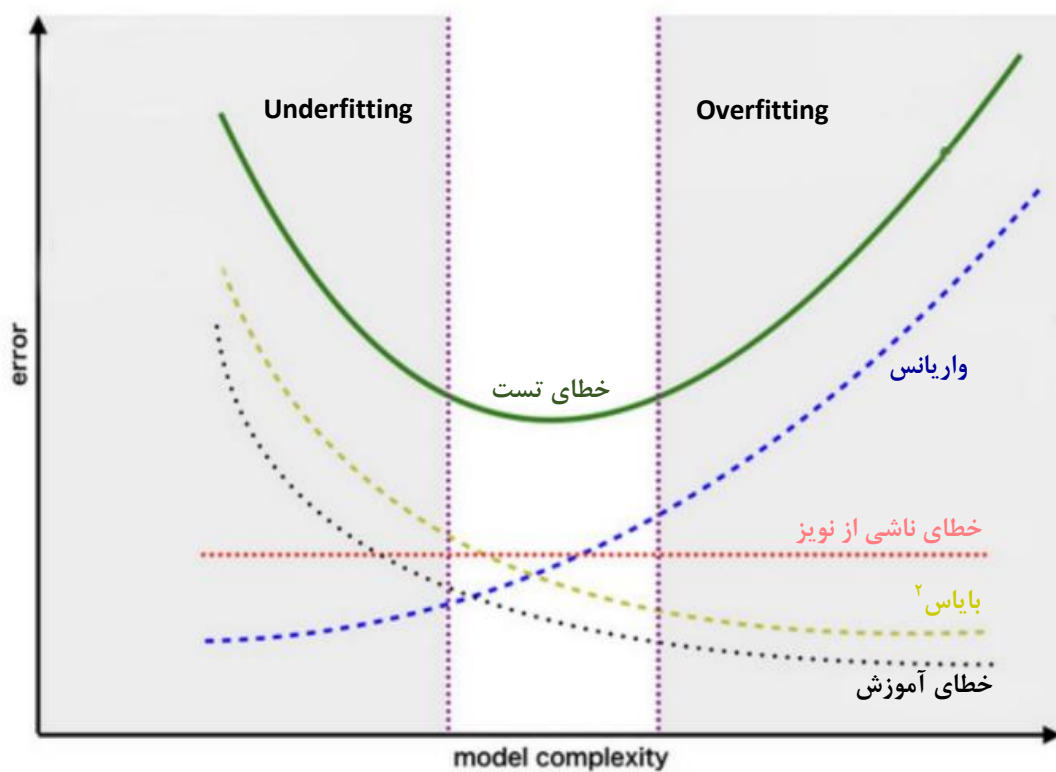
مدلی که واریانس بالایی دارد، Complexity نسبتاً بالایی دارد. می‌دانیم مدل پیچیده نیاز به دیتای آموزش زیادی دارد و از این بابت با افزایش داده‌ها کارایی مدل بهبود می‌یابد. اما باید توجه کنیم که اگر واریانس مدل بسیار زیاد باشد و دیتای زیادی هم به مدل دهیم احتمالاً مدل، دیتا را حفظ می‌کند و دچار Overfitting می‌شود. برای مقابله با بیش‌برازش و بهبود کارایی مدل می‌توانیم از راه‌هایی مانند **Regularization** و کاهش پیچیدگی مدل استفاده کنیم.

مدلی که بایاس بالایی دارد، Complexity نسبتاً پایینی دارد و فرض‌های ساده‌ای نسبت به توزیع داده‌ها می‌کند. افزایش تعداد داده‌ها در این حالت احتمالاً نمی‌تواند به مدل کمک زیادی بکند چون اندازه ذخیره‌سازی اطلاعاتِ مدل پایین است و مدل پس از مدتی اشباع می‌شود. در این مورد احتمالاً مدل دچار Underfitting شده و برای بهبود این شرایط راه‌هایی مثل اضافه کردن فیچرهای بیشتر و پیچیده‌تر و همچنین اضافه کردن تعداد پارامترهای بیشتر برای مدل یا در کل پیچیده کردن مدل می‌توانند موثر باشند.

سوال ۶)

در شکل داده شده موارد زیر را مشخص کنید.

خطای آموزشی	●
خطای ناشی از نویز	●
واریانس	●
مربع بایاس	●
خطای تست	●
محدوده Overfitting	
محدوده Underfitting	



بخش دوم: پیاده سازی

سوال (۱)

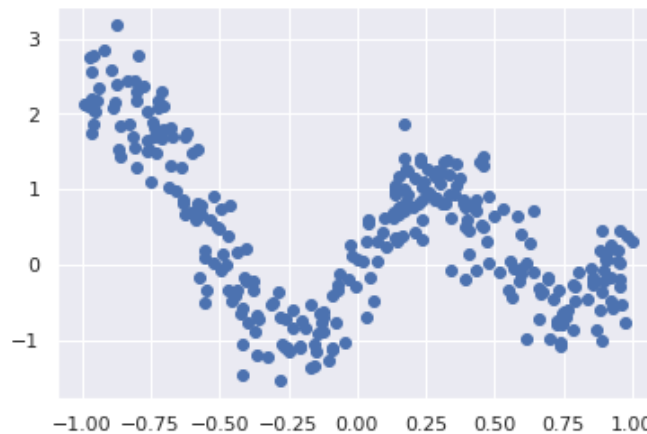
نوت بوک مربوطه به این تمرین :

https://colab.research.google.com/drive/1itjIBNliQ17kVK_gCL0o0ea8xor61Q3V?usp=sharing

مجموعه داده dataset1.csv را که در فولدر dataset1 قرار دارد در نظر بگیرید

الف) داده ها را رسم کنید.

داده ها دو ستون X, Y دارند. آنها را در یک دستگاه مختصات به طوری که $x=x', y=y'$ باشد رسم میکنیم.



شکل ۱ - رسم داده‌ها

ب) شافل کردن داده ها به چه منظور انجام می شود؟ آیا مجموعه داده این سوال نیازی به این اقدام دارد؟ در صورت لزوم این مورد را بر روی مجموعه داده اعمال کنید.

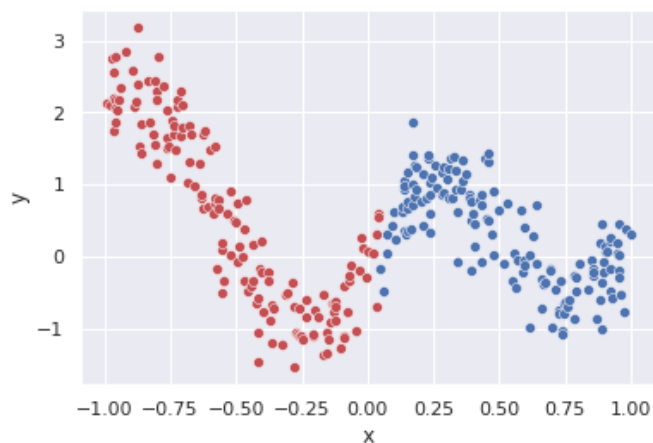
در بسیاری از دیتاست ها، داده ها ممکن است به ترتیب خاصی مرتب شده باشند. ترتیب های رایجی که مشاهده می شود مرتب بودن بر اساس نوع کلاس، مرتب بودن بر اساس یکی از ویژگی ها، مرتب بودن بر اساس تاریخ ثبت داده در دیتاست و ... هستند. این اتفاق باعث می شود هنگام تقسیم کردن داده ها به مجموعه Train, Validation, Test در داده های موجود در این مجموعه ها عدم توازن بوجود بیاید. مثلاً در مجموعه Test داده هایی با X بزرگتر و در مجموعه Train داده هایی با X کوچکتر قرار بگیرند که این امر از General بودن مدل میکاهد و خطای Overfitting را افزایش می دهد.

برای اینکه متوجه شویم ترتیب دیتا تاثیر خاصی دارد یا خیر میتوانیم نصف اول دیتاست را با یک رنگ و نصف دیگر را با رنگ دیگر رسم کنیم.



شکل ۲- نصف اول داده ها به رنگ قرمز و نصف دوم به رنگ آبی

همانطور که مشاهده می شود داده ها به خوبی Shuffle شده اند و ترتیب آمدن یک نقطه در دیتاست بر روی موقعیت مکانی نقطه تاثیر چندانی ندارد. بر خلاف این مثال اگر داده ها را به ترتیب X مرتب کنیم شکل زیر حاصل می شود.



شکل ۳- داده ها بر حسب X مرتب شده اند.

حالتی که در توضیحات اول این سوال به آن اشاره کردیم اتفاق افتاد. حالا با تقسیم کردن دیتا به دو مجموعه Train, Test با ضریب تقسیم بندی ۷۰ درصد اکثر داده های Train نقاط **قرمز** خواهند بود و اکثر داده های Test نقاط **آبی**. بنابراین نیازی به شافل کردن داده ها نمی باشد هر چند شافل کردن داده ها می تواند خیالمان را راحت تر کنند و به عنوان یک قانون کلی خوب است همیشه انجام شود.

ج) با استفاده از روش گرادیان نزولی نمودارهایی با درجه های ۵ و ۸ و ۱۰ و با تعداد تکرارهای ۵۰۰۰ و ۱۰۰۰۰ بر روی داده ها برازش دهید و نمودارهای حاصل را رسم کنید. این عمل را برای سه معیار خطای MSE و RMSE و MAE تکرار کنید و نتایج را مقایسه کنید. همچنین در صورت مشاهده ی بیش‌برازش آن را گزارش کنید. ر. ک. بخش د

د) برای هر یک از موارد قسمت قبل نمودار خطای آموزش و آزمون و نمودار اندازه قدم را رسم کنید. محور افقی نشان‌دهنده تکرارها و محور عمودی مقدار خطا و اندازه قدم را نشان دهد.

این بخش طولانی ترین قسمت سوال بود و در مراحل مختلفی انجام شد. در ادامه به این مراحل می‌پردازیم. همچنین قسمت ج سوال هم به مقدار زیادی به این بخش وابسته است پس در یک بخش توضیح داده می‌شوند.

مرحله اول : اضافه کردن یک ستون با مقادیر ۱ به داده، به عنوان Bias (طبق اطلاعات مربوط به اسلاید های درس)

	θ	x	y
0	1	0.097627	0.626964
1	1	0.430379	0.846452
2	1	0.205527	0.756017
3	1	0.089766	0.427504
4	1	-0.152690	-1.335228

شکل ۴- افزودن ستون Bias

مرحله دوم : تقسیم کردن داده ها به دو مجموعه آموزش و اعتبارسنجی (به نسبت ۷۰-۳۰)

```
Train X size = 210
Train y size = 210
Valid X size = 90
Valid y size = 90
```

شکل ۵ - Train, Valid split

مرحله سوم: تبدیل رگرسیون چند جمله ای که یک رگرسیون غیر خطی تک متغیره است به یک رگرسیون خطی چند


متغیره.

	θ	x	x^2	x^3	x^4	x^5	x^6
0	1	0.097627	0.009531	0.000930	0.000091	0.000009	8.658046e-07
1	1	0.430379	0.185226	0.079717	0.034309	0.014766	6.354843e-03
2	1	0.205527	0.042241	0.008682	0.001784	0.000367	7.537202e-05
3	1	0.089766	0.008058	0.000723	0.000065	0.000006	5.232170e-07
4	1	-0.152690	0.023314	-0.003560	0.000544	-0.000083	1.267274e-05

شکل ۶ - تبدیل رگرسیون چند جمله ای به خطی

این کار به شکل زیر انجام میپذیرد

$$h(x) = \theta_0 + \dots + \theta_i \boxed{x^i} + \dots + \theta_d x^d$$



$$h(x) = \theta_0 + \dots + \theta_i \boxed{x_i} + \dots + \theta_d x_d$$

شکل ۷ - نحوه تبدیل رگرسیون چند جمله ای به خطی

مرحله چهارم : از آنجایی که خواسته اصلی صورت سوال بسیار زمان گیر و پیچیده بود، برای شروع یک مثال ساده تر با مشخصات زیر را آموزش میدهیم. این کار در پروسه دیباگ و پیدا کردن Hyper-parameter های مناسب کمک به سزایی میکند.

جزئیات این مثال خاص :

Polynomial degree = 10

Iterations = 1000

Learning rate = 2.1

Decay = 5e-9

Loss function = MSE

پس از حدود ۳۲ ثانیه این مدل آموزش دید و خطای آموزش و اعتبارسنجی نهایی آن به شرح زیر بود :

Training loss = 0.2174

Validation loss = 0.2434

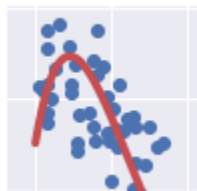
```
100% ██████████ 1000/1000 [00:32<00:00, 31.53it/s]
training_loss : 0.2174 | validation_loss : 0.2434
```

شکل ۸ - آموزش



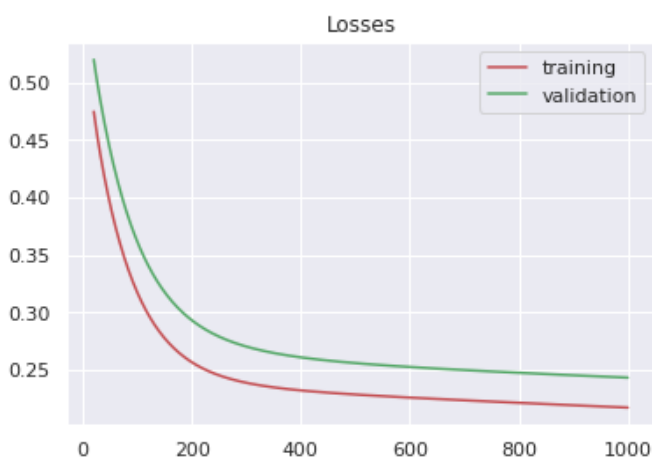
شکل ۹ - منحنی Fit شده روی داده ها

نمودار منحنی Fit شده توسط این مدل بر روی داده ها را در شکل ۹ مشاهده میکنید. منحنی با دقت بالایی به داده Fit شده. در این مرحله خوب بود کمی Regularization به مدل اضافه میکردیم تا بخش هایی مانند خمیدگی سمت چپ (شکل ۱۰) که به نظر اشتباه میرسد را یاد نمی گرفت و مدل جنرال تری بدست می آوردیم اما از آنجایی که خواسته سوال این نیست این ویژگی را اضافه نمیکنیم.



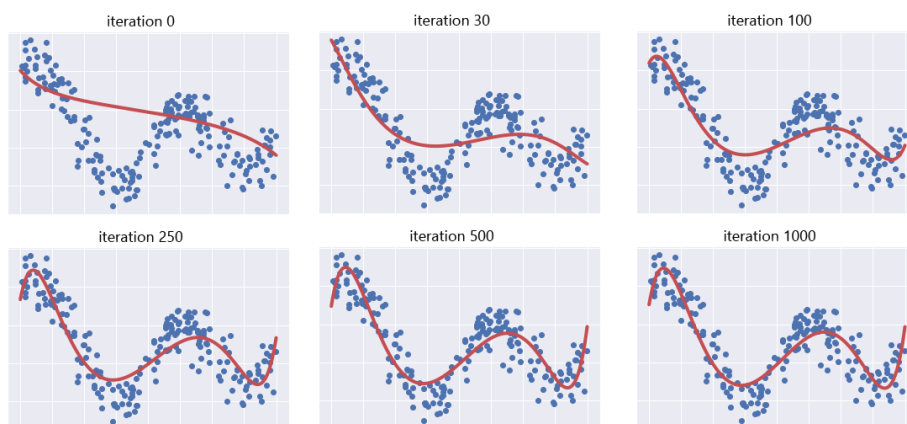
شکل ۱۰ - علائم احتمالی Overfitting

برای بررسی روند کاهش خطا در طی مراحل آموزش میتوانیم از نمودار خطا بر حسب iteration استفاده کنیم. شکل ۱۱ این نمودار را هم برای خطای آموزش و هم برای خطای اعتبارسنجی نشان میدهد. از این نمودار میتوان برای مشاهده علائم Overfitting, Underfitting استفاده کرد. همانطور که در نمودار مشاهده می شود رفته رفته خطای آموزش و اعتبارسنجی در حال کاهش هستند و خطای آموزش همیشه کمی کمتر از خطای اعتبارسنجی است که حالت مطلوب این نمودار است.



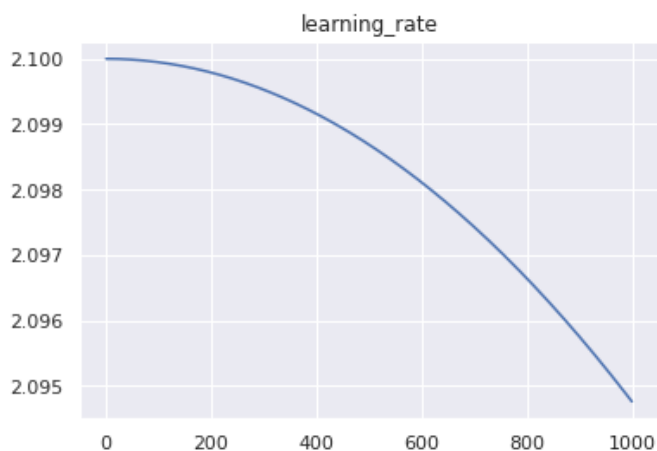
شکل ۱۱ - منحنی خطای آموزش و اعتبارسنجی بر حسب iteration

برای دیدن پیشرفت مدل به سمت ضرایب تتای بهینه میتوانیم هیستوری از θ های مختلف ذخیر کنیم و منحنی Fit شده را با استفاده از θ های از زمان های مختلف رسم کنیم.



شکل ۱۲ - روند نزدیک شدن منحنی به داده ها

در آخر روند کاهش learning rate در طی آموزش را رسم کردیم. (پیدا کردن مقدار مناسب برای این هایپر پارامتر کار دشواری بود و ممکن است هنوز هم بهترین مقدار انتخاب نشده باشد).

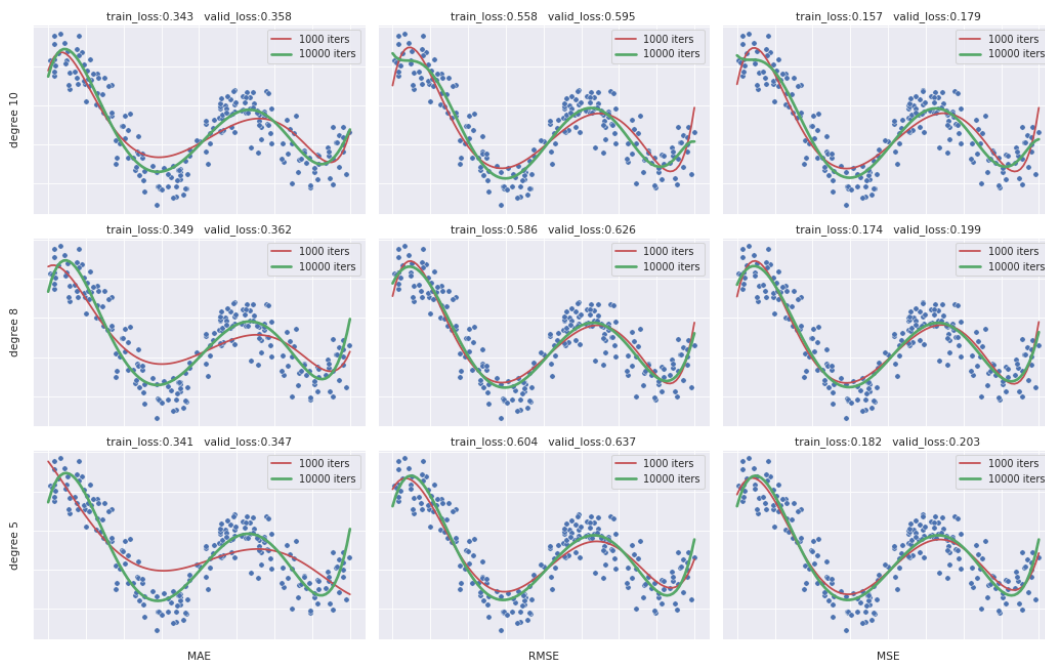


شکل ۱۳ - کاهش لرنینگ ریت بر حسب iteration

مکانیزمی که برای کاهش لرنینگ ریت انتخاب کردیم یک "Time-Based Learning Rate Schedule" ساده بود.

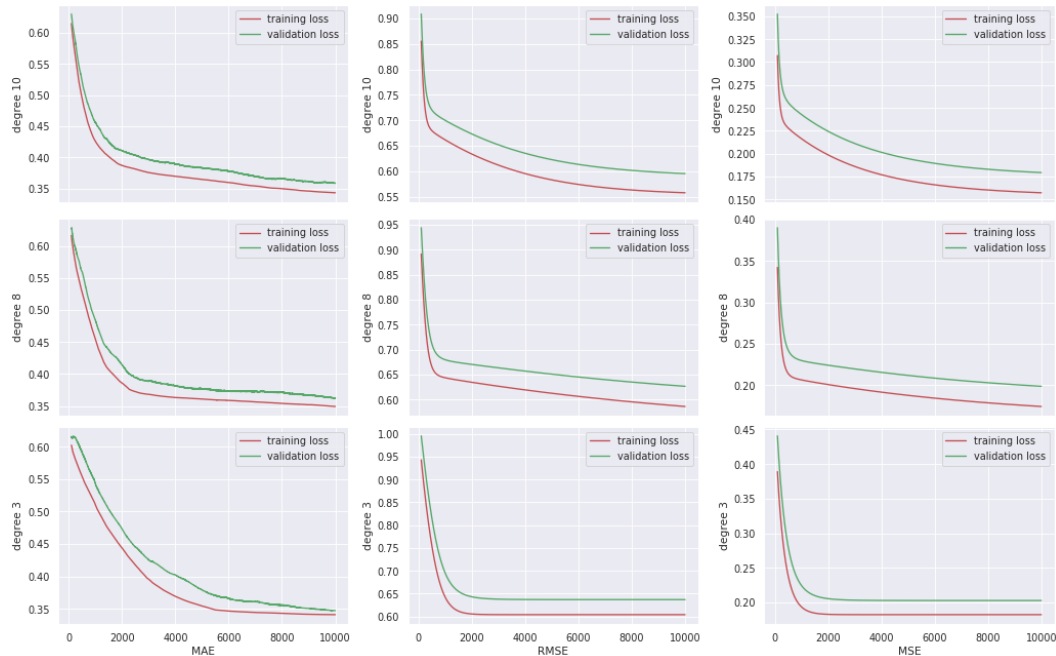
$$\text{LearningRate} = \text{LearningRate} * 1/(1 + \text{decay} * \text{iteration})$$

مرحله پنجم: در این قسمت خواسته اصلی سوال را اجرا میکنیم و تمام نمودارهایی که در مرحله قبل دیدیم را برای تمامی حالت های سوال رسم میکنیم.



شکل ۱۴ - منحنی Fit شده روی داده ها برای همه حالت ها

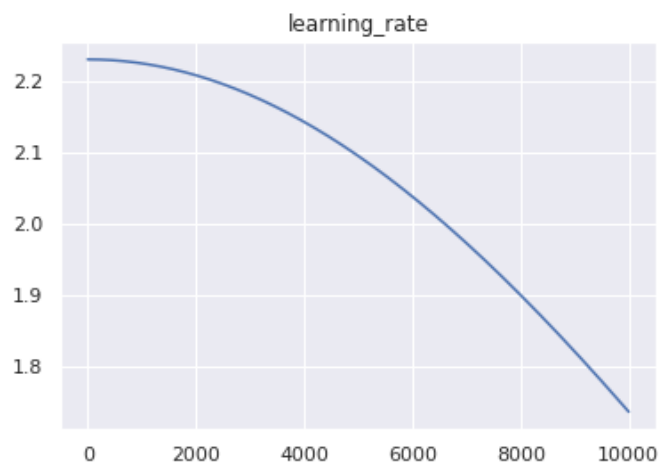
اکثر نکات مربوط به این نمودار را در بخش قبل توضیح دادیم. توجه کنید که منحنی **سبز رنگ** نشان دهنده منحنی فیت شده پس از ۱۰ هزار iteration است و منحنی **قرمز رنگ** منحنی فیت شده پس از هزار iteration (از قصد عدد ۱۰۰۰ بجای ۵۰۰۰ انتخاب شد تا تفاوت بین دو منحنی بهتر مشاهده شود) است. همچنین ستون ها به ترتیب از چپ به راست مربوط به لاس فانکشن های MAE, RMSE, MSE هستند و سطر ها از پایین به بالا مربوط به درجه چند جمله ای ۵, ۸, ۱۰ هستند.



شکل ۱۵- منحنی لاس بر حسب iteration برای همه حالات

در مورد شکل ۱۵ اگر دقت کنید منحنی درجه ۵ با لاس فانکشن های RMSE, MSE پس از ایتريشن ۲۰۰۰ بهبود چندانی نداشته اند. میتوان این اتفاق را اینگونه تفسیر کرد که Complexity مدل درجه ۵ کافی نبوده و پس از ۲۰۰۰ ایتريشن اشباع شده.

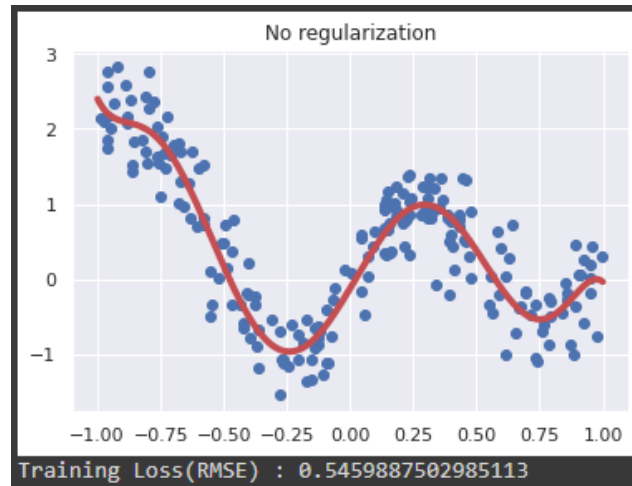
همچنین منحنی های مربوط به لاس فانکشن MAE دارای شکستگی هایی و مقداری نویز هستند. دلیل این اتفاق ممکن است بخاطر گسسته بودن تابع MAE باشد که باعث می شود مشتق خوبی نداشته باشد. در بسیاری از منابع پیشنهاد شده برای محاسبه مشتق MAE از روش های Approximation استفاده کنیم که همه جا مشتق پذیر هستند (خارج از اهداف این تمرین).



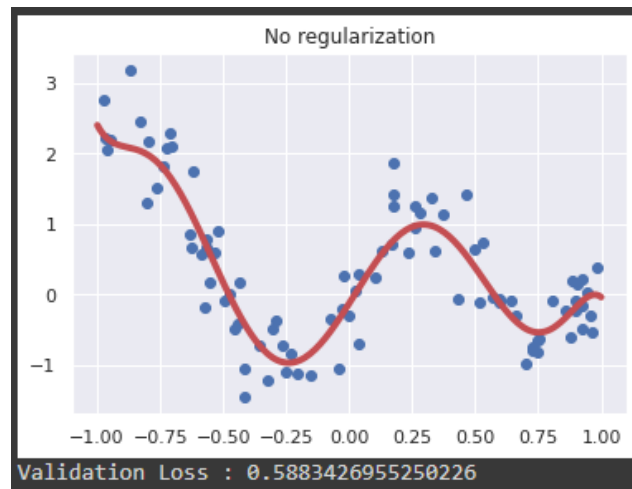
شکل ۱۶- نمودار کاهش $learning\ rate$ بر حسب $iteration$

هـ) قسمت قبل را با معادله‌ی نرمال و بدون در نظر گرفتن ضریب λ تکرار کنید و نتایج را مقایسه کنید.

در این قسمت با کمک معادله نرمال پارامترهای منحنی (theta) را محاسبه کردیم در ادامه میتوان نمودار منحنی برازش شده توسط این روش را دید



شکل ۱۷- منحنی *fit* شده روی داده آموزش با کمک معادله نرمال



شکل ۱۸- - منحنی *fit* شده روی داده اعتبارسنجی با کمک معادله نرمال

و) به ازای درجه ۸، با روش معادله‌ی نرمال و با در نظر گرفتن سه مقدار مختلف برای ضریب λ نموداری بر روی داده‌ها برازش دهید و مقدار خطای RMSE را برای داده‌های آموزش و آزمون رسم کنید. تاثیر ضریب λ را بررسی نمایید.
در این بخش به معادله نرمال جمله Regularization اضافه میکنیم تا از overfit شدن منحنی بکاهیم.

$$\theta = (X^T X + \lambda \cdot L)^{-1} X^T y$$

$$L = \begin{bmatrix} 0 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix}$$

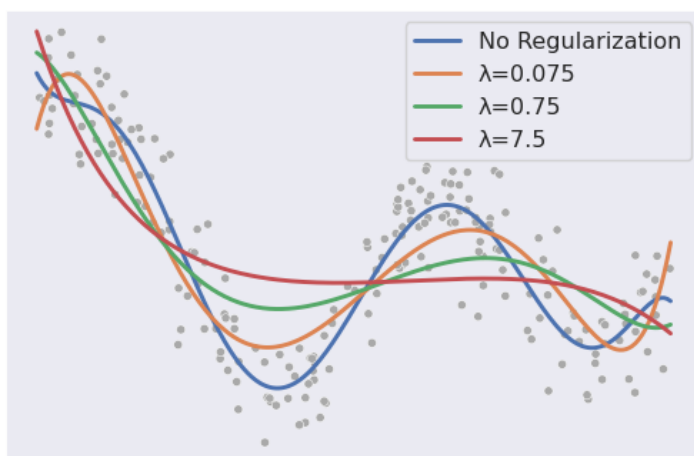
```
0 - Regularized Normal Equation (λ = 0.075)
Training Loss : 0.66061
Validation Loss : 0.70674

1 - Regularized Normal Equation (λ = 0.75)
Training Loss : 0.85344
Validation Loss : 0.90629

2 - Regularized Normal Equation (λ = 7.5)
Training Loss : 1.02587
Validation Loss : 1.06704
```

شکل ۱۹- خطاهای آموزش و اعتبار سنجی متناظر با مقادیر مختلف λ

در نهایت در شکل ۲۰ میتوانید اثر مقدار λ بر منحنی Fit شده را ببینید. هر چه مقدار λ بیشتر باشد منحنی smooth تر شده و اصطلاحاً مدل جنرال تر میشود.



شکل ۲۰- منحنی *Fit* شده روی داده با کمک معادله نرمال و مقدار *lambda* مختلف

سوال ۲)

نوت بوک مربوطه به این تمرین :

https://colab.research.google.com/drive/17PHn4Qvq45pYAqFRZ_1PmJQi1tYQy6Lb?usp=sharing

مجموعه داده "dataset_CSM" که در فولدر "dataset2" قرار دارد را در نظر بگیرید. این مجموعه داده مربوط به فیلمها و امتیاز IMDb آنها میباشد که از سه منبع Twitter، YouTube و IMDB جمع آوری شده است. در این سوال میخواهیم مقدار متغیر Ratings را پیشبینی کنیم.

الف) در برخی از ستونهای مجموعه داده مقادیر گم شده وجود دارد. با استفاده از روشهای مناسب، مقادیر گم شده را پر کنید و به طور مختصر توضیح دهید که چگونه این کار را انجام دادید. همچنین اگر داده ها نیاز به پیش پردازش دیگری دارند، در این قسمت انجام دهید و در گزارش خود ذکر کنید.

راه های متعددی برای مقابله با مقادیر گم شده وجود دارد.

یکی از راه های رایج، حذف کردن یا نادیده گرفتن سطر یا ستونی است که شامل مقدار گم شده است. به این صورت که اگر ویژگی Likes دارای تعدادی مقدار گم شده باشد، کل ویژگی Likes را از دیتاست حذف میکنیم. همینطور که مشخص است برای این مثال خاص این کار عاقلانه ای نیست چون با حذف کردن ستون Likes با وجود اینکه مقادیر گم شده دیتاست را کم میکنیم اما مقدار زیادی اطلاعات مفید مخصوصا برای پیش بینی Rating که هدف این مسئله است را از دست می دهیم. حالت دیگر زمانی است که یک سطر دارای تعدادی مقدار گم شده باشد مثلا فیلم Ant-Man ویژگی Aggregate followers را ندارد. میتوانیم کل سطر مربوط به فیلم Ant-Man را حذف کنیم و با این کار هم تعداد مقادیر گم شده را کاهش داده ایم. اما این روش هم مناسب به نظر نمی رسد زیرا ممکن بود ما بدون نیاز به داشتن مقدار Aggregate followers برای فیلم Ant-Man و تنها با توجه به سایر ویژگی های دیتاست، پیش بینی مناسبی از Rating این فیلم داشته باشیم.

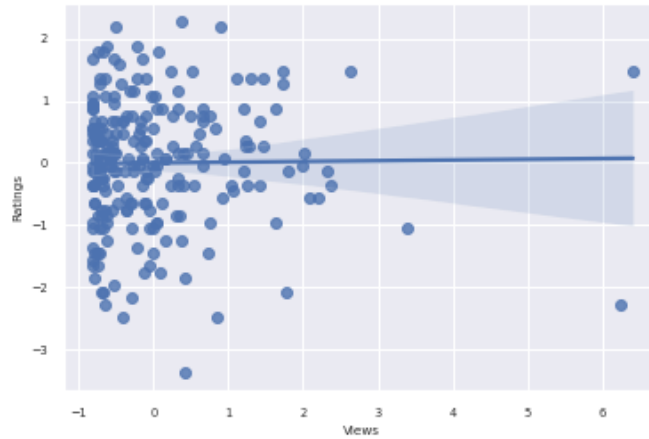
راه دیگر اصطلاحا Imputation نام دارد و به این معنی است که مقادیر گم شده دیتاست را به نحوی به طور مصنوعی پر بکنیم. استراتژی های بسیار زیادی برای پر کردن این مقادیر گم شده وجود دارد مثل جایگزینی با یک عنصر خنثی (0,1,"") یا جایگزینی با یک ویژگی آماری یک ویژگی یا کل دیتاست (Mean, Median, Min, Max, ...) و استراتژی های دیگر. استفاده از این روش برای این دیتاست به نظر روش معقول تری می آید.

- مقادیر گم شده ویژگی Budget را میتوان با میانگینی از بودجه سایر فیلم ها جایگزین کرد. از آنجایی که ژانر های مختلف از نظر بودجه تفاوت زیادی با هم دارند (بودجه یک فیلم علمی تخیلی بالاتر از فیلم درام است) می توانیم بجای استفاده از میانگین همه فیلم ها، میانگین فیلم های هم ژانر را جایگزین کنیم.
- مقادیر گم شده ویژگی Screens را با میانگین سایر سطر ها جایگزین میکنیم (شاید انتخاب بهتری هم می توانستیم بکنیم)
- ستون Aggregate followers برای تعداد زیادی از فیلم های به خصوص جدید خالی است. این اتفاق ممکن است در پیش بینی امتیاز فیلم های جدید مدل را به اشتباه بی اندازد به عنوان مثال جایگزین کردن میانگین این ویژگی برای اکثر فیلم های ۲۰۱۵ ممکن است به مدل این ایده را بدهد که فیلم های ۲۰۱۵ در سطح متوسط هستند و به اندازه فیلم های ۲۰۱۴ طرفدار ندارند. پس این ویژگی را حذف میکنیم.

از دیگر کارهایی که در این مرحله انجام میدهیم Normalization یا Feature Scaling است. بردن مقادیر تمام ویژگی ها به یک رنج کوچک و مشابه مزیت زیادی در پروسه یادگیری Gradient Descent دارد.

(ب) نمودار همبستگی بین ویژگی ها را رسم کنید و در گزارش بیاورید.

برای مشاهده میزان Correlation بین ویژگی های مختلف میتوان نموداری مشابه شکل زیر رسم کرد.



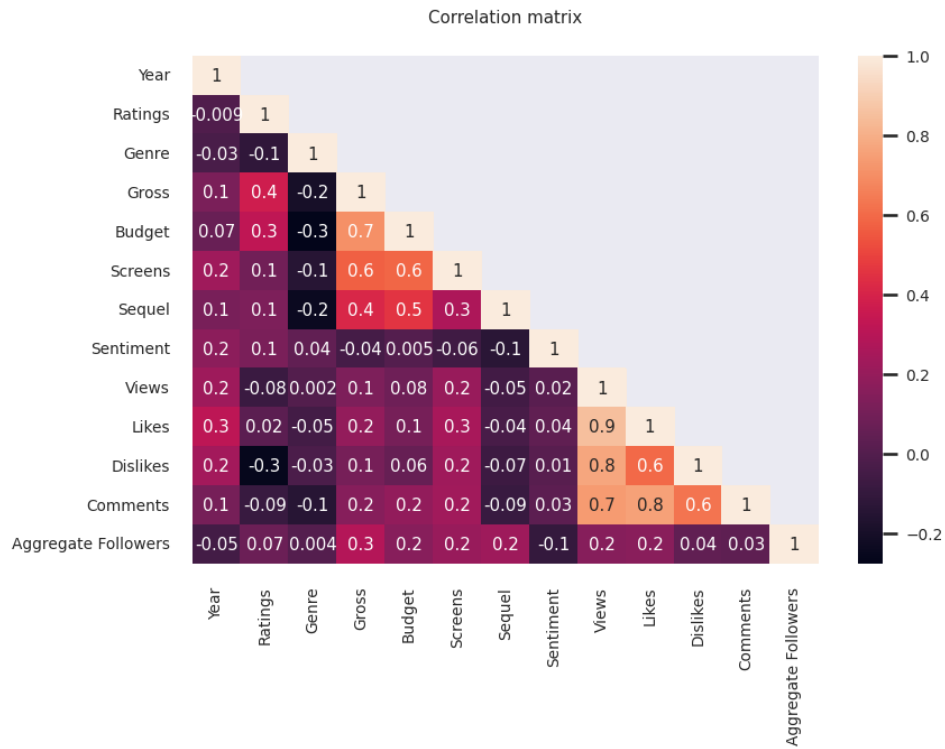
شکل ۲۱- نمودار رابطه بین Views و Ratings

اگر این کار را برای هر جفت ویژگی رسم کنیم تصویر بسیار بزرگی تولید میشود که قابلیت تحلیل کمی هم دارد.



شکل ۲۲- نمودار هم بستگی برای هر جفت ویژگی

راه بهتری که وجود دارد رسم کردن Heatmap مربوط به **Correlation Matrix** است.



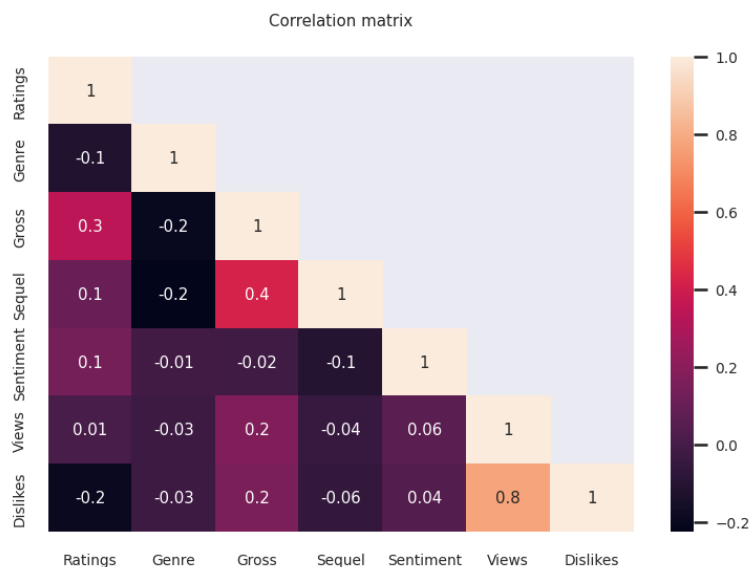
شکل ۲۳ - ماتریس درهم ریختگی^۱

^۱ Correlation Matrix

ج) با توجه به نمودار رسم شده آیا می‌توان یک یا چند ویژگی را حذف کرد؟ دلیل خود را ذکر کنید.

با دیدن ماتریس درهم ریختگی متوجه میشویم تعدادی از ویژگی‌ها هستند که همبستگی نسبتاً بالایی به هم دارند. در واقع این ویژگی‌ها اطلاعات تقریباً مشابهی به مدل می‌دهند و اصطلاحاً **Redundant** هستند. جفت ویژگی‌هایی مثل **Views-Likes** و **Comments** از این دست ویژگی‌ها هستند.

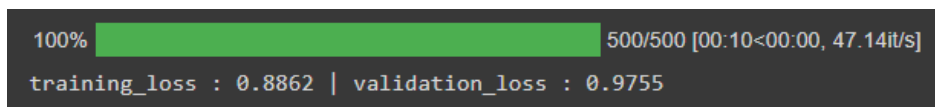
پس از حذف ویژگی‌های تعدادی از ویژگی‌ها ("Year", "Comments", "Screens", "Likes", "Budget") ماتریس درهم ریختگی به شکل زیر در خواهد آمد.



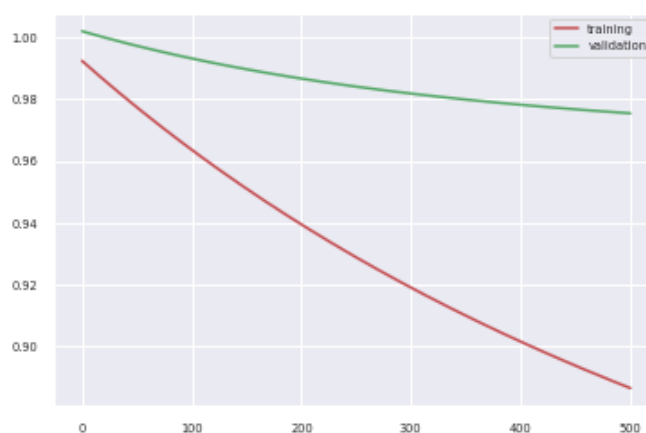
شکل ۲۴- ماتریس درهم ریختگی بعد از حذف تعدادی از ویژگی‌های *redundant*

د) با استفاده از روش گرادیان نزولی یک نمودار بر روی داده‌ها برازش دهید. (پارامترها را به گونه‌ای انتخاب کنید که به بهترین خروجی دست یابید) این عمل را یک بار با استفاده از کل ویژگی‌ها و یک بار با استفاده از ویژگی‌های منتخب تکرار کنید. نمودار خطای آموزش و آزمون و نمودار طول گام را رسم کنید.

مرحله اول (آموزش با تمام ویژگی ها)



شکل ۲۵ - آموزش با استفاده از تمام ویژگی ها

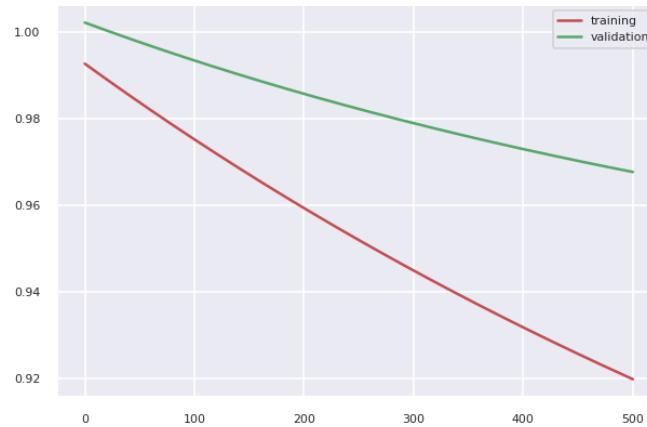


شکل ۲۶ - نمودار خطا بر حسب iteration

مرحله دوم (آموزش با ویژگی های منتخب)

```
100% ██████████ 500/500 [00:06<00:00, 72.98it/s]
training_loss : 0.9197 | validation_loss : 0.9676
```

شکل ۲۷ - آموزش با ویژگی های منتخب



شکل ۲۸ - نمودار خطا بر حسب iteration

نکته ای که در مورد این سوال وجود داشت این بود که حتی با پیاده سازی های آماده sklearn هم امتیاز بالایی نمیگرفتیم (ممکن است مشکل از دیتاست باشد) به طوریکه R2 Score مدل LinearRegression از لایبری sklearn برابر با مقدار 0.30810 بود و Score مدل SGDRegressor از لایبری sklearn برابر با مقدار 0.2739 بود. (R2 score هر چه به ۱ نزدیک تر باشد بهتر است)

سوال امتیازی)

نوت بوک مربوطه به این تمرین:

<https://colab.research.google.com/drive/1fSVpsDTRh0vFbD5yKkSLdSgTj8lYTqED?usp=sharing>

داده‌های مربوط به ویژگی‌های شخصی و هزینه‌ی پزشکی افراد در فولدر dataset3 قرار دارد. هر داده دارای ۶ ویژگی ورودی و یک خروجی می‌باشد که هزینه‌ی پزشکی فرد را نشان می‌دهد. همانطور که در داده‌های سوال دیده می‌شود، ویژگی‌های جنیست، منطقه و سیگاری بودن از نوع categorical هستند. برای تبدیل این ویژگی‌ها به ویژگی‌های عددی در این سوال می‌خواهیم از دو روش Integer Encoding و One Hot Encoding استفاده کنیم.

الف) توضیح دهید که Integer Encoding و One Hot Encoding چگونه انجام می‌شوند.

هر موقع نیاز است با داده‌های Categorical که به صورت عددی نیستند کار کنیم، باید از یک روش Encoding استفاده کنیم و به کمک آن داده‌های از نوع رشته را به عدد تبدیل کنیم. دو روش معروف Integer Encoding و OHE هستند. Integer Encoding به این صورت است که مانند یک دیکشنری به هر دیتای غیر عددی یک عدد صحیح نسبت می‌دهیم.

مثلا Bad=0, Average=1, Good=2

روش دیگر روش One Hot Encoding است که بردارهایی با طول مشخص از اعداد صفر و یک را به جای هر داده‌ی غیر عددی در نظر می‌گیرد به طوری که در هر بردار فقط یک مولفه ۱ وجود داشته باشد و سایر مولفه‌ها ۰ هستند.

مثلا Bad=[0,0,1], Average=[0,1,0], Good=[1,0,0]

ب) ویژگی‌های جنسیت و سیگاری بودن را با استفاده از Integer Encoding و ویژگی منطقه را با استفاده از OHE به مقدار عددی تبدیل کنید (برای جنسیت: female=0, male=1 و برای سیگاری بودن: no=0, yes=1)

	age	gender	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520

شکل ۲۹- قبل از encoding

	age	gender	bmi	children	smoker	charges	region_southwest	region_southeast	region_northwest	region_northeast
0	19	0	27.900	0	1	16884.92400	1.0	0.0	0.0	0.0
1	18	1	33.770	1	0	1725.55230	0.0	1.0	0.0	0.0
2	28	1	33.000	3	0	4449.46200	0.0	1.0	0.0	0.0
3	33	1	22.705	0	0	21984.47061	0.0	0.0	1.0	0.0
4	32	1	28.880	0	0	3866.85520	0.0	0.0	1.0	0.0

شکل ۳۰- بعد از encoding

ج) به نظر شما چرا برای ویژگی منطقه از OHE استفاده میکنیم و از Integer Encoding استفاده نمیکنیم؟

اگر از Integer Encoding استفاده کنیم به طور ناخودآگاه رابطه کوچکتر بزرگتری بین مقادیر مختلف یک ویژگی ایجاد میکنیم. مثلا در مثال Bad=0, Average=1, Good=2 یک توالی بین مقادیر از بد تا خوب ایجاد کردیم که مدل ممکن است در پروسه یادگیری خود بخواهد از این ویژگی استفاده کند. در مثال ویژگی Region مقادیر مختلف ارتباطی به یکدیگر ندارند و هر کدام از داده ها میتواند هر عددی به خود بگیرد. با استفاده از OHE به نوعی تعدادی بردار عمود بر هم را به مقادیر ویژگی نسبت دادیم و استقلال بین مقادیر را به صورت ریاضی به مدل نشان داده ایم.

د) رگرسیون خطی تعمیم یافته را بدون جمله‌ی منظم ساز پیاده سازی کرده، نتایج را روی داده های تست گزارش کنید. تابع basis برای ویژگی سن به صورت زیر باشد:

$$\phi_{age}(x_{age}) = x_{age}^2$$

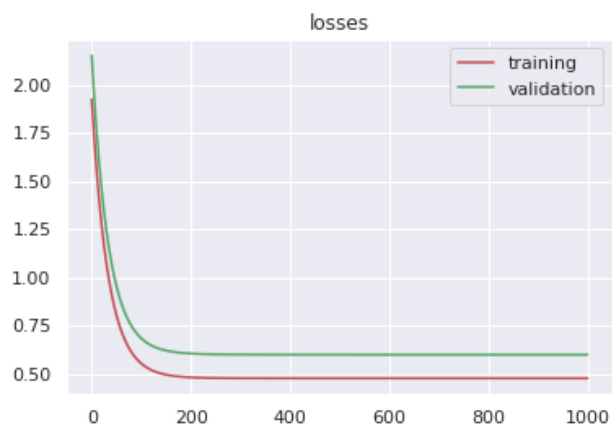
و برای سایر ویژگی‌ها از تابع همانی استفاده کنید:

$$\phi(x) = x$$

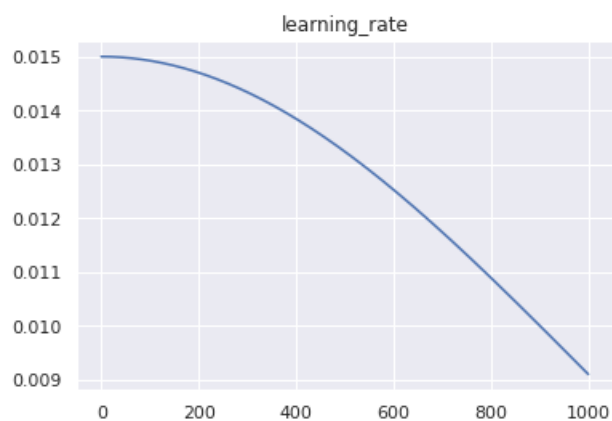
```
100% ██████████ 1000/1000 [00:20<00:00, 49.88it/s]
training_loss : 0.4753 | validation_loss : 0.5978
- Our Model

MSE on train : 0.47532531745619283
MSE on test : 0.5977550982029473
```

شکل ۳۱- خطای آموزش و تست



شکل ۳۲- نمودار خطا بر حسب iteration



شکل ۳۳- کاهش learning rate بر حسب iteration

همچنین در این بخش یک مقایسه با میزان خطای پیاده سازی SGDRegressor و Linear Regression از کتابخانه sklearn انجام شد تا از صحت پیاده سازی مطمئن شویم.

	Model Name	MSE Train	MSE Test
0	Sklearn Linear Regression	0.239332	0.304525
1	Sklearn SGDRegressor	0.239236	0.306813
2	Our Gradient Descent	0.239152	0.307154

شکل ۳۴- مقایسه میزان خطای مدل ما و مدل های دیگر

ه) در این قسمت ابتدا تعداد داده های آموزش را ۱۰ در نظر بگیرید و با گامهای ۱۰ تایی، تا ۱۰۰۰ افزایش دهید. تغییرات خطای تست و آموزش را با افزایش داده آموزش بررسی کنید. برای مقایسه پذیر بودن خطا، از MSE برای تابع هزینه استفاده کنید. نمودار این تغییرات را در گزارش بیاورید .

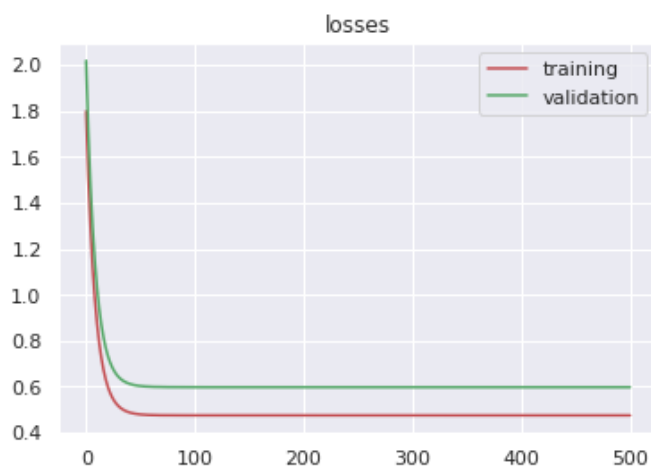
همانطور که در شکل ۳۰ مشاهده میکنید، با اندازه آموزش بسیار کم مدل ما اطلاعات جامع و کاملی از پترن های موجود در داده ها ندارد و بخاطر همین خطای تست زیادی دارد. رفته رفته با افزایش اندازه داده آموزشی مدل اطلاعات خود را بر اساس نمونه بزرگتری از داده ها بدست می آورد و این باعث می شود در مقابل دیتای تست عملکرد بهتری داشته باشد چون پترن های کلی تر و بیشتری روی داده های آموزشی دیده.

از آن طرف خطای آموزش با افزایش داده آموزش زیاد می شود. چون حفظ کردن دیتای کم کار ساده تری است و در Training size های خیلی کم مدل به راحتی overfit شده.

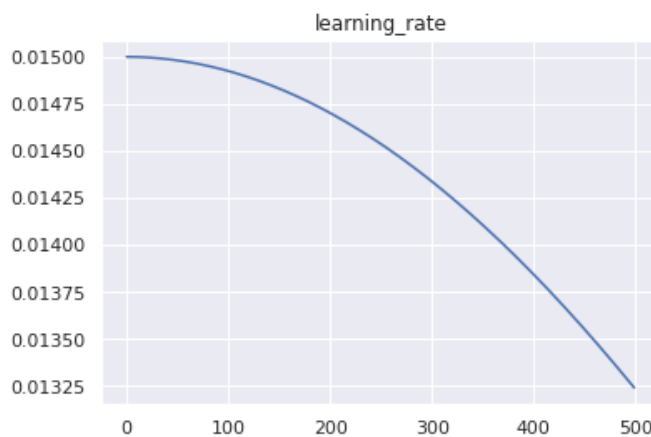


شکل ۳۵- نمودار خطا بر حسب اندازه داده آموزش

و) روش Stochastic Gradient Descent را بر روی این مجموعه داده پیاده سازی کنید و مقدار دقت و خطا را گزارش کنید. Mini-batch Stochastic Gradient Descent پیاده سازی شد. توضیحات این قسمت مانند توضیحات نمودار های قبلی است.



شکل ۳۶- نمودار خطا بر حسب iteration



شکل ۳۷- نمودار کاهش learning rate بر حسب iteration

منابع

تمرین پیاده سازی اول

https://colab.research.google.com/drive/1itjIBNliQ17kVK_gCL0o0ea8xor61Q3V?usp=sharing

تمرین پیاده سازی دوم

https://colab.research.google.com/drive/17PHn4Qvq45pYAqFRZ_1PmJQi1tYQy6Lb?usp=sharing

تمرین پیاده سازی سوم

<https://colab.research.google.com/drive/1fSVpsDTRh0vFbD5yKkSLdSgTj8IYTqED?usp=sharing>

لینک گیتهاب

<https://github.com/Gholamrezadar/machine-learning-exercises>

رسم کردن Heatmap

<https://androidkt.com/plot-correlation-matrix-and-heatmaps-between-columns-using-pandas-and-seaborn>

توضیحاتی راجع به regularization

<https://machinelearningmedium.com/2017/09/11/regularized-linear-regression>

توضیحاتی راجع به SGD

<https://realpython.com/gradient-descent-algorithm-python/#stochastic-gradient-descent-algorithms>