«به نام خدا»

Artificial Intelligence

Computer Assignment 3

The Magical Machine Learning

بخش اول : رگرسیون خطی (Linear Regression)

فرض کنید شما مسئول مدیریت و برنامه‌ریزی انبارهای یک فروشگاه اینترنتی هستید که در کل کشور خدمات ارائه می‌دهد. این فروشگاه تاکنون انبارهایی در شهرهای گوناگون داشته است. داده مربوط به جمعیت شهرهای هدف و سود حاصل شرکت در آن‌ها را نیز دارید.

شما قرار است با استفاده از این داده‌ها، بهترین شهر برای به راه انداختن انبار کالای بعدی را بیابید به طوری که بیشترین سود را برای شرکت شما داشته‌باشد.

مجموعه داده موردنیاز برای حل این مسئله رگرسیون خطی در فایل part1data.txt در اختیار شما قرار گرفته‌است. ستون اول جمعیت شهر و ستون دوم سود شرکت در آن جا است. واضح است که مقدار منفی برای سود به معنی ضرر است.

۱,۱ ترسیم داده (Visualization)

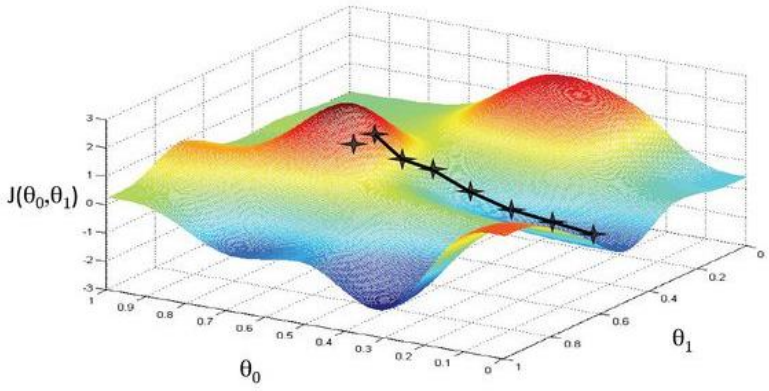
قبل از انجام هرکاری، بهتر است داده‌ها را ترسیم کنیم تا نسبت به آن حس و درک بیشتری پیدا کنیم. برای این مجموعه داده می‌توانید از scatterplot برای به تصویر کشیدن داده استفاده کنید چرا که تنها ۲ ویژگی برای ترسیم وجود دارد(جمعیت و سود). [بسیاری از مسائل دیگری که در آینده با آن‌ها در زندگی مواجه خواهید شد چند بعدی هستند و نمی‌توان آن‌ها را در یک نمودار ۲ بعدی ترسیم کرد.]

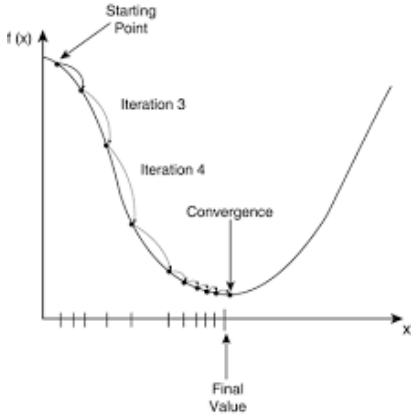
◄ برای اینکار می‌توانید از کتابخانه‌های numpy و matplotlib استفاده کنید.

تصویر scatterplot مربوطه را در گزارشکار خود قرار دهید. توجه کنید که باید هر دو محور و خود نمودار برچسب داشته باشند.

۱,۲ گرادیان کاهشی (Gradient Descent)

در این بخش، به کمک روش گرادیان





کاهشی، پارامترهای رگرسیون خطی (θ) را برای مجموعه داده خود متناسب می‌کنیم. (fit to dataset)

۱,۲,۱ به روز کردن معادلات

هدف رگرسیون خطی به حداقل رساندن تابع هزینه است.

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

که فرضیه $h_\theta(x)$با مدل خطی زیر داده شده است :

$$h_\theta(x) = \theta^T x = \theta_0 + \theta_1 x_1$$

توجه کنید که پارامترهای مدل شما مقادیر θ_j هستند و در واقع باید آن‌ها را برای حداقل کردن هزینه بیابید. یکی از روش‌های انجام این کار استفاده از الگوریتم گرادیان کاهشی دسته‌ای (batch gradient descent) می‌باشد. در این روش در هر تکرار (iteration) مقادیر به روز می‌شوند.

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} \quad \text{(simultaneously update } \theta_j \text{ for all } j).$$

در هر گام گرادیان کاهشی، پارامترهای θ_j به مقادیر بهینه نزدیک‌تر می‌شوند تا جایی که به کمترین میزان هزینه $J(\theta)$ برسد.

۱,۲,۲ پیاده‌سازی

از آنجایی که از گرادیان کاهشی به منظور به حداقل رساندن تابع هزینه استفاده می‌کنیم، خوب است بتوانیم همگرایی این الگوریتم را به کمک محاسبه مقدار تابع هزینه بررسی کنیم. (همانطور که می‌دانید روند باید کاهشی باشد) در این قسمت، ابتدا باید تابعی برای محاسبه مقدار $J(\theta)$ پیاده‌سازی کنید. (به کمک این تابع مقدار تابع هزینه را در هر بار تکرار الگوریتم نمایش دهید.)

پس از آن باید یک تابع برای محاسبه گرادیان کاهشی پیاده‌سازی کنید.

☜ جهت یادآوری به منظور در نظر گرفته شدن θ_0 یا همان intercept term باید یک ستون تمام یک به ابتدای داده اضافه کنید. در ضمن تعداد تکرارهای الگوریتم را ۱۵۰۰ و پارامتر آلفا را نیز ۰,۰۱ در نظر بگیرید.

◄ می‌توانید برای پیاده‌سازی از توابع ریاضی کتابخانه‌ی numpy و scipy کمک بگیرید.

۱,۲,۳ به تصویر کشیدن تابع هزینه(امتیازی)

برای اینکه درک بهتری نسبت به تابع هزینه پیدا کنید، در این مرحله باید تابع هزینه را نسبت به دو پارامتر θ_0 و θ_1 ترسیم کنید.

☚ تصویر نمودار خروجی را در گزارشکار خود قرار دهید.

هدف از این نمودار این است که به شما نشان دهد که $J(\theta)$ چگونه با تغییرات θ_0و θ_1 تغییر می‌کند. همانطور که می‌بینید تابع هزینه کاسه‌شکل است و یک مقدار بهینه سراسری دارد. این نقطه، مقدار بهینه θ_0و θ_1 را خواهد داشت و گرادیان کاهشی در هر گام به این نقطه نزدیک‌تر می‌شود.

بخش دوم : رگرسیون لجستیک (Logistic Regression)

در این قسمت تمرین، برای پیش‌بینی اینکه آیا یک دانشجو در یک دانشگاه پذیرفته‌می‌شود یا خیر، یک مدل رگرسیون لجستیکی خواهید ساخت.

فرض کنید که شما مسئول بخش پذیرش دانشکده برق و کامپیوتر دانشگاه تهران هستید و میخواهید شانس پذیرش درخواست‌کنندگان برای ورود به دانشکده را بر اساس نمرات دو آزمون آن‌ها بسنجید. تاریخچه‌ای از متقاضیان پیشین دانشکده در اختیار دارید که می‌توانید به عنوان داده آموزشی از آن‌ها استفاده کنید.(در فایل part2data.txt) برای هر نمونه آموزشی، نمره متقاضی در دو آزمون و وضعیت پذیرش آن به شما داده شده است.

حال شما باید یک مدل طبقه‌بندی (classification model) بسازید که به کمک آن بتوانید احتمال پذیرش دانشجو بر اساس نمرات آن دو آزمون را تخمین بزنید.

۲,۱ ترسیم داده (Visualization)

همیشه خوب است قبل از شروع پیاده‌سازی هر الگوریتم یادگیری (Learning Algorithm)، در صورت امکان داده را به تصویر کشید.

- تصویر scatterplot مربوط به داده‌های آموزشی را در گزارشکار خود قرار دهید. (نمرات دو آزمون، محورهای نمودار خواهند بود. وضعیت پذیرش یا عدم پذیرش را با نشانگرهای مختلف مشخص کنید)

۲,۲ پیاده‌سازی

۲,۲,۱ تابع سیگموید (Sigmoid Function)

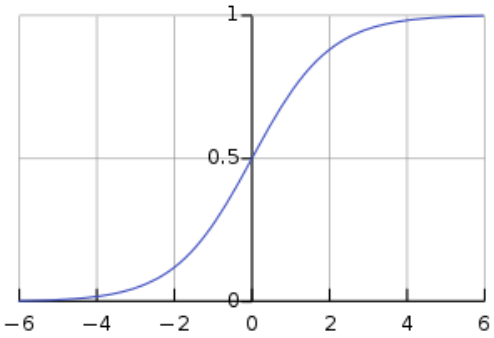
قبل از اینکه در مورد تابع هزینه واقعی صحبت کنیم، جهت یادآوری میدانیم که فرضیه رگرسیون لجستیک به صورت زیر تعریف می‌شود :

$$h_\theta(x) = g(\theta^T x)$$

که تابع g در واقع همون سیگموید است که به صورت زیر تعریف می‌شود:

$$g(z) = \frac{1}{1+e^{-z}}.$$

در گام اول این تابع را پیاده‌سازی کرده و با دادن مقادیر مختلف به آن از صحت عملکرد آن اطمینان حاصل کنید.



۲,۲,۲ تابع هزینه و گرادیان

در این قسمت باید تابعی بنویسید که مقادیر هزینه و گرادیان را برگرداند.

جهت یادآوری تابع هزینه رگرسیون لجستیک به صورت زیر تعریف می‌شود :

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log(h_\theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right]$$

و المان j-ام بردار گرادیان هزینه به صورت زیر تعریف می‌شود :

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$$

توجه داشته باشید که گرچه این رابطه شبیه مسئله گرادیان خطی بنظر می‌رسد، اما در واقع متفاوت هستند. زیرا تعریف $h(\theta)$ برای آن‌ها متفاوت است.

۲,۲,۳ یادگیری پارامترها

حال میخواهیم به کمک توابعی که نوشته‌ایم و نیز روش گرادیان کاهشی، سعی در یادگیری پارامترهای بهینه کنیم.

◄ می‌توانید این بار از توابع کتابخانه‌ای sklearn کمک بگیرید.(به جای نوشتن تابع گرادیان کاهشی به کمک توابع ریاضی)

پس از یادگیری، تصویر scatterplot مربوط به داده‌های آموزشی و خط جداکننده آن‌ها را در گزارشکار خود قرار دهید.

۲,۲,۴ ارزیابی رگرسیون لجسیتک

پس از یادگیری پارامترهای مدل، می‌توانید از آن‌ها برای پیش‌بینی اینکه آیا یک دانشجو پذیرفته خواهد شد یا خیر استفاده کنید. به عنوان مثال برای یک دانشجو با نمره امتحان۱، ۴۵ و نمره امتحان۲، ۸۵ انتظار میرود احتمال پذیرش ۰,۷۷۶ را مشاهده کنید.

در این قسمت، تابعی برای پیش‌بینی بنویسید که یک مجموعه دیتا و یک پارامتر یادگیری شده θ می‌گیرد و پیش‌بینی‌های ۰ یا ۱ باز می‌گرداند.

بخش سوم : رگرسیون لجستیک قاعده‌مند (Regularized Logistic Regression)

در این بخش تمرین، مدلی برای پیش‌بینی اینکه آیا میکروچیپ‌های تولیدی یک کارخانه، تست‌های تضمین کیفیت (QA) را به خوبی خواهند گذراند یا خیر خواهیم ساخت.

فرض کنید که شما مدیر محصول یک کارخانه هستید و نتایج تست در دو آزمون برای تعدادی میکروچیپ را در اختیار دارید. به کمک این نتایج میخواهید تعیین کنید که یک میکروچیپ قابل قبول است یا نه! برای کمک در تصمیم‌گیری به شما، مجموعه‌ی داده‌ای از نتایج تست برای نمونه‌های مشابه در گذشته به شما در فایل part3data.txt داده شده است.

۳,۱ ترسیم داده (*Visualization*)

تصویر scatterplot مربوط به داده‌های آموزشی را در گزارشکار خود قرار دهید. (نتایج دو آزمون، محورهای نمودار خواهند بود. وضعیت پذیرش یا عدم پذیرش را با نشانگرهای مختلف مشخص کنید)

همانطور که مشاهده می‌کنید مجموعه داده رسم‌شده را نمی‌توان به کمک یک خط مستقیم به دو گروه + / - تقسیم کرد. بنابراین از آن‌جایی که رگرسیون لجیستیک با تعریفی که تاکنون از آن داشتیم، تنها قادر به پیدا کردن یک مرز تصمیم‌گیری خطی (Linear Decision Boundary) است، نمی‌توانیم مانند قبل نسبت به این مجموعه داده برخورد کنیم.

۳,۲ نگاشت ویژگی‌ها (**Feature Mapping**)

یکی از راه‌های فیت کردن بهتر مدل به داده، افزودن ویژگی‌های بیشتر برای هر داده است.

$$\text{mapFeature}(x) = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_1^2 \\ x_1 x_2 \\ x_2^2 \\ x_1^3 \\ \vdots \\ x_1 x_2^5 \\ x_2^6 \end{bmatrix}$$

در نتیجه این نگاشت، بردار دو ویژگی (که نتایج دو آزمون بود) به یک بردار ۲۸ بعدی تبدیل می‌شود. رگرسیون لجیستیک طبقه‌بندی که با این بردار ویژگی با بعد بالاتر آموزش می‌یابد، مرز تصمیم‌گیری پیچیده‌تری خواهد داشت و زمانی‌که در یک نمودار دوبعدی ترسیم می‌شود، غیرخطی خواهد بود.

✍ درباره مشکل انطباق بیش از حد (overfitting) تحقیق کنید و بگویید در چه صورت بوجود می‌آید؟ چرا روش نگاشت ویژگی‌ها ممکن است به این مشکل بربخورد؟

در بخش‌های بعدی این تمرین، رگرسیون لجستیک قاعده‌مند را برای متناسب کردن مدل به داده پیاده‌سازی خواهید کرد و خواهید دید که چگونه به مقابله با مشکل overfitting کمک خواهد کرد.

۳,۳ تابع هزینه و گرادیان (Cost Function and Gradient)

در این بخش باید تابعی برای محاسبه تابع هزینه و گرادیان برای رگرسیون لجیستیک قاعده‌مند پیاده‌سازی کنید.

همانطور که میدانید، تابع هزینه قاعده‌مند در رگرسیون لجستیک به صورت زیر تعریف می‌شود :

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log(h_\theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2.$$

✍ توجه داشته باشید که نباید پارامتر θ_0 را در عبارت تنظیم (regularization term) تاثیر دهید.

همچنین گرادیان تابع هزینه برداری است که عنصر jام آن به صورت زیر تعریف می‌شود:

$$\frac{\partial J(\theta)}{\partial \theta_0} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} \qquad \text{for } j = 0$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m}\left(\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} + \lambda\theta_j\right) \quad \text{for } j \geq 1$$

✍ مقدار اولیه برای θ را می‌توانید بردار تمام صفر در نظر بگیرید. (که در این صورت هزینه ۰,۶۳ خواهد بود)

۳,۳,۱ یادگیری پارامترها

◄ می‌توانید از تابعی که در قسمت مشابه بخش قبل نوشتید استفاده کنید.

۳,۴ ترسیم مرز تصمیم (Plotting the Decision Boundary) - امتیازی

به منظور تماشای مدل یادگیری‌شده با این طبقه‌بند، مرز تصمیمی را که داده‌های + و - را از هم جدا می‌کند رسم کنید.

بخش چهارم : طبقه‌بند چند کلاسی و شبکه‌های عصبی
(Multi-class Classification and Neural Networks)

در این تمرین، برای تشخیص عددهای دست‌نوشته از رگرسیون لجستیک و شبکه‌های عصبی کمک می‌گیریم. این کار امروزه کاربردهای بسیاری مانند تشخیص کد پستی روی بسته‌های پستی، اعداد نوشته شده روی چک‌های بانکی و .... دارد. در این تمرین روش‌هایی که تاکنون آموخته‌اید را در یک کاربرد عملی طبقه‌بندی خواهید دید.
در بخش اول تمرین، پیاده‌سازی قبلی رگرسیون لجستیک خود را برای طبقه‌بندی یکی-در برابر-همه (one-vs-all classification) گسترش خواهید داد.

۴,۱ مجموعه داده (Dataset)

مجموعه داده‌های داده شده به شما از پایگاه داده MNIST گرفته شده است. که در آدرس زیر آمده است :

the MNIST database of handwritten digits

در انتهای لینک راجع به این مجموعه داده اطلاعاتی داده شده است که برای کار کردن با آن دانستن آن‌ها لازم است. ابتدا لینک بالا را مطالعه کنید. برای این بخش استفاده از ۶۰۰۰ داده آموزشی کافی است.

۴,۲ ترسیم داده‌ها (Visualizing the Data)

۱۰ سطر از مجموعه داده آموزشی را به صورت تصادفی انتخاب کرده و آن‌ها را نمایش دهید.

◄ برای ذخیره داده‌ها میتوانید از ساختارهای موجود در کتابخانه Numpy استفاده نمایید.

۴,۳ رگرسیون لجیستیک (Logistic Regression)

در این بخش از چند مدل رگرسیون لجستیک برای ساختن یک طبقه‌بند چند کلاسی استفاده خواهید کرد. از آنجایی که ۱۰ کلاس وجود دارد، باید ۱۰ طبقه‌بند رگرسیون لجستیک جداگانه را اجرا کنید.

توجه کنید که X و θ به صورت برداری به شکل زیر تعریف می‌شوند :

$$X = \begin{bmatrix} — (x^{(1)})^T — \\ — (x^{(2)})^T — \\ \vdots \\ — (x^{(m)})^T — \end{bmatrix} \quad \text{and} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix} \quad \Longrightarrow \quad X\theta = \begin{bmatrix} — (x^{(1)})^T\theta — \\ — (x^{(2)})^T\theta — \\ \vdots \\ — (x^{(m)})^T\theta — \end{bmatrix} = \begin{bmatrix} — \theta^T(x^{(1)}) — \\ — \theta^T(x^{(2)}) — \\ \vdots \\ — \theta^T(x^{(m)}) — \end{bmatrix}$$

و بر این اساس توابع هزینه و گرادیان به صورت زیر خواهند بود :

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log(h_\theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2.$$

$$\frac{\partial J}{\partial \theta_0} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} \qquad \text{for j} = 0$$

$$\frac{\partial J}{\partial \theta_j} = \frac{1}{m}\left(\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} + \lambda\theta_j\right) \qquad \text{for j} \geq 1.$$

◄ پیاده‌سازی توابع مربوطه برای انجام قسمت بعدی لازم است.

۴,۴ طبقه‌بندی یکی-در برابر-همه (One-vs-all Classification)

در این قسمت، «طبقه‌بند یکی-در برابر-همه»ای را با آموزش چند طبقه‌بند رگرسیون لجستیک تنظیم‌شده(regularized logistic regression)، به ازای هر کدام از K کلاس در مجموعه داده پیاده‌سازی خواهید کرد. در مسئله اعداد دست‌نوشته K برابر ۱۰ است ولی کد شما باید برای هر مقدار K کار کند.

کد شما باید تمام پارامترهای طبقه‌بند را در یک آرایه $\ominus \in R^{K \times (N+1)}$، که هر ردیف $\ominus$ پارامترهای رگرسیون لجستیک یادگیری‌شده برای یک کلاس است، بازگرداند.

هنگام آموزش طبقه‌بند برای کلاس‌های $k \in \{1, \ldots, K\}$، به یک بردار m-بعدی از برچسب‌های y، نیاز هست که $y_j \in 0,1$ نشان می‌دهد که آیا

داده آموزشی j-ام به کلاس k $(y_j = 1)$ تعلق دارد یا به کلاس دیگری $(y_j = 0)$

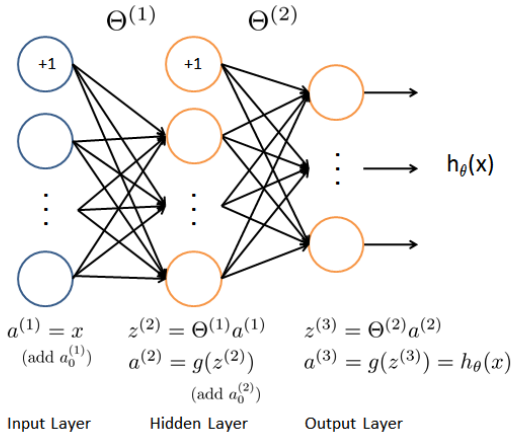
۴,۴,۱ پیش‌بینی یک-در مقابل-بقیه (One-vs-all Prediction)

بعد از آموزش طبقه‌بند خود، می‌توانید از آن برای پیش‌بینی عدد داخل یک عکس استفاده کنید. برای هر ورودی باید احتمال اینکه آن ورودی متعلق به هر یک از کلاس‌ها باشد را با استفاده از طبقه‌بند رگرسیون لجستیک محاسبه کنید و سپس محتمل‌ترین را به عنوان پیش‌بینی برای ورودی داده‌شده، خروجی دهید.

۴,۵ شبکه‌های عصبی (Neural Networks) - امتیازی

از آنجایی که رگرسیون لجستیک یک طبقه‌بند خطی است، نمی‌تواند فرضیه‌های پیچیده‌ای بسازد. در این بخش تمرین، شبکه عصبی برای تشخیص عددهای دست‌نوشته با استفاده از همان داده‌های آموزشی پیشین پیاده‌سازی خواهید کرد. شبکه عصبی قابلیت بیان مدل‌های پیچیده‌ای که فرضیه غیرخطی می‌سازند را داراست.(کلی قابلیت‌های هیجان‌انگیز دیگه هم داره البته :D)

در این بخش باید شبکه عصبی سه لایه‌ای (شامل یک لایه ورودی - یک لایه مخفی - یک لایه خروجی) طراحی کنید. توجه کنید که ورودی ما تصاویر ۲۸ پیکسل در ۲۸ پیکسل هستند و از آنجایی که ورودی ما مقادیر پیکسل‌ها برای تصاویر ارقام است، ۷۸۴ نورون برای لایه اول باید در نظر بگیرید.



کد شبکه عصبی شما باید پس از آموزش، بتواند با دریافت یک تصویر با ابعاد ۲۸*۲۸ شامل یک رقم دست‌نوشته، پیش‌بینی خود را برای رقم داخل عکس به عنوان خروجی بدهد.