محمدرضا کاری

بخش یک:

در این بخش کد ها در سه فایل :

First_part.py

Question_one.py

Simple_regression.py

نوشته شده است و دیتای ان در فایل

data.csv

است.

بخش ۲ :

سوال ۱:

رگرسیون ridge:

در حالت ساده ی رگرسیون خطی تابع هزینه ی ما بصورت زیر است:

$$\sum_{i=1}^{M}(y_i-\hat{y}_i)^2=\sum_{i=1}^{M}\left(y_i-\sum_{j=0}^{p}w_j\times x_{ij}\right)^2 \qquad (1.2)$$

که در ان هزینه بصورت توان دوی ارور محاسبه میشود حال در حالت کلی اگر تعداد زیادی feature داشته باشیم باعث پیچیدگی بیشتر مدل میشود که باعث میشود در تست بخوبی عمل نکنیم. در لاسو برای انکه از overfitting جلوگیری کنیم و یا پیچیدگی مدل را کم کنیم برای وزن ها یک جریمه تعیین میکنیم تا وزن ها را کاهش دهیم. برای این منظور تابع هزینه را بصورت زیر تغییر میدهیم:

$$\sum_{i=1}^{M}(y_i-\hat{y}_i)^2=\sum_{i=1}^{M}\left(y_i-\sum_{j=0}^{p}w_j\times x_{ij}\right)^2+\lambda\sum_{j=0}^{p}w_j^2 \qquad (1.3)$$

که همانطور که میبینیم محدودیتی برای این مدل گزاشته ایم میتوان این را بصورت زیر هم تعریف کرد:

$$\text{For some } c > 0, \sum_{j=0}^{p} w_j^2 < c$$

در اینجا اگر لامدا را صفر بگیریم الگوریتم ridge مانند رگرسیون خطی ساده میشود. بنابراین ridge با کم نگه داشتن وزن ها پیچیدگی مدل را کم میکند.

الگوریتم lasso:

تابع هزینه lasso یا least absolute shrinkage and selection operator بصورت زیر تعریف میشود:

$$\sum_{i=1}^{M} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{M} \left(y_i - \sum_{j=0}^{p} w_j \times x_{ij} \right)^2 + \lambda \sum_{j=0}^{p} |w_j| \qquad (1.4)$$

یا بعبارتی :

$$\text{For some } t > 0, \sum_{j=0}^{p} |w_j| < t$$

که شبیه به رگرسیون ridge هست با این تفاوت که بجای بتوان دو رساندن وزن ها قدر مطلق انرا بدست می اوریم در اینجا مانند رگرسیون ridge اگر لامدا برابر با صفر باشد با رگرسیون ساده تفاتی ندارد. در رگرسیون لاسو بعصی از ویژگی ها ممکن است وزن ۰ بدست اورند که بمعنی حذف این ویژگی ها است. در حالت کلی رگرسیون لاسو به ما کمک میکند تا بهترین ویژگی هارا برگزینیم در حالی که رگرسیون ridge باعث میشود تا از مشکل overfitting دوری کنیم.

https://towardsdatascience.com/ridge-and-lasso-regression-a-complete-guide-with-python-scikit-learn-e20e34bcbf0b

میتوان بصورت کلی گفت که ridge بطور پیش فرض خوب است اما اگر تنها تعداد محدودی از ویژگی ها قابل استفاده است بهتر است از lasso یا elastic استفاده کرد چون که ویژگی های بدردنخور را حذف میکنند

Hands on machine learning with scikit and tensorflow

سوال ۲:

در حالت کلی اگر لامدا برابر ۰ باشد این دو رگرسیون با رگرسیون ساده یا همان least-square برابر میشوند. اگر عدد زیاد باشد هم وزن ها به صفر میل میکنند.

Hands on machine learning with scikit and tensorflow

در اینجا برای پیدا کردن بهترین ضریب لامدا یا همان ضریب ترم regularization میتوان نخست دیتاست را نرمالایز کرده و سپس میتوان از ۰.۱ تا عددی بزرگ مانند ۵ را در نظر گرفته و سپس برای اعداد بین این دو مدل هارا اموزش دهیم و در جایی که برای تست ست بهترین مدل را پیدا کردیم ان را بعنوان لامدا یا ضریب متناسب بگیریم.

روش جستجوی دیگر هم جستجوی تصادفی است تا بتوان بهترین حالت را پیدا کرد.

سوال ۳:

افزایش تعداد فولد ها در حالت کلی باعث میشود تا واریانس افزایش یابد و بایاس کاهش پیدا کند. در حالت کلی تعداد فولد ها بستگی به اندازه داده دارد اگر اندازه داده کم باشد بهتر است تعداد فولد ها هم کم باشد تا تست ست بیش از حد کوچک نشود اما در صورتی که اندازه بسیار زیاد باشد میتوان اندازه فولد هارا بیشتر هم کرد. در صورتی اندازه فولد هارا زیاد میکنیم که بخواهیم بایاس را کاهش بدهیم اما این باعث میشود زمان اجرا بالاتر برود و واریانس هم ممکن است بیشتر شود.

سوال ۴:

در حالتی که ما داده را به S قسمت تقسیم میکنیم و هردفعه برای یادگیری مدل از S-1 قسمت استفاده میکنیم و یکی را انتخاب میکنیم تا داده تست باشد و سپس اینکار را S بار تکرار کنیم به این تکنیک S-fold cross-validation میگویند. در صورتی که S=n باشد در صورتی که n برابر تعداد کل داده ها باشد به این الگوریتم *leave-one-out میگویند*.

Pattern recognition and machine learning

در حالت کلی بخاطر هزینه محاسباتی بالا این روش باید بر روی دیتاست های کوچک استفاده شود. در حالاتی دیگر این روش برای انکه کارایی خود الگوریتم یادگیری ماشین را بر روی دیتاست ببینیم استفاده میشود.

https://machinelearningmastery.com/loocv-for-evaluating-machine-learning-algorithms/#:~:text=The%20leave%2Done%2Dout%20cross%2Dvalidation%20procedure%20is%20appro
.priate,computational%20cost%20of%20the%20method

سوال ۵:

در حالت bootstrapping ما همانند k-fold cross validation تکه ای از داده را برداشته و سپس انرا تست میکنیم با این تفاوت که در bootstrapping ما بخشی از داده را بصورت رندم برداشته و سپس انرا تست میکنیم و تعداد تکرار دفعاتی که باید اینکار را کرد هم تعیین میکنیم.

یعنی اول اندازه نمونه را تعیین میکنیم و سپس به اندازه نمونه و بطور تصادفی از داده ها برداشته و سپس به نمونه اضافه میکنیم و سپس انرا تست میکنیم.

میتوان ازین روش برای ارزیابی مدل یادگیری ماشین استفاده کرد.

https://machinelearningmastery.com/a-gentle-introduction-to-the-bootstrap-method/

سوال ۶:

در این روش به این صورت عمل میکنیم که به جای انکه از 10-fold cross validation استفاده کنیم از روش ۵*۲ استفاده میکنیم یعنی داده هارا نصف کرده و هر نصف را به پنج قسمت تقسیم مکنیم و سپس از هرکدام از بخش های نصف شده یک قسمت یک پنجم را برمیداریم و این کار را پنج بار انجام میدهیم. میتوان با استفاده از فرمول های زیر ارور را بدست بیاوریم:

$$p_A^1, p_A^2, p_B^1, p_B^2.\ p^1 = p_A^1 - p_B^1 \text{ and } p^2 = p_A^2 - p_B^2,$$

$$\overline{p} = \frac{p^1 + p^2}{2}, \text{ variance from } i^{th} \text{ replication is}$$

$$s_i^2 = (p_i^1 - \overline{p_i})^2 + (p_i^2 - \overline{p_i})^2 \text{ use statistic}$$

$$\tilde{t} = \frac{p_1^1}{\sqrt{\frac{1}{5}\sum_{i=1}^{5} s_i^2}}$$

از این روش میتوان در صورتی استفاده کرد که دو یا چند الگوریتم classifier داریم و میخواهیم انهارا با همدیگر بر روی داده ها مقایسه کنیم و ارزیابی کنیم.

منبع:

http://aivi.csee.usf.edu/~hall/dm/Eval2fold.pdf

بخش ۳:

سوال ۵:

در صورتی که imbalanced data داشته باشیم سه راه حل خوب داریم:

۱-undersampling

در این روش میتوان در صورتی که یک کلاس بیشتر از یک کلاس دیگر نمونه دارد بصورت رندم بعضی از نمونه ها از کلاس بیشتر را برداشته و حذف کنیم که به این روش undersampling میگویند. این روش جزو روش های resamplingهست و اینکار باعث میشود داده های اضافه حذف شود.

Oversampling

در این حالت یکسری از نمونه هایی که در کلاس کمتر یا کلاس اقلیت هست را برداشته و بصورت رندم دوبرابر میکنیم و یا چند برابر همان داده را تکرار میکنیم تا در داده ها به اندازه کلاس بیشتر شود. این روش نیز جزو روش های resampling هست و باعث زیاد شدن داده های تست و ترین میشود.

وزن کلاس ها:

در این روش به هر کلاس وزنی میدهیم تا کمبود داده های ان کلاس را جبران کند. این را میتوان در مدل logistic regression استفاده کرد. در حالت پیش فرض وزن همه ی کلاس ها یک میباشد اما sklearn میتوان با اضافه کردن پارامتر class_weight این مسئله را حل کرد که با گزاشتن نسبت تعداد هر کلاس بهترین وزن برای هر کلاس را میگزاریم. در صورت وزن دار بودن در مدل این وزن ها در کلاس ضرب میشود و خود مدل انرا در نظر میگیرد.

بخش ۴:

سوال ۱:

در حالت کلی به دو روش میتوان این مسیله را حل کرد:

یک کلاس در برابر همه و یک کلاس در برابر یک کلاس. در یک کلاس در برابر همه یا one-vs-all ما به اندازه تعداد کلاس ها که فرض میکنیم n است به ازای هر کلاس یک binary classifier تعریف میکنیم و تمام این کلاسیفایر ها را برای یک داده ی بطور مثال x ران میکنیم و سپس کلاسی که بیشترین امتیاز را دارد را برمیداریم.

در حالت one-vs-one تعداد n(n-1)/2 تا مدل نیاز داریم تا هر کدام از کلاس هارا در برابر همدیگر قرار دهد. سپس کلاسی که بیشترین رای هارا از مدل ها دارد را برمیداریم.

سوال ۲:

همانطور که از کد داریم برای سوال ۴ داریم:

```
precision_score
```

[0.98006645 0.98969072]

```
recall_score
```

[0.99662162 0.94117647]

```
f1_score
```

[0.98827471 0.96482412]

و برای سوال ۵ داریم:

```
precision_score
```

[.1 0.96226415]

```
recall_score
```

[0.98648649 .1]

```
f1_score
```

[0.99319728 0.98076923]

که میتوان دید که وزن دادن به داده ها میتواند تاثیر زیادی داشته باشد

سوال ۳:

برای سوال ۶ داریم برای معیار f1_score :

[0.95652174 0.88151659 0.8358209 0.92]

و برای سوال ۱ داریم:

[0.99459459 0.93333333 0.89655172 0.95959596]

که همانطور که میبینیم عملکرد سوال یک بهتر از سوال ۶ است

سوال ۴:

در حالت کلی تکنیک های انتخاب ویژگی به چهار قسمت تقسیم میشوند که دو تا از انها را داریم:

روش های فیلتری:

در این روش ها بجای انکه از روش های cross-validation و یا درست کردن مدل و تست استفاده کنیم از شاخص های اماری استفاده میکنیم. این روش ها سریعتر و از نظر محاسباتی از روش های بسته بندی کننده کم هزینه ترند. یکی از این روش ها استفاده از ضریب همبستگی است.

روش ضریب همبستگی:

ضریب همبستگی را میتوان به اینصورت دید که در صورتی که صریب همبستگی با ویژگی هدف بالا و یا پایین باشد یا به منفی یک و یک نزدیک باشد ویژگی ارزشمندیست و اطلاعات بیشتری از ویژگی هایی که ضریب همبستگیشان به صفر نزدیکتر است دارند. در صورتی هم که دو ویژگی بیش از حد ضریب همبستگی بالا با همدیگر داشته باشند میتوان یکی را حذف و دیگری را جای ان قرار داد.در اینجا میتوانیم یک مقدار مثل ۰.۵ قرار دهیم و ویژگی های ورودی که ضریب همبستگیشان با همدیگر بیشتر از ۰.۵ است را یکیشان را حذف کنیم.

روش های بسته بندی:

روش های بسته بندی به این صورت عمل میکند که از بین تمامی حالات ممکن برای ورودی ها یک زیر مجموعه را انتخاب کرده و برای ان مدلی میسازد در صورتی که مدل بخوبی کار کند انرا انتخاب میکند. روش های پیشرو و پسرو از همین روش های دسته بندی هستند. از دیگر روش های بسته بندی میتوان به exhaustive feature selection اشاره کرد. در این روش تمام حالت های ممکنی که میتوان ویژگی ها و یا ورودی ها را انتخاب کرد را انتخاب کرده و ارزیابی میکند و بهترین را برمیدارد. واضح است که در این روش برای تعداد ویژگی های زیاد نمیتوان استفاده کرد و این روش سرعت بسیار کمتر اما دقت بیشتری از دیگر روش ها دارد.

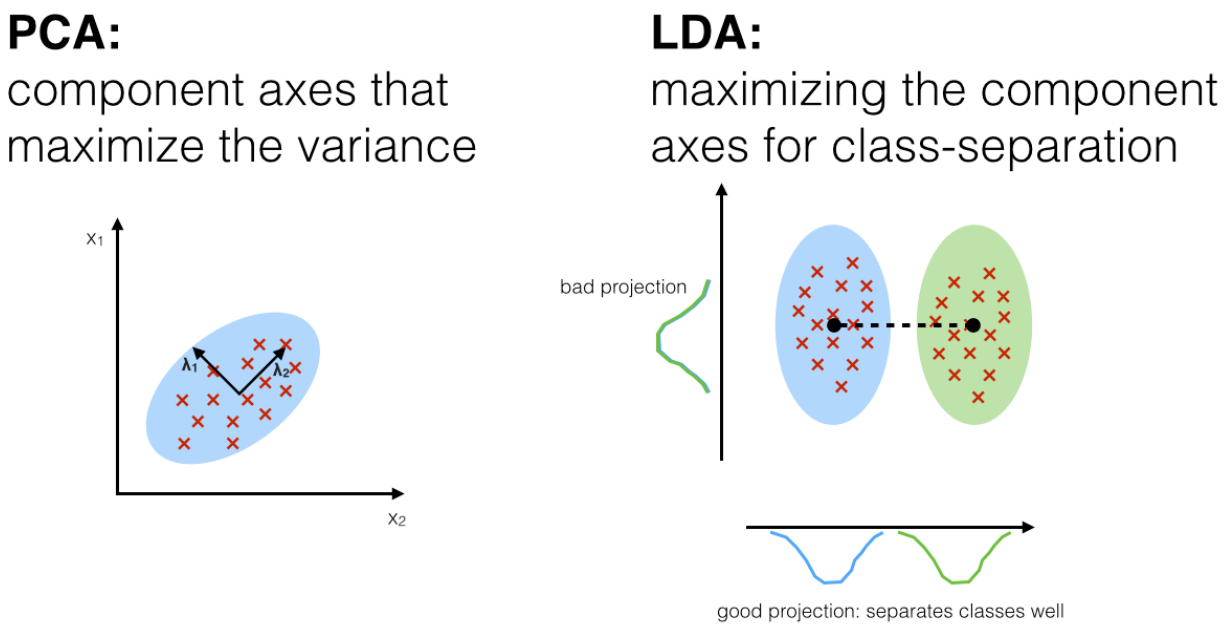
/https://www.analyticsvidhya.com/blog/2020/10/feature-selection-techniques-in-machine-learning

سوال ۵:

میتوان از معایب این دو روش به کند بودن انها اشاره کرد که میتوان با استفاده از ترکیب ان با روش های فیلتری برخی از ویژگی های بدون اطلاعات را حذف کرده و سپس روش پیشرو و پسرو رو روی باقیمانده ی ویژگی ها اعمال کرد که سرعت انرا تا حد خوبی بهبود میدهد.

سوال ۶:

روش LDA در حالت کلی بسیار به PCA شبیه است با این تفاوت که بجای انکه اجزای محورهایی را پیدا کنیم که واریانس را حداکثر میکند، ما محور هایی را انتخاب میکنیم که جدایی بین چند کلاس را افزایش میدهد. در حالت کلی هدف یک LDA اینست که ویژگی هارا به یک فضای کوچکتر تبدیل کند در حالی که اطلاعاتی که باعث میشود کلاس ها از هم جدا باشند را نگه دارد. در مقایسه هر دوی الگوریتم های PCA و LDA میتوان گفت که PCA یک الگوریتم unsupervised هست بطوریکه بدون توجه به لیبل کلاس ها محور با بیشترین واریانس را انتخاب میکند. در نقطه مقابل LDA supervised هست و جهت هایی را انتخاب میکند که در ان محور بیشتری تفاوت بین کلاس ها باشد. تصویر زیر بیانگر این تفاوت هست:



برای مقایسه میتوان گفت که در حالاتی که یک طبقه بندی چند کلاسی داریم که لیبل ها را میدانیم ممکن است LDA بهتر باشد اما همیشه این مورد درست نیست مثلا در حالاتی که موارد نمونه نسبت به هر کلاس کم باشد PDA بهتر عمل میکند. بهترین حالت استفاده از هردو روش است.

https://sebastianraschka.com/Articles/2014_python_lda.html

PCA vs. LDA, A.M. Martinez et al., 2001

سوال ۸:

از انجایی که accuracy نسبت به class imbalance اسیب پذیر است و recall F1-score هم نا متقارن است از این روش استفاده میکنیم.اگر هر دو کلاس مورد نیاز هستند یکی میتواند یک مسئله طبقه بندی دودویی را بعنوان یک مسئله چند کلاسی با دو کلاس ببینیم و سپس شاخص های چند کلاسی را بدست اوریم. در حالت کلی MCC معیاریست که فرمول زیر انرا بدست می اورد:

$$\mathrm{MCC} = \frac{\mathrm{TP}\times\mathrm{TN}-\mathrm{FP}\times\mathrm{FN}}{\sqrt{(\mathrm{TP}+\mathrm{FP})(\mathrm{TP}+\mathrm{FN})(\mathrm{TN}+\mathrm{FP})(\mathrm{TN}+\mathrm{FN})}}$$

اگر مدل بخوبی عمل کند MCC برابر ۱ میباشد و وقتی مدل همیشه بد عمل میکند این معیار برابر ۱- میباشد.