به نام خدا

دانشگاه تهران

پردیس دانشکده‌های فنی

دانشکده برق و کامپیوتر

درس سیستم‌های هوشمند

تمرین شماره ۲

نام و نام خانوادگی : محمدمهدی رحیمی

شماره دانشجویی : ۸۱۰۱۹۷۵۱۰

آبان ۱۴۰۰

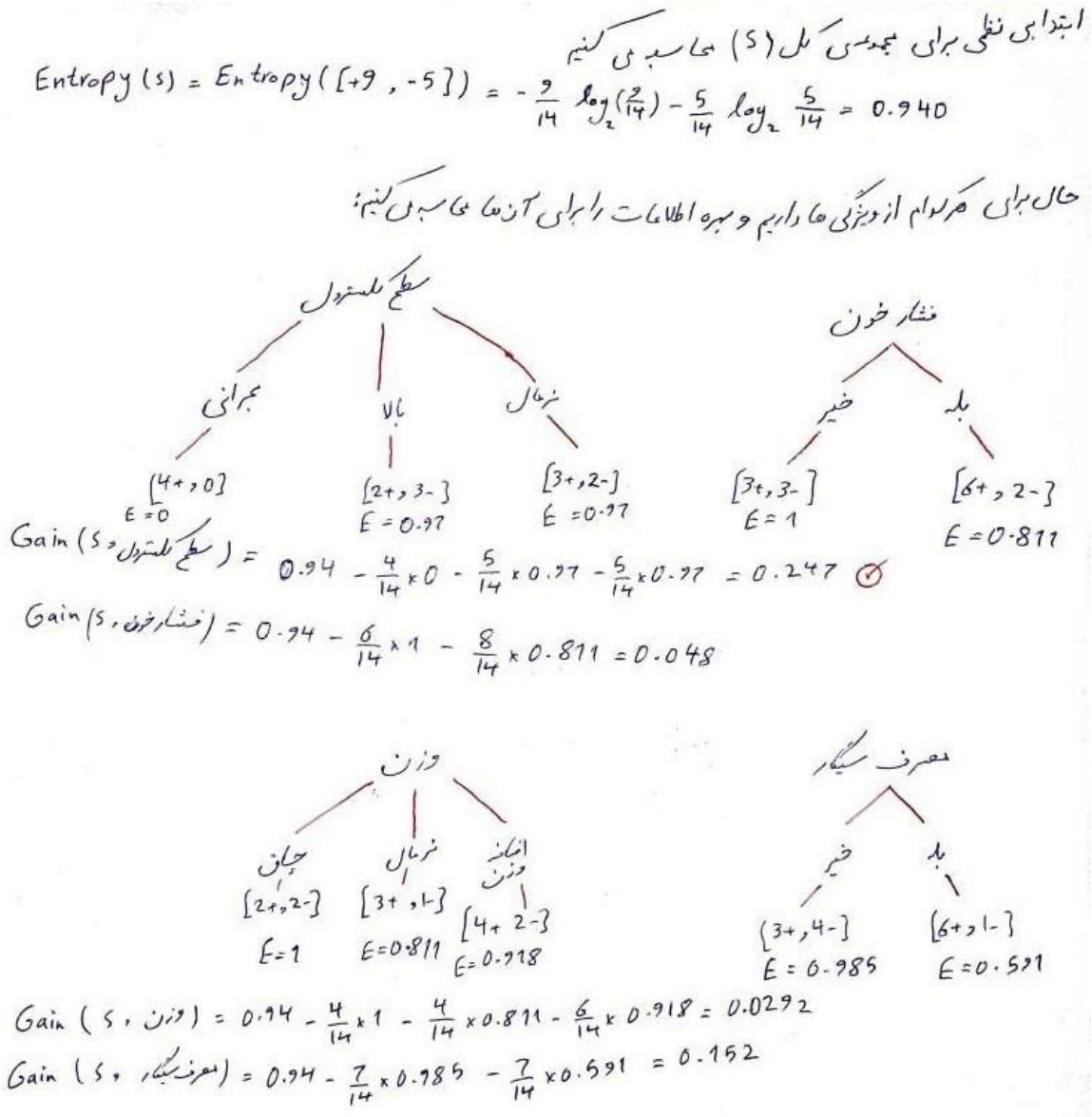
فهرست سوالات

سوال ۱

در این سوال ابتدا با استفاده از بهره اطلاعات و الگوریتم ID3 ، درخت تصمیم را آموزش می دهیم و در قسمت بعد آن را تست می کنیم و در نهایت عملکرد آن را بررسی کرده و برای بهبود آن راه پیشنهاد خواهیم داد. و در انتها راه حلی برای مشکل این الگوریتم ارائه می کنیم.

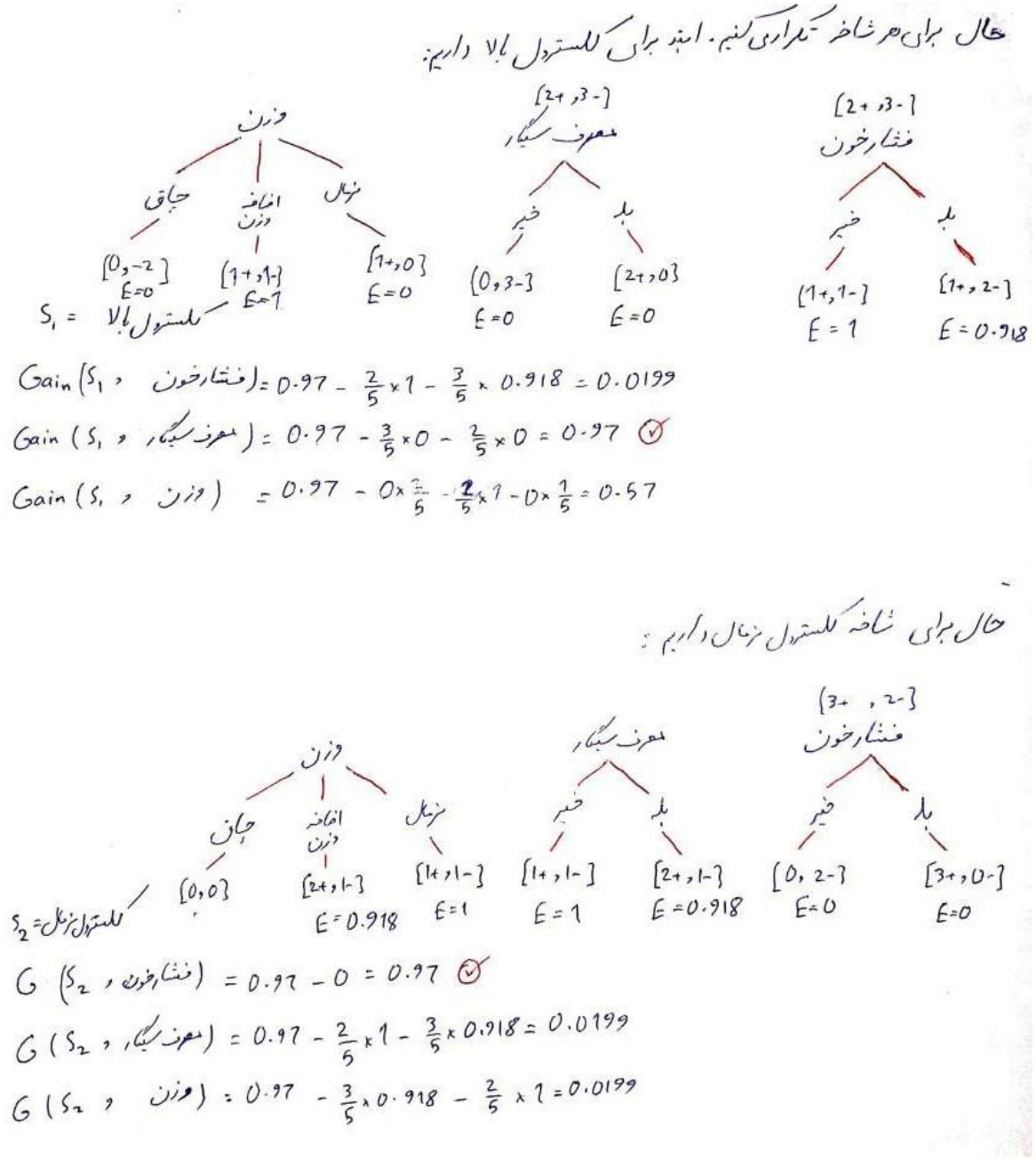
الف: طراحی طبقه بند

با استفاده از بهره اطلاعات داریم:



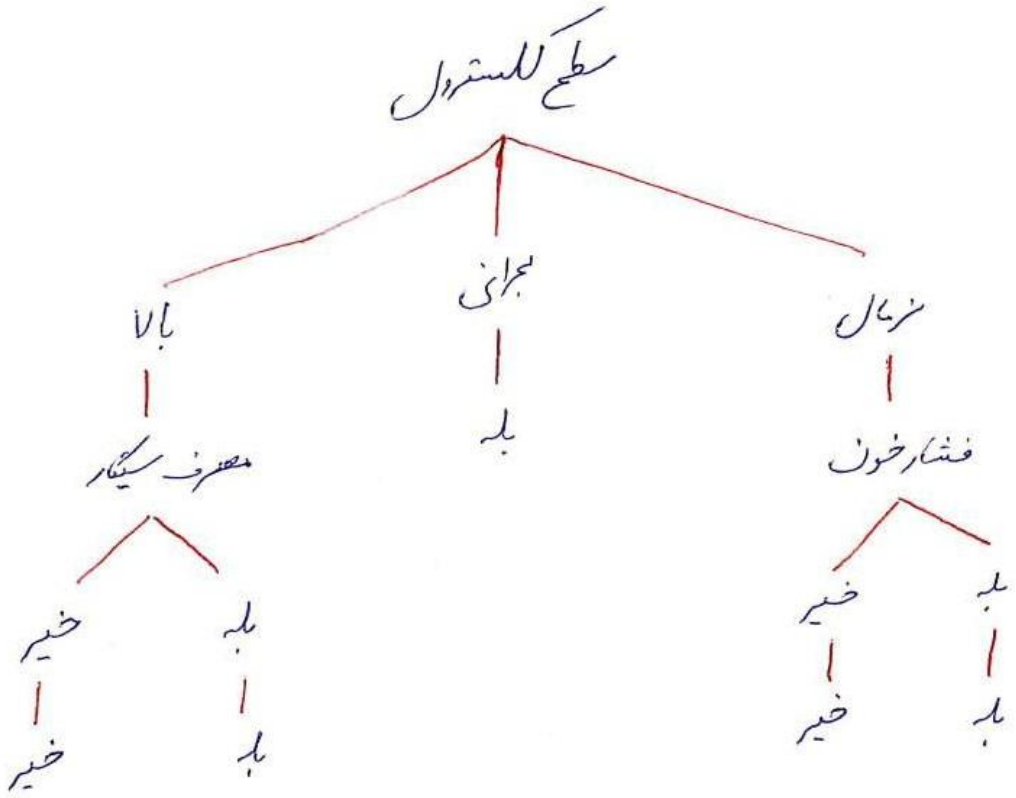
شکل ۱-۱: محاسبه بهره اطلاعات شاخه اول

همانطور که مشاهده می شودبهترین بهره اطلاعات را سطح کلسترول دارد پس آن را انتخاب می کنیم و حال سراغ مرحله بعد می رویم.



شکل ۱-۲: محاسبه بهره اطلاعات مرحله بعد

همانطور که مشاهده می شود برای برگ کلسترول بالا مصرف سیگار بیشترین بهره اطلاعات را دارد و برای برگ کلسترول نرمال فشار خون بیشترین بازده اطلاعات را دارد. برگ کلسترول بحرانی نیز خود به نقطه ی انتهایی رسیده است.



شکل ۱-۳: شکل نهایی درخت تصمیم

همانطور که مشاهده می شود کل داده ها تقسیم بندی شده اند و درخت تصمیم ب حالت نهایی خود رسیده است. آموزش درخت تصمیم به پایان رسیده است.

ب:

با توجه به درخت تصمیمی که در قسمت قبل آموزش دادیم داده هایی که به عنوان تست به ما داده شده است را با استفاده از درخت تصمیم پیش بینی می کنیم و حالت پیش بینی شده و حالت حقیقی را در جدول زیر به نمایش می گذاریم.

جدول ۱-۱ نتیجه داده تست با استفاده از درخت تصمیم آموزش داده شده

شماره	حالت پیش بینی شده	حالت حقیق
۱۵	بله	بله
۱۶	بله	بله
۱۷	نه	نه

۱۸	بله	نه
۱۹	نه	بله

و ماتریس آشفتگی آن را به شکل یک جدول نمایش می دهیم:

جدول۱-۲: ماتریس آشفتگی

	پیش بینی		
		بله	نه
حقیقی	بله	۲	۱
	نه	۱	۱

همانطور که مشاهده می شود درخت آموزش داده شده دقت کافی ندارد که می تواند به خاطر فرا برازش باشد. درخت با توجه به داده های آموزش فرابرازش شده و اگر داده ای به غیر از داده های آموزش را با درخت تصمیم تست کنیم احتمال اینکه اشتباه کند بسیار زیاد است و باید از فرابرازش جلوگیری شود.

ج:

درخت تصمیم در مقابل بیش برازش مقاوم نمی باشد و الگوریتم استفاده شده یعنی ID3 تا زمانی که به گره برسد ادامه پیدا می کند به همین دلیل است که بیش برازش اتفاق می افتد. حال به دو روش جلو گیری از این مشکل اشاره می کنیم.

روش اول:

می توان ابتدا درخت را ساخت سپس آن را حرص کرد. با این کار ابتدا واریانس را بالا می بریم اما بعد که آن را حرص میکنیم نتیجه مطلوبی خواهیم گرفت. برای انجام این کار می توان قسمتی از داده ها را برای اعتبار سنجی جدا کرده و با استفاده از آن به نتیجه مورد نظر رسید

روش دوم:

باید معیاری را انتخاب کنیم و برای درخت های متفاوت آن ار محاسبه کنیم تا درخت مطلوب پیدا شود. این معیار علاوه بر بهره اطلاعات باید ازپراکندگی و سایز درخت نیز استفاده کند تا بتوان درخت با اندازه مناسب را پیدا کرد.

سوال ۲

در این سوال ابتدا داده ها را به داده تست و آموزش تقسیم کرده سپس با استفاده از الگوریتم ID3 با عمق های متفاوت مدل را آموزش داده و خروجی را بررسی می کنیم. سپس در قسمت بعد از الگوریتم جنگل صادفی استفاده می کنیم تا با قسمت قبل مقایسه کنیم و در انتها نیز با استفاده از کتابخانه داده شده بار دیگر داده ها را تس می کنیم.

الف:

در این بخش ابتدا توابعی را ایجاد کردیم تا به توان پروژه را به شکل تمیز تری انجام داد.

ابتدا توابع بی نظمی و بهره اطلاعات را ایجاد می کنیم که با آن بتوانیم از میان چند ویژگی مختلف بهترین آن ها را پیدا کنیم. سپس تابعی را ایجاد کردیم که از بین تمام ویژگی های داده شده بهترین را پیدا کرده و به ما برگرداند. تابع بعدی درخت تصمیم ما را تولید می کند و به صورت بازگشتی خواهد بود. خروجی آن از جنس کلاسی است که در آن مقدار هر گره و شاخه های آن و اینکه گره انتهایی هست یا خیر مشخص می باشد. و دو تابع چاپ و پیش بینی نیز به صورت بازگشتی می باشد که از درخت تصمیم تولید شده استفاده می کنند. ترتیب توابع ذکر شده ترتیبی است که در ابتدای کد نوشته شده است.

سپس داده های مورد نظر را می خوانیم و به داده های تست و آموزش تقسیم می کنیم. روشی که اینجا برای جدا سازی این داده ها استفاده شده است به این شکل است که به شکل یکنواخت عدد رندومی به هر یک از سطر های داده ها نسبت داده و بعد با توجه به میزانی که خودمان می خواهیم تا حد خوبی میتوان داده ها را به همان نسبت تقسیم کرد. دلیل استفاده از این روش برای جلو گیری از استفاده از کتابخانه ها و کد های آماده است. و در انتها نیز ماتریس آشفتگی چاپ می شود.

حال برای عمق های متفاوت کد را اجرا می کنیم.

```
With depth =  3 we have:
Predict   1   0
Actual
1    1127    672
0    202    1152
Accuracy: 72.28036790358388 %
```

شکل۲-۱: ماتریس آشفتگی و دقت برای عمق ۳

```
With depth =  2 we have:
Predict   1   0
Actual
1    1076    664
0    209    1142
Accuracy: 71.75671303785182 %
```

شکل ۲-۲ : ماتریس آشفتگی و دقت برای عمق ۲

```
With depth =  4 we have:
Predict   1   0
Actual
1    1109    632
0    260    1158
Accuracy: 71.76321620766065 %
```

شکل ۲-۳: ماتریس آشفتگی و دقت برای عمق ۴

```
With depth =  5 we have:
Predict   1   0
Actual
1    1268    411
0    443    904
Accuracy: 71.77792465300728 %
```

شکل ۲-۴: ماتریس آشفتگی و دقت برای عمق ۵

همانطور که دقت و ماتریس اشفتگی نشان می دهد بهترین عمق در اینجا ۳ می باشد. زیرا از عمق ۲ به عمق ۳ دقت بالا می رود ولی اگر بیشتر از ۳ بشود دقت کاهش می یابد. در اصل در عمق های کمتر از ۳ مشکل کم برازش داریم ولی در عمق های بالاتر از ۳ بیش برازش رخ می دهد. بهترین عمق در این مسئله ۳ می باشد.

ب:

در این قسمت الگوریتم جنگل تصادفی را استفاده می کنیم. توابعی که از قبل داریم در این بخش استفاده می شوند و تنها تفاوت در ورودی و تعداد تکرار می باشد.

ابتدا باید داده هایی را به صورت bootstrap آماده کرده و تعدادی از ویژگی ها را انتخاب کرده و به الگوریتم قسمت قبل داده تا یک درخت برای ما ایجاد کند. با توجه به نکته گفته شده در درس تعداد مطلوب ویژگی ها باید جذر کل ویژگی ها باشد که در اینجا تعداد کل ویژگی ها ۱۰ می باشد . حال اگر این الگوریتم توضیح داده شده را چند بار انجام دهیم به جنگل تصادفی خواهیم رسید. در کد نوشته شده تعداد درخت، عمق درخت ها و تعداد ویژگی ها قابل تغییر است.

به دلیل سنگین بودن آن را در Google Colab انجام دادیم و نتیجه به شکل زیر می باشد:

```
Predict  1  0
Actual
1   1125   177
0   647   1127
Accuracy: 73.21196358907672
```

شکل۲-۵ : دقت و ماتریس آشفتگی برای جنگل تصادفی با ۱۰۱ درخت

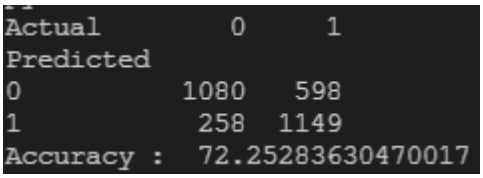
بهتر است تعداد درخت های انتخاب شده فرد باشد تا در مرحله رای اکثریا به موردی بر نخوریم که تعداد رای های مثبت و منفی برابر باشد. برای حالت هایی که این اتفاق می افتد کد پیام اختار نمایش می دهد. پس بهتر است تعداد درخت ها فرد باشد و عمق درخت از تعداد ویژگی های انتخاب شده بیشتر نباشد.

درمورد خروجی همانطور که مشاهده می کنید خروجی بهبود یافته است. به طور کلی جنگل تصادفی دقت را افزایش می دهد زیرا با میانگین گیری از درخت هایی که همبسته نیستند خطا به شدت کاهش می یابد و مشکل بیش برازش کم تر می شود

ج:

در این بخش با استفاده از کتابخانه گفته شده الگوریتم جنگل تصادفی را پیاده سازی می کنیم. در ابتدا نیز رمز گذاری برچسب استفاده کردیم تا بتوان از کتابخانه نام برده شده استفاده کرد.

در این بخش برای به دست آوردن ماتریس آشفتگی و دقت از توابع آماده کتابخانه نام برده شده استفاده کردیم:



شکل۲-۶ : خروجی جنگل تصادفی با استفاده از کتابخانه

سوال ۳:

در این سوال بعد از جدا سازی داده تست و آموزش برای مقادیر مختلف کا با استفاده از الگوریتم کا همسایه نزدیک طبقه بندی را انجام داده و داده های تست را پیش بینی می کنیم و دقت و ماتریس آشفتگی را به دست می آوریم. و در قسمت بعد به کمک یاد گیری بر اساس معیار دقت را بهبود می دهیم.

الف:

در این بخش نیز ابتدا توابعی را نوشته تا پروژه تمیز تر انجام شود. تابع محاسبه فاصله که با توجه به ویژگی های مختلف فاصله دو داده را محاسبه می کند. سپس تابعی که کا نقطه نزدیک را پیدا می کند و تابع رای اکثریت که با توجه به همسایه های نزدیک پیش بینی را انجام میدهد. تابع اصلی نیز از تمام توابع نام برده شده استفاده می کند و با توجه به مقدار حقیقی داده تست شروع به تولید ماتریس آشفتگی می کند و این عمل را برای تمام داده ها تکرار می کند.

```
For  5  nighbor is:
Accuracy =  70.27027027027027 %
Confusion Matrix:
                    c1          c2          c3
             c1     14          1           0
Actual:      c2     1           4           3
             c3     1           5           8
                         Predicted
```

شکل ۳-۱ : دقت و ماتریس آشفتگی برای k = 5

```
For  3  nighbor is:
Accuracy =  75.0 %
Confusion Matrix:
                    c1          c2          c3
             c1     10          0           3
Actual:      c2     1           11          2
             c3     1           1           3
                         Predicted
```

شکل ۳-۲ : دقت و ماتریس آشفتگی برای k = 3

```
For  7  nighbor is:
Accuracy =  70.2127659574468 %
Confusion Matrix:
                    c1          c2          c3
             c1     16          0           0
Actual:      c2     2           11          5
             c3     1           6           6
                         Predicted
```

شکل ۳-۳ : دقت و ماتریس آشفتگی برای k =7

```
For  9  nighbor is:
Accuracy =  70.0 %
Confusion Matrix:
                    c1          c2          c3
            c1      9           0           5
Actual:     c2      0           11          2
            c3      1           4           8
                        Predicted
```

شکل ۳-۴ : دقت و ماتریس آشفتگی برای k = 9

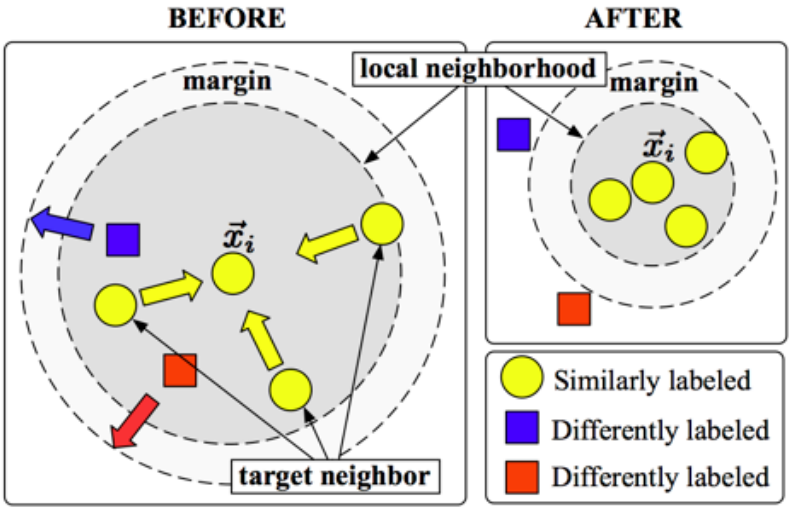
در تعداد همسایگی کم تغییر تعداد تاثیر دارد اما از جایی به بعد تاثیری ندارد و تا حدی می توان بهبود پیدا کند و چون ویژگی ها همه با یک وزن می باشند مقداری از داده را داریم از دست می دهیم و نمی توان دقت از یک مقداری بیشتر شود.

مبنای فاصله استفاده شده اقلیدسی می باشد.

ب:

روش LMNN:

این روش برای افزایش روش کا همسایه نزدیک می باشد که معیار سراسری را به صورت نظارت شده می آموزد. در این روش یک داده نمونه با کا داده که برچسب یکسان دارد محاصره می شود. در این روش همسایه های هدف باید تحت این معیار یادگیری به یکدیگر نزدیک شوند و داده های فریب دهنده را که برچسب آن با داده های همسایه متفائت است ولی براساس محاسبه فاصله همسایه تلقی می شوند از محدوده دور شود و به حداقل برسد.



شکل ۳-۵ : شماتیکی از فرایند LMNN

در این الگوریتم به دنبال مینیمم کردن فاصله نمونه و همسایه و ماکسیمم کردن فاصله با فریب دهنده ها هستیم که می توان به شکل زیر نوشت:

$$\sum_{i,j \in N_i} d(\vec{x}_i, \vec{x}_j)$$

شکل ۳-۶ : نزدیک کردن داده نمونه و همسایه های هدف

$$\sum_{i,j \in N_i, l, y_l \neq y_i} [d(\vec{x}_i, \vec{x}_j) + 1 - d(\vec{x}_i, \vec{x}_l)]_+$$

شکل۳-۷ : دور کردن داده از فریب دهنده ها

و اگر بخواهیم به هر دو هدف برسیم داریم:

$$\min_{\mathbf{M}} \sum_{i,j \in N_i} d(\vec{x}_i, \vec{x}_j) + \lambda \sum_{i,j,l} \xi_{ijl}$$
$$\forall_{i,j \in N_i, l, y_l \neq y_i}$$
$$d(\vec{x}_i, \vec{x}_j) + 1 - d(\vec{x}_i, \vec{x}_l) \leq \xi_{ijl}$$
$$\xi_{ijl} \geq 0$$
$$\mathbf{M} \succeq 0$$

شکل۳-۸:مسئله مدل شده LMNN

روش NCA:

این روش برای طبقه بندی داده های چند متغیره چند کلاسه می باشد. در این روش با استفاده از یک تبدیل خطی از داده های ورودی ، یادگیری معیار فاصله انجام می شود. برای این تبدیل ماتریسی در نظر گرفته که بدنبال آن هستیم و برای بهینه سازی هدف تلاش می کنیم.

در این روش داده های نگاشت داده شده را به عنوان همسایه های نزدیک گرفته و به کمک یک تابع از مربعات فاصله اقلیدسی محاسبات را انجام می دهیم که داریم:

$$p_{ij} = \frac{\exp(-\|Ax_i - Ax_j\|^2)}{\sum_{k \neq i} \exp(-\|Ax_i - Ax_k\|^2)} \quad , \quad p_{ii} = 0$$

شکل۳-۹ : تابع احتمال طبقه بندی

در این تابع احتمال طبقه بندی دو نقطه بر اساس یکدیگر می باشد.

$$p_i = \sum_{j \in C_i} p_{ij}$$

شکل۳-۱۰

و احتمال طبقه بندی نقطه ای برا اساس همسایه ها به صورت شکل ۳-۱۰ می باشد. پس تابع هدف به شکل زیر است:

$$f(A) = \sum_i \sum_{j \in C_i} p_{ij} = \sum_i p_i$$

شکل۳-۱۱: تابع هدف

حال بهینه سازی میکنیم:

$$\frac{\partial f}{\partial A} = -2A \sum_i \sum_{j \in C_i} p_{ij} (x_{ij} x_{ij}^\top - \sum_k p_{ik} x_{ik} x_{ik}^\top)$$

شکل ۳-۱۲: بهینه سازی تابع هدف

حال پیاده سازی عملی روش ها را انجام می دهیم.

```
For  5  nighbor is:
Accuracy with LMNN metric =  100.0 %
Confusion Matrix with LMNN metric:
                     c1          c2          c3
            c1       14          0           0
Actual:     c2        0          18           0
            c3        0           0          10
                        Predicted
Accuracy with NCA metric =  69.04761904761905 %
Confusion Matrix with NCA metric:
                     c1          c2          c3
            c1       13          0           1
Actual:     c2        0          11           7
            c3        1           4           5
                        Predicted
```

شکل ۳-۱۳ : خروجی برای k = 5

```
For  3  nighbor is:
Accuracy with LMNN metric =  91.17647058823529 %
Confusion Matrix with LMNN metric:
                      c1          c2          c3
             c1      14          0           0
Actual:      c2       1          11           1
             c3       0           1           6
                          Predicted
Accuracy with NCA metric =  70.58823529411765 %
Confusion Matrix with NCA metric:
                      c1          c2          c3
             c1      12          1           1
Actual:      c2       2          7           4
             c3       0           2           5
                          Predicted
```

شکل ۳-۱۴ : خروجی برای k = 3

```
For  7  nighbor is:
Accuracy with LMNN metric =  90.32258064516128 %
Confusion Matrix with LMNN metric:
                      c1          c2          c3
             c1      10          0           0
Actual:      c2       1          10          2
             c3       0           0          8
                          Predicted
Accuracy with NCA metric =  64.51612903225806 %
Confusion Matrix with NCA metric:
                      c1          c2          c3
             c1      8           0           2
Actual:      c2       2          7           4
             c3       0           3          5
                          Predicted
```

شکل ۳-۱۵ : خروجی برای k = 7

دقت طبقه بند نسبت به بخش قبل بسیار افزایش پیدا کرده است.

برای یافتن بهترین تعداد همسایگی باید برای تعداد همسایه های مختلف داده ها را تست کرده و نموداری از آن رسم کنیم و با توجه به رفتار نمودار می توان بهترین تعداد را پیدا کرد.