به نام خدا

دانشگاه تهران

پردیس دانشکده‌های فنی

دانشکده برق و کامپیوتر

درس سیستم‌های هوشمند

تمرین شماره 2

نام و نام خانوادگی : محمدرضا بختیاری

شماره دانشجویی : 810197468

مهر 1400

فهرست سوالات

سوال 1 : درخت تصمیم (تحلیلی) 3

الف: طراحی طبقه بند 3

ب: آزمون طبقه بند 5

ج: افزایش قوام طبقه بند 6

سوال 2 : درخت تصمیم (شبیه سازی) 8

الف: طراحی طبقه بند 8

ب: استفاده از جنگل تصادفی 9

ج: استفاده از کتابخانه 10

سوال 3 : یادگیری بر اساس معیار 11

الف: کا-همسایه نزدیک 11

ب: یادگیری بر اساس معیار 12

حاشیه بزرگ همسایه های نزدیک (LMNN) 12

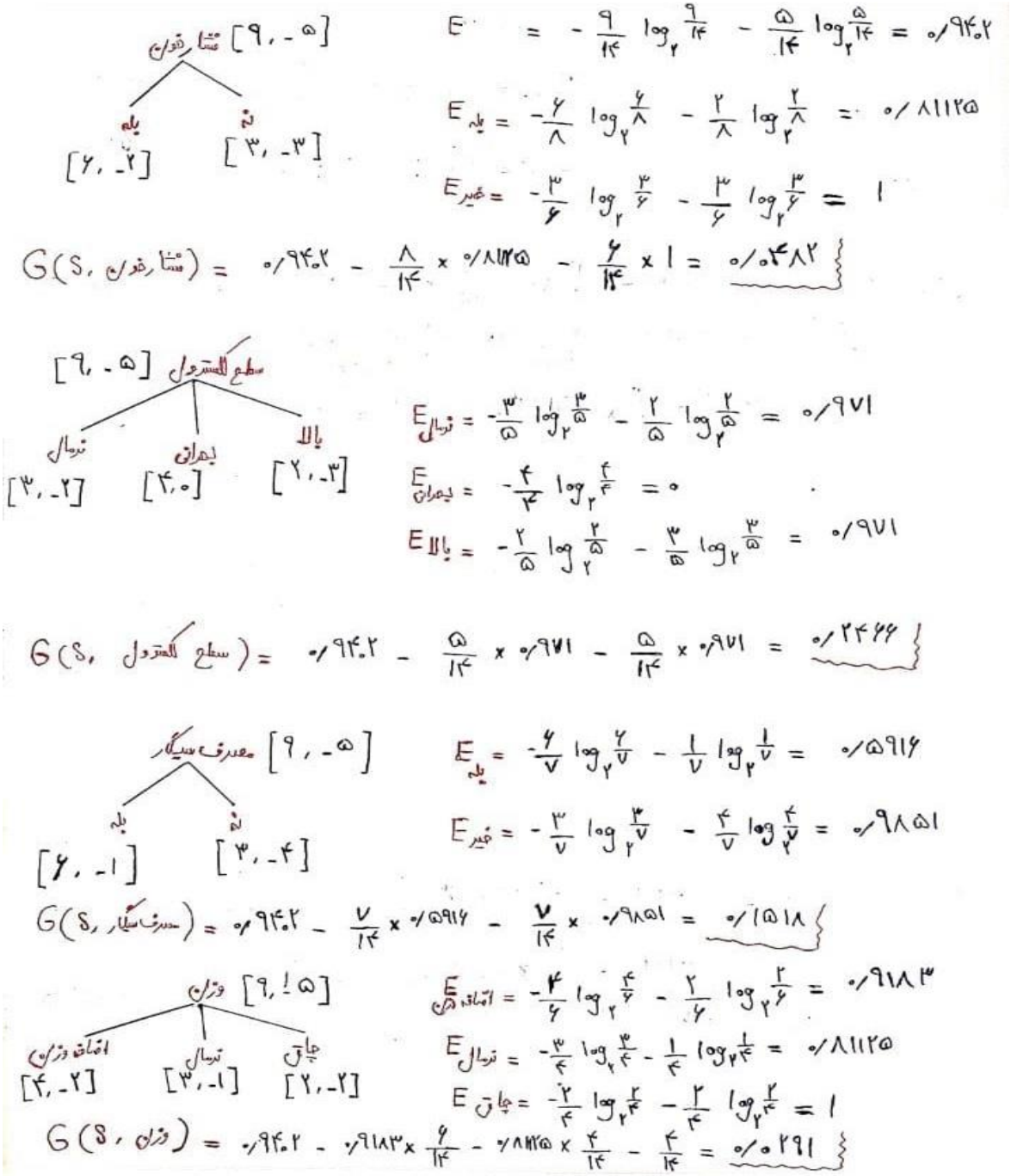
یادگیری متریک نظری اطلاعات (ITML) 13

پیوست: 16

سوال 1 : درخت تصمیم (تحلیلی)

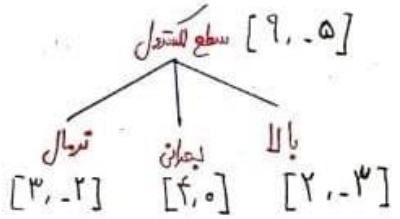
در این سوال اطلاعات چند بیمار داده شده و از ما خواسته شده است که بر اساس بهره اطلاعات یک طبقه بند درخت تصمیم را پیاده سازی کنیم و در ادامه آن را بر روی چند داده آزمون تست کنیم . درنهایت عملکرد مدل را بررسی می کنیم و دو راهکار برای جلوگیری از فرابرازش[1] ارائه می دهیم .

الف: طراحی طبقه بند



[1] Overfitting

$\max(G) = G(S, \text{سطح کلسترول}) = ۰/۲۴۶۶$

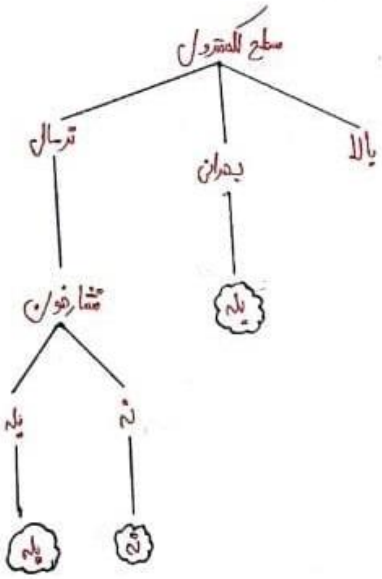


$$\begin{cases} E_{\text{نرمال}} = -\frac{۲}{۵}\log_۲\frac{۲}{۵} - \frac{۳}{۵}\log_۲\frac{۳}{۵} = ۰/۹۷۱ \\ E_{\text{بالا}} = -\frac{۳}{۵}\log_۲\frac{۳}{۵} - \frac{۲}{۵}\log_۲\frac{۲}{۵} = ۰/۹۷۱ \end{cases}$$

$G(S_{\text{نرمال}}, \text{فشارخون}) = ۰/۹۷۱ - \frac{۳}{۵}\times ۰ - \frac{۲}{۵}\times ۰ = ۰/۹۷۱ \longleftarrow \max G$

$G(S_{\text{نرمال}}, \text{مصرف سیگار}) = ۰/۹۷۱ - \frac{۳}{۵}\times ۰/۹۱۸۳ - \frac{۲}{۵}\times ۱ = ۰/۰۲$

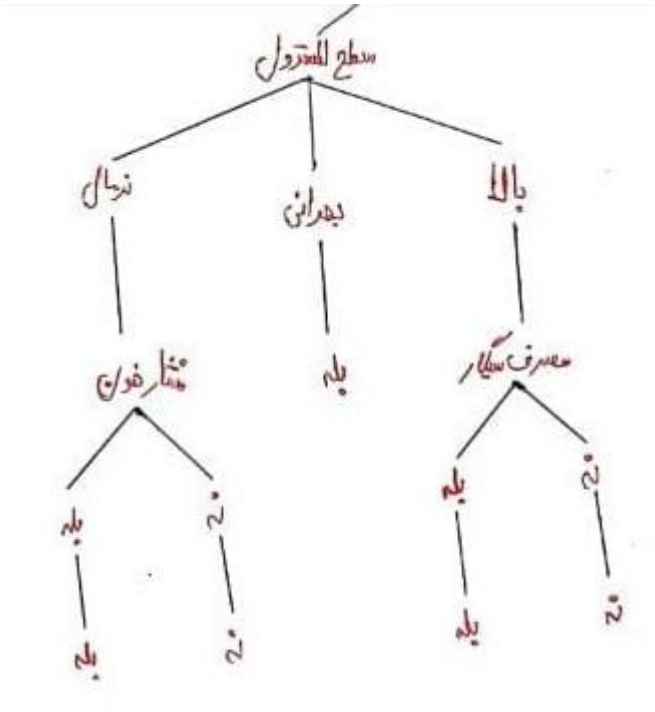
$G(S_{\text{نرمال}}, \text{وزن}) = ۰/۹۷۱ - \frac{۳}{۵}\times ۰/۹۱۸۳ - \frac{۲}{۵}\times ۱ = ۰/۰۲$



$G(S_{\text{بالا}}, \text{فشارخون}) = ۰/۹۷۱ - ۰/۹۱۸۳\times\frac{۳}{۵} - \frac{۲}{۵} = ۰/۰۲$

$G(S_{\text{بالا}}, \text{مصرف سیگار}) = ۰/۹۷۱ - \frac{۲}{۵}\times ۰ - \frac{۳}{۵}\times ۰ = ۰/۹۷۱ \longleftarrow \max G$

$G(S_{\text{بالا}}, \text{وزن}) = ۰/۹۷۱ - \frac{۲}{۵} = ۰/۵۷۱$



شکل 1-1 : درخت تصمیم نهایی

ب: آزمون طبقه بند

با استفاده از درخت تصمیم به دست آمده در قسمت قبل , دادگان آزمون جدول زیر را پیش بینی می کنیم و با مقدار واقعی آن ها مقایسه می کنیم .

جدول 2-1: اطلاعات بیماران و وضعیت ابتلا به بیماری انسداد عروق برای دادگان آزمون

شماره	فشار خون	سطح کلسترول	مصرف سیگار	وزن	انسداد شرایین	پیش بینی
15	بله	نرمال	بله	چاق	بله	بله
16	بله	بالا	بله	چاق	بله	بله
17	بله	بالا	نه	نرمال	نه	نه
18	بله	نرمال	نه	نرمال	نه	بله
19	نه	نرمال	بله	اضافه وزن	بله	نه

تعداد داده هایی که هم شبیه سازی و هم در واقعیت در گروه یک (بله) قرار دارند 2 می باشد . تعداد داده هایی که هم شبیه سازی و هم واقعیت در گروه دو (نه) قرار دارند 1 می باشد . تعداد داده هایی که در واقعیت در گروه یک می باشند اما در شبیه سازی در گروه دو قرار دارند برابر 1 می باشد و در نهایت

تعداد داده هایی که در واقعیت در گروه دو می باشند اما در شبیه سازی در گروه یک قرار دارند نیز برابر 1 می باشد .

در نتیجه ماتریس آشفتگی ما به صورت زیر می باشد :

ماتریس آشفتگی : $\begin{matrix} 2 & 1 \\ 1 & 1 \end{matrix}$

پس از بررسی 5 داده آزمون مشاهده می کنیم که نتیجه 2 داده با نتیجه واقعی متفاوت بوده و دوم بیمار شماره 18 و 19 که سطح کلسترول نرمال دارند , باید در مرحله بعدی به جای فشار خون , مصرف سیگار آن ها بررسی شود تا نتیجه با وقعیت یکی شود .

در این جا چون درخت را با داده های آموزش , درخت را تعلیم[1] می دهیم به احتمال زیاد فرابرازش رخ خواهد داد و اگر داده ای خارج از داده های آموزش باشد , ممکن است نتیجه به دست آمده با نتیجه مورد انتظار یکی نباشد که در قسمت بعد به دو روش برای جلو گیری از این اتفاق اشاره می کنیم .

ج: افزایش قوام طبقه بند

همان طور که در قسمت قبل اشاره شد , وقتی درخت را به ازای داده ها آموزش تعلیم می دهیم , فرابرازش رخ می دهد . در این حالت واریانس مدل بالا بوده و اگر داده ای خارج از داده های آموزش را بررسی کنیم ممکن است نتیجه حاصل شده نادرست باشد , برای جلو گیری از این اتفاق دو روش زیر را پیشنهاد می کنیم :

1- این روش هنگامی استفاده می شود که داده ها به قدری کم هستند که نمی توانیم داده اعتبار سنجی[2] داشته باشیم در این صورت می توانیم از رشد درخت با یک معیار ریاضی جلوگیری کنیم و اجازه ندهیم درخت از یک حدی بیشتر رشد کند در نتیجه نیاز کمتری به داده اعتبار سنجی داریم .

2- در این روش ابتدا درخت را بزرگ می کنیم و سپس به هرس کردن[3] درخت می پردازیم . سپس معیار انتخاب بهترین درخت و این که کجا را هرس کنیم به این صورت خواهد بود که دقت را روی داده های اعتبار سنجی که درخت روی آن ساخته نشده است در نظر می گیریم .
در نظر داشته باشیم که هیچ گاه داده اعتبار سنجی را برای توسعه درخت به کار نمی بریم و در اینجا از این داده ها فقط برای به دست آوردن انتخاب شاخه ها استفاده می کنیم .
به صورت خلاصه الگوریتم به این صورت است که ابتدا درخت کامل ایجاد شده و از انتها شروع به هرس

[1] train
[2] Validation data
[3] Post-prune

کردن می کنیم و هر جا دقت روی داده های اعتبار سنجی ما دقت بهتری بود یا برابر بود , در این صورت شروع به هرس کردن می کنیم .

در این حالت از Minimum Description Length (MDL) استفاده می کنیم که هدف کمینه کردن اندازه درخت بعلاوه ضریبی از خطای طبقه بندی درخت است :

Minimize [size(tree) + a*size(misclassifications(tree))]

هنگامی که تصمیم گیری بر روی یک گره باعث نشود خطا خیلی کاهش پیدا کند , گروه مورد نظر را قبول نمی کنیم , زیرا باعث شده است که اندازه درخت افزایش پیدا کند ولی خطا تغییر نکند .

سوال 2 : درخت تصمیم (شبیه سازی)

در این سوال , با استفاده از الگوریتم ID3 یک درخت تصمیم برای دادگان مورد نظر پیاده سازی می کنیم , در ادامه با تغییر عمق درخت , تغییر در دقت و ماتریس آشفتگی را تحلیل می کنیم و در ادامه با استفاده از الگوریتم جنگل تصادفی , دقت به دست آمده را با حالت قبل مقایسه می کنیم و در انتها با استفاده از کتابخانه Scikit-Learn دقت و ماتریس آشفتگی به دست آمده را با حالت قبل (جنگل تصادفی) مقایسه می کنیم .

الف: طراحی طبقه بند

با استفاده از الگوریتم ID3 و بهره گیری از بهره اطلاعات[1] برای انتخاب بهترین ویژگی در هر مرحله , طراحی طبقه بند را پیاده سازی می کنیم . به این صورت که در هر مرحله از بین صفت[2] های موجود با استفاده از بهره اطلاعات , بهترین صفت را انتخاب کرده و این فرایند را انقدر ادامه می دهیم تا یکی از دو شرط زیر اقناع شده باشند :

1- تمامی برچسب[3] های یک گره یکسان باشند (در این مثال همه یا 0 باشند یا 1)

2- از تمامی ویژگی ها استفاده شده و دیگر صفتی باقی نمانده است یا این که به عمق دلخواه خود رسیده ایم و دیگر نیازی به افزایش تعداد شاخه ها نداریم , که در این صورت اگر در گره ای همه ی برچسب های یکسان نباشد , معیار آن گره را برچسبی در نظر می گیریم که فراوانی بیشتری داشته باشد.

نکته : لازم به ذکر است , درخت تصمیم ایجاد شده پس از هر بار اجرای برنامه در فایلی به نام tree.txt ذخیره می شود .

دقت و ماتریس آشفتگی به ازای عمق 3 :

ماتریس آشفتگی : $\begin{matrix} 1075 & 656 \\ 201 & 1153 \end{matrix}$ دقت : 72.22 %

دقت و ماتریس آشفتگی به ازای عمق 2 :

ماتریس آشفتگی : $\begin{matrix} 1043 & 672 \\ 218 & 1152 \end{matrix}$ دقت : 71.15 %

[1] Information Gain
[2] Attribute
[3] Label

دقت و ماتریس آشفتگی به ازای عمق 1 :

ماتریس آشفتگی : $\begin{matrix} 1755 & 0 \\ 1330 & 0 \end{matrix}$ دقت : 56.88 %

مشاهده کردیم بر اساس دقت و ماتریس آشفتگی به دست آمده , هر چقدر عمق درخت را کمتر کنیم (در واقع از ویژگی ها کمتری برای طراحی درخت استفاده کنیم) به دقت کمتری دست پیدا خواهیم کرد.

ب: استفاده از جنگل تصادفی

این الگوریتم برای هر نمونه یک درخت تصمیم گیری می کند . سپس نتیجه پیش بینی را از هر درخت تصمیم به دست می آورد , در مرحله بعدی رای گیری برای هر نتیجه پیش بینی شده انجام می شود . در انتها بیشترین نتیجه پیش بینی را به عنوان نتیجه پیش بینی نهایی انتخاب می کنیم .

دقت و ماتریس آشفتگی به ازای 5 درخت :

ماتریس آشفتگی : $\begin{matrix} 1585 & 110 \\ 1088 & 302 \end{matrix}$ دقت : 61.16 %

دقت و ماتریس آشفتگی به ازای 10 درخت :

ماتریس آشفتگی : $\begin{matrix} 1045 & 639 \\ 215 & 1186 \end{matrix}$ دقت : 72.31 %

دقت و ماتریس آشفتگی به ازای 15 درخت :

ماتریس آشفتگی : $\begin{matrix} 1089 & 619 \\ 200 & 1177 \end{matrix}$ دقت : 73.45 %

دقت و ماتریس آشفتگی به ازای 20 درخت :

ماتریس آشفتگی : $\begin{matrix} 1321 & 422 \\ 451 & 891 \end{matrix}$ دقت : 71.1 %

مشاهده می کنیم در ابتدا دقت به دست آمده به ازای 5 درخت در الگوریتم جنگل تصادفی از حالت قبل کمتر هم می باشد , اما با افزایش تعداد درختان دقت بالاتر می رود و از حالت قبل کمی بهتر می شود , علت این است در درخت تصمیم (به خصوص وقتی تعداد شاخه ها زیاد شود) فرابرازش[1] رخ می دهد (چون واریانس بالا است) ولی هنگامی که از جنگل تصادفی استفاده می کنیم , با اینکه واریانس هر کدام از درخت ها کماکان زیاد است اما هنگامی که از خروجی این درخت ها میانگین می گیریم واریانس کاهش پیدا کرده و دقت نسبت به حالت قبل افزایش پیدا می کند .

ج: استفاده از کتابخانه

در این سوال برای حل مشکل ذکر شده در صورت سوال از sklearn.preprocessing.LabelEncoder کمک می گیریم و از این لینک استفاده می کنیم .

ماتریس آشفتگی و دقت به دست آمده با استفاده از کتابخانه Scikit-Learn :

ماتریس آشفتگی : $\begin{matrix} 1071 & 225 \\ 630 & 1159 \end{matrix}$ دقت : 72.28 %

مشاهده می کنیم دقت به دست آمده از طریق پیاده سازی الگوریتم با استفاده از کتابخانه به صورت تقریبی برابر با قسمت ب (بدون استفاده از کتابخانه) می شود .

[1] Overfitting

سوال 3 : یادگیری بر اساس معیار

در این سوال ابتدا با استفاده از طبقه بند کا-همسایه نزدیک , طبقه بندی را انجام می دهیم , و با تغییر تعداد همسایگی , تغییرات احتمالی دقت را تحلیل می کنیم , سپس با استفاده از دو روش بر اساس یادگیری معیار , بار دیگر طبقه بند را پیاده سازی کرده و دقت را با حالت قبل مقایسه می کنیم . در انتها با تغییر تعداد همسایگی و تحلیل دقت های به دست آمده سعی در ارائه یک روش سیستماتیک برای تعیین تعداد همسایه بهینه می کنیم .

الف: کا-همسایه نزدیک

با استفاده از نرم اقلیدسی فاصله هر داده آزمون را از داده های آزمایش پیدا کرده و در اولین گام , 5 تا از نزدیک ترین داده ها را انتخاب کرده و با استفاده از رای اکثریت[1] , کلاس داده آموزش مورد نظر را پیدا می کنیم و در ادامه ماتریس آشفتگی و دقت را برای تعداد همسایگی های مختلف گزارش می کنیم .

دقت و ماتریس آشفتگی به ازای 5 همسایگی نزدیک :

ماتریس آشفتگی :

$$\begin{matrix} 9 & 1 & 0 \\ 0 & 15 & 4 \\ 0 & 4 & 3 \end{matrix}$$

دقت : 75 %

```
Accuracy =  75.0  %
Confusion Matrix =
[9, 1, 0]
[0, 15, 4]
[0, 4, 3]
```

شکل 3-1 : دقت و ماتریس آشفتگی به ازای k = 5

دقت و ماتریس آشفتگی به ازای 3 همسایگی نزدیک :

ماتریس آشفتگی :

$$\begin{matrix} 13 & 1 & 0 \\ 0 & 10 & 3 \\ 1 & 3 & 4 \end{matrix}$$

دقت : 77.14 %

```
Accuracy =  77.14285714285715  %
Confusion Matrix =
[13, 1, 0]
[0, 10, 3]
[1, 3, 4]
```

شکل 3-2 : دقت و ماتریس آشفتگی به ازای k = 3

[1] Majority voting

دقت و ماتریس آشفتگی به ازای 9 همسایگی نزدیک :

ماتریس آشفتگی : $\begin{matrix} 8 & 2 & 0 \\ 0 & 12 & 1 \\ 0 & 8 & 4 \end{matrix}$ دقت : 68.57 %

```
Accuracy =  68.57142857142857  %
Confusion Matrix =
[8, 2, 0]
[0, 12, 1]
[0, 8, 4]
```

شکل 3-3 : دقت و ماتریس آشفتگی به ازای k = 9

مشاهده می کنیم به ازای تعداد همسایگی های بیشتر دقت نیز کمتر می شود .

اگر تعداد همسایگی ها را از 5 به 3 برسانیم دقت به میزان اندکی افزایش پیدا می کند اما باعث بهبود چشم گیر آن نمی شود .

توجه 1 : مقادیر دقت و ماتریس آشفتگی پس از هر بار اجرا به علت تصادفی انتخاب شدن داده های تست و آموزش , ممکن است کمی با مقادیر ذکر شده تفاوت داشته باشند و دقیقا معادل آن ها نباشند , اما کماکان تحلیل های ذکر شده پا برجا باقی خواهند ماند .

توجه 2 : در برخی حالت ها ممکن است تعداد همسایگی های دو کلاس برابر شوند و برای حل این موضوع از randrange استفاده می کنیم , به این حالت که به صورت تصادفی یکی از کلاس ها را برای داده مورد نظر در نظر می گیریم .

ب: یادگیری بر اساس معیار

حاشیه بزرگ همسایه های نزدیک (LMNN)

متریک آموخته شده تلاش می کند تا k نزدیک ترین همسایه ها را از یک کلاس نگه دارد , در حالی که نمونه هایی از کلاس های مختلف را با یک حاشیه بزرگ از هم جدا نگه می دارد .

مسافت مورد نظر با حل مساله بهینه سازی زیر به دست می آید :

$$\min_{\mathbf{L}} \sum_{i,j} \eta_{ij} ||\mathbf{L}(\mathbf{x_i} - \mathbf{x_j})||^2 + c \sum_{i,j,l} \eta_{ij}(1 - y_{ij})[1 + ||\mathbf{L}(\mathbf{x_i} - \mathbf{x_j})||^2 - ||\mathbf{L}(\mathbf{x_i} - \mathbf{x_l})||^2]_+)$$

که در آن xi یک نقطه داده است , xj یکی از k نزدیکترین همسایگان آن است که برچسب یکسانی را به اشتراک می گذارد , و xl ها همه نمونه های دیگر در آن منطقه با برچسب های مختلف هستند , و $\eta ij, yij \in \{0,1\}$ هر دو نشانگر هستند , ηij نشان دهنده ی xj نزدیک ترین همسایه های (با برچسب

یکسان) از xi است . معادله yij=0 نشان دهنده ی تعلق xi , xj به کلاس های مختلف است . در نهایت $[\cdot]_+=\max(0,\cdot)$. نشان دهنده ی اتلاف هینگ[1] است .

یادگیری متریک نظری اطلاعات (ITML)

ما مسئله یادگیری متریک را به عنوان به حداقل رساندن آنتروپی نسبی دیفرانسیل بین دو گاوسی چند متغیره تحت محدودیت های تابع فاصله Mahalanobis فرموله می کنیم . از طریق یک معادله , نشان می دهیم که این مشکل می تواند به عنوان یک مشکل یادگیری هسته رتبه پایین حل شود . به طور خاص ما واگرایی Burg یک هسته با رتبه پایین را به یک هسته ورودی , مشروط به محدودیت های فاصله زوجی , به حداقل می رسانیم . رویکرد ما نسبت به روش های موجود چندید مزیت دارد . ابتدا , یک فرمول نظری اطلاعات طبیعی برای مسئله ارائه می کنیم . دوم , الگوریتم از روش های توسعه یافته توسط Kulis استفاده می کند که شامل هیچ محاسبات بردار ویژه نیست . به ویژه زمان اجرای روش ما سریع تر از بسیاری از تکنیک های موجود است . سوم , این فرمول بینش هایی را در مورد ارتباط بین یادگیری متریک و یادگیری هسته ارائه می دهد .

در نهایت هدف بهینه سازی تابع هزینه زیر می باشد :

$$\mathrm{KL}(p(\boldsymbol{x};\boldsymbol{m},A_1)\|p(\boldsymbol{x};\boldsymbol{m},A_2)) = \int p(\boldsymbol{x};\boldsymbol{m},A_1)\log\frac{p(\boldsymbol{x};\boldsymbol{m},A_1)}{p(\boldsymbol{x};\boldsymbol{m},A_2)}\,d\boldsymbol{x}.$$

$$\begin{aligned} \min \quad & \mathrm{KL}(p(\boldsymbol{x};\boldsymbol{m},A)\|p(\boldsymbol{x};\boldsymbol{m},I)) \\ \text{subject to} \quad & d_A(\boldsymbol{x}_i,\boldsymbol{x}_j)\le u \qquad (i,j)\in S, \\ & d_A(\boldsymbol{x}_i,\boldsymbol{x}_j)\ge l \qquad (i,j)\in D. \end{aligned}$$

در ادامه به ازای 5 همسایگی در روش LMNN دقت زیر حاصل می شود :

```
Accuracy for lmnn is =  96.11726857851123  %
```

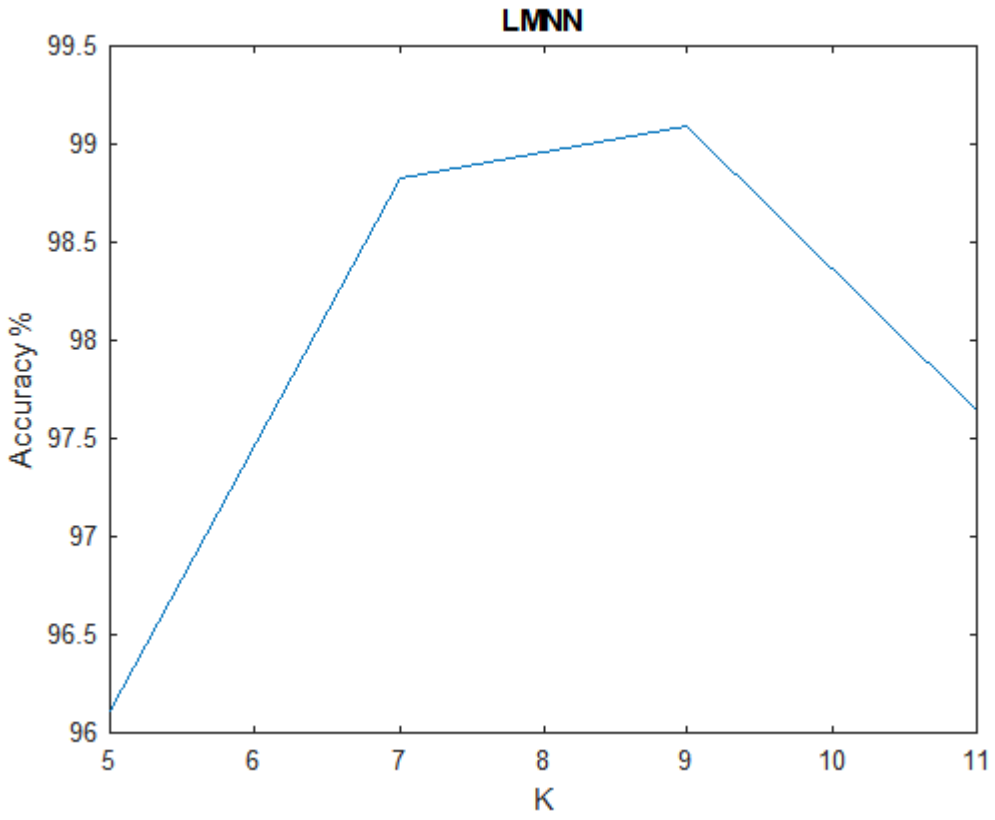
همچنین برای روش ITML به ازای 5 همسایگی دقت زیر را داریم :

```
Accuracy for itml is =  84.6733810001537  %
```

مشاهده می کنیم به علت قوی تر بودن الگوریتم و استفاده از یادگیری بر اساس معیار دقت به دست آمده به مراتب از حالت قبل بهتر بوده .

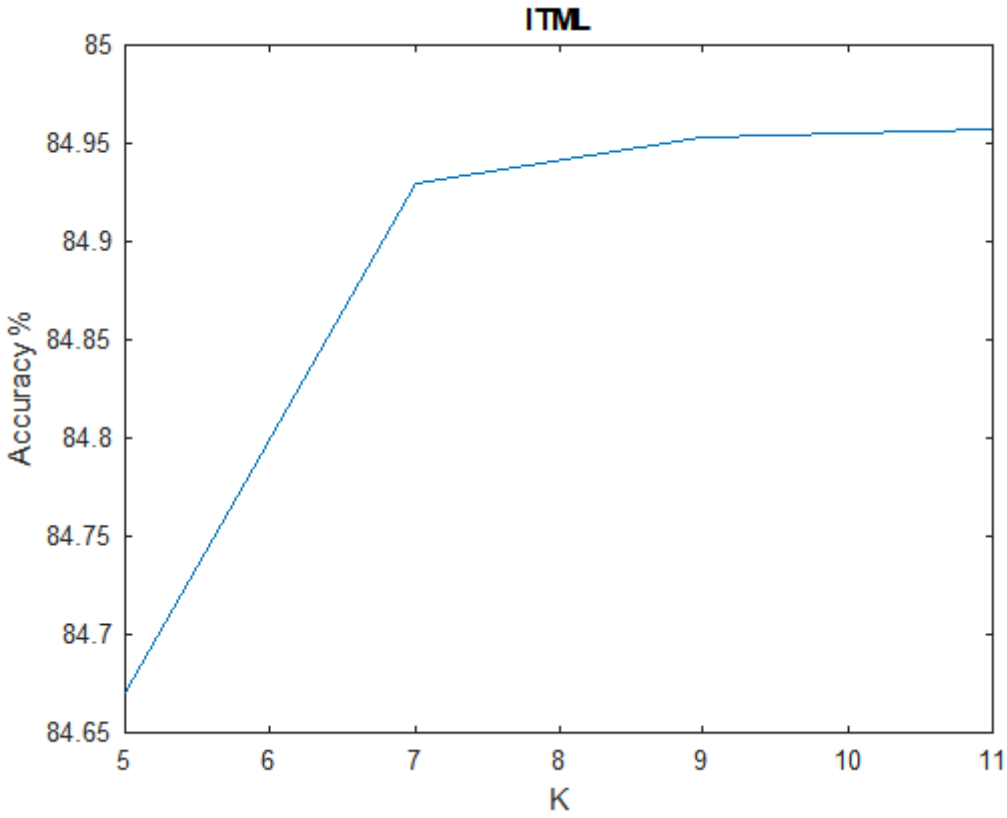
[1] Hinge loss

حال با تغییر تعداد همسایگی در هر دو روش به تحلیل سیستماتیک برای تعیین تعداد همسایه بهینه ارائه می کنیم :



شکل 4-3 : تغییرات دقت به ازای تغییرات تعداد همسایگی در روش LMNN

مشاهده می کنیم لزوما به ازای افزایش تعداد همسایگی , دقت افزایش نمیابد , اگر چه اختلاف دقت به دست آمده به ازای همسایگی های مختلف بسیار ناچیز است .



شکل 5-3 : تغییرات دقت به ازای تغییرات تعداد همسایگی در روش ITML

مشاهده می کنیم در روش ITML ابتدا با افزایش تعداد همسایگی دقت کمی افزایش پیدا کرده ولی در ادامه با افزایش همسایگی دقت تقریبا ثابت بوده و به نوعی به اشباع رسیده است .

پیوست:

در سوال 2 و 3 برای خوانا شدن هر چه بیشتر کد , فایل جداگانه برای توابع مورد نیاز قرار داده شده است و لازم است پیش از اجرای کد اصلی , یک بار فایل توابع اجرا شوند و سپس کد اصلی اجرا شود .