Data Mining

Keivan Ipchi Hagh - 9831073

بخش تئوری

سوال اول

اگر مدل درخت تصمیم را از روی داده ای که تابه حال ندیده است هرس کنیم (داده ای جدا از داده آموزشی اولیه)، کمکی برای جلوگیری بیش برازش مدل نکردیم (مگر به تصادف) و صرفا مدل طوری تغییر دادیم تا روی مجموعه داده جدید نتیجه خوب اما غیرقابل استناد بدهد، زیرا مدل را برای مجموعه داده خاصی تغییر دادیم.

از طرفی هرس کردن روی مجموعه داده جدید که می‌تواند توزیع و مشخصات متفاوتی از داده آموزشی داشته باشد، می‌تواند منجر به از دست رفتن اطلاعات بواسطه هرس اشتباه شود.

سوال دوم

در ابتدا باید آنتروپی ستون تصمیم را محاسبه کنیم (در اینجا، "آیا به مهمانی دعوت میشود؟"). برای اینکار:

$$E(is\_invited) = -\sum_{1}^{5} p(i).\log_2 p(i) = -(\frac{2}{5}\log\frac{2}{5}) - \left(\frac{3}{5}\log\frac{3}{5}\right) = 0.67$$

در قدم بعد باید آنتروپی ستون تصمیم را بعد از دسته بندی کردن با استفاده از هر ویژگی بدست آوریم. در ادامه محاسبات آورده شده است:

$$E(is\_invited, weight) = \frac{3}{5}E(weight_{low}) + \frac{2}{5}E(weight_{high}) = \frac{3}{5}\left(-\frac{1}{3}\log\frac{1}{3} - \frac{2}{3}\log\frac{2}{3}\right) + \frac{2}{5}\left(-\frac{1}{2}\log\frac{1}{2} - \frac{1}{2}\log\frac{1}{2}\right)$$

$$=$$

$$E(is\_invited, color) = \frac{1}{5}E(color_{red}) + \frac{2}{5}E(color_{blue}) + \frac{2}{5}E(color_{green})$$

$$= \frac{1}{5}\left(-\log\frac{1}{1}\right) + \frac{2}{5}\left(-\frac{2}{2}\log\frac{2}{2}\right) + \frac{2}{5}\left(-\frac{1}{2}\log\frac{1}{2} - \frac{1}{2}\log\frac{1}{2}\right) = 0 + 0 + 0.4 = 0.4$$

برای محاسبه آنتروپی ستون تصمیم بر اساس قد که یک ویژگی *continues* بوده، ابتدا اعداد را بر حسب قد مرتب می‌کنیم:

160 – yes

162 – yes

165 – no

170 – no

172 – yes

و سپس در مرز های تغییر ستون تصمیم، آنها را دسته بندی می‌کنیم:

$$E(is\_invited, height) = \frac{2}{5}\left(-\frac{2}{2}\log\frac{2}{2} - 0\right) + \frac{2}{5}\left(-\frac{2}{2}\log\frac{2}{2} - 0\right) + \frac{1}{5}\left(\frac{1}{1}\log\frac{1}{1}\right) = 0$$

حال میزان Gain را با استفاده از فرمول، برای هر ویژگی محاسبه می‌کنیم:

Data Mining

Keivan Ipchi Hagh - 9831073

$$Gain(weight) = Info(is_{invited}) - Info(is_{invited}, weight) = 0.67 - 0.54 = 0.13$$

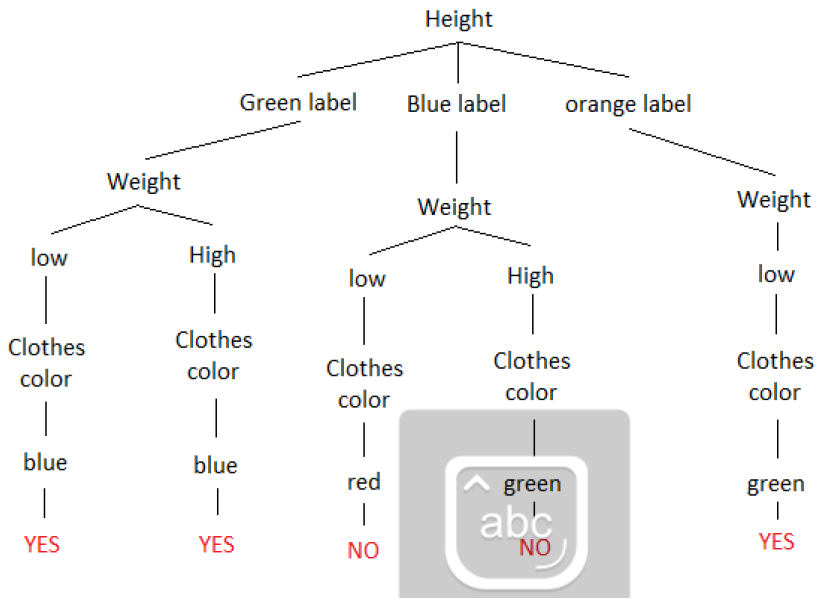
$$Gain(color) = Info(is_{invited}) - Info(is_{invited}, color) = 0.67 - 0.4 = 0.27$$

$$Gain(height) = Info(is_{invited}) - Info(is_{invited}, height) = 0.67 - 0 = 0.67$$

طبق محاسبات بالا، بهترین ویژگی قد است. پس مجموعه داده را بر حسب این ویژگی تقسیم بندی می‌کنیم. از طرفی این ویژگی اولین گره درخت ما است.

آیا به مهمانی دعوت می‌شود؟	وزن	قد	رنگ لباس
بله	چاق	۱۶۲	آبی
بله	لاغر	۱۶۰	آبی
خیر	لاغر	۱۷۰	قرمز
خیر	چاق	۱۶۵	سبز
بله	لاغر	۱۷۲	سبز

از این به بعد، نیازی به محاسبات نیست و می‌توان شهودی درخت را تشکیل داد، زیرا ویژگی بهتر آن است که دسته های مشابه بیشتری بسازد. پس برای دسته سبز و آبی تفاوتی نمیکند زیرا هر دو ویژگی خروجی یکسان خواهند داشت. برای دسته نارنجی نیز یک رکورد بیشتر نداریم. درخت نهایی:



سوال سوم

$$P(pass = \text{"yes"}) = \frac{4}{6} = 0.6$$

$$P(pass = \text{"no"}) = \frac{2}{6} = 0.3$$

Data Mining

Keivan Ipchi Hagh - 9831073

$$P(gpa = "great" \mid pass = "yes") = \frac{2}{4} = 0.5$$

$$P(gpa = "great" \mid pass = "no") = \frac{0+1}{4+1} = 0.25$$

$$P(practice = "yes" \mid pass = "yes") = \frac{3}{4} = 0.75$$

$$P(practice = "yes" \mid pass = "no") = \frac{0+1}{2+1} = 0.3$$

$$P(present = "no" \mid pass = "yes") = \frac{1}{4} = 0.25$$

$$P(present = "no" \mid pass = "no") = \frac{2}{2} = 1$$

$$X = (gpa = \text{"great"} \ \& \ practice = "yes" \ \& \ present = "no")$$

$$P(X|pass = "yes") = 0.5 * 0.75 * 0.25 = 0.09375$$

$$P(X|pass = "no") = 0.25 * 0.3 * 1 = 0.075$$

$$P(X|pass = "yes") * P(pass = "yes") = 0.09375 * 0.6 = 0.05625$$

$$P(X|pass = "no") * P(pass = "no") = 0.075 * 0.3 = 0.0225$$

پس طبق محاسبات بالا، قبول خواهد شد!

سوال چهارم

معیار صحت برای مجموعه داده unbalanced خطا دارد. برای مثال زمانی که ۱۰۰ رکورد داشته باشیم و ۹۰ رکورد برای دسته اول و ۱۰ رکورد باقی مانده برای دسته دوم باشند، صحت برابر 90 درصد خواهد شد. این دقت درست نیست چون لزوما 90 درصد داده برای دسته اول هستند و مدل به ناچار 90 درصد مواقع این دسته را انتخاب می‌کند، پس معیار صحت در واقعیت برای مجموعه داده تست، عددی بسیار پایین تر خواهد داشت.

سوال پنجم

بهترین مدل، میانگین هر ۱۰ مدل آموزش داده شده است، به طوری که داده تست را به هر ۱۰ مدل آموزش داده شده بدهیم و خروجی را بر حسب نوع مسئله میانگین بگیریم و یا max voting انجام دهیم. از طرفی برای محاسبه خطا نهایی نیز می‌توان میانگین گرفت و یا max voting انجام داد. یکی از روش های معروف استفاده از bootstrap است به گونه ای که طبق فرمول زیر خطای نهایی محاسبه می‌شود:

$$Acc(M) = \frac{1}{k}\sum_{i=0}^{k}(0.632 * Acc(M_i)_{test} + 0.358 * Acc(M_i)_{train})$$

Data Mining

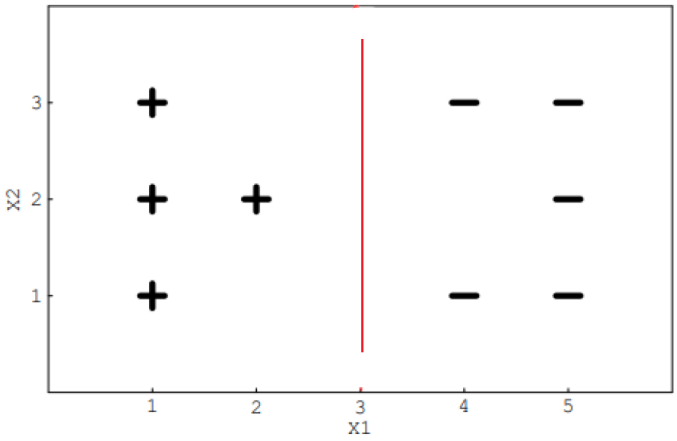
Keivan Ipchi Hagh - 9831073

سوال ششم

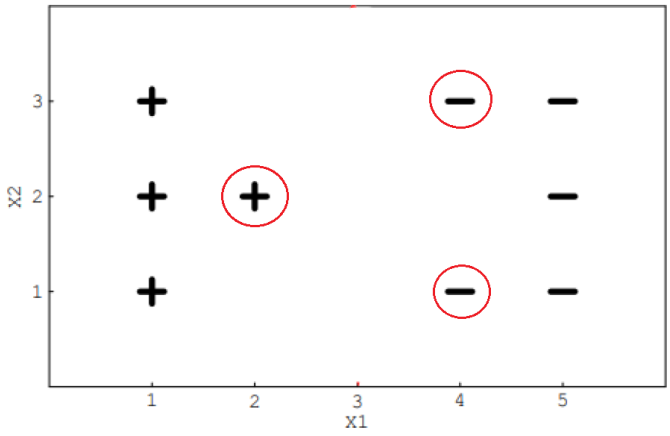
- بله. در این صورت یعنی مدل همه داده های آموزشی را به درستی پیشبینی کرده و نیازی به آموزش بیشتر ندارد. کماکان آموزش بیشتر می‌تواند منجر به overfit شده و مدل ضرر بزند.
- خیر. مدل باید قادر باشد بر روی داده از پیش دیده نشده توانایی پیشبینی داشته باشد و داده وزن دار آموزشی معیار مناسبی برای توقف آموزش نیست. ادامه فرایند آموزش ممکن است به شناسایی الگوهای اضافی در داده ها کمک کند که باعث بهبود توانایی مدل برای تعمیم به داده های جدید می‌شود. علاوه بر این، رسیدن به خطای 0 بر روی داده های وزن دار ممکن است نشانه ای از بیش برازش باشد، بنابراین مهم است که آموزش ادامه پیدا کند تا اطمینان حاصل شود که مدل به داده های آموزشی بیش برازش نشده است.

سوال هفتم

الف) خط کشیده شده توسط SVM باید به گونه ای باشد که میزان خطا کمترین مقدار ممکن باشد یا به عبارتی بیشترین فاصله از تمامی نقاط داشته باشد. داشتن بیشترین فاصله لزوما به معنای دسته بندی با دقت 100 درصد نیست، هرچند در مثال این سوال، تمامی نقاط درست دسته بندی شده اند.



ب) از آنجایی که خط SVM به دلیل C بسیار بزرگ تمایل دارد در وسط قرار گیرد، پس با تغییر سه نقطه پایین لزوما خط جا به جا می‌شود:



Data Mining

Keivan Ipchi Hagh - 9831073

سوال هشتم

الف) درست. الگوریتم بیز ساده فرض می‌کند تمام ویژگی ها یکتا و مستقل از یکدیگر هستند، یا به عبارتی متغیر های ورودی وابستگی به یکدیگر ندارند. پس قادر به تشخیص و تشکیل وابستگی ها نیست.

ب) نادرست. هدف درخت تصمیم جداسازی داده نویز از داده مفید است به گونه ای که با دسته بندی کردن سعی بر تبدیل داده نویز به داده مفید تر دارد. بنابراین، احتمال پوشش دادن نویز توسط درخت تصمیم بیشتر نمی‌شود.

ج) نادرست. هرچه k کمتر باشد نسبت به نویز حرکت بیشتری داریم زیرا برای مثال در k=1 فقط یک نقطه را معیار قرار داده و اگر نویز باشد باعث حرکت centroid ها می‌شود. هرچه k بیشتر شود نویز تاثیر کمتری روی حرکت centroid ها می‌گذارد.

سوال نهم

معیار recall بهترین گزینه است، زیرا هدف تشخیص خستگی راننده حتی به غلط است. زیرا هدف ما جلوگیری از هر حادثه ای است. تشخیص غلط خستگی بهتر از تشخیص ندادن خستگی درست است (از نظر خسارت به بار آمده و هزینه نهایی).

سوال دهم

این متریک gini index نام دارد که میزان تنوع در توضیح ویژگی را برسی می‌کند.

تفاوت اصلی بین شاخص آنتروپی و جینی در روش محاسبه آنها است. شاخص آنتروپی مبتنی بر لگاریتم است، ولی شاخص جینی مبتنی بر مربعات احتمالات است. همچنین، شاخص جینی احتمال اشتباه طبقه‌بندی را بیشتری از آنتروپی در نظر می‌گیرد (به دلیل مربع کردن)، بنابراین وقتی که شاخص جینی برای یک گره بسیار کم باشد، اعتماد بیشتری به مدل داریم تا نسبت به شاخص آنتروپی.

$$Gini = \sum i = pi(1 - pi) \cong p_i^2$$

گر شاخص جینی بسیار کم باشد، به این معنی است که بیشترین اطلاعات به دست آمده در مورد کلاس‌ها در آن گره موجود است و در نتیجه، گره فشرده شده است. اگر شاخص جینی بسیار بزرگ باشد، عکس این موضوع صادق است.

سوال یازدهم

الف) هر دو مدل، روش های regression بوده اما در تعداد متغیر های وابسته مورد استفاده تفاوت هایی دارند. به گونه ای که در simple linear regression دو متغیر مورد برسی قرار گرفته و یک متغیر وابسته دارد که بر اساس آن متغیر دیگر پیشبینی می‌شود. خروجی این مدل یک خط صاف است. اما در multiple linear regression بیش از یک متغیر وابسته داریم که باعث میشود متغیر نهایی بر اساس چندین متغیر وابسته پیشبینی شود. خروجی این مدل معمولا یک صفحه یا فرا صفحه است.

ب) هر دو روش برای کاهش بیش برازش استفاده میشود به طوری که یک penalty برای تابع هزینه اضافه می‌کنند تا آن را افزایش دهد در نتیجه وزن های بزرگ پنالتی بیشتری گرفته و به مرور کوچک (کم اهمیت تر) می‌شوند تا تعادل بین همه ویژگی ها برقرار شود. در L1 پنالتی به صورت جمع قدر مطلق وزن ها و در L2 به صورت جمع مربعات وزن ها اعمال می‌شوند. از طرفی L1 موجب میشود برخی وزن ها به صفر میل کنند که در feature selection نیز کاربرد دارد زیرا ویژگی های غیر مهم را حذف می‌کند.

Data Mining

Keivan Ipchi Hagh - 9831073

ج)

$$XB = Y, \quad X = \begin{matrix} 1 & 42 \\ 1 & 74 \\ 1 & 48 \\ 1 & 35 \\ 1 & 56 \\ 1 & 26 \\ 1 & 60 \end{matrix}, \quad Y = \begin{matrix} 98 \\ 130 \\ 120 \\ 88 \\ 182 \\ 80 \\ 135 \end{matrix}$$

$$X^T XB = X^T Y \quad => \quad B = (X^T X)^{-1} X^T Y$$

$$(X^T X) = \begin{matrix} 7 & 341 \\ 341 & 18181 \end{matrix}, \quad X^T Y = \begin{matrix} 833 \\ 42948 \end{matrix}$$

$$B = \begin{matrix} 45.46 \\ 1.50 \end{matrix}, Y_{pred} = 105.8$$

بخش عملی

دسته بندی دادهها با استفاده از یادگیری عمیق

ماتریس درهم ریختگی و دقت تمام iteration ها به صورت زیر است:

Data Mining

Keivan Ipchi Hagh - 9831073

```
Epoch 1/20
276/276 [==============================] - 1s 2ms/step - loss: 1.1823 - accuracy: 0.4783 - val_loss: 1.0761 - val_accuracy: 0.5290 - lr: 0.0010
Epoch 2/20
276/276 [==============================] - 0s 1ms/step - loss: 1.0227 - accuracy: 0.5865 - val_loss: 1.0297 - val_accuracy: 0.5580 - lr: 0.0010
Epoch 3/20
276/276 [==============================] - 0s 1ms/step - loss: 0.9889 - accuracy: 0.5969 - val_loss: 0.9987 - val_accuracy: 0.5996 - lr: 0.0010
Epoch 4/20
276/276 [==============================] - 0s 1ms/step - loss: 0.9702 - accuracy: 0.6014 - val_loss: 0.9801 - val_accuracy: 0.5815 - lr: 0.0010
Epoch 5/20
276/276 [==============================] - 0s 1ms/step - loss: 0.9551 - accuracy: 0.5996 - val_loss: 0.9663 - val_accuracy: 0.5851 - lr: 0.0010
Epoch 6/20
276/276 [==============================] - 0s 1ms/step - loss: 0.9423 - accuracy: 0.6055 - val_loss: 0.9660 - val_accuracy: 0.5906 - lr: 0.0010
Epoch 7/20
276/276 [==============================] - 0s 1ms/step - loss: 0.9345 - accuracy: 0.6173 - val_loss: 0.9445 - val_accuracy: 0.6141 - lr: 0.0010
Epoch 8/20
276/276 [==============================] - 0s 1ms/step - loss: 0.9265 - accuracy: 0.6119 - val_loss: 0.9372 - val_accuracy: 0.6159 - lr: 0.0010
Epoch 9/20
276/276 [==============================] - 0s 1ms/step - loss: 0.9206 - accuracy: 0.6159 - val_loss: 0.9262 - val_accuracy: 0.6069 - lr: 0.0010
Epoch 10/20
276/276 [==============================] - 0s 1ms/step - loss: 0.9134 - accuracy: 0.6182 - val_loss: 0.9241 - val_accuracy: 0.6159 - lr: 0.0010
Epoch 11/20
276/276 [==============================] - 0s 1ms/step - loss: 0.9088 - accuracy: 0.6209 - val_loss: 0.9231 - val_accuracy: 0.6159 - lr: 0.0010
Epoch 12/20
276/276 [==============================] - 0s 1ms/step - loss: 0.9028 - accuracy: 0.6223 - val_loss: 0.9214 - val_accuracy: 0.6159 - lr: 0.0010
Epoch 13/20
276/276 [==============================] - 0s 1ms/step - loss: 0.9029 - accuracy: 0.6245 - val_loss: 0.9191 - val_accuracy: 0.6123 - lr: 0.0010
Epoch 14/20
276/276 [==============================] - 0s 1ms/step - loss: 0.8980 - accuracy: 0.6200 - val_loss: 0.9150 - val_accuracy: 0.6159 - lr: 0.0010
Epoch 15/20
276/276 [==============================] - 0s 1ms/step - loss: 0.8922 - accuracy: 0.6259 - val_loss: 0.9148 - val_accuracy: 0.6105 - lr: 0.0010
Epoch 16/20
276/276 [==============================] - 0s 1ms/step - loss: 0.8905 - accuracy: 0.6300 - val_loss: 0.9095 - val_accuracy: 0.6069 - lr: 0.0010
Epoch 17/20
276/276 [==============================] - 0s 1ms/step - loss: 0.8849 - accuracy: 0.6223 - val_loss: 0.9133 - val_accuracy: 0.6268 - lr: 0.0010
Epoch 18/20
276/276 [==============================] - 0s 1ms/step - loss: 0.8818 - accuracy: 0.6350 - val_loss: 0.9081 - val_accuracy: 0.6178 - lr: 0.0010
Epoch 19/20
276/276 [==============================] - 0s 1ms/step - loss: 0.8786 - accuracy: 0.6264 - val_loss: 0.9097 - val_accuracy: 0.6069 - lr: 0.0010
Epoch 20/20
276/276 [==============================] - 0s 1ms/step - loss: 0.8769 - accuracy: 0.6336 - val_loss: 0.9104 - val_accuracy: 0.6268 - lr: 0.0010
```

```
confusion_matrix(y_test.to_numpy().argmax(axis=1), y_pred.argmax(axis=1))
```

```
[48]  ✓ 0.0s
```

```
... array([[112,   6,  35,  18],
           [ 48,  67,   5,  44],
           [ 36,   2, 147,   0],
           [ 33,  13,  12, 112]], dtype=int64)
```

همانطور که در کد قابل مشاهده است، از optimizer به نام adam استفاده شده که انتخاب مناسبی برای مدل های multi-label classification است و نیز تابع فعال سازی leaky_relu بهتر از relu عادی عمل می‌کند. با اضافه کردن batch_size مدل را متعادل تر نسبت به bias ها کرده و با استفاده از callback ها ضریب آموزش را به مرور کاهش دادیم تا همگرا شود.