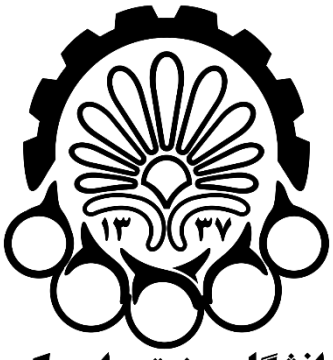
به نام خدا



دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

تمرین درس یادگیری ماشین-سری سوم

فردین آیار

شماره دانشجویی: ۹۹۱۳۱۰۴۰

استاد: دکتر ناظرفرد

دانشکده کامپیوتر- زمستان ۹۹

سوالات تشریحی

۱) در مسائل دسته بندی با بیز ساده، در برخی موارد ممکن است به دلیل عدم وجود داده های کافی، یکی از احتمالات likelihood صفر باشد. به بیان بهتر، $p(x_i|w)$ به ازای یکی از مقادیر محتمل x_i صفر است. در چنین شرایطی، از آنجا که احتمالات posterior، از ضرب $p(x_i|w)$ به ازای i های مختلف به دست می‌آید؛ کل احتمال posterior صفر می‌شود. در این حالت تاثیر سایر x_i ها صرفنظر از مقدارشان از بین می‌رود. برای غلبه بر این مشکل رویکرد smoothing پیشنهاد می‌شود. یکی از روش های این رویکرد Laplace Smoothing نام دارد که احتمالات Likelihood را به صورت زیر تغییر می‌دهد:

$$P(X_i = x_{ik}|Y = y_j) = \frac{count(X_i = x_{ik}, Y = y_j) + l}{count(Y = y_j) + lk}$$

در رابطه بالا K تعداد مقادیر مختلف X است.

۲) مهم‌ترین تفاوت این دو الگوریتم در مکانیزم یادگیری آن‌ها می‌باشد. الگوریتم بیز ساده احتمال $p(y|x)$ را براساس احتمال توام x وy حساب می‌کند؛ درحالی که الگوریتم رگرسیون لاجستیک احتمال $p(y|x)$ را مستقیماً یاد می‌گیرد. از این نظر الگوریتم بیز ساده یک الگوریتم Generative و رگرسیون لاجستیک یک الگوریتم Discriminative است.

الگوریتم بیز ساده ویژگی‌ها را مستقل فرض می کند بنابراین اگر این فرض برقرار نباشد، الگوریتم نتایجی ضعیفی خواهدداشت. در طرف مقابل الگوریتم رگرسیون لاجستیک، چنین فرضی ندارد بنابراین در صورت وابسته بودن ویژگی ها نیز به خوبی عمل خواهد کرد.

ازآن‌جا که بیز ساده احتمالات توام را حساب می‌کند و سعی می‌کند توزیع داده ها را تخمین بزند، در شرایطی که تعداد داده‌های آموزش کم باشند بهتر عمل خواهدکرد. اما رگرسیون لاجستیک چون مستقیما از داده ها یاد می‌گیرد ممکن است در این شرایط دچار بیش برازش شود. همچنین استفاده بیز ساده از احتمالات اولیه(prior) نیز باعث می‌شود دقت آن در شرایط کم بودن داده‌های آموزش بهبود یابد. به صورت کمی Ng & Jordan (2002) اثبات کردند که در صورت میل تعداد داده‌های آموزش به سمت بی‌نهایت، رگرسیون لاجستیک بهتر عمل خواهد کرد. اگرچه الگوریتم بیزساده‌گاووسی، که در آن مقاله بررسی می‌شود، با تعداد داده های بسیار کمتری به نتایج حدی خود می‌رسد.

۳) داده تست را t می نامیم. در روش GNB ابتدا توزیع نرمال داده های تست برای هر دو ویژگی x_1 و x_2 و به ازای هردو کلاس تعیین می‌شود. سپس با توجه به توزیع های بدست آمده احتمالات $p(x_i|w_j)$ ، را برای داده تست تعیین می کنیم. نکته خاصی که در این دیتاست وجود دارد این است که واریانس داده های کلاس A در جهت x_1 حدوداً صفر می باشد. و نقطه t در این راستا از میانگین داده ها فاصله زیادی دارد؛ بنابراین $p(x_1=t_1|A)$ مقدار کمی خواهد بود. در مورد کلاس B اگرچه فاصله نقطه t در این راستا از میانگین داده ها زیاد است؛ اما واریانس نقطه در این راستا بسیار زیاد است بنابراین به

وضوح $p(x_1=t_1|A)< p(x_1=t_1|B)$. در راستای x_2 نیز نقطه t در نزدیکی میانگین هر دو کلاس قرار دارد اما چون واریانس B در این راستا کمتر است داریم $p(x_2=t_2|A)< p(x_2=t_2|B)$. در انتها چون تعداد داده های کلاس B بیشتر است، احتمال پیشین آن نیز بیشتر خواهد بود. به طور خلاصه:

$$\begin{cases} \mathrm{p}(\mathrm{x}_1 = \mathrm{t}_1|\mathrm{A}) < \mathrm{p}(\mathrm{x}_1 = \mathrm{t}_1|\mathrm{B}) \\ \mathrm{p}(\mathrm{x}_2 = \mathrm{t}_2|\mathrm{A}) < \mathrm{p}(\mathrm{x}_2 = \mathrm{t}_2|\mathrm{B}) \Rightarrow \mathrm{p}(\mathrm{A}|\mathrm{t}) < \mathrm{p}(\mathrm{B}|\mathrm{t}) \\ p(A) < p(B) \end{cases}$$

بنابراین GNB برچسب داده تست را B تعیین می‌کند.

۴) با توجه به شکل:

$$\begin{cases} P(C=T) = \sum_A P(C=T|A)P(A) = 0.3 \times 0.2 + 0.25 \times 0.8 = 0.26 \\ P(C=F) = 1 - P(C=T) = 0.74 \\ P(B=T) = \sum_A P(B=T|A)P(A) = 0.37 \times 0.2 + 0.21 \times 0.8 = 0.242 \\ P(B=F) = 1 - P(B=T) = 0.758 \end{cases}$$

$$P(B=T|D=T) = \frac{P(D=T, B=T)}{P(D=T)}$$

$$P(D=T) = \sum_{C,B} P(D=T|\mathrm{C},\mathrm{B})\, P(C,B) = \sum_{C,B} P(D=T|\mathrm{C},\mathrm{B})\, P(C)P(B)$$
$$= 0.5 \times 0.26 \times 0.242 + 0.15 \times 0.242 \times 0.74 + 0.67 \times 0.758 \times 0.26 + 0.95 \times 0.74 \times 0.758 = 0.723$$

$$P(D=T, B=T) = \sum_C P(D=T, B=T, C)$$
$$= \sum_C P(D=T|B=T, C)\, P(B=T)P(C)$$
$$= 0.5 \times 0.242 \times 0.26 + 0.15 \times 0.242 \times 0.74 = 0.058$$

$$\boldsymbol{P(B=T|D=T) = \frac{0.058}{0.723} = 0.08}$$

$$\boldsymbol{P(B=F|D=T) = 1 - 0.08 = 0.92}$$

۵) مقدار cut-off در الگوریتم رگرسیون لاجستیک(و سایر الگوریتم های دسته بندی) به اهمیت خطای کلاس های T و F بستگی دارد. به بیان بهتر اگر هزینه خطای fn بیشتر باشد(تشخیص کلاس T برای ما مهم تر است) باید cut-off به نحوی انتخاب شود که tpr افزایش یابد. اگرچه این مسئله باعث افزایش fp خواهد شد. به عنوان مثال وقتی هدف تشخیص بیماری کرونا باشد، تخصیص برچسب منفی به موارد مثبت(fn) بسیار بحرانی تر از تخصیص برچسب مثبت به موارد منفی(fp) خواهد بود. عکس توضیحات فوق برای وقتی خطای fp بحرانی باشد برقرار است.

۶) نسبت بخت ارتباط یک نتیجه را با یک عامل نشان می‌دهد. نسبت بخت، شانس رخداد نتیجه A را در حضور عاملX، با شانس رخداد نتیجه A بدون حضور عامل X، مقایسه می‌کند. به صورت رسمی این نسب به صورت زیر تعریف می شود.

$$OR = \frac{\frac{a}{c}}{\frac{b}{d}}$$

a= تعداد مواردی که در آنها عامل X وجود داشته و نتیجه A رخ داده است.P(A=1|X=1)

b= تعداد مواردی که در آنها عامل X وجود داشته و نتیجه A رخ نداده است.P(A=0|X=1)

c= تعداد مواردی که در آنها عامل X وجود نداشته و نتیجه A رخ داده است.P(A=1|X=0)

d= تعداد مواردی که در آنها عامل X وجود نداشته و نتیجه A رخ نداده است.P(A=0|X=0)

اگر این نسبت برابر با یک باشد، نشان می‌دهد عامل X تاثیری روی رخداد نتیجه A نداشته است. درصورتی که نسبت بزرگتر از یک باشد، یعنی عامل X روی رخداد نتیجه A تاثیر مثبت داشته است. و درنهایت اگر این نسبت کمتر از یک باشد، یعنی عامل X روی رخداد نتیجه A تاثیر منفی داشته است.(باعث شده نتیجه A کمتر رخ دهد)

براساس تعاریف فوق، برای بدست آوردن نسبت بخت برای عامل X در بردار ویژگی Z و رخداد A به وسیله رگرسیون لاجستیک داریم:

$$odds\ ratio_x = \frac{\frac{P(A=1|Z_x=1)}{1-P(A=1|Z_x=1)}}{\frac{P(A=1|Z_x=0)}{1-P(A=1|Z_x=0)}} = \frac{e^{a_0+a_1z_1+\cdots+a_x\times 1+\cdots+a_nz_n}}{e^{a_0+a_1z_1+\cdots+a_x\times 0+\cdots+a_nz_n}} = e^{a_x}$$

بنابراین نسبت بخت X برابر است با e^{a_x} که a_x ضریب X در تابع رگرسیون می‌باشد.

۷)

$$P(Buy=+|X_1) \sim P(Buy=+)\prod P(X_{1i}|Buy=+) = \frac{9}{14}\times\frac{2}{9}\times\frac{2}{9}\times\frac{6}{9}\times\frac{6}{9}$$
$$= 0.014$$

$$P(Buy=-|X_1) \sim P(Buy=-)\prod P(X_{1i}|Buy=-) = \frac{5}{14}\times\frac{3}{5}\times\frac{2}{5}\times\frac{1}{5}\times\frac{2}{5}$$
$$= 0.006$$

بنابراین برچسب اولین داده مثبت می‌باشد.

$$P(Buy=+|X_2) \sim P(Buy=+)\prod P(X_{2i}|Buy=+) = \frac{9}{14}\times\frac{3}{9}\times\frac{3}{9}\times\frac{3}{9}\times\frac{3}{9}$$
$$= 0.007$$

$$P(Buy=-|X_2) \sim P(Buy=-)\prod P(X_{2i}|Buy=-) = \frac{5}{14}\times\frac{2}{5}\times\frac{1}{5}\times\frac{4}{5}\times\frac{3}{5}$$
$$= 0.013$$

بنابراین برچسب دومین داده منفی می‌باشد.

$$P(Buy=+|X_3) \sim P(Buy=+)\prod P(X_{3i}|Buy=+) = \frac{9}{14}\times\frac{4+1}{9+3}\times\frac{4+1}{9+3}$$
$$\times\frac{3+1}{9+2}\times\frac{6+1}{9+2} = 0.025$$

$$P(Buy=-|X_3) \sim P(Buy=-)\prod P(X_{3i}|Buy=-) = \frac{5}{14}\times\frac{0+1}{5+3}\times\frac{2+1}{5+3}$$
$$\times\frac{4+1}{5+2}\times\frac{2+1}{5+2} = 0.005$$

بنابراین برچسب سومین داده مثبت می‌باشد.

سوالات پیاده‌سازی

۱) کد مربوط به این سوال در فایل Q1.py قرار دارد. از آنجا که به نظر می‌رسد داده های موجود در دیتاست دارای نظم خاصی می‌باشند؛ به وسیله کد shafling.py ، شافل شده و در فایل s_car.csv ذخیره شده‌اند. برای اجرای برنامه فایل s_car.csv استفاده شده است.

الف) ماتریس درهم‌ریختگی برای همه کلاس‌ها همزمان رسم شده، اما سایر پارامترهای خواسته‌شده به روش one vs all برای هر کلاس محاسبه شده اند. لازم به ذکر است برای رسم ماتریس درهم‌ریختگی، ردیف‌ها برچسب واقعی، و ستون‌ها برچسب پیش‌بینی‌شده است.

نتایج داده‌های آموزش:

```
metrics fo train
         acc  unacc  vgood  good
acc    213.0   57.0    0.0   8.0
unacc   36.0  806.0    0.0   2.0
vgood   17.0    0.0   22.0   0.0
good    31.0    0.0    0.0  17.0
       sensitivity  specificity    fn    fp
acc       0.766187     0.909774  65.0  84.0
unacc     0.954976     0.843836  38.0  57.0
vgood     0.564103     1.000000  17.0   0.0
good      0.354167     0.991387  31.0  10.0
```

نتایج داده‌های آزمون:

```
         acc  unacc  vgood  good
acc    74.0   30.0    0.0   2.0
unacc  18.0  348.0    0.0   0.0
vgood  14.0    0.0   10.0   2.0
good   15.0    0.0    1.0   5.0
metrics for test
       sensitivity  specificity    fn    fp
acc       0.698113     0.886199  32.0  47.0
unacc     0.950820     0.803922  18.0  30.0
vgood     0.384615     0.997972  16.0   1.0
good      0.238095     0.991968  16.0   4.0
```

مطابق نتایج، sensitivity الگوریتم برای دو کلاس vgood و good نسبتا پایین است. دلیل این امر می‌تواند نبود داده کافی برای این دو کلاس باشد.(فراوانی این دو کلاس در دیتاست کمتر است)

ب) نتایج برای l=1 ارائه می‌شود. طبق نظر استاد smooting روی همه ویژگی‌ها اعمال می‌شود.

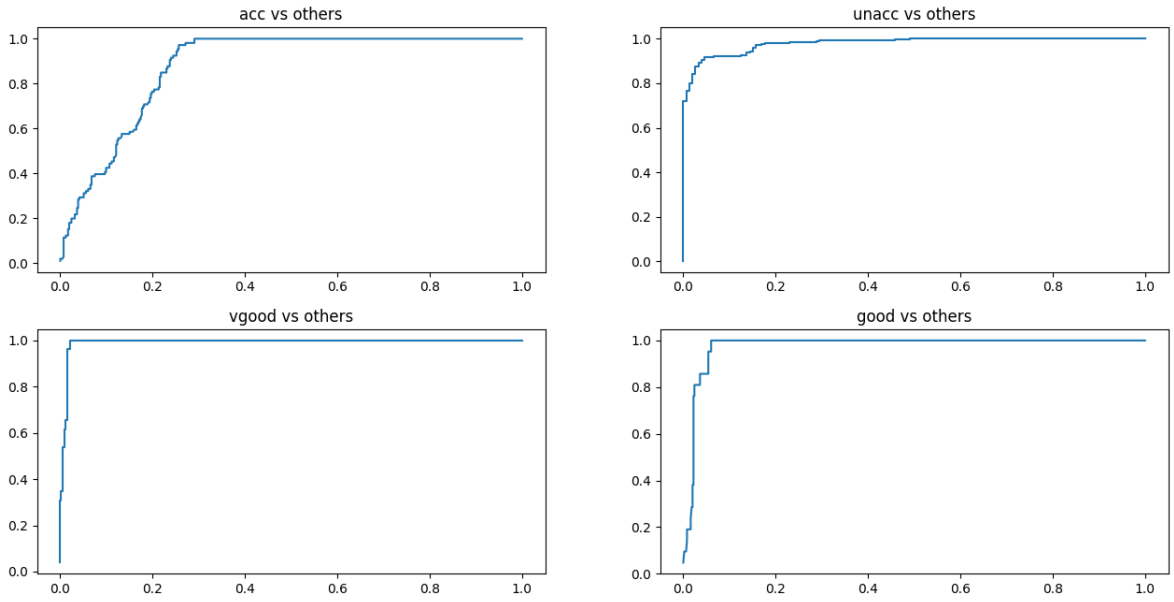
نتایج داده‌های آموزش:

```
metrics fo train
         acc  unacc  vgood  good
acc    213.0   57.0    0.0   8.0
unacc   36.0  806.0    0.0   2.0
vgood   21.0    0.0   18.0   0.0
good    32.0    0.0    0.0  16.0
       sensitivity  specificity    fn    fp
acc       0.766187     0.904404  65.0  89.0
unacc     0.954976     0.843836  38.0  57.0
vgood     0.461538     1.000000  21.0   0.0
good      0.333333     0.991387  32.0  10.0
```

نتایج داده‌های آزمون:

```
         acc  unacc  vgood  good
acc     74.0   30.0    0.0   2.0
unacc   18.0  348.0    0.0   0.0
vgood   16.0    0.0    8.0   2.0
good    17.0    0.0    0.0   4.0
metrics for test
       sensitivity  specificity    fn    fp
acc       0.698113     0.876513  32.0  51.0
unacc     0.950820     0.803922  18.0  30.0
vgood     0.307692     1.000000  18.0   0.0
good      0.190476     0.991968  17.0   4.0
```

ج)نمودار roc به روش one vs all برای همه کلاس ها رسم می‌شود:



مطابق نمودار های فوق، دسته بندی کننده عملکرد مطلوبی داشته است. نکته بسیار مهم در مورد دو کلاس good و vgood این است که علارغم کم بودن sensitivity برای این دو کلاس، نمودار آن‌ها بسیار بهتر از دو کلاس دیگر است. دلیل این امر کم بودن داده های مربوط به این دو کلاس است. به عبارت دیگر دیتاست برای این دوکلاس نامتقارن است و می‌دانیم roc در موارد این‌چنینی، عملکرد را بهتر از واقعیت نشان می‌دهد. رسم نمودار precision recall در چنین مواقعی ترجیح دارد.

۲) کد مربوط به این سوال در فایل Q2.py قراردارد. دیتاست مدنظر سوال به صورت csv از این لینک دانلود شده و نام ستون ها به صورت دستی به آنها اضافه شد. فایل دیتاست نهایی که در پروژه استفاده شده در پوشه dataset موجود است.

الف) پیاده سازی مطابق خواسته سوال به صورت one_vs_all است. در هر مرحله کلاس مورد نظر به عنوان کلاس ۱ و سایر کلاس‌ها به عنوان کلاس۰ در نظر گرفته می شود.یعنی در مجموع ۱۰ دسته بندی کننده رگرسیون لاجستیک خواهیم داشت.هر ۱۰ تابع را روی تمام داده ها اعمال می‌کنیم و درنهایت دسته‌بندی‌کننده‌ای که بیشترین احتمال را دارد به عنوان برچسب داده درنظر گرفته می‌شود.مشابه سوال قبلی، در ماتریس درهم‌ریختگی، ردیف ها برچسب واقعی و ستون ها برچسب پیش‌بینی شده است. نتایج به شرح زیر است:

نتایج داده های آموزش:

	0	1	2	3	4	5	6	7	8	9
0	5742.0	0.0	17.0	11.0	10.0	34.0	30.0	7.0	60.0	12.0
1	34.0	4969.0	304.0	73.0	251.0	52.0	104.0	162.0	709.0	84.0
2	71.0	2.0	5375.0	76.0	66.0	16.0	100.0	57.0	172.0	23.0
3	58.0	1.0	149.0	5419.0	20.0	149.0	28.0	90.0	136.0	81.0
4	17.0	4.0	42.0	12.0	5392.0	23.0	50.0	15.0	71.0	216.0
5	103.0	0.0	33.0	203.0	64.0	4583.0	104.0	31.0	221.0	79.0
6	68.0	0.0	46.0	3.0	29.0	100.0	5632.0	2.0	35.0	3.0
7	23.0	0.0	55.0	31.0	83.0	11.0	3.0	5788.0	26.0	245.0
8	90.0	2.0	77.0	148.0	88.0	209.0	59.0	48.0	5032.0	98.0
9	57.0	3.0	17.0	100.0	239.0	44.0	2.0	219.0	68.0	5200.0

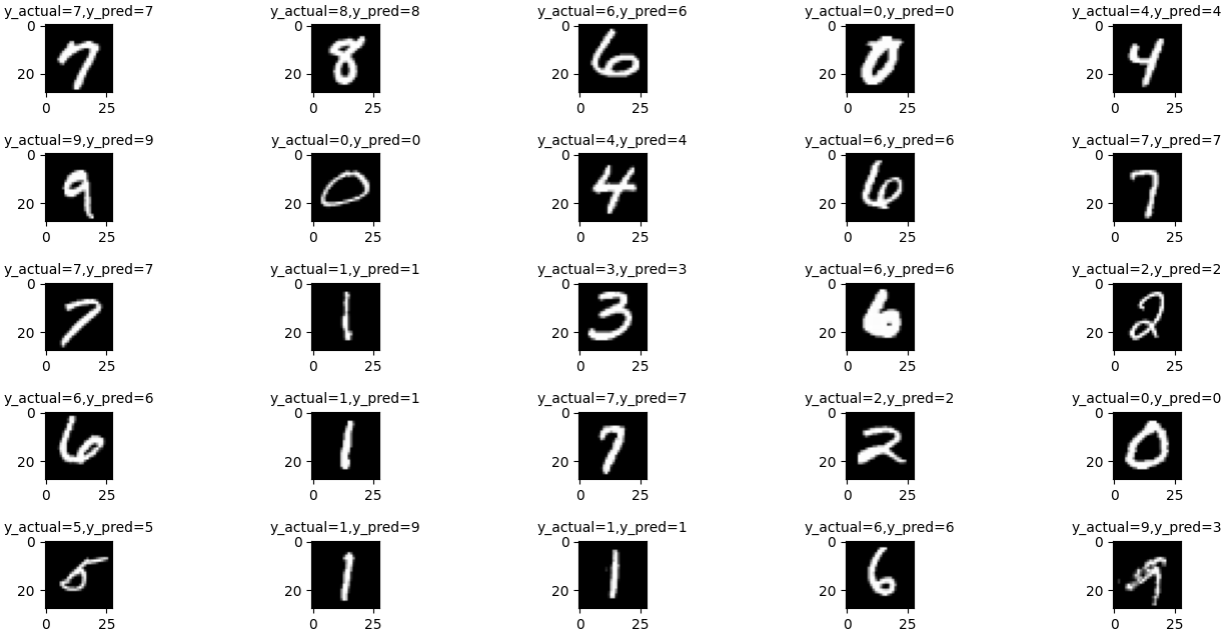
error for train:
0.11446666666666672

نتایج داده‌های آزمون:

	0	1	2	3	4	5	6	7	8	9
0	952.0	0.0	2.0	2.0	0.0	5.0	8.0	2.0	7.0	2.0
1	8.0	864.0	36.0	12.0	47.0	6.0	19.0	20.0	114.0	9.0
2	13.0	0.0	910.0	17.0	11.0	3.0	18.0	11.0	46.0	3.0
3	9.0	0.0	17.0	906.0	3.0	23.0	5.0	14.0	22.0	11.0
4	1.0	0.0	8.0	2.0	910.0	0.0	12.0	2.0	14.0	33.0
5	15.0	1.0	1.0	34.0	12.0	746.0	16.0	14.0	43.0	10.0
6	14.0	1.0	10.0	1.0	5.0	25.0	898.0	1.0	3.0	0.0
7	3.0	0.0	24.0	8.0	12.0	2.0	2.0	931.0	6.0	40.0
8	12.0	1.0	10.0	21.0	20.0	30.0	10.0	20.0	838.0	12.0
9	20.0	2.0	1.0	13.0	47.0	8.0	1.0	38.0	12.0	867.0

error for test:
0.11780000000000002

ب)



ج)

۱) الگوریتم knn، یک الگوریتم غیرپارامتریک و غیرخطی است بنابراین برای داده‌های غیرخطی بهتر عمل می‌کند. از آنجایی که این الگوریتم غیرپارامتریک است با افزایش داده های آموزش، می تواند تعداد پارامترها(همسایه ها) خود را افزایش دهد و در نتیجه می‌تواند بهتر عمل کند.

۲) knn می تواند مستقیماً در مسائل چندکلاسه استفاده شود. در صورتی که LR باید با استفاده از روش هایی مانند one vs all در مسائل چندکلاسه اعمال شود.

۳) LR احتمال تعلق داده بر هر کلاس را مشخص می‌کند اما knn تنها برچسب پیش‌بینی شده را برمی‌گرداند.

۴) الگوریتم knn تنبل است و مرحله آزمون آن بسیار کندتر است. بنابراین در شرایطی که نرخ پرسش از الگوریتم بالا باشد، این الگوریتم به شدت کند عمل می‌کند.

در مجموع وقتی داده ها دارای مرز پیچیده هستند هستند ، تعداد کلاس ها زیاد است ، نرخ و تعداد پرسش از الگوریتم پایین است یا نیازی به استفاده از احتمالات نیست استفاده از knn ترجیح دارد و در غیر اینصورت LR مناسب تر است.