Machine Learning Course

Assignment #3:

Ensemble Algorithms

آرمین خیاطی

۹۹۳۱۱۵۳

پویان انصاری راد

۹۹۶۰۵۶۵

Spring 2021

فهرست

مقدمه

در این تمرین دو متد از الگوریتم های Ensemble پیاده سازی شده اند:

- Bagging
- AdaBoost.M۱

کلاس بند پایه Decision Tree در نظر گرفته شده است که به کمک دو متد فوق سعی در حصول نتیجه بهتری در دسته بندی دیتاست ها شده است. در قسمت Bagging از پارامتر های پیشفرض کتابخانه ScikitLearn برای ایجاد مدل پایه درخت تصمیم استفاده شده ولی در قسمت AdaBoost.M۱ مقدار پارامتر عمق درخت بصورت تجربی انتخاب شده است(طبق موارد درخواست شده در متن تمرین).

ابتدا داده های به نسبت ۷۰٪ داده آموزشی و ۳۰٪ داده تست برای هر دیتاست جدا میشوند و سپس فراخور هر قسمت، Bootstrap Sampling with Replacement از روی داده ها آموزشی ایجاد میگردند. همچنین یکمرتبه نیز عمل آموزش را با داده های آموزش نویزی انجام داده و نتایج را بررسی میکنیم. در متن تمرین درخواست شده بود که برای هر دیتاست ۱۰ اجرای مستقل گرفته شود و میانگین دقت ها گزارش شود که در این پیاده سازی برای هر دیتاست ۱۰ اجرای مستقل در هر قسمت گرفته شده است.

مقدار ایتریشن T در هر دو متد Bagging و AdaBoost.M۱ به ترتیب از مجموعه های {۱۱،۲۱،۳۱،۴۱} و {۲۱،۳۱،۴۱،۵۱} با آزمون و خطا انتخاب شد. شایان ذکر است دیتاست های استفاده شده در این پیاده سازی عبارتند از:

BreastTissue, Diabetes, Glass, Ionosphere, Sonar, Wine

بخش اول : آموزش مدل ها با داده های بدون نویز

در این بخش برای هر دو متد انسمبل Bagging و AdaBoost.M۱ برای آموزش مدل ها از داده های آموزشی بدون نویز استفاده شده است و از روش Bootstrap Sampling With Replacement در آنها استفاده شده است. مقدار بیشینه عمق درخت تصمیم برای آدابوست مطابق زیر انتخاب شده:

datasets = [['BreastTissue',4],['Diabetes',1],['Glass',7],['Ionosphere',1],['Sonar',1],['Wine',1]]

T_bagging = 21

T_boosting = 51

برای هر دیتاست ۱۰ اجرای مستقل گرفته شد که نتایج از قرار زیر هستند:

```
----------------------DataSet : BreastTissue--------------------------
x_train.shape = (74, 9)
y_train.shape = (74,)
m = 74
n = 9
x_test.shape = (32, 9)
y_test.shape = (32,)
m_test = 32
c = 6
y_unique = ['1' '2' '3' '4' '5' '6']

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  4
Runs =  10

Normal Test Accuracy =  0.58
Bagging Test Accuracy =  0.68
Boosting Test Accuracy =  0.69
--------------------------------------------------------------------------

--------------------------DataSet : Diabetes----------------------------
x_train.shape = (537, 8)
y_train.shape = (537,)
m = 537
n = 8
x_test.shape = (231, 8)
y_test.shape = (231,)
m_test = 231
c = 2
y_unique = ['-1' '1']

T(Bagging) =  21
```

```
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  1
Runs =  10

Normal Test Accuracy =  0.75
Bagging Test Accuracy =  0.76
Boosting Test Accuracy =  0.78
------------------------------------------------------------------------

---------------------------DataSet : Glass----------------------------
x_train.shape = (149, 9)
y_train.shape = (149,)
m = 149
n = 9
x_test.shape = (65, 9)
y_test.shape = (65,)
m_test = 65
c = 6
y_unique = ['1' '2' '3' '4' '5' '6']

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  7
Runs =  10

Normal Test Accuracy =  0.52
Bagging Test Accuracy =  0.60
Boosting Test Accuracy =  0.61
------------------------------------------------------------------------

-------------------------DataSet : Ionosphere---------------------------
x_train.shape = (245, 34)
y_train.shape = (245,)
m = 245
n = 34
x_test.shape = (106, 34)
y_test.shape = (106,)
m_test = 106
c = 2
y_unique = ['b' 'g']

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  1
Runs =  10

Normal Test Accuracy =  0.82
Bagging Test Accuracy =  0.93
Boosting Test Accuracy =  0.93
------------------------------------------------------------------------

---------------------------DataSet : Sonar----------------------------
x_train.shape = (145, 60)
y_train.shape = (145,)
m = 145
n = 60
x_test.shape = (63, 60)
```

```
y_test.shape = (63,)
m_test = 63
c = 2
y_unique = ['M' 'R']

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  1
Runs =  10

Normal Test Accuracy =  0.73
Bagging Test Accuracy =  0.77
Boosting Test Accuracy =  0.81
------------------------------------------------------------------------

---------------------------DataSet : Wine-------------------------------
x_train.shape = (124, 13)
y_train.shape = (124,)
m = 124
n = 13
x_test.shape = (54, 13)
y_test.shape = (54,)
m_test = 54
c = 3
y_unique = ['1' '2' '3']

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  1
Runs =  10

Normal Test Accuracy =  0.69
Bagging Test Accuracy =  0.91
Boosting Test Accuracy =  0.93
------------------------------------------------------------------------
```

بخش دوم : آموزش مدل ها با داده های نویزی

در این بخش برای هر دو متد انسمبل Bagging و AdaBoost.M۱ برای آموزش مدل ها از داده های آموزشی نویزی استفاده شده است و از روش Bootstrap Sampling With Replacement در آنها استفاده شده است.

برای اعمال نویز با سه مقدار متفاوت {۱۰و۲۰و۳۰} درصد از فیچر های هر دیتاست بصورت رندوم انتخاب میشوند و سپس از توزیع احتمال نرمال با میانگین ۰ و واریانس ۱ نویز به مقادیر این فیچر های در تمام نمونه های صرفا آموزشی اضافه یا کم میشود و در نهایت مدل با داده تست بدون نویز آزمایش میگردد.

مقدار بیشینه عمق درخت تصمیم برای آدابوست مطابق زیر انتخاب شده:

datasets = [['BreastTissue',4],['Diabetes',1],['Glass',7],['Ionosphere',1],['Sonar',1],['Wine',1]]

T_bagging = 21

T_boosting = 51

برای هر دیتاست ۱۰ اجرای مستقل گرفته شد که نتایج از قرار زیر هستند:

```
-------------------------DataSet : BreastTissue--------------------------
x_train.shape = (74, 9)
y_train.shape = (74,)
m = 74
n = 9
x_test.shape = (32, 9)
y_test.shape = (32,)
m_test = 32
c = 6
y_unique = ['1' '2' '3' '4' '5' '6']

Noise(%) =  10

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  4
Runs =  10

Normal Test Accuracy =  0.58
Bagging Test Accuracy =  0.69
Boosting Test Accuracy =  0.72
```

```
--------------------------------------------------------------------------

-------------------------DataSet : BreastTissue--------------------------
x_train.shape = (74, 9)
y_train.shape = (74,)
m = 74
n = 9
x_test.shape = (32, 9)
y_test.shape = (32,)
m_test = 32
c = 6
y_unique = ['1' '2' '3' '4' '5' '6']

Noise(%) =  20

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  4
Runs =  10

Normal Test Accuracy =  0.58
Bagging Test Accuracy =  0.69
Boosting Test Accuracy =  0.68
--------------------------------------------------------------------------

-------------------------DataSet : BreastTissue--------------------------
x_train.shape = (74, 9)
y_train.shape = (74,)
m = 74
n = 9
x_test.shape = (32, 9)
y_test.shape = (32,)
m_test = 32
c = 6
y_unique = ['1' '2' '3' '4' '5' '6']

Noise(%) =  30

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  4
Runs =  10

Normal Test Accuracy =  0.58
Bagging Test Accuracy =  0.66
Boosting Test Accuracy =  0.68
--------------------------------------------------------------------------
```

```
--------------------------DataSet : Diabetes-----------------------------
x_train.shape = (537, 8)
y_train.shape = (537,)
m = 537
n = 8
x_test.shape = (231, 8)
y_test.shape = (231,)
m_test = 231
c = 2
y_unique = ['-1' '1']

Noise(%) =  10

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  1
Runs =  10

Normal Test Accuracy =  0.69
Bagging Test Accuracy =  0.77
Boosting Test Accuracy =  0.77
------------------------------------------------------------------------------

--------------------------DataSet : Diabetes-----------------------------
x_train.shape = (537, 8)
y_train.shape = (537,)
m = 537
n = 8
x_test.shape = (231, 8)
y_test.shape = (231,)
m_test = 231
c = 2
y_unique = ['-1' '1']

Noise(%) =  20

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  1
Runs =  10

Normal Test Accuracy =  0.73
Bagging Test Accuracy =  0.76
Boosting Test Accuracy =  0.77
------------------------------------------------------------------------------
```

```
--------------------------DataSet : Diabetes-----------------------------
x_train.shape = (537, 8)
y_train.shape = (537,)
m = 537
n = 8
x_test.shape = (231, 8)
y_test.shape = (231,)
m_test = 231
c = 2
y_unique = ['-1' '1']

Noise(%) =  30

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  1
Runs =  10

Normal Test Accuracy =  0.72
Bagging Test Accuracy =  0.75
Boosting Test Accuracy =  0.77
------------------------------------------------------------------------

--------------------------DataSet : Glass--------------------------------
x_train.shape = (149, 9)
y_train.shape = (149,)
m = 149
n = 9
x_test.shape = (65, 9)
y_test.shape = (65,)
m_test = 65
c = 6
y_unique = ['1' '2' '3' '4' '5' '6']

Noise(%) =  10

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  7
Runs =  10

Normal Test Accuracy =  0.52
Bagging Test Accuracy =  0.59
Boosting Test Accuracy =  0.60
------------------------------------------------------------------------
```

```
--------------------------DataSet : Glass---------------------------------
x_train.shape = (149, 9)
y_train.shape = (149,)
m = 149
n = 9
x_test.shape = (65, 9)
y_test.shape = (65,)
m_test = 65
c = 6
y_unique = ['1' '2' '3' '4' '5' '6']

Noise(%) =  20

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  7
Runs =  10

Normal Test Accuracy =  0.53
Bagging Test Accuracy =  0.56
Boosting Test Accuracy =  0.54
-------------------------------------------------------------------------

--------------------------DataSet : Glass---------------------------------
x_train.shape = (149, 9)
y_train.shape = (149,)
m = 149
n = 9
x_test.shape = (65, 9)
y_test.shape = (65,)
m_test = 65
c = 6
y_unique = ['1' '2' '3' '4' '5' '6']

Noise(%) =  30

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  7
Runs =  10

Normal Test Accuracy =  0.51
Bagging Test Accuracy =  0.54
Boosting Test Accuracy =  0.55
-------------------------------------------------------------------------
```

```
--------------------------DataSet : Ionosphere---------------------------
x_train.shape = (245, 34)
y_train.shape = (245,)
m = 245
n = 34
x_test.shape = (106, 34)
y_test.shape = (106,)
m_test = 106
c = 2
y_unique = ['b' 'g']

Noise(%) =  10

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  1
Runs =  10

Normal Test Accuracy =  0.82
Bagging Test Accuracy =  0.93
Boosting Test Accuracy =  0.92
------------------------------------------------------------------------------

--------------------------DataSet : Ionosphere---------------------------
x_train.shape = (245, 34)
y_train.shape = (245,)
m = 245
n = 34
x_test.shape = (106, 34)
y_test.shape = (106,)
m_test = 106
c = 2
y_unique = ['b' 'g']

Noise(%) =  20

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  1
Runs =  10

Normal Test Accuracy =  0.82
Bagging Test Accuracy =  0.92
Boosting Test Accuracy =  0.93
------------------------------------------------------------------------------
```

```
--------------------------DataSet : Ionosphere---------------------------
x_train.shape = (245, 34)
y_train.shape = (245,)
m = 245
n = 34
x_test.shape = (106, 34)
y_test.shape = (106,)
m_test = 106
c = 2
y_unique = ['b' 'g']

Noise(%) =  30

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  1
Runs =  10

Normal Test Accuracy =  0.81
Bagging Test Accuracy =  0.92
Boosting Test Accuracy =  0.91
------------------------------------------------------------------------------

-----------------------------DataSet : Sonar-------------------------------
x_train.shape = (145, 60)
y_train.shape = (145,)
m = 145
n = 60
x_test.shape = (63, 60)
y_test.shape = (63,)
m_test = 63
c = 2
y_unique = ['M' 'R']

Noise(%) =  10

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  1
Runs =  10

Normal Test Accuracy =  0.73
Bagging Test Accuracy =  0.77
Boosting Test Accuracy =  0.79
------------------------------------------------------------------------------
```

```
------------------------------DataSet : Sonar------------------------------
x_train.shape = (145, 60)
y_train.shape = (145,)
m = 145
n = 60
x_test.shape = (63, 60)
y_test.shape = (63,)
m_test = 63
c = 2
y_unique = ['M' 'R']

Noise(%) =  20

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  1
Runs =  10

Normal Test Accuracy =  0.71
Bagging Test Accuracy =  0.78
Boosting Test Accuracy =  0.78
--------------------------------------------------------------------------------

------------------------------DataSet : Sonar------------------------------
x_train.shape = (145, 60)
y_train.shape = (145,)
m = 145
n = 60
x_test.shape = (63, 60)
y_test.shape = (63,)
m_test = 63
c = 2
y_unique = ['M' 'R']

Noise(%) =  30

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  1
Runs =  10

Normal Test Accuracy =  0.71
Bagging Test Accuracy =  0.76
Boosting Test Accuracy =  0.76
--------------------------------------------------------------------------------
```

```
--------------------------DataSet : Wine---------------------------------
x_train.shape = (124, 13)
y_train.shape = (124,)
m = 124
n = 13
x_test.shape = (54, 13)
y_test.shape = (54,)
m_test = 54
c = 3
y_unique = ['1' '2' '3']

Noise(%) =  10

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  1
Runs =  10

Normal Test Accuracy =  0.69
Bagging Test Accuracy =  0.91
Boosting Test Accuracy =  0.94
-------------------------------------------------------------------------

--------------------------DataSet : Wine---------------------------------
x_train.shape = (124, 13)
y_train.shape = (124,)
m = 124
n = 13
x_test.shape = (54, 13)
y_test.shape = (54,)
m_test = 54
c = 3
y_unique = ['1' '2' '3']

Noise(%) =  20

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  1
Runs =  10

Normal Test Accuracy =  0.69
Bagging Test Accuracy =  0.91
Boosting Test Accuracy =  0.93
-------------------------------------------------------------------------
```

```
---------------------------DataSet : Wine----------------------------------
x_train.shape = (124, 13)
y_train.shape = (124,)
m = 124
n = 13
x_test.shape = (54, 13)
y_test.shape = (54,)
m_test = 54
c = 3
y_unique = ['1' '2' '3']

Noise(%) =  30

T(Bagging) =  21
T(Boosting) =  51
tree_max_dept for (Normal, Boosting) =  1
Runs =  10

Normal Test Accuracy =  0.69
Bagging Test Accuracy =  0.91
Boosting Test Accuracy =  0.92
---------------------------------------------------------------------------
```

جمع بندی :

چکیده نتایج را برای دو قسمت داده آموزشی بدون نویز و نویزی میتوان بصورت زیر نمایش داد:

Datasets	Algorithms	
	Bagging	AdaBoost.M1
Wine	۰,۹۱	
Glass	۰,۶۰	
BreastTissue	۰,۶۸	
Diabetes	۰,۷۶	۰,۷۸
Sonar	۰,۷۷	۰,۸۱
Ionosphere	۰,۹۳	۰,۹۳

Datasets	Algorithms	
	Bagging + noise 10%, 20%, 30%	AdaBoost.M1 + noise 10%, 20%, 30%
Wine	۰,۹۱ , ۰,۹۱ , ۰,۹۱	
Glass	۰,۵۹ , ۰,۵۶ , ۰,۵۴	
BreastTissue	۰,۶۹ , ۰,۶۹ , ۰,۶۶	
Diabetes	۰,۷۷ , ۰,۷۶ , ۰,۷۵	۰,۷۷ , ۰,۷۷ , ۰,۷۷
Sonar	۰,۷۷ , ۰,۷۸ , ۰,۷۶	۰,۷۹ , ۰,۷۸ , ۰,۷۶
Ionosphere	۰,۹۳ , ۰,۹۲ , ۰,۹۲	۰,۹۲ , ۰,۹۳ , ۰,۹۱

در ادامه سوالاتی در تمرین مطرح شده بود که با توجه به نتایج حاصل میتوان توضیحاتی را برای آن ها از قرار زیر ارایه داد:

Questions: (Answer these questions in your report)

١- Why should we set max_depth parameter in AdaBoost.M1 so that the base classifiers become a little better than random?

٢- What do we mean by stable, unstable, and weak classifier?

٣- What kind of classifiers should be used in Bagging? How about AdaBoost.M1?

٤- Compare the results in noiseless and noisy settings and say which algorithm's performance degrades with noisy features. And why?

جواب ها :

١- در Boosting ممکن است که مشکل Overfitting اتفاق بیفتد به همین دلیل از کلاسیفایر پایه ای استفاده میکنیم که دقت کلاسبندی آن کمی بیشتر از رندوم باشد تا از Overfitting جلوگیری کنیم.

٢- یک کلاسیفایر Stable نسبت به داده های نویزی و ترتیب ارایه نمونه ها حساسیت ندارد و مقاوم است ولی در کلاسیفایر Unstable مدل به داده های نویزی و ترتیب ارایه نمونه های حساس میباشد. هم چنین یک کلاسیفایر Weak توانایی کمی در دسته بندی داده در مقایسه با سایر کلاسیفایر ها دارد.

٣- برای متد Bagging باید کلاسیفایر Unstable مورد استفاده قرار بگیرد چون این متد با کاهش واریانس و میانگین گیری در آرا، حساسیت به داده های نویزی را کاهش میدهد در نتیجه برای کلاسیفایر هایی که به داده های نویزی و واریانس حساس هستند کاربردی تر است.

برای متد AdaBoost.M١ چون تا حدودی به داده های نویزی حساس است بنابر این بهتر است از کلاسیفایر های Weak استفاده شود.

۴- طبق نتایج مشخص است که دقت در روش Bagging با اضافه شدن نویز تغییر قابل ملاحظه ای نمیکند چون با کاهش واریانس و رای گیری نسبت به نویز مقاوم است.