МАШИННОЕ ОБУЧЕНИЕ
ЛАБОРАТОРНЫЙ ПРАКТИКУМ
(предварительный вариант)

Н.Ю. Золотых, А.Н. Половинкин

11 ноября 2007 г.

1. *Практическое машинное обучение*

1.1. Bootstrap

Используется пакет `bootstrap` Роба Тибширани.

```
bootstrap(x,nboot,theta,..., func=NULL)
```

Аргументы

`x` — вектор, содержащий данные. Чтобы bootstrap более сложные структуры данных (например, двумерные данные), смотрите пример ниже.

`nboot` — число требуемых bootstrap наборов данных.

`theta` — функция, которую надо bootstrap. Принимает `x` в качестве аргумента, также может иметь дополнительные аргументы.

`...` — дополнительные аргументы, передаваемые функции `theta`

`func` — дополнительный аргумент, определяющий желаемую функцию распределения $\hat{\theta}$. Если аргумент `func` определен, возвращается оценка стандартной ошибки для данной величины.

Возвращаемые значения

`thetastar` — `nboot` bootstrap значений `theta`.

`func.thetastar` — функционал `func` bootstrap распределения thetastar, если func определен.

`jack.boot.val` – the jackknife-after-bootstrap значения для `func`, если `func` определен.

`jack.boot.se` — the jackknife-after-bootstrap оценка стандартной ошибки `func`, если `func` определен.

`call` — строка, представляющая собой «разобранный» вызов функции.

```
abcnon(x, tt, epsilon=0.001,
       alpha=c(0.025, 0.05, 0.1, 0.16, 0.84, 0.9, 0.95, 0.975))
```

Аргументы

`x` — данные. Могут быть как вектором, так и матрицей, строки которой являются наблюдениями.

`tt` — функция, определяющая параметр в resamling form `tt(p,x)`, где `p` — вектор отношений и `x` — дополнительный аргумент, определяющий размер шага для метода конечных разностей.

`alpha` — дополнительный аргумент, определяющий желаемые доверительные интервалы.

Возвращаемое значение

`limits` — оцененные confidence points, полученные с помощью ABC и стандартных нормальных методов.

`stats` — список, состоящий из `t0` — наблюдаемое значение `tt`, `sighat` — infinitesimal jackknife оценка стандартной ошибки для `tt`, `bhat` — оцененный bias.

`constants` — список, состоящий из `a` — константа ускорения, `z0` — регулировка bias, `cq` — компонента искривления.

`tt.inf` — аппроксимированные коэффициенты влияния для `tt`.

`pp` — матрица, строки которой являются resampling points в наименьшей степени подходящем семействе. The abc confidence points являются вычисленными функцией `tt` в данных точках.

```
abcpar(y, tt, S, etahat, mu, n=rep(1,length(y)),lambda=0.001,
       alpha=c(0.025, 0.05, 0.1, 0.16))
```

Аргументы

`y` — вектор данных.

`tt` — функция оцениваемого параметра `mu`, определяющая изучаемый параметр.

`S` — оценка максимального правдоподобия матрицы ковариации для `x`.

`etahat` — оценка максимального правдоподобия natural параметра `eta`.

`mu` — функция, дающая оценку `x` в терминах `eta`.

`n` — дополнительный аргумент, содержащий знаменатели для бинома (вектор длины `length(x)`).

`lambda` — дополнительный аргумент, определяющий размер шага для метода конечных разностей.

`alpha` — дополнительный аргумент, определяющий желаемые доверительные уровни.

Возвращаемые значения

`call` — вызов к `abcpar`

`limits` — номинальный доверительный уровень, точка `ABC`, квадратичная точка `ABC`, стандартная нормальная точка.

`stats` — список, содержащий наблюдаемые значения `tt`, оцененную стандартную ошибку и оцененный bias.

`constants` — список, состоящий из `a` — константа ускорения, `z0` — регулировка bias, `cq` — компонента искривления.

`asym.05`

```
bcanon(x, nboot, theta, ...,
       alpha=c(0.025, 0.05, 0.1, 0.16, 0.84, 0.9, 0.95, 0.975))
```

Аргументы

`x` — вектор, содержащий данные.

`nboot` — число bootstrap повторений.

`theta` — функция, определяющая estimator, используемый в построении доверительных точек.

`...` — дополнительные аргументы для `theta`.

`alpha` — дополнительный аргумент, определяющий требуемые доверительные уровни.

Возвращаемые значения

`confpoint` — оцененные `bca` границы доверительного интервала.

`z0` — оцененная bias коррекция.

`acc` — оцененная константа ускорения.

`u` — jackknife значения влияния.

`call` — строка, представляющая собой «разобранный» вызов функции.

```
bootpred(x,y,nboot,theta.fit,theta.predict,err.meas,...)
```

Аргументы

`x` — матрица, содержащая predictor (regressor) значения. Каждая строка соответствует наблюдению.

`y` — вектор ответов.

`nboot` — число bootstrap повторений.

`theta.fit` — функция, которая должна быть подвергнута процедуре cross-validation. Принимает в качестве аргументов `x` и `y`.

`theta.predict` — функция, порождающая предсказываемые значения для `theta.fit`. Аргументами является матрица `x` и fit объект, порожденный `theta.fit`.

`err.meas` — функция, определяющая измерение ошибки для одиночного ответа `y` и предсказания `yhat`.

`...` — дополнительные аргументы, которые должны быть переданы функции `theta.fit`.

Возвращаемые значения

`app.err` — среднее значение ошибки `err.meas` когда функция `theta.fit` применяется к `x` и `y` и затем используется для предсказания `y`.

`optim` — bootstrap оценка of optimism в функции `app.err`. Полезной оценкой ошибки предсказания является `app.err+optim`.

`err.632` — «.632» bootstrap оценка ошибки предсказания.

`call` — the строка, представляющая собой «разобранный» вызов функции.

```
crossval(x, y, theta.fit, theta.predict, ..., ngroup=n)
```

Аргументы

`x` — матрица, содержащая predictor (regressor) значения. Каждая строка соответствует наблюдению.

`y` — вектор ответов.

`nboot` — число bootstrap повторений.

`theta.fit` — функция, которая должна быть подвергнута процедуре cross-validation. Принимает в качестве аргументов `x` и `y`.

`theta.predict` — функция, порождающая предсказываемые значения для `theta.fit`. Аргументами является матрица `x` и fit объект, порожденный `theta.fit`.

`...` — дополнительные аргументы, которые должны быть переданы функции `theta.fit`.

`ngroup` — дополнительный аргумент, определяющий число групп. Значение по умолчанию равно числу прецедентов, что соответствует leave-one out cross-validation.

Возвращаемые значения

`cv.fit` — the cross-validated fit для каждого наблюдения. Числа от `1` до `n` (число прецедентов) делятся на `ngroup` взаимно непересекающихся групп размера `leave.out`. `leave.out` — число наблюдений в каждой группе (равно целой части величины `n/ngroup`). Группы выбираются случайно, если `ngroup < n`. (если `n/leave.out` не является целым числом, последняя группа будет содержать `> leave.out` наблюдений). Затем `theta.fit` применяется с удаленной k-ой группой наблюдений для `k = 1,2,...,ngroup`. Окончательно вычисляется fitted значение для k-ой группы, используя `theta.predict`.

`ngroup` — число групп.

`leave.out` — число наблюдений в каждой группе.

`groups` — список длины `ngroup`, содержащий индексы наблюдений в каждой группе. Возвращается в случае, если `leave.out > 1`.

`call` — строка, представляющая собой «разобранный» вызов функции.

```
boott(x,theta, ..., sdfun=sdfunboot, nbootsd=25, nboott=200,
        VS=FALSE, v.nbootg=100, v.nbootsd=25, v.nboott=200,
        perc=c(.001,.01,.025,.05,.10,.50,.90,.95,.975,.99,.999))
```

Аргументы

`x` — вектор, содержащий данные. Используется непараметрический bootstrap sampling.

`theta` — функция, которую необходимо bootstrap. Принимает `x` в качестве входного аргумента, может также иметь дополнительные параметры.

`...` — дополнительные параметры, передаваемые `theta`.

`sdfun` — дополнительный параметр, представляющий собой имя функции для вычисления стандартной ошибки для `theta`. Данная функция должна задаваться в виде `sdmean <- function(x,nbootsd,theta,...)`, где `nbootsd` — неиспользуемый аргумент. Если `theta` является математическим ожиданием, для примера, `sdmean <- function(x,nbootsd,theta,...) {sqrt(var(x)/length(x))}`. Если `sdfun` не задан, то `boott` использует внутренний bootstrap цикл, чтобы оценить стандартную ошибку для `theta(x)`.

`nbootsd` — число bootstrap samples, используемых, чтобы оценить стандартную ошибку для `theta(x)`.

`nboott` — число bootstrap samples, используемых, чтобы оценить распределение bootstrap T-статистики. 200 — это абсолютный минимум и 1000 или более необходимо для достоверной $\alpha\%$ доверительной точки. Общее число bootstrap samples равно `nboot*nbootsd`.

`VS` — если данный параметр равен `TRUE`, то вычисляется преобразование, стабилизирующее дисперсию; доверительный интервал строится в преобразованном пространстве и затем отображается обратно в исходное пространство. Это может ускорить как статистические свойства интервалов, так и ускорить вычисления. Если параметр равен `FALSE`, то стабилизация дисперсии не выполняется.

`v.nbootg` — число bootstrap samples, используемых для вычисления преобразования `g`, стабилизирующего дисперсию. Используется, если только `VS=TRUE`.

`v.nbootsd` — число bootstrap samples, используемых для оценки стандартного отклонения для `theta(x)`. Используется, если только `VS=TRUE`.

`v.nboott` — число bootstrap samples, используемых, чтобы оценить bootstrap T-статистику. Используется, если только `VS=TRUE`. Общее число bootstrap samples равно `v.nbootg*v.nbootsd + v.nboott`.

`perc` — вычисляемые (требуемые) доверительные точки.

Возвращаемые значения

Список, содержащий следующие компоненты:

`confpoints` — вычисленные (оцененные) доверительные точки.

`theta, g` — `theta` и `g` возвращаются, только если `VS=TRUE`. `(theta[i],g[i]), i=1,length(theta)` представляют собой оценку значения функции — преобразования `g`, стабилизирующего дисперсию, в точках `theta[i]`.

`call` — строка, представляющая собой «разобранный» вызов функции.

Пример использования функции bootstrap

Оценим математическое ожидание и 0.05-квантили для набора данных, полученного из равномерного распределения.

Сгенерируем 1000 реализаций случайной величины, равномерно распределенной на отрезке $[0; 1]$:

```
> x <- runif(1000)
```

Зададим функцию, соответствующую 0.05-квантилю:

```
> perc05 <- function(x){quantile(x, .05)}
```

Вызовем процедуру `bootstrap` со значением параметра `nboot=100`:

```
> results <- bootstrap(x, 100, perc05)
```

В `results$thetastar` будет помещено `100` bootstrap оценок для значения 0.05-квантиля.

Чтобы вычислить математическое ожидание для полученных bootstrap оценок, передадим в качестве значения параметра `func` встроенную функцию `mean`:

```
> results <- bootstrap(x, 100, perc05, func=mean)
```

В `results$func.thetastar` будет помещено математическое ожидание для полученных bootstrap оценок.

Пример использования функций boott, bcanon, abcnon

Построим доверительные интервалы для математического ожидания для набора данных, полученного из экспоненциального распределения.

Сгенерируем 20 реализаций случайной величины, экспоненциально распределенной со значением параметра $\lambda = 2$:

```
> x <- rexp(1000, rate = 2)
```

Зададим функцию, используемую для вычисления стандартной ошибки для `theta(x)`:

```
> sdmean <- function(x,nbootsd,theta){sqrt(var(x)/length(x))}
```

Вызовем функцию `boott` для вычисления 0.05 и 0.95 confidence points с использованием преобразования, стабилизирующего дисперсию:

```
> results <- boott(x, mean, sdfun=sdmean, VS=TRUE,
                   perc=c(0.05,0.95), nboott = 2000)
```

В `results$confpoints` будут помещены 0.05 и 0.95 confidence points.

Оценим те же confidence points с использованием функции `bcanon` и `abcnon`:

```
> results <- bcanon(x, theta=mean, nboot=2000, alpha=c(0.05, 0.95))
```

В `result$z0` будет помещено вычисленное (оцененное) значение смещения; в `result$acc` – значение константы ускорения.

Для применения функции `abcnon` следует задать функцию для математического ожидания как функцию от вектора данных `x` и вектора частот встречаемости каждого элемента x_i в каждом из bootstrap samples — `p`:

```
> tt <- function(p,x) {sum(p*x)/sum(p)}
> results<-abcnon(x, tt, alpha=c(0.05,0.95))
```

1.1.1. Задания для лабораторной работы

1) Для `N=10,100,1000` сгенерируйте вектор длины `N`, состоящий из реализаций нормально распределенной случайной величины `N(0,1)`. Вычислите стандарную ошибку для статистического мат. ожидания, используя bootstrap с числом повторений `B=200`, исследуйте ее зависимость от N, сравните с теоретической оценкой.

2) Загрузите набор данных `mouse.t`, вычислите статистическое мат. ожидание и медиану и оцените стандартную ошибку, используя bootstrap (`B=50,100,250,500,1000`).

3) Загрузите набор данных `law` и вычислите корреляцию между средним баллом, набранным учениками одной школы на тесте по правоведению (`LSAT`), и их средним баллом по всем предметам (`GPA`). Оцените стандартную ошибку коэффициента корреляции, используя bootstrap. Вычислите коэффициент корреляции на наборе данных `law82` и сравните полученные результаты.

4) Загрузите набор данных `spatial`. Вычислите 90%-доверительные интервалы для статистической дисперсии для A, используя `1)bootstrap-t`, 2)BC_α, 3)ABC, `4)bootstrap percentile`, и сравните полученные интервалы.