ТОМСКИЙ
ПОЛИТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ

Статистическое моделирование и прогнозирование
Лекция 8. Машинное обучение. Классификация.

Семёнов Михаил Евгеньевич
к. ф.-м. н., доцент ОЭФ ИЯТШ

Томский политехнический университет
9 июня 2020 г.

Задача кластерного анализа - заданную совокупность объектов, для каждой пары которых определена мера сходства, разбить на однородные в некотором смысле группы объектов. Полученные в результате группы объектов называются *кластерами*.

Области применения кластерного анализа чрезвычайно разнообразны. Это приводит к тому, что общее число алгоритмов, упоминающихся в литературе по автоматической классификации и кластерному анализу, варьируется от 200 до 500 и с каждым годом это число возрастает. По способу обработки данных *алгоритмы классификации* можно разделить на две большие группы:

1. *иерархические*
2. *неиерархические*

Иерархические алгоритмы характеризуются последовательным объединением исходных элементов и соответствующим уменьшением числа кластера. В начале работы алгоритма все объекты являются отдельными кластерами. На первом шаге наиболее похожие объекты объединяются в кластер. На последующих шагах объединение продолжается до тех пор, пока все объекты не будут составлять один кластер.

Характерной особенностью иерархических алгоритмов является графическая форма представления результатов в виде **дендрограмы** – древовидного графа.

Преимуществом иерархических методов кластеризации является их **наглядность**, но при большом количестве наблюдений они трудны в восприятии результатов.

Неиерархические алгоритмы основаны на дроблении исходной совокупности многомерных данных на определенное число классов. Одним из наиболее распространенных методов кластерного анализа такого типа являются метод k-средних.

Замечание: Неиерархические алгоритмы более удобны при большом количестве наблюдений, чем иерархические.

Алгоритм 1 Метод k-средних

Вход: k – число классов разбиения.

1: Произвольным образом выбираются k исходных центров классов z_{j0} для $j = 1, 2, ..., k$. Для этой выборки удобнее всего использовать множество подлежащих классификации объектов x_i при $i = 1, 2, ..., n$.

2: Все объекты x_i распределяются по k классам в соответствии с правилом $x_i \in P_n$, если $\rho(x_i, z_{n0}) = \min\limits_{1 \leqslant j \leqslant k} \rho(x_i, z_{j0})$, то есть объект x_i относится к классу P_n, если расстояние от него до центра класса z_{n0} является наименьшим среди всех возможных.

3: Центры классов пересчитываются $z_{p1} = \frac{1}{n_p} \sum_{i=1}^{n_p} x_i, \quad p = 1, 2, ..., k$.

4: Выполнение равенств $z_{j1} = z_{j0}$ для $j = 1, 2, ..., k$ с заранее выбранной точностью является условием окончания работы алгоритма. При нарушении хотя бы одного из указанных равенств выполняется присваивание $z_{j0} \leftarrow z_{j1}$ и переход к шагу 2.

Выход: Объекты x_i, $i = 1,2, \ldots, n$ разбитые на k классов

Обобщенные модели регрессии

Обобщенные модели расширяют класс общих линейных и нелинейных моделей регрессии, связывая зависимую переменную с факторами и ковариатами посредством задаваемой функции связи (link function), причем допускается наличие у отклика произвольного распределения.

Иными словами, **обобщенные линейные модели** (generalized linear models) расширяют применимость линейных моделей, делая возможным анализ зависимых переменных, имеющих отличное от нормального распределение.

Примеры использующихся статистических моделей:

1. линейная регрессия – для откликов с нормальным распределением;
2. логистические модели – для двоичных данных;
3. логлинейные модели – для счетных данных;
4. модели с дополняющим двойным логарифмированием – для интервал-цензурированных данных выживания.

Модель логистической регрессии

Логистическая регрессия полезна для предсказания значений бинарной зависимой переменной по набору непрерывных и/или категориальных зависимых переменных.

Обобщенная логит-линейная модель (или логистическая регрессия на m предикторов) будет иметь вид

$$g(y) = \log \frac{p(y)}{1 - p(y)} = \beta_0 + \sum_{k=1}^{m} \beta_k x_k, \tag{1}$$

а оценки ее параметров можно трактовать следующим образом: при изменении значения предиктора на единицу, значение логарифма отношения шансов зависимой переменной y изменится на величину соответствующего коэффициента β_k.

Постановка заданий

1. Используя выборочные данные по ирисам Фишера $iris\{(x_1, x_2, x_3, x_4)\}$ для $i = 5, 9, \ldots, 149$ провести их автоматическую классификацию (кластерный анализ) различными методами: единственной связи и k-средних. Визуализировать и сравнить полученные результаты. Сделать выводы. Сохранить скрипт в файле (*.r).
2. Для векторов выборочных данных о пассажирах построить логистическую регрессию для прогнозирования выживаемости. Представьте результаты дисперсионного анализа в виде классической таблицы ANOVA. Оцените качество классификации (прогнозирования) построенной модели с помощью ROC-кривой.
   Сохранить скрипт в файле (*.r).

Постановка заданий

3 Сгенерируйте реализации двухмерной выборки: $X \sim N(0,1)$ и $Y \sim N(3,1)$. Постройте график полученных данных. Разделите данные на обучающее множество (80%) и текстовое (20%). Постройте график полученных данных с учетом их принадлежности к множествам. Установите пакет kernlab и проведите обучение линейной SVM. Построить прогноз и проанализировать результаты с использованием ROC-кривой.

Алгоритм 2 Автоматическая классификация методом k-средних

Вход: *data* – вектор выборочных данных.

```
 1: data(iris)
 2: smp <- seq(5, 150, 4)
 3: dat <- scale(iris[smp, -5])
 4: dat[c(1:2, 18:19, 36:37),]
 5: fit1 <- kmeans(dat, centers=3); mps <- fit1[[1]]
 6: library(cluster); library(fpc)
 7: plotcluster(dat, fit1$cluster)
 8: clusplot(dat, fit1$cluster, color=TRUE, shade=TRUE, labels=2, lines=0)
 9: with(iris, pairs(dat, col=c(1:3)[fit1$cluster]))
10: library(cluster); library(HSAUR)
11: dissE <- daisy(dat)
12: dE2 <- dissE^2
13: sk2 <- silhouette(fit1$cl, dE2)
14: plot(sk2)
```

Выход: Классификация выборочных данных методом и k-средних

Результат

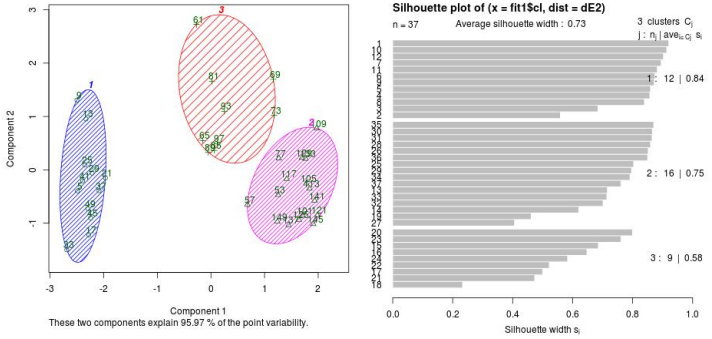

Рисунок 1 – Результат выполнения алгоритма №2

Алгоритм 3 Классификация методом единственной связи

Вход: *data* – вектор выборочных данных.

```
1: data(iris)
2: smp <- seq(5, 150, 4)
3: dat <- scale(iris[smp, -5])
4: sps <- as.numeric(iris[smp, 5])
5: dat[c(1:2, 18:19, 36:37),]
6: dst <- dist(dat, method = "euclidean")
7: fit2 <- hclust(dst, method = "single"); fit2
8: gps <- cutree(fit2, k=3)
```

Выход: Классификация выборочных данных методом единственной связи

Алгоритм 4 Построение логистической регрессии

Вход: *data* – вектор выборочных данных.

```
 1: train <- data[1:800,]
 2: test <- data[801:889,]
 3: model <- glm(Survived ~., family=binomial(link='logit'), data=train)
 4: summary(model)
 5: anova(model, test="Chisq")
 6: library(pscl);    pR2(model)
 7: fitted.results <- predict(model, newdata=[...], type='response')
 8: library(ROCR);    p <- predict(model, newdata=[...], type="response")
 9: pr <- prediction(p, test$Survived)
10: prf <- performance(pr, measure = 'tpr', x.measure = 'fpr');    plot(prf)
11: auc <- performance(pr, measure = 'auc');    auc <- auc$y.values[[1]];   auc
```

Выход: ROC-кривая

Результат

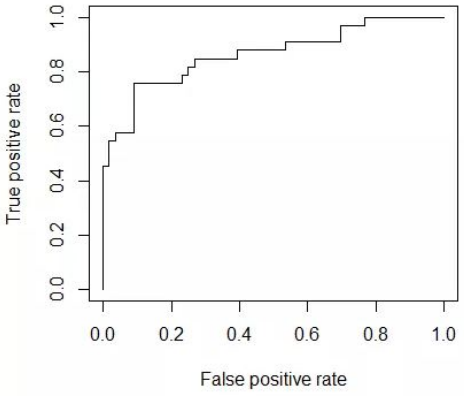

Рисунок 2 – Пример результата выполнения алгоритма №4

Алгоритм 5 Обучение линейной SVM

Вход: $sigma$, $meanX$, $meanY$ – среднеквадратическое отклонение и средние значения генерируемых выборок.

```
sigma <- 1;    meanX <- 0;    meanY <- 3
n <- 150; p <- 2 #number of data points and dimension
nX <- round(n/2);    nY <- n-nX
xX <- matrix(rnorm(nX*p, mean=meanX,sd=sigma), nX, p)
xY <- matrix(rnorm(nY*p, mean=meanY,sd=sigma), nY, p)
x <- rbind(xX, xY)
y <- matrix(c(rep(1,nX),rep(-1,nY))) #Generate the labels
ntrain <- round(n*0.8) #number of training examples
tindex <- sample(n,ntrain) #indices of training samples
xtrain <- x[tindex,];    xtest <- x[-tindex,]
ytrain <- y[tindex];    ytest <- y[-tindex]
istrain=rep(0,n);    istrain[tindex]=1
library(kernlab)
svp <- ksvm(xtrain, ytrain, type='C-svc', kernel='vanilladot', C=100, scaled=c()) #train the SVM
plot(svp, data=xtrain)
```

Скрипт на R [Vert,]


```
16: ypred = predict(svp, xtest);    table(ytest, ypred) #Predict labels on test
17: sum(ypred==ytest)/length(ytest) #Compute accuracy
18: ypredscore = predict(svp, xtest, type="decision") #Compute at the prediction scores
19: table(ypredscore > 0, ypred) #Check that the predicted labels are the signs of the scores
20: library(ROCR)
21: pred <- prediction(ypredscore, ytest)
22: perf <- performance(pred, measure = 'tpr', x.measure = 'fpr');    plot(perf) #Plot ROC curve
23: perf <- performance(pred, measure = 'prec', x.measure = 'rec');    plot(perf) #Plot precision/recall
    curve
24: perf <- performance(pred, measure = "acc");    plot(perf) #Plot accuracy as function of threshold
```

Выход: ROC-кривая

Результат

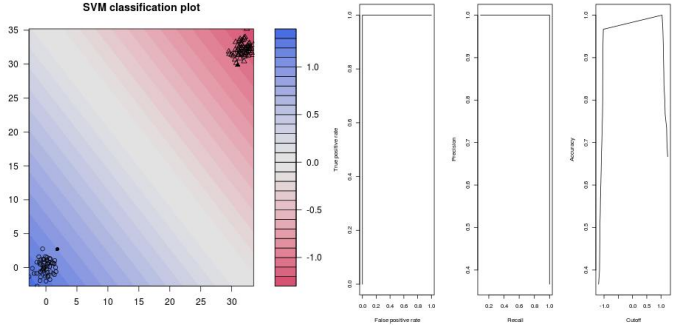

Рисунок 3 – Результат выполнения алгоритма № 5

Rousseeuw, P. (1987).
Silhouettes: A graphical aid to the interpretation and validation of cluster analysis.
J. Comput. Appl. Math., 20:53–65.

Vert, J.-P.
Practical session: Introduction to SVM in R.

Буховец, and Москалев, (2015).
Алгоритмы вычислительной статистки в системе R.

Мастицкий, and Шитиков, (2014).
Статистический анализ и визуализация данных с помощью R.