Спецкурс «Машинное обучение» (2016)

1. Понятие о прецедентном обучении

Имеется множество *объектов* (ситуаций) и множество возможных *ответов* (откликов, реакций). Существует некоторая зависимость между ответами и объектами, но она неизвестна. Известна только конечная совокупность *прецедентов* — пар «объект, ответ», называемая *обучающей выборкой*. На основе этих данных требуется восстановить зависимость, то есть построить алгоритм, способный для любого объекта выдать достаточно точный ответ. Для измерения точности ответов определённым образом вводится *функционал качества*.

Данная постановка является обобщением классических задач [аппроксимации](https://ru.wikipedia.org/wiki/%D0%90%D0%BF%D0%BF%D1%80%D0%BE%D0%BA%D1%81%D0%B8%D0%BC%D0%B0%D1%86%D0%B8%D1%8F) функций. В классических задачах аппроксимации объектами являются действительные числа или векторы. В реальных прикладных задачах входные данные об объектах могут быть неполными, неточными, нечисловыми, разнородными. Эти особенности приводят к большому разнообразию методов машинного обучения.

2. Типы информационных признаков (численные, категорные и т.д.)

|  |
| --- |
| **Описательные** признаки выражаются словесно: национальность человека, разновидность почв, материал стен здания. Описательные признаки подразделяются на номинальные и порядковые. Отличие между ними в том что номинальные – это признаки, по которым можно ранжировать, упорядочивать данные.  **Количественные –**признак, отдельные варианты которого имеют числовое выражение отражают размеры, масштабы изучаемого объекта или явления.  **Первичные** признаки характеризуют единицу совокупности в целом.  **Вторичные,** или расчетные, признаки не измеряются непосредственно, а рассчитываются. Вторичные признаки представляют собой соотношения первичных признаков: деление объема выпущенной продукции на численность работников дает показатель производительности труда; деление суммы затрат на произведенную продукцию на число единиц данной продукции дает себестоимость и т.д  **Прямые**(непосредственные) признаки – это свойства, непосредственно присущие тому объекту, который ими характеризуется.  **Альтернативные признаки –**признак, имеющий только два варианта значений.  **Дискретные** относятся количественные признаки, которые могут принимать только отдельные значения, без промежуточных значений между ними.  **Непрерывные,**точнее, непрерывно варьирующие признаки способны принимать любые значения, конечно, в определенных границах.  **Моментные** признаки характеризуют изучаемый объект в какой-то момент времени, установленный планом статистического исследования.  **Интервальные** признаки, характеризующие результаты процессов. |

3. Задачи классификации и регрессии

Отнести новый объект к классу

Определение зависимости между переменными

Построение функции

4. Феномен переобучения

**Переобучение** (*переподгонка*, *пере-* в значении «слишком», [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *overfitting*) в [машинном обучении](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) и [статистике](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D1%82%D0%B8%D1%81%D1%82%D0%B8%D0%BA%D0%B0) — явление, когда построенная модель хорошо объясняет примеры из обучающей выборки, но относительно плохо работает на примерах, не участвовавших в обучении (на примерах из тестовой выборки).

Это связано с тем, что при построении модели («в процессе обучения») в обучающей выборке обнаруживаются некоторые случайные закономерности, которые отсутствуют в [генеральной совокупности](https://ru.wikipedia.org/wiki/%D0%93%D0%B5%D0%BD%D0%B5%D1%80%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D0%BE%D0%B2%D0%BE%D0%BA%D1%83%D0%BF%D0%BD%D0%BE%D1%81%D1%82%D1%8C).

5. Линейная регрессия

**Регре́ссия** ([лат.](https://ru.m.wikipedia.org/wiki/%D0%9B%D0%B0%D1%82%D0%B8%D0%BD%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *regressio* — обратное движение, отход) в [теории вероятностей](https://ru.m.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B2%D0%B5%D1%80%D0%BE%D1%8F%D1%82%D0%BD%D0%BE%D1%81%D1%82%D0%B5%D0%B9) и [математической статистике](https://ru.m.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%81%D1%82%D0%B0%D1%82%D0%B8%D1%81%D1%82%D0%B8%D0%BA%D0%B0) — [математическое выражение](https://ru.m.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%BE%D0%B5_%D0%B2%D1%8B%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5), отражающее зависимость зависимой переменной *у* от независимых переменных *х* при условии, что это выражение будет иметь [статистическую значимость](https://ru.m.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D1%82%D0%B8%D1%81%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%B7%D0%BD%D0%B0%D1%87%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D1%8C)

Линейная зависимость

df <- data.frame(h = c(100,115,160,160,175,190,190,195),

w = c(10,15,65,80,75,100,110,80))

View(df)

plot(df)

mod\_lin <- lm(w~h-1, df)

#lm(formula=w~h-1, data = df)

mod\_lin

summary(mod\_lin)

mod\_cube <- lm(w~ I(h^3) - 1,df)

mod\_cube

summary(mod\_cube)

res\_lin = predict(mod\_lin, h = df$h)

res\_lin

?predict

res\_cubic = predict(mod\_cube, h = df$h)

res\_cubic

residual\_lin <- res\_lin - df$w

residual\_lin

residuals(mod\_lin)

residual\_cubic <- res\_cubic - df$w

residual\_cubic

accuracy\_lin <- sqrt(mean(residual\_lin^2)) #sum(residual\_lin^2)/length(residual\_lin)

accuracy\_lin

length(residual\_lin)

accuracy\_cubic <- sqrt(mean(residual\_cubic^2)) #sum(residual\_cubic^2)/length(residual\_cubic)

accuracy\_cubic

6. Логистическая регрессия

**Логистическая регрессия** или **логит-регрессия** ([англ.](https://ru.m.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *logit model*) — это [статистическая](https://ru.m.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%81%D1%82%D0%B0%D1%82%D0%B8%D1%81%D1%82%D0%B8%D0%BA%D0%B0) модель, используемая для предсказания вероятности возникновения некоторого события путём подгонки данных к [логистической кривой](https://ru.m.wikipedia.org/wiki/%D0%9B%D0%BE%D0%B3%D0%B8%D1%81%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F).

Y содержит 0 и 1.

Верятность того, что y = 1, равна f(z)

z = t1\*x1 + .... + tn \* xn

df <- data.frame(h = c(100,115,160,160,175,190,190,195),

w = c(10,15,65,80,75,100,110,80))

View(df)

plot(df)

mod\_log <- glm(w~h, data = df, family = "binomial")

summary(mod\_log)

pr = predict(glm,type="response")

library(ISLR)

#Ex.1

?Weekly

l1 = glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,

data = Weekly, family = binomial)

summary(l1)

probs = predict(l1, type="response")

probs[1:5]

contrasts(Weekly$Direction)

pred = ifelse(probs > 0.5, "Up", "Down")

table(pred, Weekly$Direction)

mean(pred == Weekly$Direction)

#200

auto = read.csv("Auto.csv")

med = median(auto$mpg)

med

mpg01 = ifelse(auto$mpg > med, 1, 0)

mpg01

auto = data.frame(auto, mpg01)

lrnSet = sample(nrow(auto),200)

l1 = glm(mpg01 ~ displacement + horsepower + weight + acceleration,

data = auto, family = binomial, subset = lrnSet)

summary(l1)

probs = predict(l1, newdata = auto[-lrnSet,],type = "response")

pred = ifelse(probs > 0.5, 1, 0)

#pred

mean(pred == mpg01[-lrnSet])

k = table(pred, mpg01[-lrnSet])

k

k[1,1]/sum(k[,1])

k[2,2]/sum(k[,2])

7. Решающие деревья

**Дерево принятия решений** (также может называться деревом классификации или регрессионным деревом) — средство поддержки [принятия решений](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%BF%D1%80%D0%B8%D0%BD%D1%8F%D1%82%D0%B8%D1%8F_%D1%80%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D0%B9), использующееся в [статистике](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D1%82%D0%B8%D1%81%D1%82%D0%B8%D0%BA%D0%B0) и [анализе данных](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) для прогнозных моделей. Структура дерева представляет собой «листья» и «ветки». На ребрах («ветках») дерева решения записаны атрибуты, от которых зависит целевая функция, в «листьях» записаны значения [целевой функции](https://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BB%D0%B5%D0%B2%D0%B0%D1%8F_%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F), а в остальных узлах — атрибуты, по которым различаются случаи. Чтобы классифицировать новый случай, надо спуститься по дереву до листа и выдать соответствующее значение. Подобные деревья решений широко используются в интеллектуальном анализе данных. Цель состоит в том, чтобы создать модель, которая предсказывает значение целевой переменной на основе нескольких переменных на входе.

Каждый лист представляет собой значение целевой переменной, измененной в ходе движения от корня по листу. Каждый внутренний узел соответствует одной из входных переменных. Дерево может быть также «изучено» разделением исходных наборов переменных на подмножества, основанные на тестировании значений атрибутов. Это процесс, который повторяется на каждом из полученных подмножеств. Рекурсия завершается тогда, когда подмножество в узле имеет те же значения целевой переменной, таким образом, оно не добавляет ценности для предсказаний. Процесс, идущий «сверху вниз», индукция деревьев решений (TDIDT)[[1]](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_%D0%BF%D1%80%D0%B8%D0%BD%D1%8F%D1%82%D0%B8%D1%8F_%D1%80%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D0%B9#cite_note-1), является примером поглощающего «жадного» алгоритма, и на сегодняшний день является наиболее распространенной стратегией деревьев решений для данных, но это не единственная возможная стратегия. В интеллектуальном анализе данных, деревья решений могут быть использованы в качестве математических и вычислительных методов, чтобы помочь описать, классифицировать и обобщить набор данных, которые могут быть записаны следующим образом:

Зависимая переменная Y является целевой переменной, которую необходимо проанализировать, классифицировать и обобщить. Вектор x состоит из входных переменных и т. д., которые используются для выполнения этой задачи.

library(rpart)

train\_url <- "train.csv"

train <- read.csv(train\_url)

tree1 <- rpart(Survived ~ Sex + Age,data = train, method ="class")

tree1

plot(tree1)

text(tree1)

prediction1 <- predict(tree1, test, type = "class")

solution1 <- data.frame(PassengerId = test$PassengerId, Survived = prediction1)

write.csv(solution1, file = "solution1.csv", row.names = FALSE)

8. Нейронные сети

9. Проверка модели, кроссвалидация

Функция потерь

Обучающее и тестовое множество

**Перекрёстная прове́рка** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *cross-validation*) — метод оценки аналитической модели и её поведения на независимых данных. При оценке модели имеющиеся в наличии данные разбиваются на k частей. Затем на k−1 частях данных производится обучение модели, а оставшаяся часть данных используется для тестирования. Процедура повторяется k раз; в итоге каждая из k частей данных используется для тестирования. В результате получается оценка эффективности выбранной модели с наиболее равномерным использованием имеющихся данных.

10.Метрики для проверки работы регрессионной модели

Функция потерь

Метод наименьших квадратов

R-squared

Коэффициент детерминации ( {\displaystyle R^{2}} R^2 — R-квадрат) — это доля дисперсии зависимой переменной, объясняемая рассматриваемой моделью зависимости, то есть объясняющими переменными. Более точно — это единица минус доля необъяснённой дисперсии (дисперсии случайной ошибки модели, или условной по факторам дисперсии зависимой переменной) в дисперсии зависимой переменной. Его рассматривают как универсальную меру зависимости одной случайной величины от множества других. В частном случае линейной зависимости {\displaystyle R^{2}} R^2 является квадратом так называемого множественного коэффициента корреляции между зависимой переменной и объясняющими переменными. В частности, для модели парной линейной регрессии коэффициент детерминации равен квадрату обычного коэффициента корреляции между y и x.

11.ROC-кривая для бинарного классификатора

**ROC-кривая** ([англ.](https://ru.m.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *receiver operating characteristic*, *рабочая характеристика приёмника*) — график, позволяющий оценить качество [бинарной классификации](https://ru.m.wikipedia.org/w/index.php?title=%D0%91%D0%B8%D0%BD%D0%B0%D1%80%D0%BD%D0%B0%D1%8F_%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F&action=edit&redlink=1), отображает соотношение между долей объектов от общего количества носителей признака, верно классифицированных, как несущих признак, ([англ.](https://ru.m.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *true positive rate*, TPR, называемой *чувствительностью* алгоритма классификации) и долей объектов от общего количества объектов, не несущих признака, ошибочно классифицированных, как несущих признак ([англ.](https://ru.m.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *false positive rate*, FPR, величина 1-FPR называется *специфичностью* алгоритма классификации) при варьировании порога решающего правила.

Также известна как **кривая ошибок**. Анализ классификаций с применением ROC-кривых называется **ROC-анализом**.

Количественную интерпретацию ROC даёт показатель *AUC* ([англ.](https://ru.m.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *area under ROC curve*, *площадь под ROC-кривой*) — площадь, ограниченная ROC-кривой и осью доли ложных положительных классификаций. Чем выше показатель AUC, тем качественнее классификатор, при этом значение 0,5 демонстрирует непригодность выбранного метода классификации (соответствует случайному гаданию). Значение менее 0,5 говорит, что классификатор действует с точностью до наоборот: если положительные назвать отрицательными и наоборот, классификатор будет работать лучше.

12.Стандартные этапы процесса моделирования

# Load the raw training data and replace missing values with NA

training.data.raw <- read.csv('train.csv',header=T,na.strings=c(""))

# Output the number of missing values for each column

sapply(training.data.raw,function(x) sum(is.na(x)))

# Quick check for how many different values for each feature

sapply(training.data.raw, function(x) length(unique(x)))

# A visual way to check for missing data

library(Amelia)

missmap(training.data.raw, main = "Missing values vs observed")

# Subsetting the data

data <- subset(training.data.raw,select=c(2,3,5,6,7,8,10,12))

# Substitute the missing values with the average value

data$Age[is.na(data$Age)] <- mean(data$Age,na.rm=T)

# R should automatically code Embarked as a factor(). A factor is R's way of dealing with

# categorical variables

is.factor(data$Sex) # Returns TRUE

is.factor(data$Embarked) # Returns TRUE

# Check categorical variables encoding for better understanding of the fitted model

contrasts(data$Sex)

contrasts(data$Embarked)

# Remove rows (Embarked) with NAs

data <- data[!is.na(data$Embarked),]

rownames(data) <- NULL

# Train test splitting

train <- data[1:800,]

test <- data[801:889,]

# Model fitting

model <- glm(Survived ~.,family=binomial(link='logit'),data=train)

summary(model)

# Analysis of deviance

anova(model,test="Chisq")

# McFadden R^2

library(pscl)

pR2(model)

#-------------------------------------------------------------------------------

# MEASURING THE PREDICTIVE ABILITY OF THE MODEL

# If prob > 0.5 then 1, else 0. Threshold can be set for better results

fitted.results <- predict(model,newdata=subset(test,select=c(2,3,4,5,6,7,8)),type='response')

fitted.results <- ifelse(fitted.results > 0.5,1,0)

misClasificError <- mean(fitted.results != test$Survived)

print(paste('Accuracy',1-misClasificError))

# Confusion matrix

library(caret)

confusionMatrix(data=fitted.results, reference=test$Survived)

library(ROCR)

# ROC and AUC

p <- predict(model, newdata=subset(test,select=c(2,3,4,5,6,7,8)), type="response")

pr <- prediction(p, test$Survived)

# TPR = sensitivity, FPR=specificity

prf <- performance(pr, measure = "tpr", x.measure = "fpr")

plot(prf)

auc <- performance(pr, measure = "auc")

auc <- auc@y.values[[1]]

auc