Data Analytics

Yong Zheng

Illinois Institute of Technology
Chicago, IL, 60616, USA

School of Applied Technology
ILLINOIS INSTITUTE OF TECHNOLOGY

Summary: KNN

- ❑ K-Nearest Neighbor (KNN) Classifier

A simple classifier, a lazy learner

1).Choose an odd number for K

2).Calculate distances between target and instances in training set

3).Pick the top KNN and assign the majority label as prediction

- ❑ Extended Problems in Classification Algorithms

Q1. Is it able to take categorical features? If Yes, how to treat them

Q2. Is normalization required?

Q3. How to alleviate overfitting problem?

Note: they are general concerns in classification, not only KNN.

Summary: Naïve Bayes Classifier

1). How to treat numerical and categorical data in Naïve Bayes?

In KNN, we need to transform nominal data to numerical ones.
In Naïve Bayes, we need to transform numerical data to nominal data.

2). Is normalization required in Naïve Bayes?

It is not necessary in Naïve Bayes, but required in KNN

3). Overfitting in Naïve Bayes?

Be careful about the imbalanced data in Naïve Bayes.

4). Which one is better?

It depends. You'd better try Naïve Bayes if there are multiple nominal features

Logistic Regression

Simple Logistic regression model

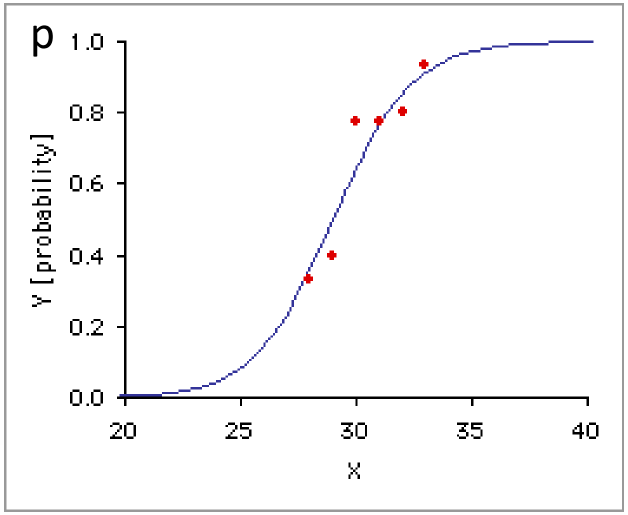
Simple case: Relationship between qualitative binary variable Y and one x-variable:

Model for probability p=Pr(Y=1) for each value x.

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x$$

Odds= $$\frac{p}{1-p} = \frac{P(Y=1)}{P(Y=0)}$$

measures the odds that event Y = 1 occurs



Note: the logistic regression model

is linear in log(odds).

Interpreting odds $\frac{p}{1-p} = \frac{P(Y=1)}{P(Y=0)}$

Let p=Pr(Y=1) the probability of “success”

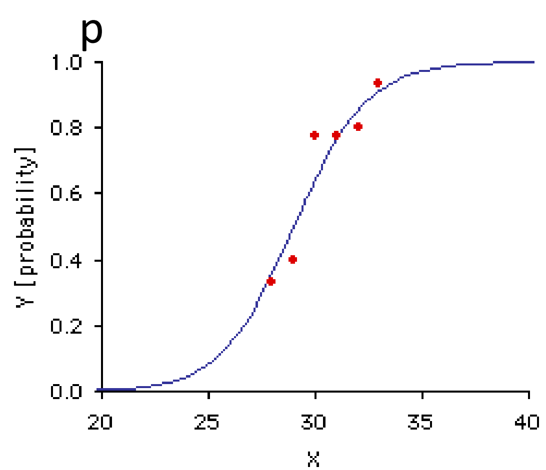
- If odd>1 then pr(Y=1) **>** Pr(Y=0) ➔ Pr(Y=1) > 0.5
- If odd=1 then Pr(Y=1) **=** Pr(Y=0) ➔ Pr(Y=1)=0.5
- If odd<1 then p=pr(Y=1) **<** Pr(Y=0) ➔ Pr(Y=1) < 0.5

Simple Logistic regression model

Simple case: Relationship between qualitative binary variable Y and one x-variable:

Model for probability p=Pr(Y=1) for each value x.

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x$$



The slope parameter measures the degree of association between the probability p =pr(Y=1) and the value of X.

If β_1 **> 0**, then the odds of success **increases** with an increase in X.
If β_1 **< 0**, then the odds of success **decreases** with an increase in X.

$e^{\beta_1} - 1$ = percentage change in odds of success for every unit increase in X.

Example of logistic model

The predictive model for *p=Pr(Y=task completed)* is

$$\log\left(\frac{\hat{p}}{1-\hat{p}}\right) = -3.0597 + 0.1615\,\textit{months of experience}$$

or

$$\hat{p} = \frac{\exp(-3.0597 + 0.1615\,\textit{months of experience})}{1 + \exp(-3.0597 + 0.1615\,\textit{months of experience})}$$

What does the slope β_1 mean?

The log(p/(1-p)) increases of 0.1615, for each additional month of experience.

Or using the anti-log function $\exp(0.1615) = 1.17$

The odds p/(1-p) of success increase of 17%, for each additional month of experience.

Multiple logistic regression models

- Relationship between a qualitative binary variable Y and several X-variables
- Model for probability p=Pr(Y=1):

$$\log(\frac{p}{1-p}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + e$$

- The x-variable can be quantitative or qualitative (represented by dummy variables), just the similar process of incorporating dummy variables when we introduce multiple linear regression models.

Estimation procedures

- Parameter estimates are computed using Maximum Likelihood Estimation (MLE). It is no longer Least Squares!
- Results hold only for large samples
- You can still use different feature selection methods, such as backward, forward, stepwise, etc, by using the step function in R
- The residuals in logistic regression are difficult to be interpreted. You can ignore residual analysis

R code

```
# Logistic Regression
# where F is a binary factor and
# x1-x3 are continuous predictors
# mydata is the dataframe containing the dataset
# glm(…,family=binomial()) fit the logistic regression model
fit <- glm(F~x1+x2+x3,data=mydata, family=binomial())
summary(fit) # display results
confint(fit) # 95% CI for the coefficients
exp(coef(fit)) # compute exp(coefficients) to analyze
             change in odds for changes in X
exp(confint(fit)) # 95% CI for exp(coefficients), that is
             change in odds
```

R code

```
predict(fit, type="response", newdata=dataframe)
```

https://stat.ethz.ch/R-manual/R-devel/library/stats/html/predict.glm.html

`type`

the type of prediction required. The default is on the scale of the linear predictors; the alternative `"response"` is on the scale of the response variable. Thus for a default binomial model the default predictions are of log-odds (probabilities on logit scale) and `type = "response"` gives the predicted probabilities. The `"terms"` option returns a matrix giving the fitted values of each term in the model formula on the linear predictor scale.

```
residuals(fit, type="deviance") # residuals
```

N-fold cross validation ➔ you can still use cv.glm()
The output is not longer prediction errors, but error rate
Accuracy = 1 – error rate

R Code

- Special Note: the glm function with family = binominal(), can only be used for binary classifications.
- If you would like to perform multi-class classifications, you need to use other functions

R Code for multi-class Classifications

1. If you use hold-out evaluation

```
library(nnet)
library(Metrics)
model=multinom(class~.,data=train)
pd=predict(model,newdata=test,type="class")
accuracy(class,pd)
```

2. If you use N-fold cross validation

```
library(caret)
library(nnet)
model = train(x,y,'multinom',trControl=trainControl(method='cv',number=5))
print(model)
```

Example: Logistic Regression

- Data: Case Study 3 - Admissions

```
admit,gre,gpa,rank
0,380,3.61,3
1,660,3.67,3
1,800,4,1
1,640,3.19,4
0,520,2.93,4
1,760,3,2
1,560,2.98,1
0,400,3.08,2
```

Example: Logistic Regression

- Example: hold-out evaluations, load and split data

```
> mydata=read.csv("case3_admission.csv",header=T)
> mydata=mydata[sample(nrow(mydata)),]
> select.data = sample (1:nrow(mydata), 0.8*nrow(mydata))
> train.data = mydata[select.data,]
> test.data = mydata[-select.data,]
> head(mydata)
    admit gre  gpa rank
265     1 520 3.90    3
140     1 600 3.58    1
120     0 340 2.92    3
172     0 540 2.81    3
247     0 680 3.34    2
168     0 720 3.77    3
> train.label=train.data$admit
> test.label=test.data$admit
```

Example: Logistic Regression

- Example: hold-out evaluations, build model by FS

```
> full=glm(admit~gre+gpa+rank, data=train.data, family=binomial())
> base=glm(admit~gpa, data=train.data, family=binomial())
> library(leaps)
Warning message:
package 'leaps' was built under R version 3.5.2
> step(base, scope=list(upper=full, lower=~1), direction="both", trace=F)

Call:  glm(formula = admit ~ gpa + rank + gre, family = binomial(),
    data = train.data)

Coefficients:
(Intercept)          gpa         rank          gre
  -2.861669     0.683853    -0.594686     0.002019

Degrees of Freedom: 319 Total (i.e. Null);  316 Residual
Null Deviance:      402.1
Residual Deviance: 370.7        AIC: 378.7
```

Example: Logistic Regression

- Example: hold-out evaluations, produce probabilities

```
> predict(full, type="response", newdata=test.data)
       168        153        147        392        349        184        339        399 
0.35114554 0.48386137 0.31932437 0.48369815 0.27974173 0.41706093 0.45565929 0.46461888 
       182        297        159          7        283          1        274        243 
0.17113226 0.45889138 0.41777034 0.42851806 0.17527994 0.19625871 0.53655916 0.22292808 
        95        266         12        108        217        271        187        224 
0.40947750 0.16875397 0.40949799 0.28059366 0.31286434 0.48556940 0.25990833 0.34126702 
        86        358         19        112        385          3        293        330 
0.27620968 0.56483108 0.53208385 0.11299118 0.21580479 0.70974819 0.46308072 0.09732698 
       379        176        113        109        255         29         25        304 
0.22793249 0.37877506 0.13384615 0.13770473 0.20762858 0.43184987 0.44379247 0.51088138 
        54        242         18        150        226        245        123        131 
0.39126377 0.54957227 0.10263655 0.57472852 0.31031481 0.42867812 0.16152450 0.34716567 
> prob=predict(full, type="response", newdata=test.data)
```

Example: Logistic Regression

- Example: hold-out evaluations
- Next, choose cut-off value to calculate accuracy

```
> for(i in 1:length(prob)){
+     if(prob[i]>0.5){
+         prob[i]=1
+      }else{
+ prob[i]=0
+ }
+ }
> library(Metrics)
> accuracy(test.label, prob)
[1] 0.6875
```

```
> prob=predict(full, type="response", newdata=test.data)
> for(i in 1:length(prob)){
+     if(prob[i]>0.4){
+         prob[i]=1
+      }else{
+ prob[i]=0
+ }
+ }
> accuracy(test.label, prob)
[1] 0.7
```

Example: Logistic Regression

- Example: 5-fold cross validation

```
> fit=glm(admit~gre+gpa+rank, data=mydata)
> cv.glm(fit, data=mydata, K=5)$delta
[1] 0.2013244 0.2007128
> |
```

- Use $delta to get the error rate
- Always read the first one which is the raw rate
- The error rate in this case is 0.2013244
- The accuracy = 1 – error rate = 0.79868

Classification Algorithms

- Classification algorithm is the key component in the process
- They are able to learn from training and build models

There are many (supervised) classification algorithms:

- K-nearest neighbor classifier
- Naïve Bayes classifier
- Decision tress
- Logistic regression
- Support Vector Machines
- Ensemble classifiers (e.g., random forest)
- …

In-Class Practice

- UCI Data: Teaching Assistant Evaluation Data

https://archive.ics.uci.edu/ml/datasets/Teaching+Assistant+Evaluation

- UCI Data: Adult Data

https://archive.ics.uci.edu/ml/datasets/Adult

- Task

Build models by using KNN, Naïve Bayes and Logistic regression
Compare the models and find the best performing one.

- Practice

Use Adult data (binary classification) for in-class practice
Use TA evaluation data (multi-class) as your assignment

In-Class Practice

- Hints

Make a decision about features and labels

Make a decision about evaluations (strategy and metrics)

Make a decision about which algorithms to be used

Preprocess your data according to the requirements in each algorithm

Run classifications on the preprocessed data