semana 4

luis

19/7/2021

Contents

Regresión regularizada

Idea básica

1. Ajustar un modelo de regresión
2. Penalizar (o reducir) los coeficientes grandes

Pros:

- Puede ayudar con la compensación de sesgo / varianza
- Puede ayudar con la selección del modelo

Contras:

- Puede ser computacionalmente exigente en grandes conjuntos de datos
- No funciona tan bien como bosques random forests y boosting.

ejemplo motivacional:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon$$

donde X_1 y X_2 están casi perfectamente correlacionados (colineales). Puede aproximar este modelo por:

$$Y = \beta_0 + (\beta_1 + \beta_2) X_1 + \epsilon$$

El resultado es:

- Obtendrá una buena estimación de Y
- La estimación (de Y) estará sesgada

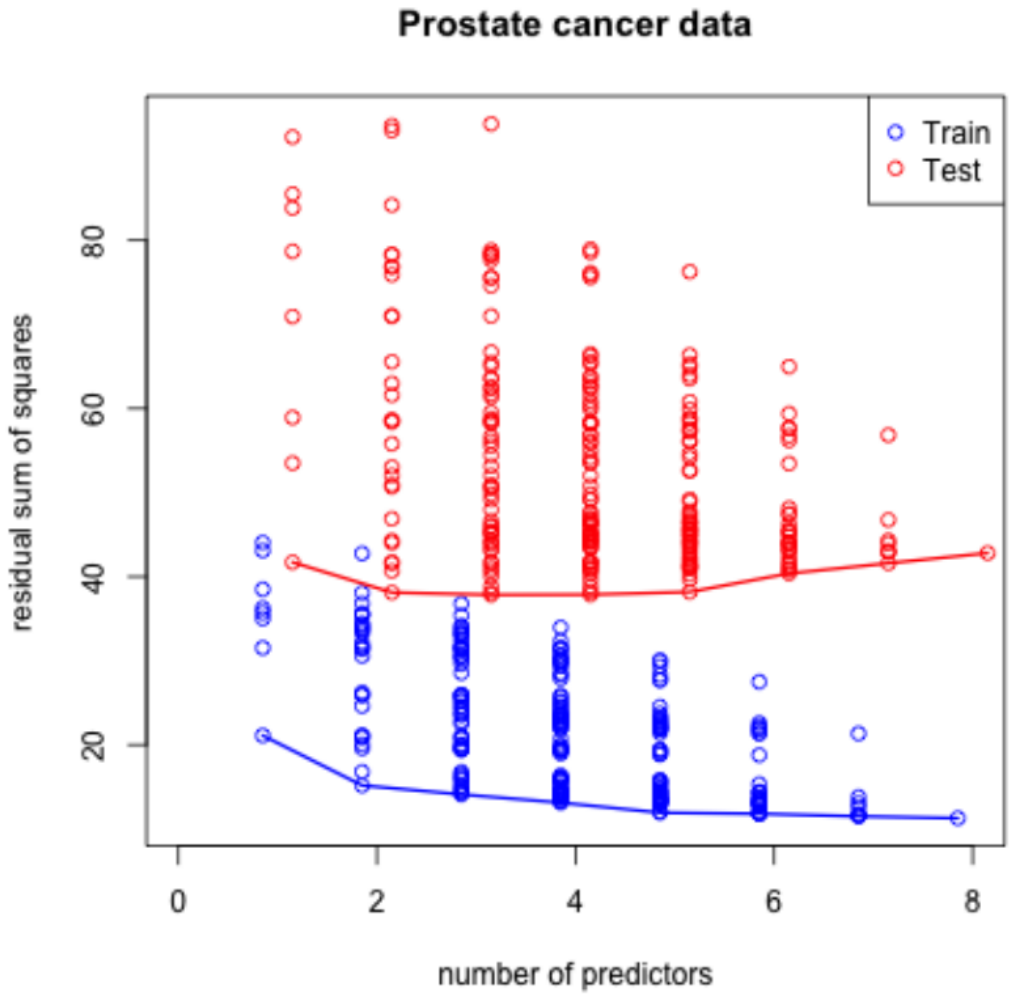
- Podemos reducir la variación en la estimación.

con datos de cancer de prostata

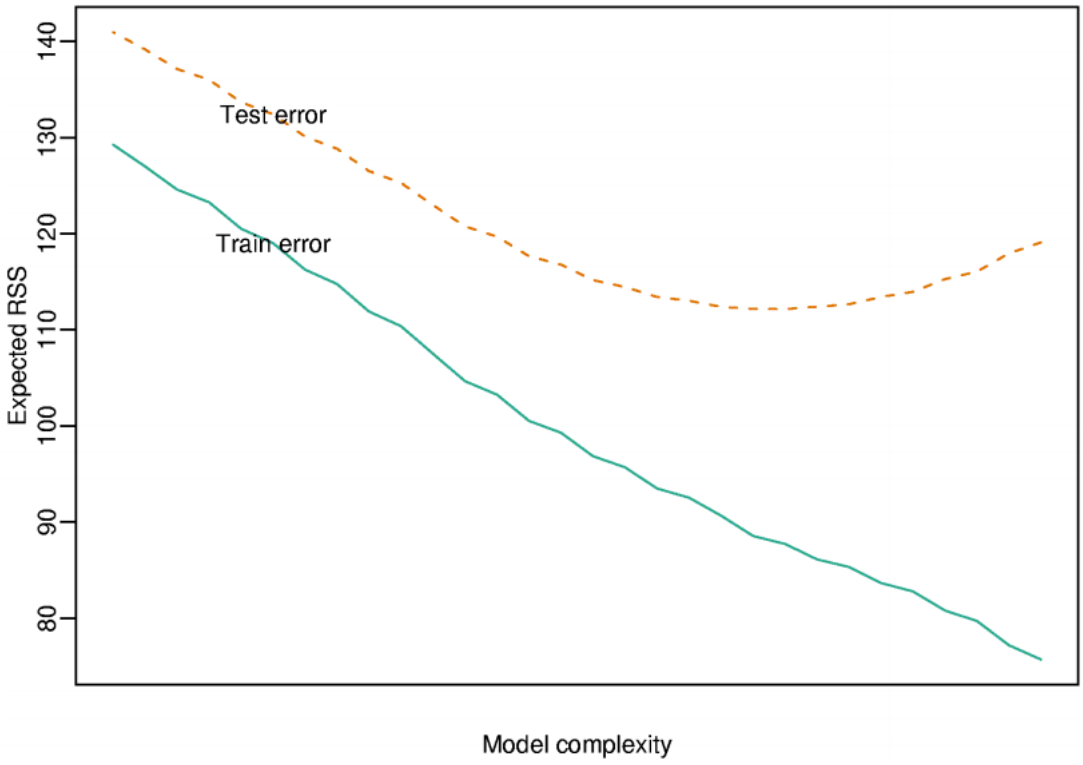
```
library(ElemStatLearn); data(prostate)
str(prostate)
```

```
## 'data.frame':    97 obs. of  10 variables:
##  $ lcavol : num  -0.58 -0.994 -0.511 -1.204 0.751 ...
##  $ lweight: num  2.77 3.32 2.69 3.28 3.43 ...
##  $ age    : int  50 58 74 58 62 50 64 58 47 63 ...
##  $ lbph   : num  -1.39 -1.39 -1.39 -1.39 -1.39 ...
##  $ svi    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ lcp    : num  -1.39 -1.39 -1.39 -1.39 -1.39 ...
##  $ gleason: int  6 6 7 6 6 6 6 6 6 6 ...
##  $ pgg45  : int  0 0 20 0 0 0 0 0 0 0 ...
##  $ lpsa   : num  -0.431 -0.163 -0.163 -0.163 0.372 ...
##  $ train  : logi  TRUE TRUE TRUE TRUE TRUE TRUE ...
```

podemos ver que llega un punto en que el error aumenta si aumenta el número de variables



entonces el patron mas comun es



Enfoque de selección de modelos: muestras divididas

- Acercarse
  1. Divida los datos en entrenamiento / prueba / validación
  2. Trate la validación como datos de prueba, entrene todos los modelos competidores en los datos del train y elija el mejor para la validación.
  3. Para evaluar adecuadamente el rendimiento sobre nuevos datos, aplique al conjunto de prueba.
  4. Puede volver a dividir y volver a realizar los pasos 1-3
- Dos problemas comunes
  - Datos limitados
  - Complejidad computacional

http://www.biostat.jhsph.edu/~ririzarr/Teaching/649/ http://www.cbcb.umd.edu/~hcorrada/PracticalML/

descomposición de Error de predicción esperado

sea $Y_i = f(X_i) + \epsilon_i$

$EPE(\lambda) = E\left[\{Y - \hat{f}_\lambda(X)\}^2\right]$

supongamos que $\hat{f}_\lambda$ es la estimación de los datos de entrenamiento y mira un nuevo punto de datos $X = x^*$

$$E\left[\{Y - \hat{f}_\lambda(x^*)\}^2\right] = \sigma^2 + \{E[\hat{f}_\lambda(x^*)] - f(x^*)\}^2 + var[\hat{f}_\lambda(x_0)]$$
$$= \text{Error irreducible} + \text{sesgo}^2 + \text{varianza}$$

Umbral duro

- Modelo $Y = f(X) + \epsilon$
- Establecer $\hat{f}_\lambda(x) = x'\beta$

- Restrinja solo los coeficientes λ para que sean distintos de cero.
- El problema de selección es después de elegir λ averiguar qué coeficientes $p - \lambda$ hacen distintos de cero

Regularización para regresión

Si los β_j no están restringidos: * Pueden explotar * Y, por tanto, son susceptibles a variaciones muy elevadas

Para controlar la varianza, podríamos regularizar/reducir los coeficientes.

$$PRSS(\beta) = \sum_{j=1}^{n}(Y_j - \sum_{i=1}^{m}\beta_{1i}X_{ij})^2 + P(\lambda;\beta)$$

donde $PRSS$ es una forma penalizada de la suma de cuadrados. Cosas que se buscan comúnmente

- La penalización reduce la complejidad
- La penalización reduce la varianza
- La penalización respeta la estructura del problema.

Regresion Ridge

Resuelve:

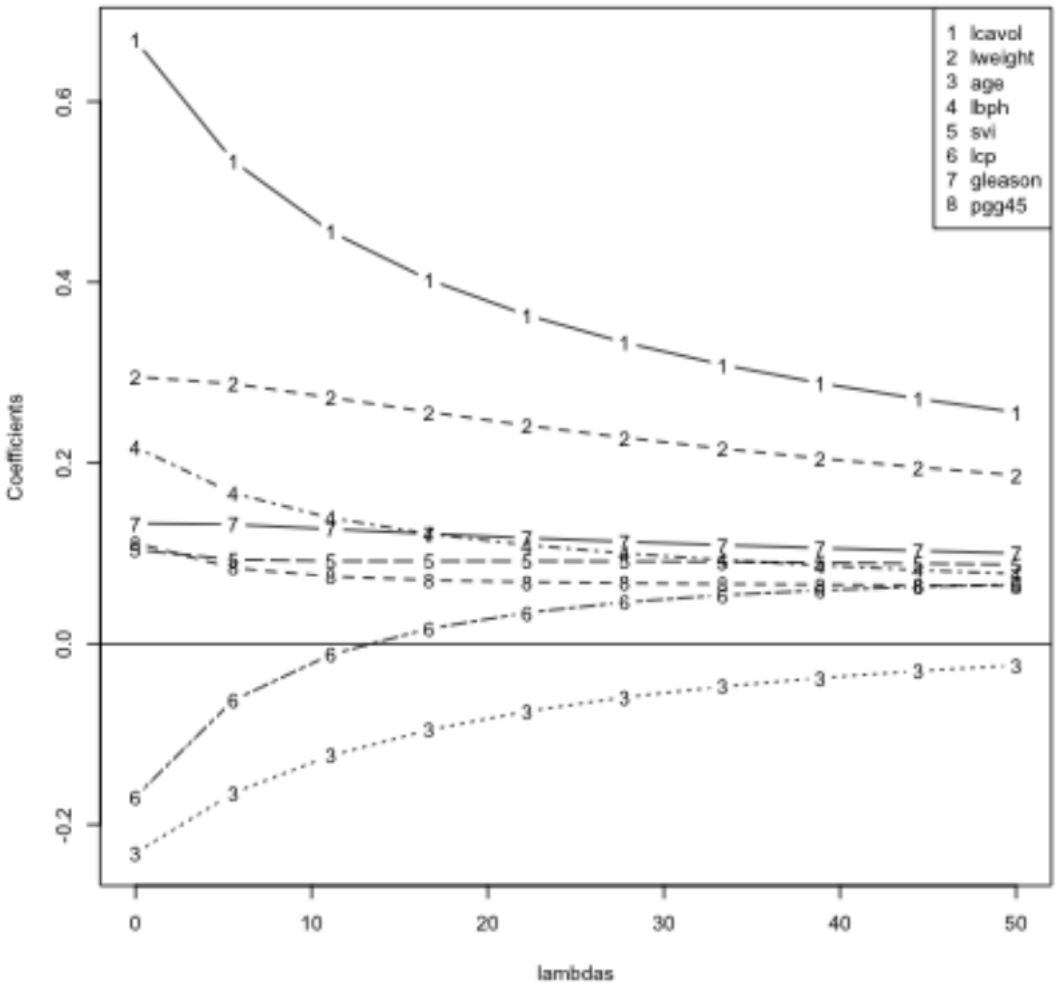
$$\sum_{i=1}^{N}\left(y_i - \beta_0 + \sum_{j=1}^{p}x_{ij}\beta_j\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2$$

equivalente a resolver

$\sum_{i=1}^{N}\left(y_i - \beta_0 + \sum_{j=1}^{p}x_{ij}\beta_j\right)^2$ sujeto a $\sum_{j=1}^{p}\beta_j^2 \leq s$ donde s es inversamente proporcional a λ

La inclusión de λ hace que el problema no sea singular incluso si X^TX no es invertible.

trayectorias de los coeficentes al aumentar λ



Parámetro de ajuste λ

- λ controla el tamaño de los coeficientes
- λ controla la cantidad de regularización {**}**
- **cuando** $\lambda \to 0$ **obtenemos la solución de mínimos cuadrados**
- **cuando** $\lambda \to \infty$ **tenemos** $\hat{\beta}^{ridge}_{\lambda=\infty} = 0$

regresion lasso

$\sum_{i=1}^{N} \left(y_i - \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j \right)^2$ sujeto a $\sum_{j=1}^{p} |\beta_j| \leq s$

equivalente a

$$\sum_{i=1}^{N} \left(y_i - \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

Para matrices de diseño ortonormal (¡no la norma!), Esto tiene una solución de forma cerrada

$$\hat{\beta}_j = sign(\hat{\beta}^0_j)(|\hat{\beta}^0_j - \gamma)^+$$

but not in general.

http://www.biostat.jhsph.edu/~ririzarr/Teaching/649/ http://www.cbcb.umd.edu/~hcorrada/PracticalML/

Notas y futuras lecturas

- Hector Corrada Bravo's Practical Machine Learning lecture notes
- Hector's penalized regression reading list
- Elements of Statistical Learning
- In `caret` methods are:
  - `ridge`
  - `lasso`
  - `relaxo`

combinando predictores

Ideas claves

- Puede combinar clasificadores promediando / votando
- La combinación de clasificadores mejora la precisión
- La combinación de clasificadores reduce la interpretabilidad.
- Boosting, bagging, y random forests son variantes de este tema.

_Supongamos que tenemos 5 clasificadores completamente independientes

Si la precisión es del 70% para cada uno: * $10 \times (0.7)^3(0.3)^2 + 5 \times (0.7)^4(0.3)^2 + (0.7)^5$ * 83,7% de precisión del voto mayoritario

Con 101 clasificadores independientes * 99,9% de precisión del voto mayoritario_

como se calculo

Enfoques para combinar clasificadores

1. Bagging, boosting, random forests

- Suelen combinar clasificadores similares

2. Combinando diferentes clasificadores
   - Modelo de apilamiento (stacking)
   - Modelos de ensamble (ensembling)

ejemplo con datos wage

creando conjuntos de training, test y validation

```
library(ISLR); data(Wage); library(ggplot2); library(caret);
Wage <- subset(Wage,select=-c(logwage))

# Create a building data set and validation set
inBuild <- createDataPartition(y=Wage$wage,
                               p=0.7, list=FALSE)
validation <- Wage[-inBuild,]; buildData <- Wage[inBuild,]

inTrain <- createDataPartition(y=buildData$wage,
                               p=0.7, list=FALSE)
training <- buildData[inTrain,]; testing <- buildData[-inTrain,]

dim(training)
```

```
## [1] 1474   10
```

```
dim(testing)
```

```
## [1] 628  10
```

```
dim(validation)
```

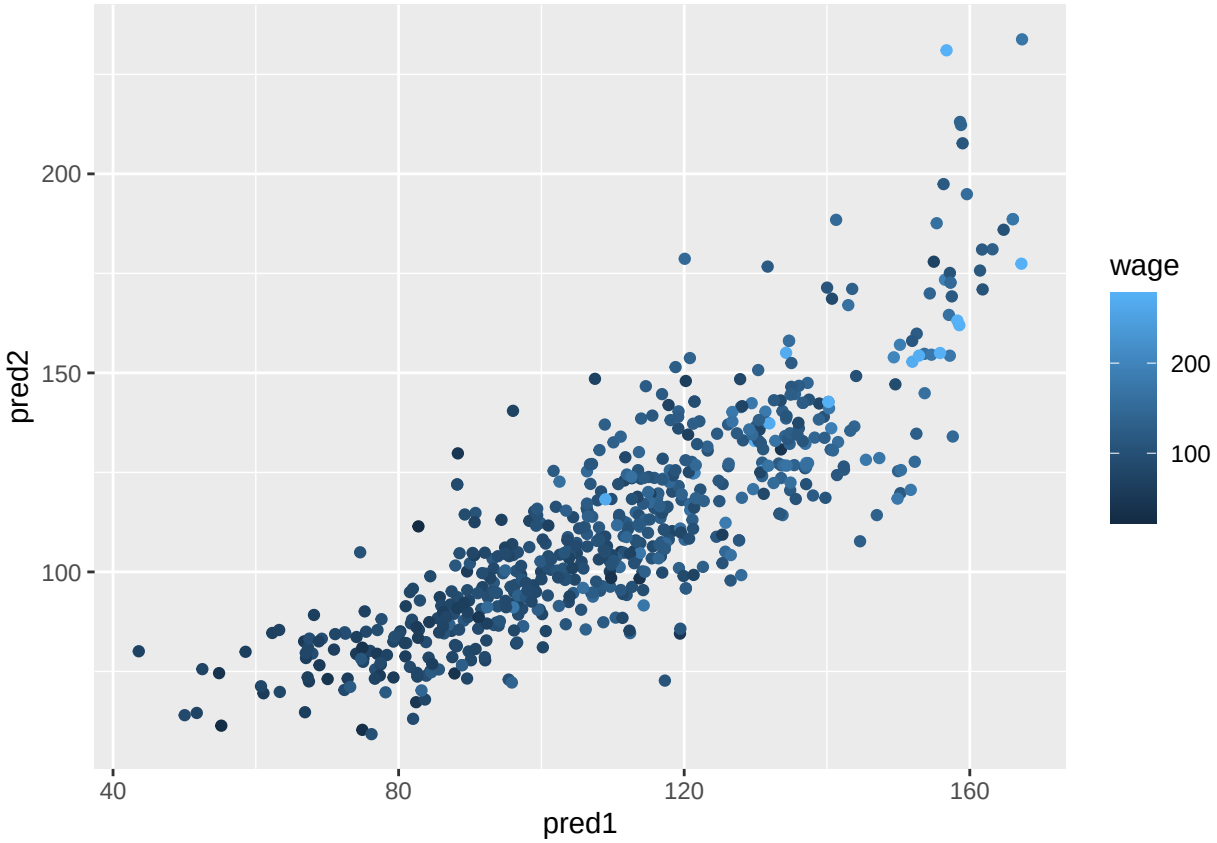
```
## [1] 898  10
```

despues creamos 2 diferentes modelos

```
mod1 <- train(wage ~.,method="glm",data=training)
mod2 <- train(wage ~.,method="rf",
              data=training,
              trControl = trainControl(method="cv"),number=3)
```

luego predecimos en el connjunto de testing, por el color vemos que no funciona del todo bien

```
pred1 <- predict(mod1,testing); pred2 <- predict(mod2,testing)
qplot(pred1,pred2,colour=wage,data=testing)
```



ahora construimos un modelos que combine los dos predictores

```
predDF <- data.frame(pred1,pred2,wage=testing$wage)
combModFit <- train(wage ~.,method="gam",data=predDF)
combPred <- predict(combModFit,predDF)
```

errores en testing

```
sqrt(sum((pred1-testing$wage)^2))
```

```
## [1] 774.6612
```

```
sqrt(sum((pred2-testing$wage)^2))
```

```
## [1] 835.0531
```

```
sqrt(sum((combPred-testing$wage)^2))
```

```
## [1] 753.6058
```

prediciendo en el conjunto de validacion

```
pred1V <- predict(mod1,validation); pred2V <- predict(mod2,validation)
predVDF <- data.frame(pred1=pred1V,pred2=pred2V)
combPredV <- predict(combModFit,predVDF)
```

evaluando en validacion

```
sqrt(sum((pred1V-validation$wage)^2))
```

```
## [1] 1045.166
```

```
sqrt(sum((pred2V-validation$wage)^2))
```

```
## [1] 1091.822
```

```
sqrt(sum((combPredV-validation$wage)^2))
```

```
## [1] 1063.127
```

Notas y otros recursos

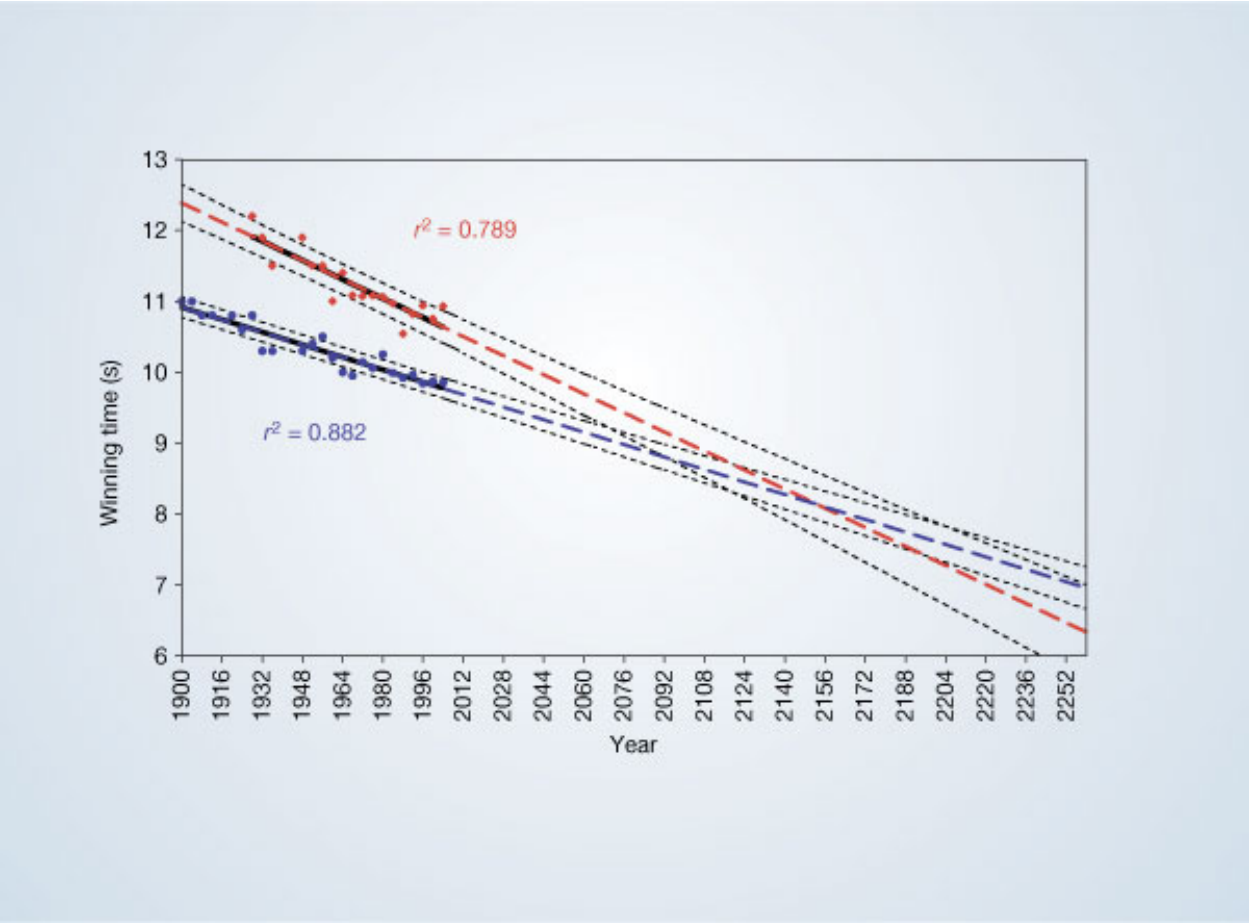
- Incluso una simple mezcla puede ser útil
- Modelo típico para datos binarios/multiclase
  - Construye un número impar de modelos.
  - Predecir con cada modelo
  - Predecir la clase por mayoría de votos
- Esto puede volverse mucho más complicado
  - Mezcla simple de intercalación: caretEnsemble (¡úsala bajo tu propio riesgo!)
  - Wikipedia ensemlbe learning

Pronosticos

¿Que es diferente?

- Los datos dependen del tiempo
- Tipos de patrones específicos
  - Tendencias: aumento o disminución a largo plazo
  - Patrones estacionales: patrones relacionados con la época de la semana, mes, año, etc.
  - Ciclos: patrones que suben y bajan periódicamente
- Submuestreo en entrenamiento / prueba es más complicado
- Surgen problemas similares en los datos espaciales
  - Dependencia entre observaciones cercanas
  - Efectos específicos de la ubicación
- Normalmente, el objetivo es predecir una o más observaciones en el futuro.
- Se pueden utilizar todas las predicciones estándar (¡con precaución!)
- ¡Cuidado con las correlaciones falsas!

- También es común en análisis geográficos ** Beware extrapolation!*



ejemplo con datos de google

```
library(quantmod)
from.dat <- as.Date("01/01/08", format="%m/%d/%y")
to.dat <- as.Date("12/31/13", format="%m/%d/%y")
getSymbols("GOOG", src="yahoo", from = from.dat, to = to.dat)
```

```
## [1] "GOOG"
```

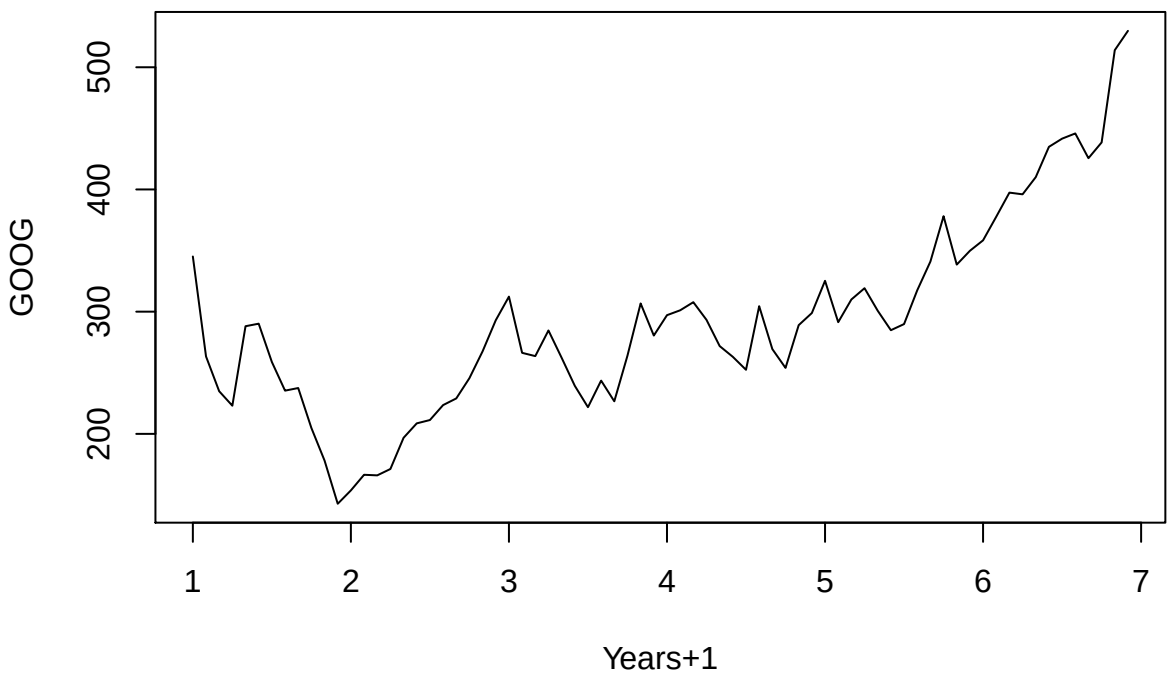
```
head(GOOG)
```

```
##            GOOG.Open GOOG.High GOOG.Low GOOG.Close GOOG.Volume GOOG.Adjusted
## 2008-01-02  345.1413  347.3829 337.5996   341.3157     8646087      341.3157
## 2008-01-03  341.3505  342.1426 336.9969   341.3854     6529382      341.3854
## 2008-01-04  338.5759  339.2086 326.2770   327.2733    10759780      327.2733
## 2008-01-07  325.7490  329.9034 317.4850   323.4128    12854803      323.4128
## 2008-01-08  325.2808  328.7478 314.3218   314.6606    10718225      314.6606
## 2008-01-09  313.8436  325.4501 310.0927   325.3804    13529924      325.3804
```

Resumir mensualmente y almacenar como series de tiempo

```
mGoog <- to.monthly(GOOG)
googOpen <- Op(mGoog)
ts1 <- ts(googOpen,frequency=12)
```

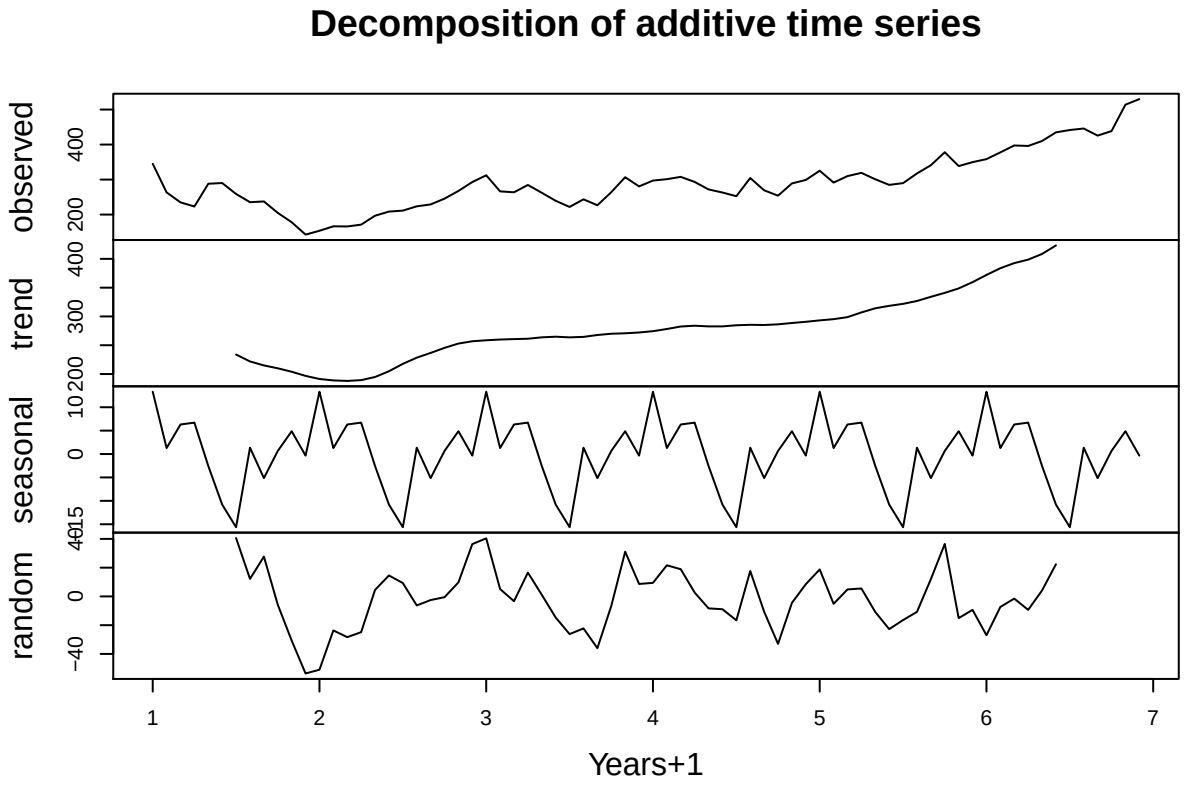
```
plot(ts1,xlab="Years+1", ylab="GOOG")
```



Ejemplo de descomposición de series de tiempo

- **Tendencia**: patrón en constante aumento a lo largo del tiempo
- **estacional**: cuando hay un patrón durante un período de tiempo fijo que se repite.
- **ciclo**: cuando los datos aumentan y disminuyen durante períodos no fijos https://www.otexts.org/fpp/6/1

```
plot(decompose(ts1),xlab="Years+1")
```



creando conjuntos de prueba y de entrenamiento

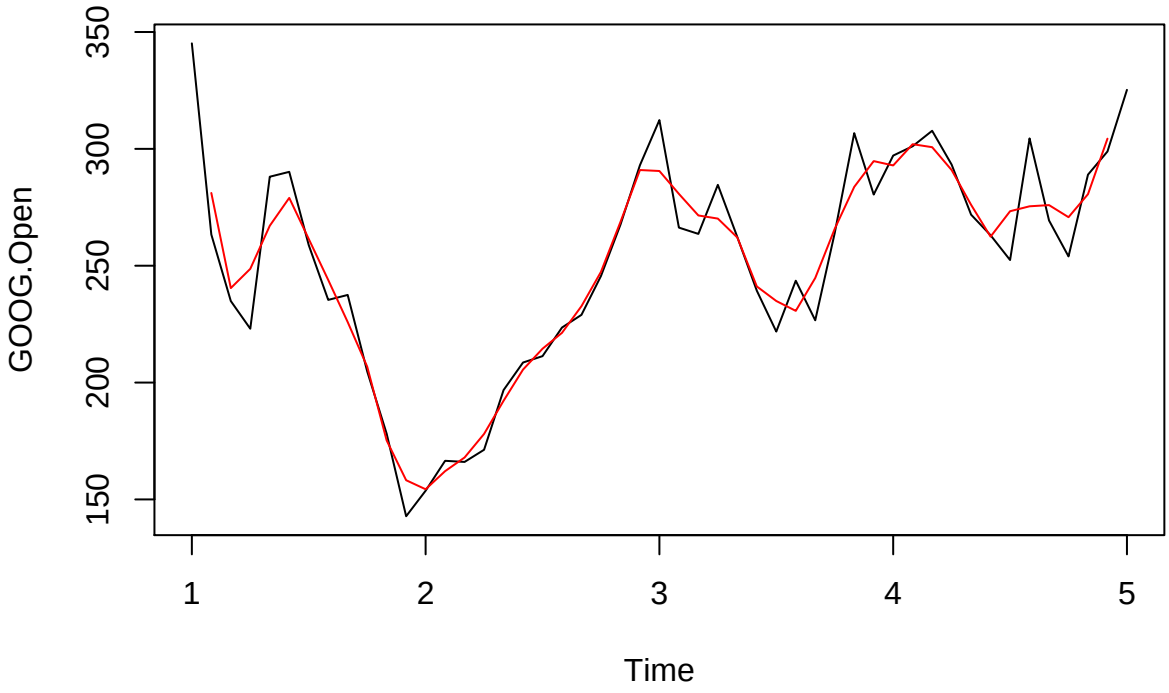
```
ts1Train <- window(ts1,start=1,end=5)
ts1Test <- window(ts1,start=5,end=(7-0.01))
ts1Train
```

```
##         Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 1 345.1413 263.3479 234.8746 223.0340 288.0752 290.1624 258.8199 235.3728
## 2 153.7238 166.5208 166.0426 171.2481 196.7774 208.5832 211.3080 223.5322
## 3 312.3044 266.3018 263.6119 284.6082 262.2670 239.3180 221.8136 243.5820
## 4 297.1263 301.1163 307.7365 293.2807 271.8311 263.0341 252.4239 304.4688
## 5 325.2509
##         Sep      Oct      Nov      Dec
## 1 237.4948 204.8073 178.1224 142.8047
## 2 228.9817 245.5795 267.5372 292.9669
## 3 226.6405 264.0104 306.7154 280.4488
## 4 269.3654 253.9731 288.9669 298.8797
## 5
```

medias moviles simples

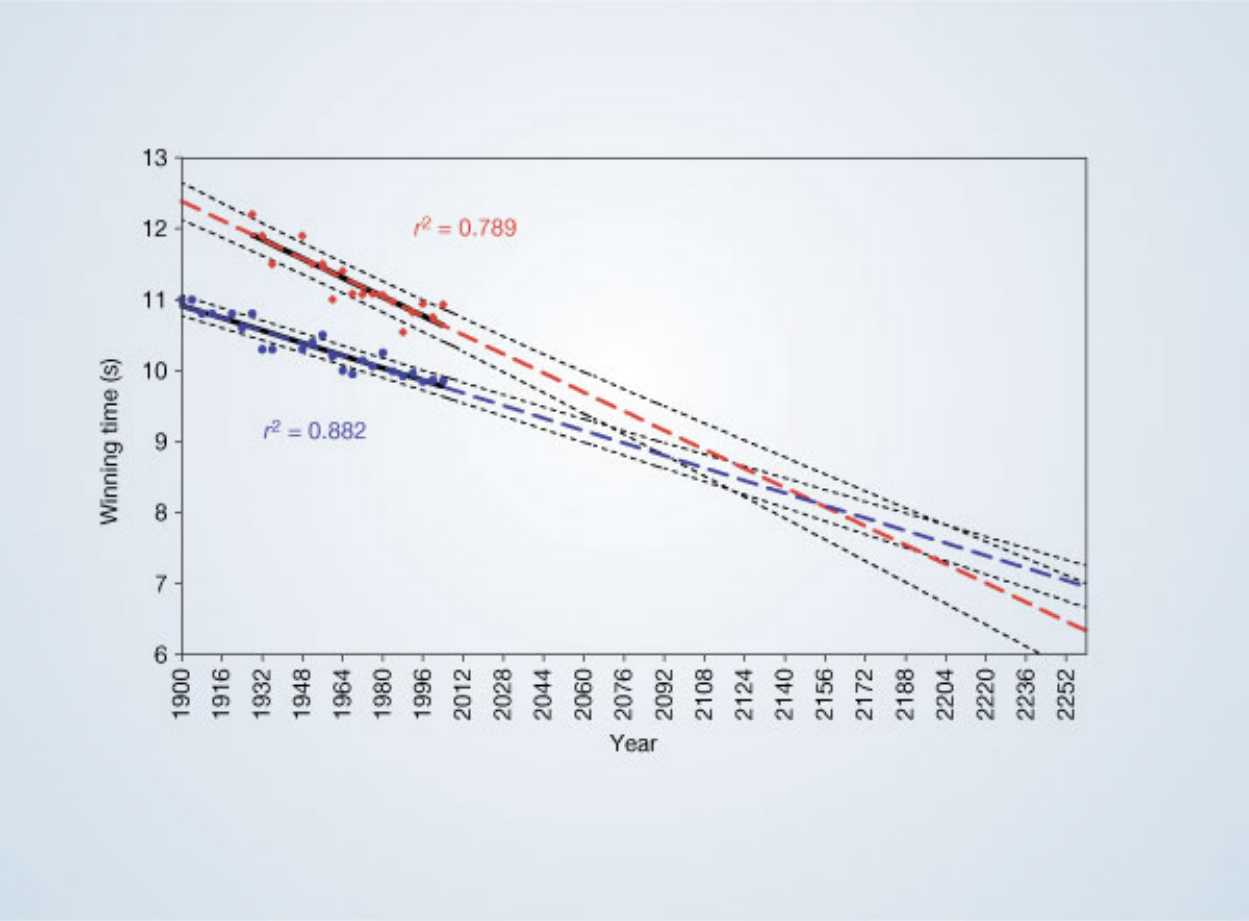
$$Y_t = \frac{1}{2 * k + 1} \sum_{j=-k}^{k} y_{t+j}$$

```
library(forecast)
plot(ts1Train)
lines(ma(ts1Train,order=3),col="red")
```



suavizado exponencial

$$\hat{y}_{t+1} = \alpha y_t + (1-\alpha)\hat{y}_{t-1}$$




```
#ets1 <- ets(ts1Train,model="MMM")
#fcast <- forecast(ets1)
#plot(fcast); lines(ts1Test,col="red")
```

obteniendo la precision

```
#accuracy(fcast,ts1Test)
```

Notas y otros recursos

- Pronóstico y predicción de series temporales es un campo completo
- Rob Hyndman Pronóstico: principios y práctica es un buen lugar para comenzar
- Precauciones
  - Tenga cuidado con las correlaciones falsas
  - Tenga cuidado con lo lejos que predice (extrapolación)
  - Tenga cuidado con las dependencias a lo largo del tiempo
- Ver paquetes de quantmod o [quandl] (http://www.quandl.com/help/packages/r) para problemas relacionados con las finanzas.

prediccion sin supervision

Ideas claves

- A veces no conoce las etiquetas para la predicción
- Para construir un predictor

  - Crear clústeres
  - nombrar lis clústeres
  - Construir predictor para clústeres
- En un nuevo conjunto de datos
  - Predecir clústeres

Ejemplo de iris ignorando las etiquetas de las especies

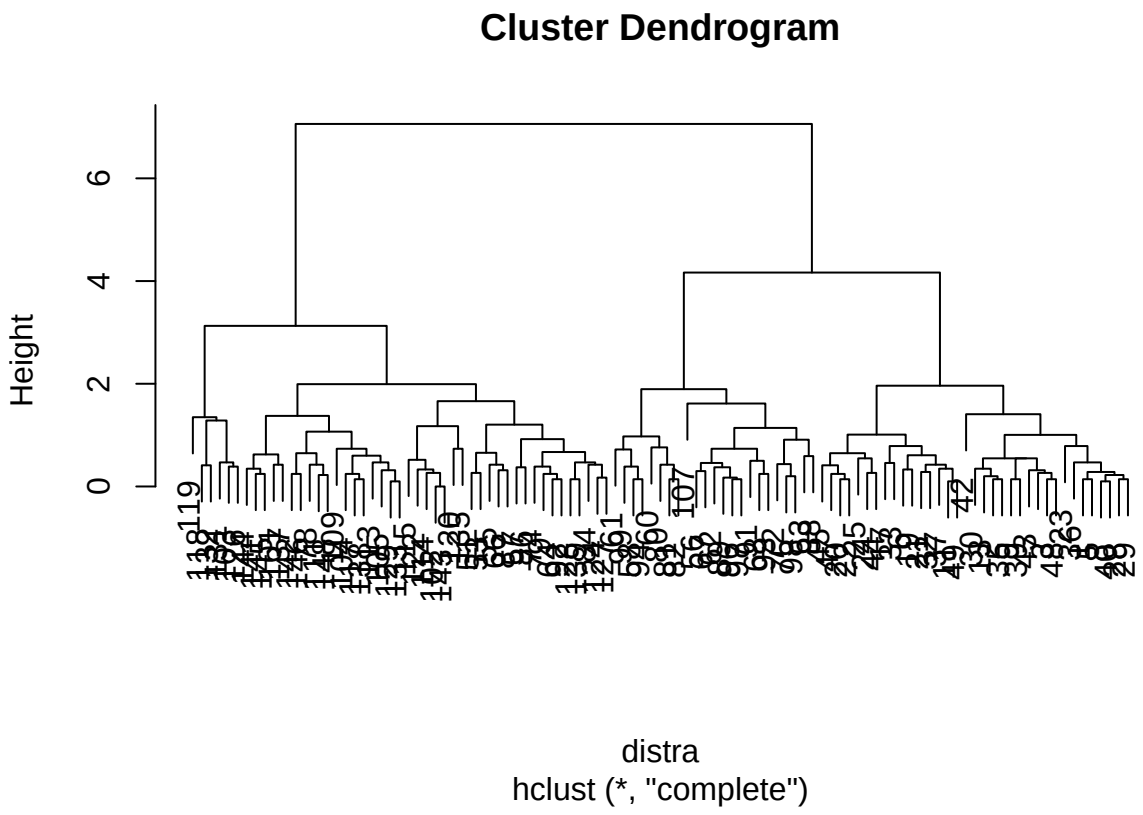
```
data(iris); library(ggplot2)
inTrain <- createDataPartition(y=iris$Species,
                               p=0.7, list=FALSE)
training <- iris[inTrain,]
testing <- iris[-inTrain,]
dim(training); dim(testing)
```

```
## [1] 105   5
```

```
## [1] 45  5
```

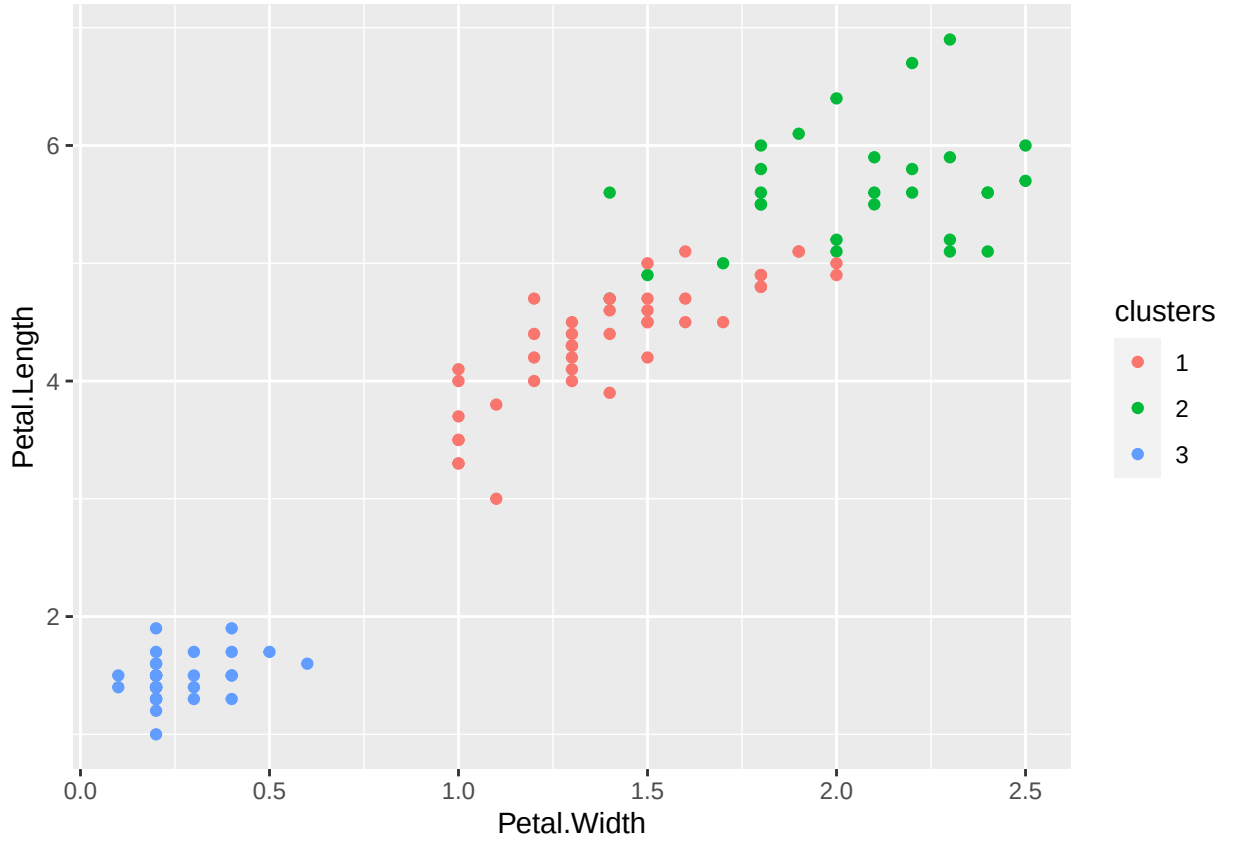
en el siguiente dendograma vemos que con una distancia de 3.5 podemos crear 3 grupos

```
distra<-dist(subset(training,select=-c(Species)))
hc<-hclust(distra)
plot(hc)
```



ahora creemos 3 grpos con k-means

```
kMeans1 <- kmeans(subset(training,select=-c(Species)),centers=3)
training$clusters <- as.factor(kMeans1$cluster)
qplot(Petal.Width,Petal.Length,colour=clusters,data=training)
```



comparando con las etiquetas reales

```
table(kMeans1$cluster,training$Species)
```

```
##
##     setosa versicolor virginica
##   1      0         32        10
##   2      0          3        25
##   3     35          0         0
```

construyendo un predictor

```
modFit <- train(clusters ~.,data=subset(training,select=-c(Species)),method="rpart")
table(predict(modFit,training),training$Species)
```

```
##
##     setosa versicolor virginica
##   1      0         35        13
##   2      0          0        22
##   3     35          0         0
```

aplicando en el conjunto de prueba

```
testClusterPred <- predict(modFit,testing)
table(testClusterPred ,testing$Species)
```

```
##
## testClusterPred setosa versicolor virginica
```

```
##                1       0          15          3
##                2       0           0         12
##                3      15           0          0
```

Notas y lectura adicional

- La función cl_predict en el paquete clue proporciona una funcionalidad similar
- ¡Tenga cuidado con la interpretación excesiva de los grupos!
- Este es un enfoque básico para motores de recomendación
- Elementos del aprendizaje estadístico
- Introducción al aprendizaje estadístico