Prediccion de subida/bajada del mercado a travez de rendimientos bursatiles

Santiago Eliges

Introducción

En este análisis, exploraremos un conjunto de datos financieros que registra los rendimientos porcentuales diarios de varios activos a lo largo de cinco años (2001-2005). El objetivo principal es predecir el comportamiento futuro del mercado utilizando modelos de clasificación basados en variables históricas. Los datos incluyen rendimientos de los cinco días de negociación previos a cada observación, además de una variable de respuesta que indica el movimiento del mercado (subida o bajada).

Este proyecto cubre: - Análisis exploratorio de datos (EDA). - Limpieza y preprocesamiento de datos. - Entrenamiento y evaluación de varios modelos de clasificación. - Comparación de métricas de rendimiento para seleccionar el mejor modelo.

Herramientas y tecnologías utilizadas: - R - Librerías: `caret`, `ggplot2`, `dplyr`

Entendiendo el dataset

Dentro del archivo 'stocks.txt' se encuentran los datos divididos por fechas desde el año 2001 hasta el 2005 donde para cada dia, se han registrado los rendimientos porcentuales de cada uno de los cinco días de negociación anteriores, disponibles en las variables lag1, lag2, lag3, lag4, lag5. En la variable Volumen se registra la cantidad de acciones negociadas en el día anterior en miles de millones. Por ultimo, cuenta con una variable cualitativa que indica si el mercado esta subiendo o bajando ese día, esta será la variable que se intentará predecir con los distintos modelos.

Analisis exploratorio de los datos

Antes de comenzar a realizar modelos predictivos de clasificación con los datos, me parece que es importante analizar la correlación entre las variables predictoras ya que el modelo de clasificación Bayes Naive usa fuertemente el supuesto de independencia de las covariables y además podría resultar inutil -o incluso generar un efecto de sobreajuste- el uso de dos variables altamente correlacionadas en el modelo de regresión logistica.

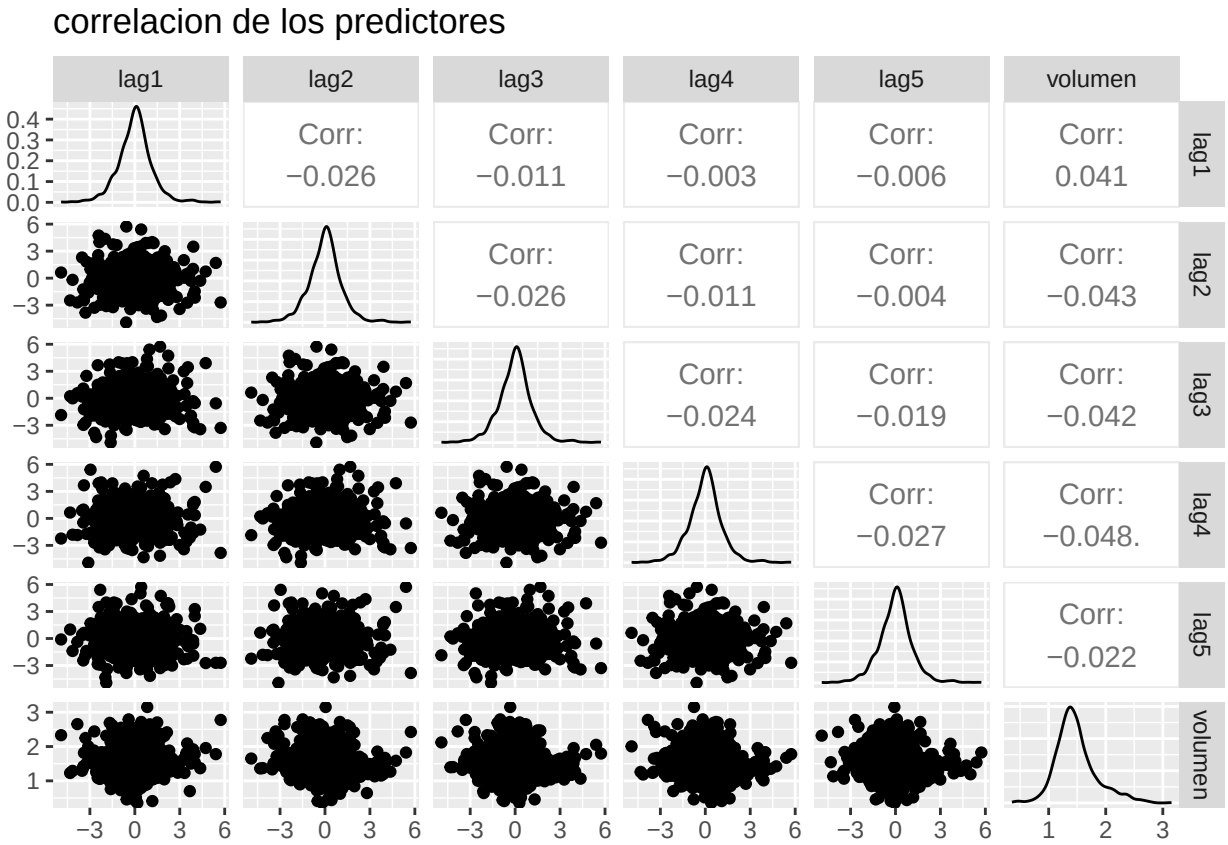
```
datos <-read.table('./stocks.txt', header = TRUE)

library(GGally)
```

```
## Loading required package: ggplot2
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
predictores <- datos[,2:7]
ggpairs(predictores, title = "correlacion de los predictores")
```



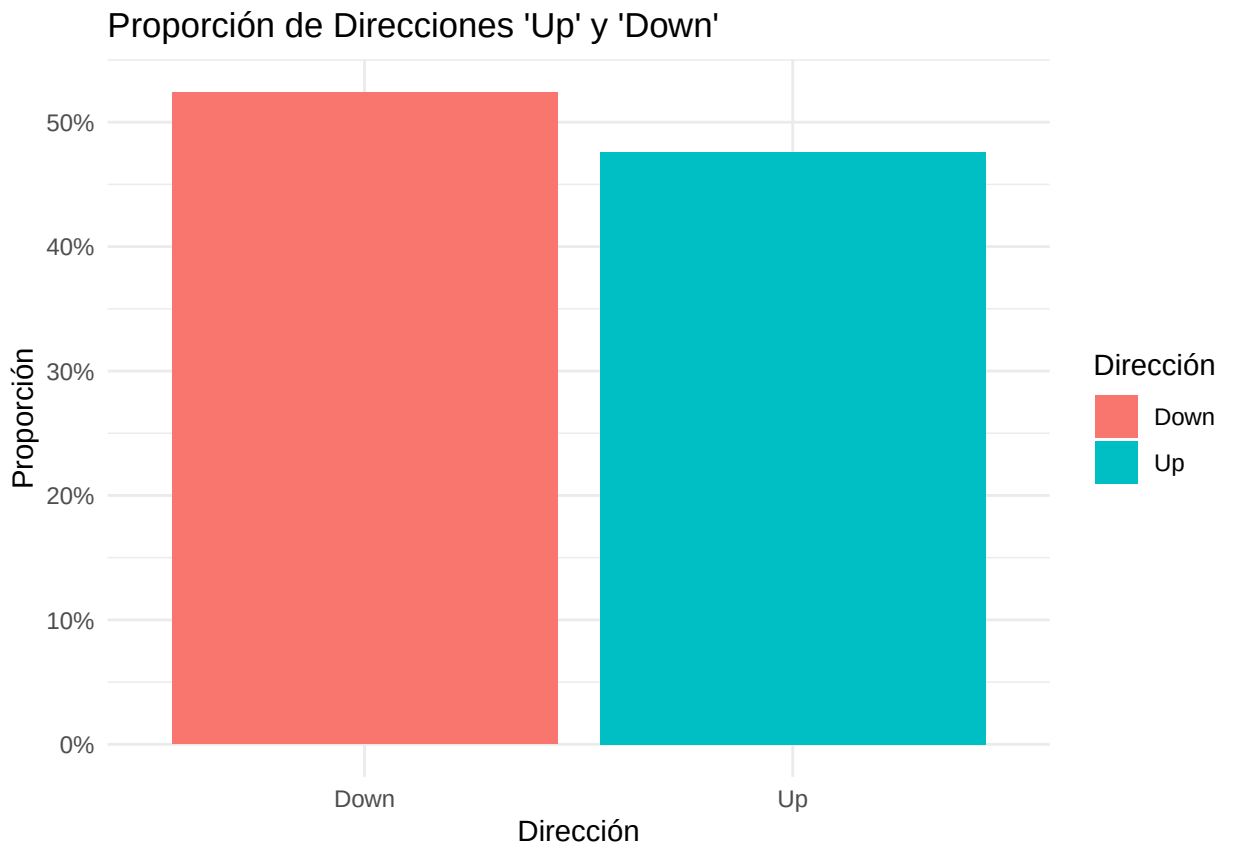
Observamos que los valores de correlación entre las variables son muy bajos y además las nubes de puntos entre las covariabes parecen formar clusters dispersos y centrados, por lo que a priori no parece necesario descartar el supuesto de independencia de las covariables (ACLARACIÓN: Se que no mostrar correlación no implica independencia, pero si hay correlación ya se podría descartar este supuesto).

También considero importante conocer la cantidad de datos de cada clase ya que un desbalance tendrá que ser tenido en cuenta a la hora de hablar de accuaracy, especificidad y sensibildiad.

```
library(ggplot2)

proporciones <- as.data.frame(table(datos$direc))
colnames(proporciones) <- c("Dirección", "Frecuencia")
proporciones$Proporción <- proporciones$Frecuencia / sum(proporciones$Frecuencia)

ggplot(proporciones, aes(x = Dirección, y = Proporción, fill = Dirección)) +
  geom_bar(stat = "identity") +
  labs(title = "Proporción de Direcciones 'Up' y 'Down'",
       x = "Dirección",
       y = "Proporción") +
  theme_minimal() +
  scale_y_continuous(labels = scales::percent)
```



Separación en datos de test y training

Para la separación de datos, considero razonable tomar a los de test como los datos de un año entero para poder capturar cualquier posible dependencia entre el tiempo y la variable respuesta (por ejemplo, podría existir una tendencia a la baja del mercado en invierno).

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
datos <- datos %>%
  mutate(direc = ifelse(direc == 'Up', 1, 0))
datos_train <- datos %>% filter(anio < 2005)
datos_2005 <- datos %>% filter(anio == 2005)
```

modelo RL (Regresión Logistica)

```
rl.modelo <- glm(direc ~ lag1 + lag2 + lag3 + lag4 + lag5 + volumen, data = datos_train)
predicciones_test <- ifelse(predict(rl.modelo, newdata = datos_2005) >= 0.5, 1, 0)
rl.accuracy <- sum(ifelse(datos_2005$direc == predicciones_test , 1, 0))/length(predicciones_test)
print(paste('Accuaracy: ', rl.accuracy))
```

```
## [1] "Accuaracy:  0.698412698412698"
```

```
matriz_confus <- as.matrix(table(Prediccion = predicciones_test, Real = datos_2005$direc))
sencibilidad <- matriz_confus[2,2]/(matriz_confus[1,2] + matriz_confus[2,2])
print(paste('sencibilidad: ', sencibilidad))
```

```
## [1] "sencibilidad:  0.947368421052632"
```

```
especificidad <- matriz_confus[1,1]/(matriz_confus[1,1]+matriz_confus[2,1])
print(paste('especificidad: ', especificidad))
```

```
## [1] "especificidad:  0.32"
```

```
summary(rl.modelo)
```

```
##
## Call:
## glm(formula = direc ~ lag1 + lag2 + lag3 + lag4 + lag5 + volumen,
##     data = datos_train)
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.0539004  0.0820257   0.657    0.511
## lag1        -0.0064129  0.0127126  -0.504    0.614
## lag2         0.0056018  0.0127224   0.440    0.660
## lag3         0.0046665  0.0127006   0.367    0.713
## lag4         0.0005936  0.0127110   0.047    0.963
## lag5         0.0011694  0.0125721   0.093    0.926
## volumen      0.2851995  0.0588931   4.843 1.49e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.2427946)
##
##     Null deviance: 246.36  on 997  degrees of freedom
## Residual deviance: 240.61  on 991  degrees of freedom
## AIC: 1428.5
##
## Number of Fisher Scoring iterations: 2
```

Observamos que los lags no parecen ser significativos en el modelo de regresión logística, por lo que parece razonable emplear un modelo de regresión regularizada, como una penalización del tipo elastic net, para evitar el sobreajuste debido a variables insignificantes.

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-8
```

```
rl.regularizado <- cv.glmnet(as.matrix(datos_train[,2:7]), datos_train$direc)
rl.regularizado.predicciones <- predict(rl.regularizado, newx = as.matrix(datos_2005[,2:7]), s = rl.regu
rl.regularizado.predicciones <- ifelse(rl.regularizado.predicciones >= 0.5,1,0)
rl.regularizado.difenpredicciones <- ifelse(rl.regularizado.predicciones==datos_2005$direc, 1, 0)


rl.accuracy <- sum(rl.regularizado.difenpredicciones)/length(rl.regularizado.difenpredicciones)
print(paste('Accuaracy: ', rl.accuracy))
```

```
## [1] "Accuaracy:  0.722222222222222"
```

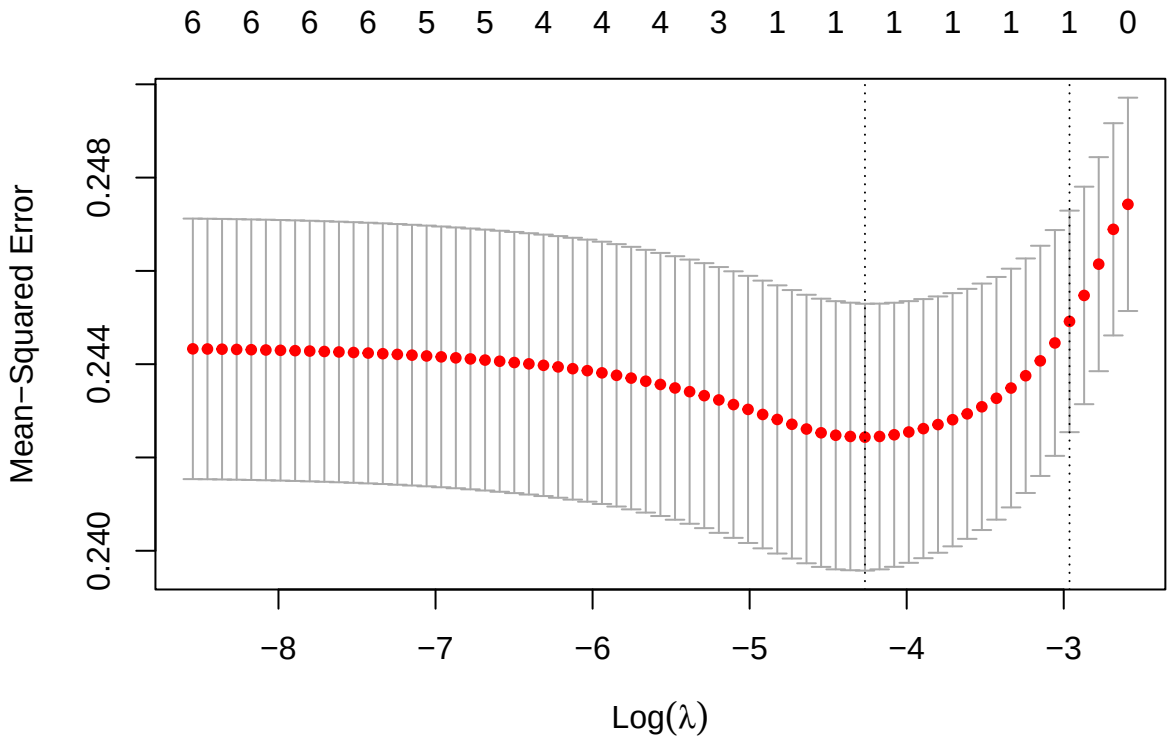
```
matriz_confus <- as.matrix(table(Prediccion = rl.regularizado.predicciones, Real = datos_2005$direc))
sencibilidad <- matriz_confus[2,2]/(matriz_confus[1,2] + matriz_confus[2,2])
print(paste('sencibilidad: ', sencibilidad))
```

```
## [1] "sencibilidad:  0.927631578947368"
```

```
especificidad <- matriz_confus[1,1]/(matriz_confus[1,1]+matriz_confus[2,1])
print(paste('especificidad: ', especificidad))
```

```
## [1] "especificidad:  0.41"
```

```
plot(rl.regularizado)
```



Es notable la mejora en las métricas al usar un modelo de regresión logística con penalización de elastic net. Esto sugiere que algunas de las covariables no son representativas en el modelo logístico, dando lugar al fenómeno de sobreajuste. Se observa un aumento en la especificidad del modelo con elastic net y validación cruzada en comparación con el original, aunque la sensibilidad disminuye ligeramente. Esto indica que el modelo original tiende a predecir más de lo deseado que los datos pertenecen a la clase 'up', fenómeno que se

intenta mitigar en el modelo regularizado.

modelo LDA

```
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
lda_model <- lda(direc ~ lag1 + lag2 + lag3 + lag4 + lag5 + volumen, data = datos_train)

lda.predicciones_test <- predict(lda_model, newdata =  datos_2005)
lda.accuracy <- sum(ifelse(datos_2005$direc == lda.predicciones_test$class,1,0))/length(datos_2005$direc
print(paste('Accuaracy: ', lda.accuracy))
```

```
## [1] "Accuaracy:  0.698412698412698"
```

```
lda.matriz_confus <- as.matrix(table(Prediccion = lda.predicciones_test$class, Real = datos_2005$direc)
lda.sencibilidad <- lda.matriz_confus[2,2]/(lda.matriz_confus[1,2] + lda.matriz_confus[2,2])
print(paste('sencibilidad: ', lda.sencibilidad))
```

```
## [1] "sencibilidad:  0.947368421052632"
```

```
lda.especificidad <- lda.matriz_confus[1,1]/(lda.matriz_confus[1,1]+lda.matriz_confus[2,1])
print(paste('especificidad: ', lda.especificidad))
```

```
## [1] "especificidad:  0.32"
```

modelo QDA

```
qda_model <- qda(direc ~ lag1 + lag2 + lag3 + lag4 + lag5 + volumen, data = datos_train)

qda.predicciones_test <- predict(qda_model, newdata =  datos_2005)
qda.accuracy <- sum(ifelse(datos_2005$direc == qda.predicciones_test$class,1,0))/length(datos_2005$direc
print(paste('Accuaracy: ', qda.accuracy))
```

```
## [1] "Accuaracy:  0.821428571428571"
```

```
qda.matriz_confus <- as.matrix(table(Prediccion = qda.predicciones_test$class, Real = datos_2005$direc)
qda.sencibilidad <- qda.matriz_confus[2,2]/(qda.matriz_confus[1,2] + qda.matriz_confus[2,2])
print(paste('sencibilidad: ', qda.sencibilidad))
```

```
## [1] "sencibilidad:  0.703947368421053"
```

```
qda.especificidad <- qda.matriz_confus[1,1]/(qda.matriz_confus[1,1]+qda.matriz_confus[2,1])
print(paste('especificidad: ', qda.especificidad))
```

```
## [1] "especificidad:  1"
```

modelo Bayes Naive

```
library(e1071)
nb_model <- naiveBayes(direc ~ lag1 + lag2 + lag3 + lag4 + lag5 + volumen, data = datos_train)

nb.predicciones_test <- predict(nb_model, newdata =  datos_2005)
nb.accuracy <- sum(ifelse(datos_2005$direc == nb.predicciones_test,1,0))/length(datos_2005$direc)
print(paste('Accuaracy: ', nb.accuracy))
```

```
## [1] "Accuaracy:  0.817460317460317"
```

```
nb.matriz_confus <- as.matrix(table(Prediccion = nb.predicciones_test, Real = datos_2005$direc))
nb.sencibilidad <- nb.matriz_confus[2,2]/(nb.matriz_confus[1,2] + nb.matriz_confus[2,2])
print(paste('sencibilidad: ', nb.sencibilidad))
```

```
## [1] "sencibilidad:  0.703947368421053"
```

```
nb.especificidad <- nb.matriz_confus[1,1]/(nb.matriz_confus[1,1]+nb.matriz_confus[2,1])
print(paste('especificidad: ', nb.especificidad))
```

```
## [1] "especificidad:  0.99"
```

Comparación

Dado que el objetivo es predecir la dirección del mercado, tiene sentido utilizar los modelos con mayor precisión, que podrían ser tanto Bayes Naive como QDA, considerando que ambos presentan valores similares y relativamente altos de especificidad y sensibilidad. Sin embargo, pueden darse casos específicos en los que se obtienen altos valores de precisión y, a pesar de ello, los modelos predicen mal. Por ejemplo, si se cuenta con muchos datos de una clase específica, un modelo que siempre predice que será de esa clase tendrá una alta precisión, pero en realidad será un modelo deficiente.

En este caso, QDA muestra una precisión y especificidad ligeramente superiores a las de Bayes Naive, por lo que sugeriría considerar QDA como el modelo más adecuado.

Es interesante observar que existe un pequeño desbalance en los datos, ya que hay más registros del tipo 'down' que del tipo 'up'. Este desbalance podría influir en la diferencia de precisión y sensibilidad entre los modelos. También es notable que, por esta razón, los modelos de regresión lineal y LDA presenten valores tan bajos de precisión, ya que tienden a clasificar un número excesivo de positivos, lo que se convierte en el principal factor de error.

```
library(ROCR)
rl.densidad_test<-predict(rl.regularizado, newx = as.matrix(datos_2005[,2:7]), s = rl.regularizado$lambo
rl.pred <- prediction(rl.densidad_test, datos_2005$direc)
rl.perf <- performance(rl.pred, "tpr", "fpr")
rl.auc <- as.numeric(performance(rl.pred, "auc")@y.values)
rl.auc
```

```
## [1] 0.9034868
```

```
lda.densidad_test <- predict(lda_model, newdata =  datos_2005)$posterior
lda.pred <- prediction(lda.densidad_test[,2], datos_2005$direc)
lda.perf <- performance(lda.pred, "tpr", "fpr")
lda.auc <- as.numeric(performance(lda.pred, "auc")@y.values)
lda.auc
```

```
## [1] 0.9006579
```

```
qda.densidad_test <- predict(qda_model, newdata =  datos_2005)$posterior
qda.pred <- prediction(qda.densidad_test[,2], datos_2005$direc)
```

```
qda.perf <- performance(qda.pred, "tpr", "fpr")
qda.auc <- as.numeric(performance(qda.pred, "auc")@y.values)
qda.auc
```

```
## [1] 0.9936184
```

```
nb.densidad_test <- predict(nb_model, newdata =  datos_2005, type = 'raw')
nb.pred <- prediction(nb.densidad_test[,2], datos_2005$direc)
nb.perf <- performance(nb.pred, "tpr", "fpr")
nb.auc <- as.numeric(performance(nb.pred, "auc")@y.values)
nb.auc
```

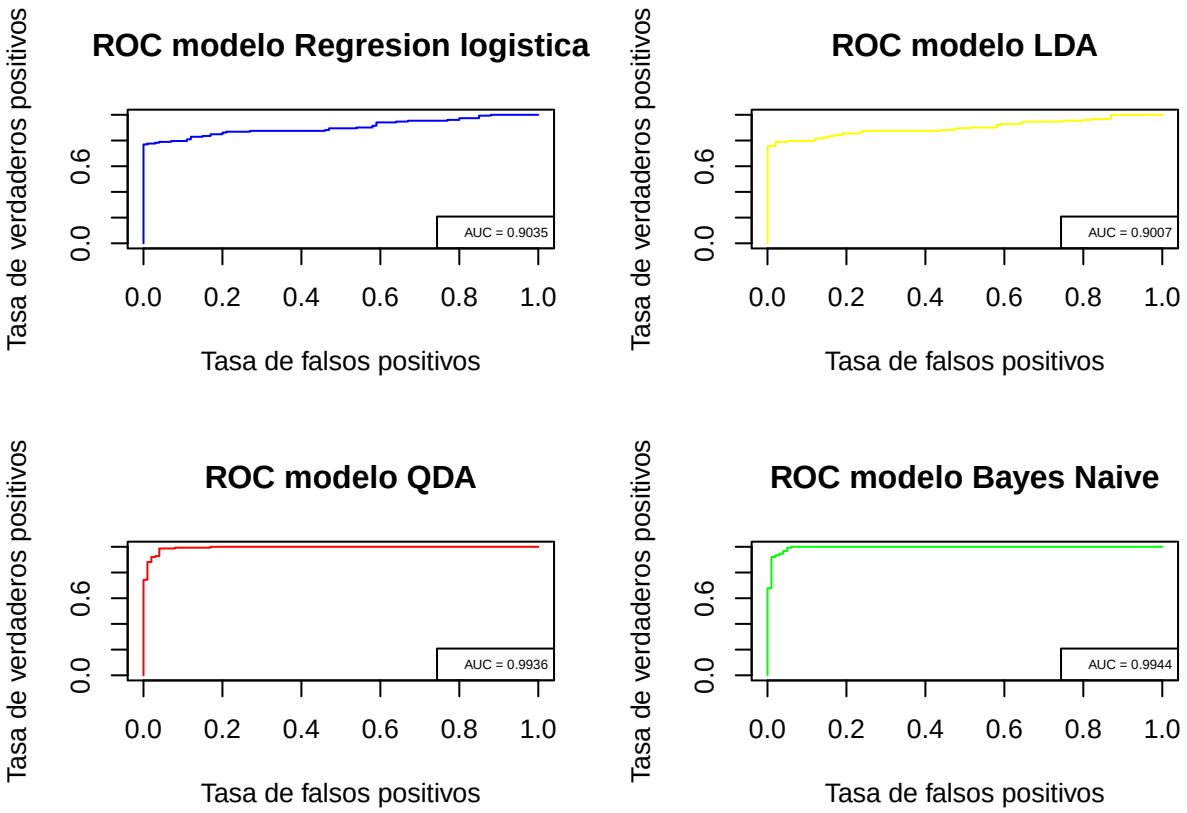
```
## [1] 0.9944079
```

```
par(mfrow=c(2,2))
plot(rl.perf,
     col = 'blue',
     main = "ROC modelo Regresion logistica",
     xlab="Tasa de falsos positivos",
     ylab="Tasa de verdaderos positivos")
legend("bottomright",legend=paste(" AUC =",round(rl.auc,4)),cex=0.5)

plot(lda.perf,
     col = 'yellow',
     main = "ROC modelo LDA",
     xlab="Tasa de falsos positivos",
     ylab="Tasa de verdaderos positivos")
legend("bottomright",legend=paste(" AUC =",round(lda.auc,4)),cex=0.5)

plot(qda.perf,
     col = 'red',
     main = "ROC modelo QDA",
     xlab="Tasa de falsos positivos",
     ylab="Tasa de verdaderos positivos")
legend("bottomright",legend=paste(" AUC =",round(qda.auc,4)),cex=0.5)

plot(nb.perf,
     col = 'green',
     main = "ROC modelo Bayes Naive",
     xlab="Tasa de falsos positivos",
     ylab="Tasa de verdaderos positivos")
legend("bottomright",legend=paste(" AUC =",round(nb.auc,4)),cex=0.5)
```



ado el análisis de las curvas ROC, parece que el modelo más adecuado es Bayes Naive, que presenta un AUC más alto que los demás. Esto sugiere que es el modelo que mejor logra discriminar entre la densidad de los positivos y los negativos (en nuestro caso, los positivos son los 'up' y los negativos son los 'down').

He cambiado la sugerencia del modelo en comparación con el punto anterior, ya que se dio el caso en el que, para un umbral específico (entendido como el valor de probabilidad que debe superarse para clasificar un dato como 'up'), el modelo de QDA obtuvo un puntaje de especificidad mayor que el de Bayes Naive. Sin embargo, existe un umbral máximo para el cual el modelo de Bayes Naive resulta ser mejor que el modelo de QDA (para cualquier umbral de QDA).

De todas formas, en la práctica considero que ambos modelos logran clasificar de manera esencialmente similar en los datos, siendo prácticamente indistinguible la mejora en rendimiento.

Conclusiones

Finalmente, comparando el rendimiento de los modelos de regresión logistica y LDA con los de QDA y Bayes Naive, aprece ser el modelado lineal de la dependecia de la variable respuesta (dirección) con las variables predictoras no es el más acertado. Siguiendo esta linea de razonamineto, consideraría usar metodos de clasificación no lineales como QDA, Bayes Naive y KNN para la predicción de los datos.

La mejora de rendimiento al emplear QDA frente a LDA puede implicar que la varianza del predictor NO es igual en todas las clases de la variable respuesta.