UNIVERSIDAD DE LOS ANDES
Departamento de Ingeniería Industrial – Segundo Semestre de 2017
Statistical Learning for Data Analytics – IIND 4123
Profesor: Carlos Valencia

PROYECTO FINAL

Equipo en Kaggle: Los Esclavos Analíticos

Laura Quiñones	201327381
Julián Uribe Giraldo	201225538
Paula Rodríguez Díaz	201327494

El presente proyecto pretende **predecir el nivel de crimen en unidades residenciales de una región** de Estados Unidos a partir de información demográfica, socioeconómica, gasto en seguridad, presencia del estado, entre otros. Con el fin de llevar a cabo esta predicción, se utilizarán dos estrategias: predecir directamente el índice de criminalidad como un problema de regresión y predecir si el sector es peligroso o no como un problema de clasificación, a partir de datos obtenidos del *Federal Bureau of Investigation* que hace parte del Departamento de Justicia de los Estados Unidos. A continuación, se presenta el procedimiento realizado para cada una de las estrategias.

Extracción de Variables y Limpieza de Datos

Antes de estimar funciones, se consideró pertinente evaluar la validez de algunos datos con el fin de decidir qué información no era útil. Para esto, se realizaron procedimientos para tratar los *datos faltantes*, la *correlación de variables* y los *datos atípicos*.

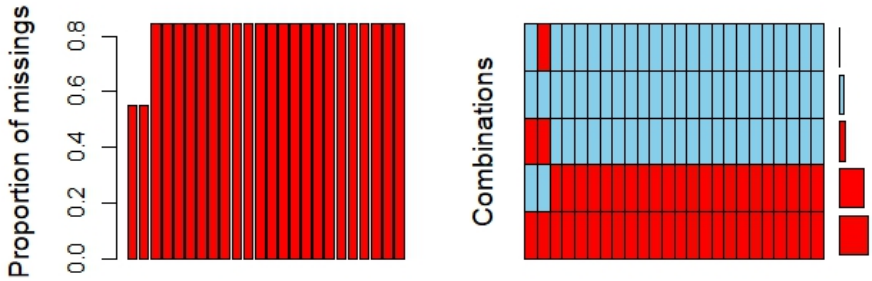
i. Datos faltantes

En cuanto al primer problema, si bien en principio se pensó en eliminar todos los datos faltantes, esto habría implicado reducir el número de observaciones de 1506 a 103. Por lo tanto, se optó por calcular el número de datos faltantes por variable y por observación con el fin de eliminar aquellas observaciones y variables que no contaran con un número significativo de datos. Sabiendo que contábamos con 24 variables cuya proporción de faltantes era mayor a 0.8, se optó por eliminarlas. Una vez realizado este procedimiento, se redujo la base de datos a 1506 observaciones y 105 variables, quedando con cero faltantes dentro de la misma. Cabe resaltar que con este procedimiento no se eliminaron observaciones e igual se solucionó el problema de los datos faltantes.

ii. Variables altamente correlacionadas

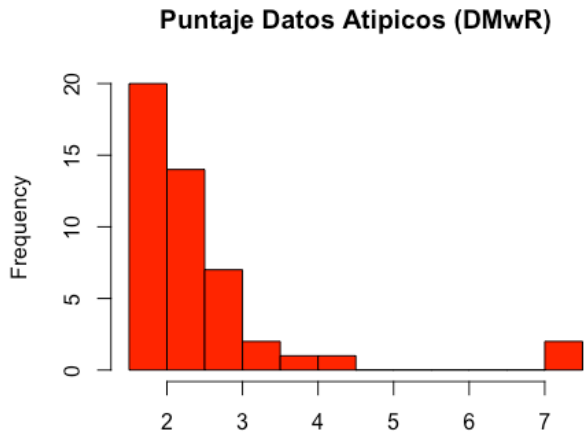
Posteriormente, en busca de eliminar las variables redundantes, se tomaron tan solo las variables numéricas – se separaron de las categóricas – y se calculó su coeficiente de correlación; aquellas que presentaban un valor cercano a 1 o a menos 1 fueron eliminadas, quedando con 66 variables. Finalmente, queriendo obtener una visualización de los datos, se realizaron

histogramas de las variables y aquellas que evidenciaban un sesgo significativo fueron corregidas utilizando funciones logarítmicas. Cabe resaltar que este procedimiento de limpieza fue realizado tanto para el grupo de *train* como para el de *test*, que se dividieron en una proporción 3/4 para el primero y 1/4 para el segundo grupo. Este procedimiento se muestra a continuación:



iii. Datos Atípicos

Para la detección de datos atípicos se usó el paquete 'DMwR' de R. Utilizando la función lofactor se asigna un puntaje a cada observación donde puntajes mayores corresponden a los datos que se consideran atípicos. Inicialmente consideramos que los datos atípicos era aquellos con un puntaje mayor o igual a 1.6, lo cuál correspondía a 77 datos. Al correr un modelo de regresión lineal nos dimos cuenta que el MSE era mejor si se consideraba un puntaje de 1.7 o mayor pues al escogerlo de 1.6 o mayor se perdía mucha información. A continuación se presenta un histograma que muestra los puntos que obtuvieron puntajes mayores o iguales a 1.7.



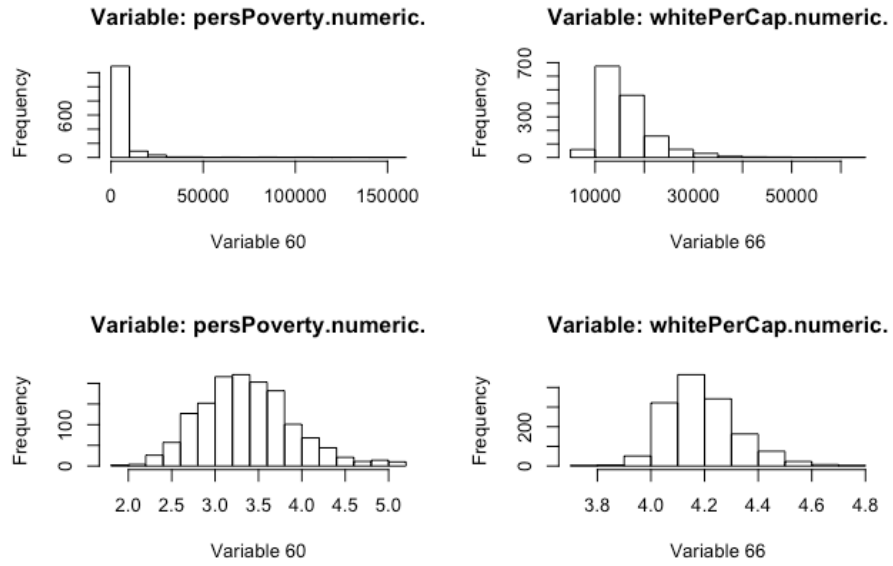
Finalmente determinamos que los datos atípicos eran aquellos con un puntaje mayor o igual 1.7, lo cuál correspondía a **47 datos atípicos**. Por lo tanto, nuestra base de entrenamiento queda con **1459 observaciones**.

Transformación de Variables

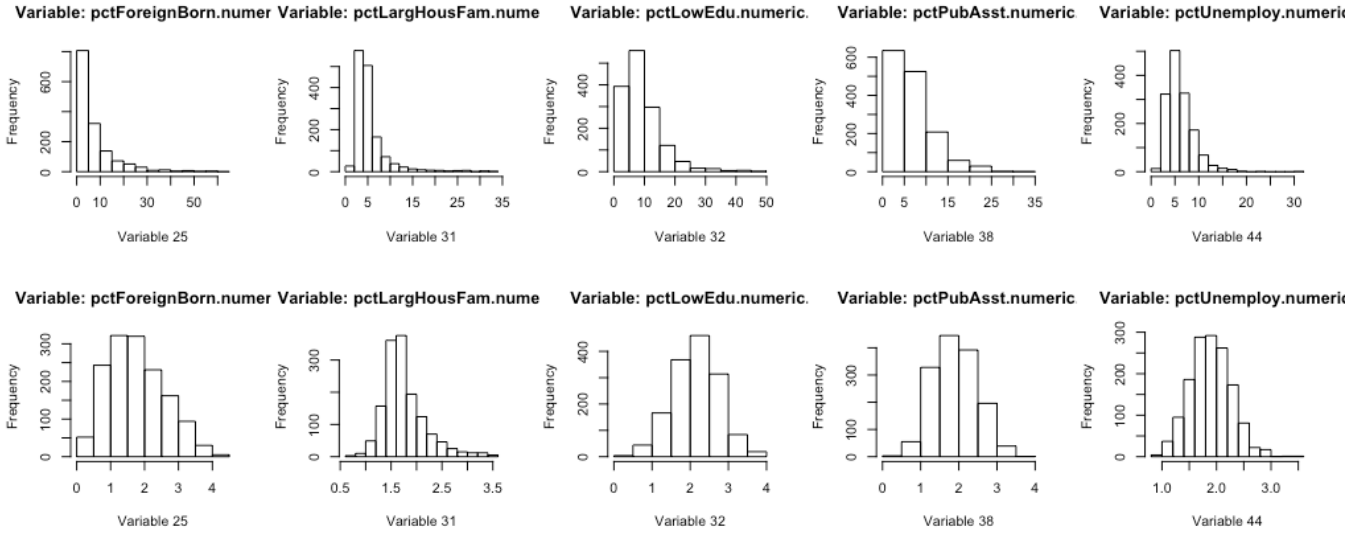
Es bastante conocido que en algunos casos los modelos de **regresión** tienen mejores desempeños cuando la distribución de las variables se acerca a normalidad. Es decir, queremos que los histogramas de las variables sean lo más simétricos posibles. Para esto modificamos algunas

variables aplicando *funciones logarítmicas*. Algunas de las variables modificadas se pueden ver a continuación. En la primera fila se pueden ver los histogramas de las variables originales y en la segunda fila los histogramas de las variables transformadas.

Transformación de variables aplicando log10



Transformación de variables aplicando log (x+1)



Es importante resaltar que la transformación de variables no implica una mejora de todos los modelos de regresión. Por lo tanto, los modelos de regresión presentados más adelante se corrieron tanto con las modificaciones de variables como con las variables originales y así tomamos la predicción que tuviera menor error cuadrático medio.

Modelos de Regresión

A continuación se presentan los métodos empleados para llevar a cabo el problema de regresión que pretende predecir el Índice de criminalidad.

i. Regresión Lineal

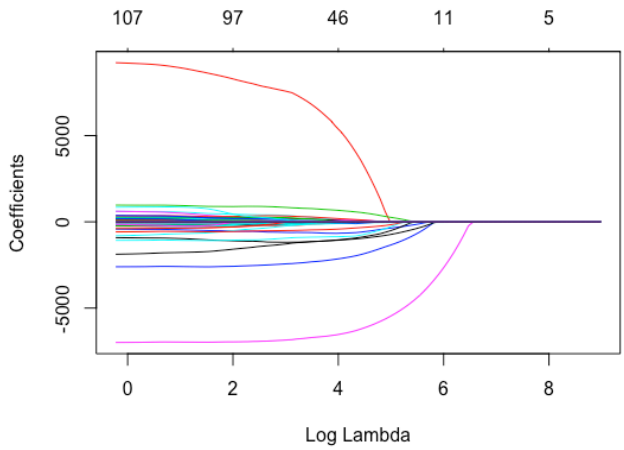
Inicialmente se llevó a cabo una regresión lineal para predecir el Índice teniendo en cuenta todos los predictores con valores reales. Para poder tener en cuenta el predictor de *State,* que originalmente es una variable categórica, se crearon variables *dummies*, una por cada estado. El modelo de regresión lineal se corrió teniendo en cuenta los estados y sin tenerlos en cuenta. En la siguiente tabla se presenta el error cuadrático medio obtenido para cada configuración hecha de este modelo.

Descripción Configuración	*MSE*
Sin modificar variables, Sin meter Estado	951,445
Sin modificar variables, Metiendo Estado	1,221,554
Modificando variables, Metiendo Estado	1,028,617

Tenemos que el mejor modelo de regresión lineal es en el cual no se modifican las variables y no se mete la variable de estado. Se esperaba que el resultado fuera contrario pues regularmente modificar variables de la forma en que se hizo mejora el desempeño de regresiones lineales. Aún así, el resultado es contundente pues precisamente este modelo es el que obtuvo el mejor puntaje final de nuestras predicciones en *Kaggle* con un score de 1,273,280.7 mientras que los demás obtuvieron puntajes (errores) significativamente mayores.

ii. Regresión lineal con penalización Lasso

Se realizaron los modelos de penalización Ridge y Lasso. El modelo Lasso tuvo menor MSE en todos los casos observados por lo tanto solo reportamos los resultados de este modelo y no de Ridge. A continuación se puede ver la gráfica correspondiente a los coeficientes estandarizados de Lasso como función de la constante de penalización.



El mejor MSE de este modelo se obtuvo sin hacer modificación de variables teniendo un **MSE de 1,032,452** en la base de prueba que partimos de la base que teníamos inicialmente. Para la base de predicción del concurso en Kaggle este fue el modelo que tuvo el segundo mejor desempeño con un puntaje (error) final de 1,460,862.57.

iii. Principal Component Regression (PCR)

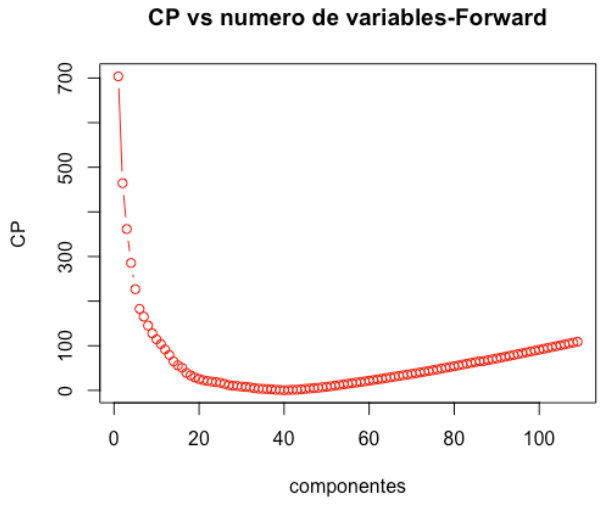
En este modelo seguimos haciendo una regresión lineal pero disminuimos la dimensionalidad del modelo utilizando componentes principales. Al igual que los casos anteriores corrimos el modelo tanto para las variables originales como las variables transformadas. Calibramos el número de componentes principales a tener en cuenta en el modelo y encontramos que la cantidad de componentes principales que minimizan el MSE son 12 y 11 para variables originales y variables transformadas respectivamente. En la siguiente tabla se pueden ver los MSE obtenidos para la base de validación que partimos de la base original.

Descripción Configuración	*MSE*
Sin modificar variables	647,470.5
Modificando variables (Log10, Log(x+1))	701,948

El MSE se reduce considerablemente en comparación con los modelos anteriores (Regresión lineal y penalización con Lasso). Sin embargo, este modelo fue no tuvo mejores resultados en *Kaggle*; es decir que su error fue mayor al predecir en la base de predicción. El puntaje final que se obtuvo fue de 1,706,611.6. Por lo tanto podemos concluir que el modelo tiene *overfitting* y por lo tanto no generaliza bien.

iv. Forward

Corrimos un modelo de regresión forward. Éste tiene mínimo error CP de Mallows cuando se consideran 40 componentes principales (variables forward). Esto mismo se puede ver en la siguiente gráfica.



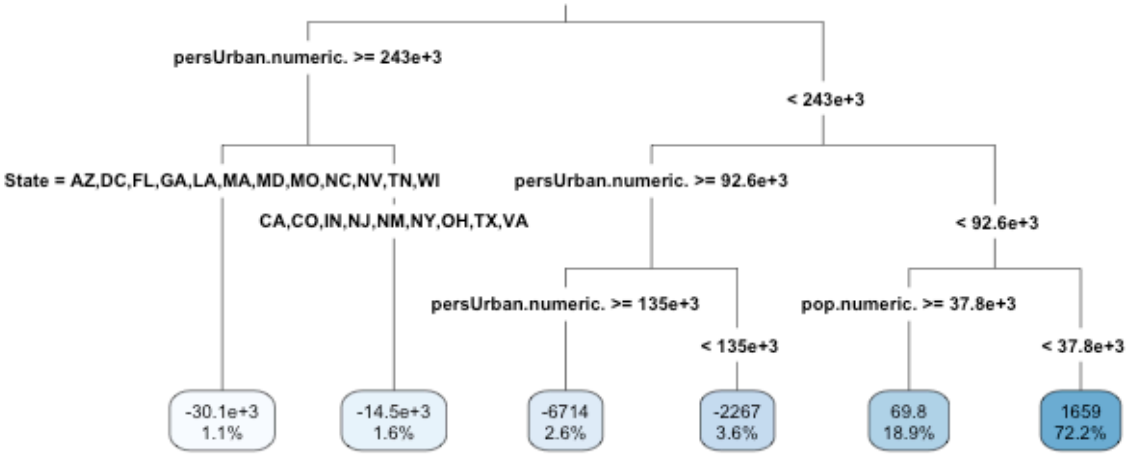
La siguiente tabla presenta el error cuadrático medio para los modelos forward implementados.

Descripción Configuración	*MSE*
Sin modificar variables	2,686,576
Modificando variables	32,468,279

Descartamos este modelo inmediatamente pues su error cuadrático medio es significativamente alto en comparación de los otros modelos ya presentados.

v. Árboles de Regresión (CART)

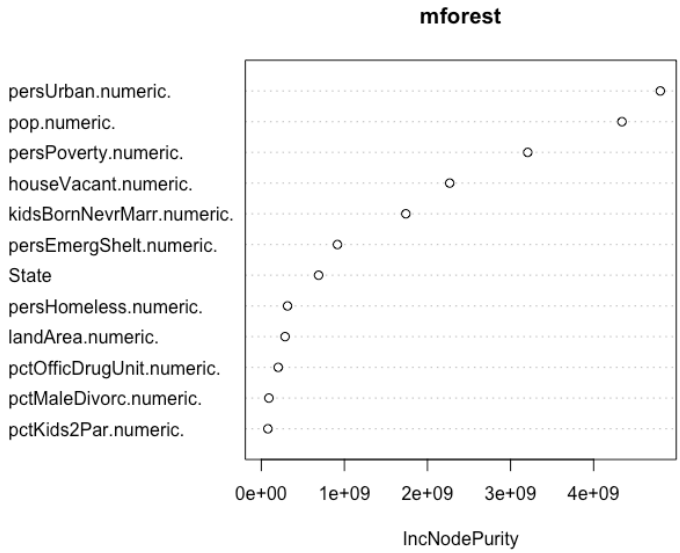
Luego de haber hecho distintos modelos lineales corrimos un árbol de regresión esperando que el MSE mejorara aún más. Utilizando los paquetes 'rpart' y 'rpart.plot' de R corrimos un árbol de regresión obteniendo así la siguiente partición.



El MSE para este modelo fue de **2,736,429**. Este error es considerablemente alto en comparación con los modelos lineales anteriormente expuestos. Esto nos da un indicio de que nuestra predicción se explica de forma lineal. Aún así intentamos mejorar este modelo de árboles corriendo un *Random Forest* a la espera de obtener mejores resultados.

vi. Random Forest

Corrimos un Random Forest con 1000 árboles y para esta cantidad calibramos el parámetro de *nodesize*. El valor de nodesize que minimiza el MSE fue 5. Para este modelo se obtuvo un mejor MSE teniendo las variables transformadas como se mostró en el inicio del documento. Según este modelo las variables con mayor importancia son las que se muestran en el siguiente gráfico.



Para este modelo el MSE sobre la base de validación partida de la base original fue de **1,246,566.** Para la predicción del concurso en Kaggle el puntaje (error) final de este modelo fue de **1,232,254.55.** Por lo tanto, con este modelo obtuvimos los mejores resultados en la predicción. Sin embargo, esta predicción fue subida a la plataforma luego de que el concurso cerrara.

vii. Conclusiones

A partir de los resultados obtenidos en los modelos implementados creemos que el Índice se puede explicar de forma casi lineal en términos de los predictores considerados. Es interesante ver que el MSE obtenido por regresión lineal es considerablemente cercano al MSE obtenido por Random Forest, siendo que éste último es un método más robusto. Así, se evidencia la cercana linealidad del modelo y se entiende el buen desempeño que obtuvieron los modelos lineales frente a un modelo más robusto como CART.

Si solo nos basáramos en el error cuadrático medio obtenido en la base de validación que partimos de la base de entrenamiento que nos entregaron para el ejercicio, habríamos escogido hacer un *Principal Component Regression* pues ya vimos que el modelo real parece explicarse de forma lineal y además con este modelo podemos tener en cuenta la información de las variables más pertinente. Sin embargo, vemos que el desempeño de este modelo no fue tan bueno en la base de predicción como lo fue en la de validación. Finalmente el mejor desempeño en la base de predicción lo tuvo el modelo de *Random Forest*, el cuál además tiene un MSE consistente en ambas bases (1,246,566 y 1,232,254.55). Lo cuál nos indica que el modelo generaliza bastante bien. Por lo tanto consideramos que el mejor modelo para modelar el Índice de criminalidad en función de las variables que tenemos es un *Random Forest.*

Modelos de Clasificación

A continuación se presentan los métodos empleados para llevar a cabo el problema de predicción que pretende determinar si el sector es peligroso o no.

i. Regresión Logística:

Antes de presentar los resultados, es importante aclarar que este método sirve para determinar la probabilidad de que una observación en particular sea asignada a algún cluster y sea clasificada con base en el valor de un *threshold* de 0.5. Así, cuando la predicción se hace a través de un threshold, la función discriminante es lineal, es decir un hiperplano.

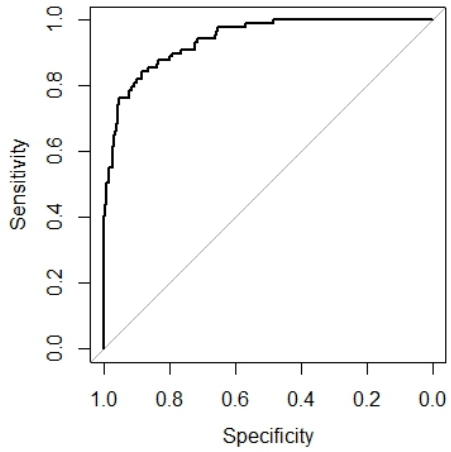
```
          Reference
Prediction   0   1
         0 259  13
         1  23  66

               Accuracy : 0.9003
                 95% CI : (0.8646, 0.9292)
    No Information Rate : 0.7812
    P-Value [Acc > NIR] : 2.058e-09

                  Kappa : 0.721
 Mcnemar's Test P-Value : 0.1336

            Sensitivity : 0.9184
            Specificity : 0.8354
         Pos Pred Value : 0.9522
         Neg Pred Value : 0.7416
             Prevalence : 0.7812
         Detection Rate : 0.7175
   Detection Prevalence : 0.7535
      Balanced Accuracy : 0.8769

       'Positive' Class : 0
```



Una vez se corrió el modelo, se puede medir el desempeño del mismo haciendo uso de la curva ROC y la matriz de confusión mostrada anteriormente. Sabiendo que el AUC se define como el área bajo la curva ROC, este toma un valor de 0.939, donde entre más cercano es a 1, mejor se considera el modelo. Simultáneamente, haciendo uso de la matriz de confusión se puede medir:

$$Sensibilidad = \frac{259}{259 + 23} = 0.9184$$

$$Especificidad = \frac{66}{66 + 13} = 0.8354$$

$$Exactitud = \frac{259 + 66}{259 + 23 + 13 + 66} = 0.9003$$

$$Error\ de\ predicción = 1 - 0.9003 = 0.0997\ (9.97\%)$$

Es importante notar que el error de predicción no es una medida muy precisa o suficiente, pues no tiene en cuenta la prevalencia (proporción de positivos) o el imbalanceo de clases, razón por

la cual es pertinente tener en cuenta el AUC también. Por esto, en todos los métodos presentados para clasificación se mostrarán ambas medidas.

ii. Linear Discriminant Analysis (LDA)

Ahora, se hace uso del método de LDA, que básicamente separa linealmente las clases proyectando al plano que mejor las divide y depende de la diferencia entre las medias y de la matriz de covarianzas. Es importante resaltar que este modelo asume normalidad y varianzas iguales entre las clases.

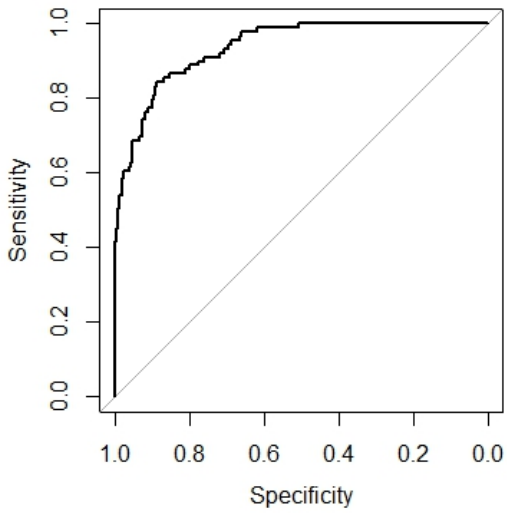
```
          Reference
Prediction   0   1
         0 259  13
         1  28  61

               Accuracy : 0.8864
                 95% CI : (0.8491, 0.9173)
    No Information Rate : 0.795
    P-Value [Acc > NIR] : 3.127e-06

                  Kappa : 0.6759
 Mcnemar's Test P-Value : 0.02878

            Sensitivity : 0.9024
            Specificity : 0.8243
         Pos Pred Value : 0.9522
         Neg Pred Value : 0.6854
             Prevalence : 0.7950
         Detection Rate : 0.7175
   Detection Prevalence : 0.7535
      Balanced Accuracy : 0.8634

       'Positive' Class : 0
```



Siguiendo el mismo procedimiento que en Regresión Logística para las medidas de desempeño del modelo, se obtiene un AUC de 0.936 y haciendo uso de la matriz de confusión se puede medir:

$$Sensibilidad = \frac{259}{259 + 28} = 0.9024$$

$$Especificidad = \frac{61}{61 + 13} = 0.8243$$

$$Exactitud = \frac{259 + 61}{259 + 28 + 13 + 61} = 0.8864$$

$$Error\ de\ predicción = 1 - 0.8864 = 0.1136\ (11.36\%)$$

iii. Quadractic Discriminant Analysis (QDA)

Igualmente, se utiliza el método QDA cuya diferencia principal con LDA es que no asume que las matrices de varianza y covarianza sean iguales, lo que implica que se deben estimar muchos más parámetros y es más flexible. Los resultados se presentan a continuación.

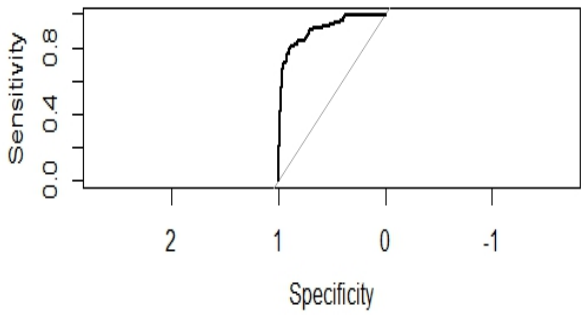
```
          Reference
Prediction   0   1
         0 252  20
         1  23  66

               Accuracy : 0.8809
                 95% CI : (0.8429, 0.9124)
    No Information Rate : 0.7618
    P-Value [Acc > NIR] : 8.51e-09

                  Kappa : 0.6757
 Mcnemar's Test P-Value : 0.7604

            Sensitivity : 0.9164
            Specificity : 0.7674
         Pos Pred Value : 0.9265
         Neg Pred Value : 0.7416
             Prevalence : 0.7618
         Detection Rate : 0.6981
   Detection Prevalence : 0.7535
      Balanced Accuracy : 0.8419

       'Positive' Class : 0
```



Siguiendo el mismo procedimiento anterior para las medidas de desempeño del modelo, se obtiene un AUC de 0.908 y haciendo uso de la matriz de confusión se puede medir:

$$Sensibilidad = \frac{252}{252 + 23} = 0.9163$$

$$Especificidad = \frac{66}{66 + 20} = 0.7674$$

$$Exactitud = \frac{252 + 66}{252 + 23 + 20 + 66} = 0.8809$$

$$Error\ de\ predicción = 1 - 0.8809 = 0.1191\ (11.91\%)$$

iv. Boosting

Este es un método que secuencialmente incrementa la flexibilidad del modelo, sumando en cada iteración un nuevo predictor que hace énfasis en los puntos de la muestra donde no se predecía bien hasta ese momento. Es decir que este nuevo predictor es de generalmente mucho sesgo y poca varianza.

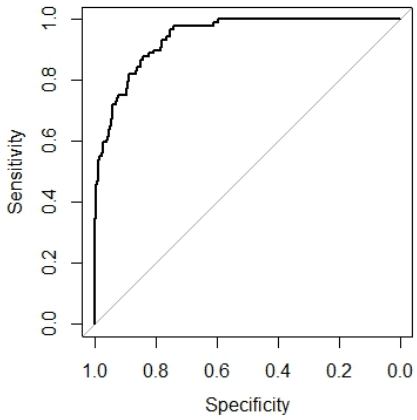
```
          Reference
Prediction   0   1
         0 256  16
         1  29  60

               Accuracy : 0.8753
                 95% CI : (0.8368, 0.9076)
    No Information Rate : 0.7895
    P-Value [Acc > NIR] : 1.555e-05

                  Kappa : 0.6471
 Mcnemar's Test P-Value : 0.07364

            Sensitivity : 0.8982
            Specificity : 0.7895
         Pos Pred Value : 0.9412
         Neg Pred Value : 0.6742
             Prevalence : 0.7895
         Detection Rate : 0.7091
   Detection Prevalence : 0.7535
      Balanced Accuracy : 0.8439

       'Positive' Class : 0
```



Siguiendo el mismo procedimiento anterior para las medidas de desempeño del modelo, se obtiene un AUC de 0.943 y haciendo uso de la matriz de confusión se puede medir:

$$Sensibilidad = \frac{256}{256 + 29} = 0.8982$$

$$Especificidad = \frac{60}{60 + 16} = 0.7894$$

$$Exactitud = \frac{256 + 60}{256 + 29 + 16 + 60} = 0.8753$$

$$Error\ de\ predicción = 1 - 0.8753 = 0.1247\ (12.47\%)$$

v. Random Forest

A diferencia de los anteriores, este método se basa en la combinación de modelos de árboles para clasificación, con el fin de disminuir su varianza individual y mejorar la predicción del modelo. Esto se hace a través de la estimación de diferentes arboles sobre datos de entrenamiento generados a partir de un proceso de *bootstrap*. Ahora bien, hasta ahora hemos descrito un método denominado *Bagging*; la mejora que *Random Forest* incorpora es que al usar un parámetro m que define la cantidad de predictores que se pueden escoger en cada división del árbol, los arboles resultan estar menos correlacionados y la combinación final, con mayor poder predictivo; es decir, en el caso en el que $m = p$, los métodos *Bagging* y *Random Forest* son los mismos. Dado que los resultados para *Random Forest* fueron mejores en términos del AUC, solo se presentan estos resultados.

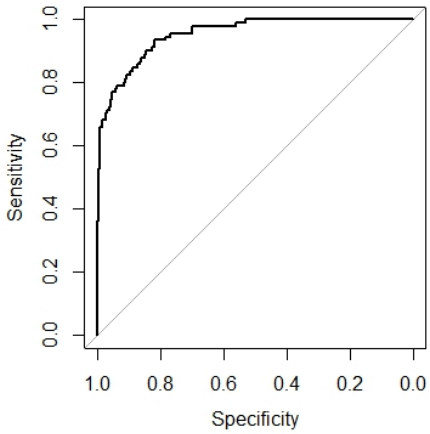
```
          Reference
Prediction   0   1
         0 261  13
         1  23  68

               Accuracy : 0.9014
                 95% CI : (0.8661, 0.93)
    No Information Rate : 0.7781
    P-Value [Acc > NIR] : 5.149e-10

                  Kappa : 0.7265
 Mcnemar's Test P-Value : 0.1336

            Sensitivity : 0.9190
            Specificity : 0.8395
         Pos Pred Value : 0.9526
         Neg Pred Value : 0.7473
             Prevalence : 0.7781
         Detection Rate : 0.7151
   Detection Prevalence : 0.7507
      Balanced Accuracy : 0.8793

       'Positive' Class : 0
```



Siguiendo el mismo procedimiento anterior para las medidas de desempeño del modelo, se obtiene un AUC de 0.941 y haciendo uso de la matriz de confusión se puede medir:

$$Sensibilidad = \frac{261}{261 + 23} = 0.919$$

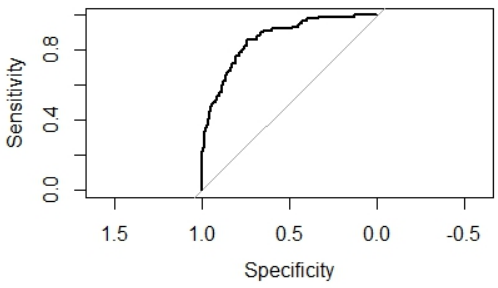
$$Especificidad = \frac{68}{68 + 13} = 0.8395$$

$$Exactitud = \frac{261 + 68}{261 + 23 + 13 + 68} = 0.9014$$

$$Error\ de\ predicción = 1 - 0.9014 = 0.0986\ (9.86\%)$$

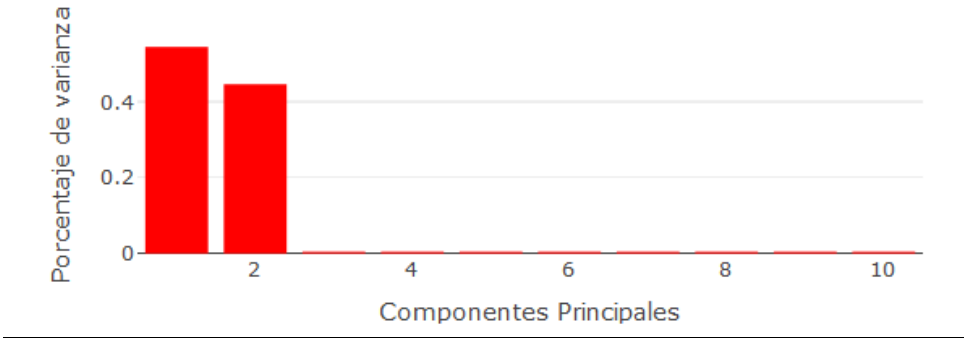
vi. Support Vector Machine

El último modelo que se utilizó consiste en crear un limite de decisión de clasificación no linear, a través de la extensión del espacio a través de *kernels* – lineal, radial o polinomial – sin aumentar la dimensionalidad. La ventaja de usar este método más allá de la flexibilidad que fronteras no lineales provee, es la computacional pues se computan los *kernels* sin necesidad de trabajar explícitamente en el espacio extendido. Para este caso en particular, el *kernel* que mejor resultados arrojó fue el radial, con un valor del AUC igual a 0.8659, una exactitud de 0.935 y la siguiente curva ROC.



vii. Regresión Logística con Componentes Principales

Si bien durante el procedimiento de limpieza de datos se eliminaron las variables cuya correlación en términos absolutos resultaba ser mayor a 0.9, se considera importante realizar un procedimiento adicional con el fin de tener la mayor cantidad de información, con la menor cantidad de variables. Para esto, se optó por transformar las variables predictivas con el objetivo de eliminar la información redundante y las variables cuya información no era relevante para predecir, a través del método *Análisis de componentes principales.* A continuación, se muestran los diferentes componentes resultantes:



Se hace evidente que los dos primeros componentes contienen el mayor porcentaje de varianza, es decir que retienen la mayor parte de la información y no existe colinealidad entre ellos. Ahora, se procede a correr los modelos de clasificación presentados inicialmente utilizando los primeros dos componentes para poder comparar los resultados y mostrar el efecto de esta transformación, en busca de una mejor predicción.

Utilizando solo los dos primeros componentes, dado que son estos los que tienen mayor variabilidad, se obtiene un AUC de 0.8707 y las siguientes medidas de desempeño:

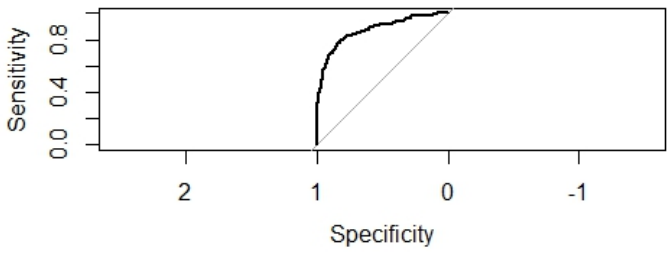
```
          Reference
Prediction   0   1
         0 264   8
         1  51  38

               Accuracy : 0.8366
                 95% CI : (0.7943, 0.8732)
    No Information Rate : 0.8726
    P-Value [Acc > NIR] : 0.9808

                  Kappa : 0.4747
 Mcnemar's Test P-Value : 4.553e-08

            Sensitivity : 0.8381
            Specificity : 0.8261
         Pos Pred Value : 0.9706
         Neg Pred Value : 0.4270
             Prevalence : 0.8726
         Detection Rate : 0.7313
   Detection Prevalence : 0.7535
      Balanced Accuracy : 0.8321

       'Positive' Class : 0
```



$$Sensibilidad = \frac{264}{264 + 51} = 0.8380$$

$$Especificidad = \frac{38}{38 + 8} = 0.8260$$

$$Exactitud = \frac{264 + 38}{264 + 51 + 8 + 38} = 0.8366$$

$$Error\ de\ predicción = 1 - 0.8366 = 0.1634\ (16.34\%)$$

Ahora, se hace evidente que utilizar tan solo los dos primeros componentes no representa una mejora en el nivel de predicción del modelo. Lo anterior, quizá debido a que al eliminar las variables cuyo nivel de correlación era alto antes de realizar el método de componentes principales, la información redundante dentro del modelo fue eliminada en su mayoría. Si bien se realizó esta comparación para todos los métodos de clasificación, en ninguno se presentó mejora y por ende no se mostrarán los resultados en este informe.

viii. Conclusión modelos de clasificación

Tras correr los modelos de clasificación presentados anteriormente, se puede recopilar sus resultados – en términos de medidas de desempeño y error – en la siguiente tabla:

Modelo	*AUC*	*Error de predicción*
Regresión Logística	0.939	9.97%
LDA	0.936	11.36%
QDA	0.908	11.91%
Boosting	0.943	12.47%
Random Forest	0.941	9.86%
SVM	0.8659	6.50%
Componentes Principales & RL	0.8707	16.34%

Ahora, teniendo en cuenta igualmente las limitaciones del error de predicción como medida de desempeño de los modelos de clasificación nombradas anteriormente y el valor del AUC arrojado para cada modelo realizado, el equipo optó por usar las predicciones realizadas a través de *Random Forest*. Lo anterior, debido a su atractivo teórico pues constituye un método que combina modelos de árboles de clasificación sobre datos de entrenamiento generados a partir de un proceso de *boosting* y simultáneamente incorpora un parámetro que reduce la correlación entre los mismos. En términos prácticos y observando tanto el AUC como el error de predicción, presenta un AUC muy cercano al más alto obtenido – diferencia de 0.002 – y el segundo menor error de predicción. Igualmente, al probar su predicción en la plataforma *Kaggle*, resultó ser el mejor modelo predictivo.