# FACULTAD DE ESTUDIOS ESTADÍSTICOS

## MÁSTER EN MINERÍA DE DATOS E INTELIGENCIA DE NEGOCIOS

**Curso 2019/2020**

## Trabajo de Fin de Máster

***TITULO: UNA HERRAMIENTA PARA MEJORAR LA JERARQUIZACIÓN DE LAS INSPECCIONES DE SANIDAD EN RESTAURANTES DE LA CIUDAD NUEVA YORK.***


***Alumno*: Ruth Maly Olivera Kalafatovich**

***Tutor*: Aida Calviño Martínez**


Septiembre de 2020


UNIVERSIDAD COMPLUTENSE
MADRID

# ÍNDICE

## TABLAS:



## ILUSTRACIONES:

1. Introducción

La inocuidad es de vital importancia en la calidad de los alimentos, por tanto, es obligación de quienes lo comercializan el garantizar que estos sean inocuos, pero a pesar de esto muchos establecimientos de comercialización de alimentos descuidan este aspecto de vital importancia, lo cual puede producir un daño en los consumidores.

Por lo comentado en el párrafo anterior, varios países en estas últimas décadas han implementado un sistema de inspección a estos establecimientos y en este trabajo hablaremos en particular de los restaurantes (Yuniesky González Muñoz,Carolina Esthela Palomino Camargo, 2012).

Todos los años el departamento de Salud de Nueva York inspecciona aproximadamente 27.000 restaurantes en dicha ciudad para controlar el cumplimiento de las normas de seguridad alimentaria, para ello el Departamento ha desarrollado un sistema de evaluación. Un restaurante que obtenga de 0 a 13 puntos de violación en su primera inspección recibe una tarjeta de calificación de A. Si el restaurante obtiene de 14 a 27 puntos obtiene una tarjeta de calificación B, y si tiene de 28 a más puntos de violaciones sanitarias se le asigna la tarjeta de calificación C (Michael,Farly, 2016).

La frecuencia de las inspecciones depende de la puntuación del restaurante. Por lo tanto, los restaurantes con calificaciones A son inspeccionados con menor frecuencia que aquellos con calificaciones B y C.

Estas inspecciones se realizan de manera inopinada y los resultados con los puntajes obtenidos son almacenados en NYC Open Data. Así mismo se ha observado que la página web, si bien proporciona datos totales actualizados acerca de las inspecciones, brinda un escaso análisis de los mismos. Por ello, el objetivo de este Trabajo de Fin de Master es proporcionar al Departamento de Salud de Nueva York un análisis descriptivo de los datos, factores que influyan en la calificación y una herramienta que permita establecer un orden para realizar las inspecciones de sanidad, ya que no cuentan con una cantidad suficiente de inspectores para realizar dicha tarea, con ello, se buscaría optimizar los recursos del personal disponible.

Además de trabajar con la base de datos obtenidos de la página web NYC Open Data, se realizará la técnica de Web Scraping en el software Python para extraer información de la calificación que asignan los clientes a los restaurantes en Google Maps.

**Antecedentes**

Cada establecimiento de servicio de alimento en la ciudad de Nueva York recibe una inspección in situ sin previo aviso al menos una vez al año para verificar si cumple con los requisitos de seguridad alimentaria del Código de Salud. El

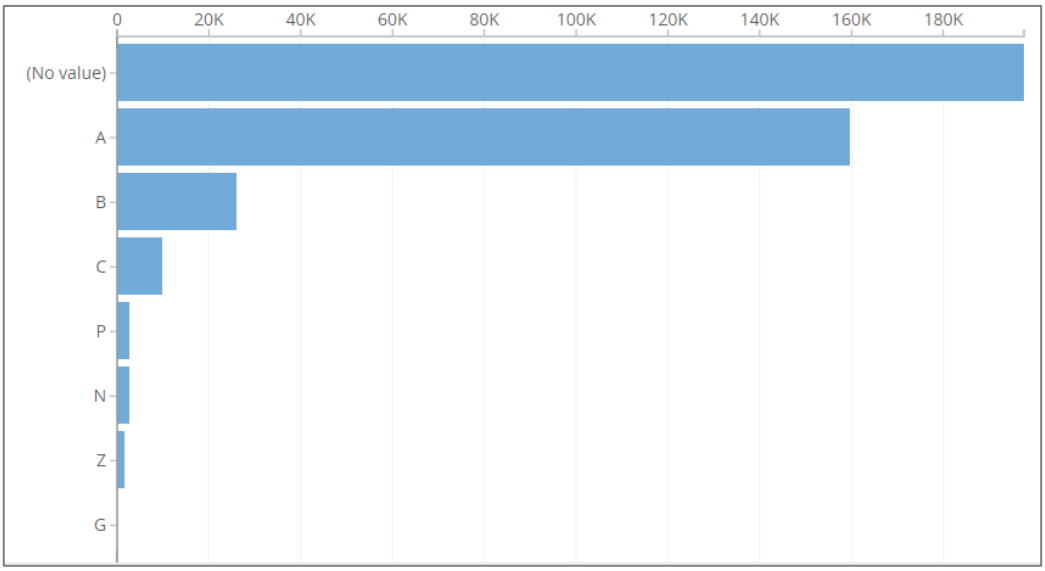
inspector puede visitar en cualquier momento que el restaurante este recibiendo o preparando comida o bebida, o esté abierto al público.  (Bloomberg, 2016).

Durante las inspecciones a los establecimientos de comida, los inspectores sanitarios de salud publica, prestan principal atención a las violaciones graves del codigo de Salud. Estos incluyen:

- Controles inadecuados de tiempo y temperatura
- Enfriamiento inadecuado
- Mala higiene personal
- Enfermedades de los empleados
- Comida lista para servir en contacto directo con la piel de las manos
- Presencia de roedores u otras alimañas
- Equipamiento e instalaciones inadecuadas

Además el departamento tiene implementado una plataforma virtual, donde se aprecian algunos gráficos de los resultados de las inspecciones, tales como el número de restaurantes que obtienen la calificación de grado A, grado B, grado C y demás denotaciones (Ilustración 1).

**Ilustración 1: Número restaurantes según calificación**




Fuente: NYC OpenData

También podremos encontrar información sobre los tipos de restaurantes o el número de restaurantes por ciudades.

2. Objetivos

- El objetivo general de este trabajo es predecir la probabilidad de que un restaurante de la ciudad de Nueva York pase una inspección de sanidad o no.

- Alimentar la base de datos de la página web de Nueva York con variables extraídas de Google Maps, mediante la técnica de web scraping.

- Identificar las principales variables que influyen para que un restaurante pase o no una inspección de sanidad.

- Aplicar algoritmos de Machine Learning y elegir el mejor para la predicción.

- Proporcionar una herramienta que permita establecer un orden para realizar las inspecciones.

3. Estado del arte

El junio del 2019, se presentó el trabajo de fin de master "*Minería de datos para la mejora de la gestión de las inspecciones de sanidad en restaurantes de la ciudad de Chicago"* el cual tuvo como objetivo facilitar un ranking de aquellos restaurantes con mayor riesgo de no pasar la inspección de sanidad en la Ciudad de Chicago, para ello aplicaron diversos métodos de Machine Learning, obteniendo como mejor modelo al Gradient Boosting y como principales variables de mayor aporte al modelo: número de opiniones, horario de atención en el restaurant, latitud, longitud y el número medio de burbujas en las 10 últimas opiniones (Ramírez, 2019).

El departamento de salud pública de la Ciudad de Chicago, cuenta con el apoyo del departamento de Innovación y Tecnología, el cual emitió un informe titulado *Pronósticos de inspección de alimentos*, donde se estimó la probabilidad de que cada establecimiento tuviera una violación crítica o no. Algunas de las variables utilizadas para es te estudio fueron: establecimientos que previamente registraron violaciones críticas, quejas respecto a basura y saneamiento, tipo de instalación, si el restaurante cuenta con licencia de tabaco, o de consumo de alcohol, tiempo transcurrido de la última inspección, entre otras. Este estudio facilitó para que el departamento de salud pública de Chicago identificara 37 restaurantes adicionales por violaciones en el primer mes, en lugar de ser identificados más tarde, lo que redujo el riesgo potencial de producir posibles daños en los comensales (Chicago, 2017).

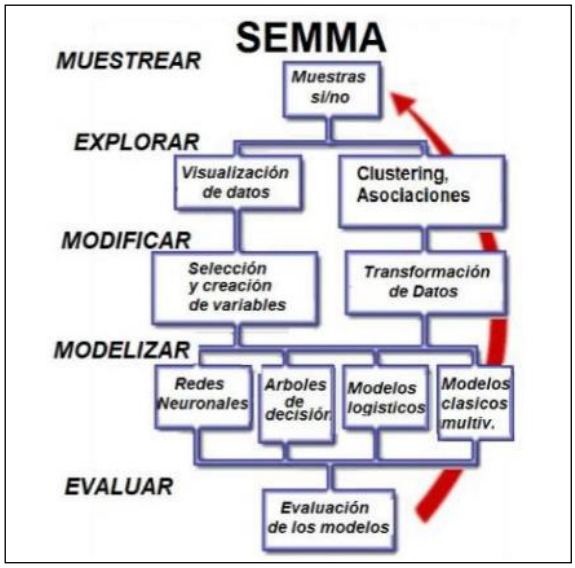
4. Fuente de datos y Metodología

## 4.1 Metodología SEMMA

En esta sección explicaremos la metodología en la que nos hemos basado para el desarrollo de este trabajo.

Para lograr desarrollar los objetivos del presente trabajo se ha seguido las fases establecidas por la metodología SEMMA. La metodología SEMMA está compuesta de 5 fases: Sample (Muestreo), Explore (Explorar), Modify (Modificar) y Assess (Evaluar).

**Ilustración 2: Tareas de la metodología SEMMA**




Fuente: Curso de Técnicas y Metodología de la Minería de Datos

La metodología se centra principalmente en las tareas de modelado, minado y análisis de datos, y aunque en la Ilustración 2 se muestra el patrón a seguir no siempre intervienen todas las tareas del proceso, Además, las fases pueden repetirse y el orden de las mismas puede modificarse.

## 4.2 Recopilación de la información

Los datos fueron extraídos de la página web de Nueva York NYC OpenData, este conjunto de datos incluye los resultados de la inspección de establecimientos de expendio de alimentos en la ciudad de Nueva York desde el año 2013 hasta la actualidad.

La base de datos contiene el tipo de infracción y la fecha de la inspección, entre otros atributos. Cuando a un restaurante se le ha inspeccionado más de una vez en diferentes fechas, se repiten los campos asociados a la identificación del restaurante. Los establecimientos se identifican de forma única por su número CAMIS (ID de registro). Considerando que cada año cientos de restaurantes empiezan a trabajar o salen del negocio, en este conjunto de datos solo los

restaurantes en estado activo son incluidos. También están incluidos registros para restaurantes que han solicitado permiso, pero aún no ha sido inspeccionados. Los establecimientos con fecha 1/1/1900 son nuevos establecimientos que aún no han recibido una inspección.

Dado que este conjunto de datos se compila a partir de varios sistemas de datos administrativos de gran tamaño, es posible que contenga valores erróneos o datos faltantes.

Por lo tanto, la base de datos de la que se parte para la realización de este trabajo es un subconjunto de lo comentado anteriormente. A continuación, en la Tabla 1 describiremos las variables del conjunto de datos obtenido de la página web NYC OpenData:

**Tabla 1: Descripción de variables – NYC OpenData**

| N° | Variable | Descripción |
|---|---|---|
| 1 | CAMIS | ID único por restaurante |
| 2 | DBA | Nombre de restaurante |
| 3 | BORO | Ciudad |
| 4 | BUILDING | Numero de la calle donde se ubica el establecimiento |
| 5 | STREET | Nombre de la calle del establecimiento |
| 6 | ZIPCODE | Código ZIP del establecimiento |
| 7 | PHONE | Número de teléfono |
| 8 | CUISINE DESCRIPTION | Tipo de comida |
| 9 | INSPECTION DATE | Fecha de la inspección |
| 10 | ACTION | Acción asociada con cada establecimiento |
| 11 | VIOLATION CODE | Código de la violación |
| 12 | VIOLATION DESCRIPTION | Descripción de la violación cometida |
| 13 | CRITICAL FLAG | Indica si la violación fue critica o no |
| 14 | SCORE | Puntuación total obtenida en la inspección |
| 15 | GRADE | Grado asignado según su puntuación |
| 16 | GRADE DATE | Fecha en la se emitió el grado al establecimiento |
| 17 | RECORD DATE | Fecha cuando se agregó el registro a la base de datos |
| 18 | INSPECTION TYPE | Tipo de inspección (combinación del programa de inspección y el tipo de inspección realizada) |
| 19 | LATITUD | Latitud del restaurante |
| 20 | LONGITUD | Longitud del restaurante |

Los campos que seleccionaremos por cada inspección serán: CAMIS, nombre del restaurante, ciudad, tipo de cocina, fecha de inspección, grado, tipo de inspección, puntación, critical flag, latitud y longitud.

## 4.3 Preparación de los datos

La base de datos inicial consta de 389.802 registros en los cuales se pueden encontrar más de un registro por restaurante, por tanto, con el software estadístico R se procedió a filtrar por la última fecha de inspección, de tal manera que cada restaurante tenga un único registro (Anexo A: Modificación de la base de datos).

Como se mencionó anteriormente, de todo el set de variables iniciales solo seleccionamos: CAMIS (ID del restaurante), nombre del restaurante, ciudad, tipo de cocina, fecha de inspección, grado, tipo de inspección, puntación, violaciones críticas, latitud y longitud, a partir de esas variables creamos las siguientes:

- Inspecciones (variable objetivo): esta variable dicotómica se crea a partir de la variable grado, donde los restaurantes calificados con grado A serán asignados como “P” y los restaurantes con grado B o C serán “NP”.
- MesInspecc: se crea a partir de la variable fecha de inspección.
- DiaInspecc: se crea a partir de la variable fecha de inspección.
- EstacionInspecc: variable creada a partir de la variable fecha:
  - Primavera (del 21 de marzo al 21 de junio)
  - Verano (del 22 de junio al 21 de setiembre)
  - Otoño (del 22 de setiembre al 21 de diciembre)
  - Invierno (del 22 de diciembre al 20 de marzo)
- NumRestCiudad: esta variable es creada desde la base de datos original a partir de la variable Boro, en donde se realiza el conteo del número de restaurantes ubicados en cada una de las cinco ciudades de Nueva York.
- TotalGradoA: de la base de datos original, se cuenta el número de veces que el ID del restaurante obtuvo una calificación de grado A.
- TotalGradoB: de la base de datos original, se cuenta el número de veces que el ID del restaurante obtuvo una calificación de grado B.
- TotalGradoC: de la base de datos original, se cuenta el número de veces que el ID del restaurante obtuvo una calificación de grado C.
- TotalInsp: información extraída de la base de datos original, donde indica el número de inspecciones que tuvo el ID del restaurante en el pasado.
- NumViolacionesCriticas_N: esta variable es creada a partir de la variable critical flag, donde se realiza un conteo solo de las violaciones consideradas no críticas.
- NumViolacionesCriticas_Y: teniendo en cuenta todo el historial de inspecciones, esta variable también es creada a partir de la variable critical flag, donde se realiza un conteo solo de las violaciones consideradas críticas.
- NumViolaciones: variable creada a partir de la variable violation code, donde se hace el conteo del total de violaciones cometidas por el restaurante.

Una vez creada las nuevas 12 variables, procedemos a eliminar los registros vacíos para la variable objetivo “Inspección”, quedando 23.697 registros.

Luego retiramos a establecimientos cuyo registro en la variable tipo de comida son: panaderías, café, ensalada de frutas, donuts, heladerías, entre otros, de tal manera que solo nos quedemos con establecimiento cuyo rubro son restaurantes quedándonos con 18.800 registros.

El siguiente paso es retroalimentar la base de datos adicionando más atributos por establecimiento, para ello utilizaremos una técnica de extracción de información, el cual es detallado en la sección 5.1.

## 4.4 Técnicas a utilizar

Para modelizar el comportamiento de la variable objetivo, utilizaremos diferentes técnicas estadísticas. Una vez ejecutada las técnicas, debemos elegir el mejor.

A continuación, definiremos a groso modo cada una de las técnicas a utilizar:

### 4.4.1. Regresión Logística:

Los modelos de regresión se han convertido en parte integrante de muchos análisis de datos relacionados con la descripción de datos, causal entre una variable dependiente y una o más variables independiente.

En nuestro caso, al ser la variable a predecir binaria, se puede construir un modelo de regresión lineal cuya variable objetivo sea una variable dummy, obtenida a partir de la variable original. La variable dependiente (Y) representa la ocurrencia o no de un suceso.

Este modelo se representa de la siguiente manera: siendo $p_1$ la probabilidad de que ocurra Y=1 (Portela, Curso Machine Learning, 2020).

$$p_1 = P(Y = 1 \,\|\, x_1, x_2, x_3, \dots, x_m) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \cdots + \beta_m x_m)}}$$

De lo que se deduce:

$$p_0 = 1 - p_1 = P(Y = 0 \,\|\, x_1, x_2, x_3, \dots, x_m) = \frac{e^{-(\beta_0 + \beta_1 x_1 + \cdots + \beta_m x_m)}}{1 + e^{-(\beta_0 + \beta_1 x_1 + \cdots + \beta_m x_m)}}$$

Con lo que se obtiene el logaritmo de la razón de probabilidades (logit) o también llamado odds ratio:

$$\log\left(\frac{p_1}{1 - p_1}\right) = \beta_0 + \beta_1 x_1 + \cdots + \beta_m x_m;\ p_1 = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 (x=1))}}$$

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

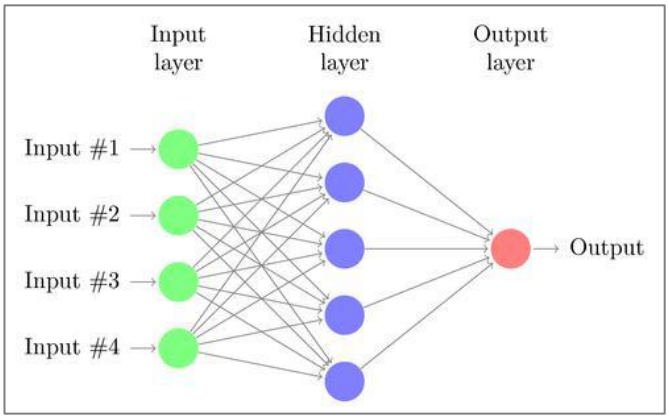
$$odds(x=0) = \left(\frac{p(x=0)}{1-p(x=0)}\right) = e^{\beta_0}$$

$$\boldsymbol{odds\ Ratio} = \frac{\boldsymbol{odds(x=1)}}{\boldsymbol{odds(x=0)}} = \boldsymbol{e^{\beta_1}}$$

4.4.2. Redes Neuronales:

Las redes neuronales están basadas en el funcionamiento de las redes de neuronas biológicas, a nivel esquemático una neurona artificial se representa de la siguiente manera:

**Ilustración 3: Red neuronal artificial**




Fuente: Curso de Machine Learning (Portela, Curso Machine Learning, 2020).

Las neuronas de la primera capa reciben como entrada los datos reales que alimentan a la red neuronal, por ello la primera capa se conoce como capa de entrada o input layer. La salida de la última capa es el resultado visible de la red, por lo que la última capa se conoce como capa de salida u output layer, las capas que se sitúan entre la capa de salida y la de entrada se conocen como capas ocultas ya que desconocemos tanto los valores de entrada como los de salida.

Por tanto, una red neuronal es un modelo que presenta la siguiente forma:

$$y = f(x_1, x_2, x_3, \dots)$$

Donde la función f es por lo general no lineal (Portela, Curso Machine Learning, 2020).

4.4.3. Random Forest

Random Forest es una técnica de aprendizaje automática supervisada basada en los árboles de decisiones. Los métodos de random forest y bagging siguen el mismo algoritmo con la única diferencia de que en, random forest, antes de cada división se seleccionan aleatoriamente m

predictores. La diferencia en el resultado dependerá del valor m escogido. Si m=p los resultados de random forest y bagging son equivalentes (Gareth, Witten, Hastie and Tibshirani, 2009).

Sus principales parámetros a controlar son (Portela, Curso Machine Learning, 2020):

- El tamaño o porcentaje de las muestras **n** y si se va a utilizar bootstrap (con reemplazo) o sin remplazamiento.
- El número de iteraciones **m** a promediar.
- El número de variables **p** a muestrar en cada nodo (si es igual al número inicial de variables k el Random Forest es equivalente al bagging).
- Características de los árboles. Son bastante influyentes:

    - ✓ El número de hojas final o, en su defecto, la profundidad del árbol.
    - ✓ El maxbranch (número de divisiones máxima en cada nodo. Por defecto se dejará en 2, árboles binarios).
    - ✓ El p-valor para las divisiones en cada nodo (más alto implica árboles menos complejos con más sesgo y menos varianza)
    - ✓ El número de observaciones mínimo en una rama- nodo. Se puede ampliar para evitar sobreajuste (reducir varianza) o reducir para ajustar mejor (reducir sesgo).

#### 4.4.4. Gradient Boosting

El algoritmo gradient boosting consiste en repetir la construcción de árboles de regresión/clasificación, modificando ligeramente las predicciones iniciales cada vez, intentando ir minimizando los residuos en la dirección de decrecimiento (Smolyakov, 2017).

Al plantear diferentes árboles cada vez, el proceso va ajustando las predicciones cada vez más a los datos, y de alguna manera unos árboles corrigen a otros con lo cual la flexibilidad y adaptación del método mejora respecto a la construcción de un único árbol. Este proceso ha de ser monitorizado en principio mediante early stopping para determinar el número de iteraciones. Por lo tanto, necesitará datos de validación. Aunque a menudo el early stopping no es necesario pues la convergencia es lenta y van a la par los errores en training y validación (no sobreajuste).

Este algoritmo es sensible ante missing y outlier, pero bien tuneado funciona bien.

Sus parámetros a controlar son:

- ✓ shrinkage: es el parámetro más importante, parámetro de regulación, los valores normalmente se sitúan entre 0.001 a 0.3.

- ✓ ntree: número de iteraciones(arboles), si le damos un valor muy grande puede sobre ajustar
- ✓ nodesize: tamaño mínimo de nodos finales, si es muy pequeño podemos estar sobre ajustando, si es muy grande hacemos arboles pocos finos
- ✓ sampsize: tamaño de cada muestra, si se quiere hacer bagging (no necesario)

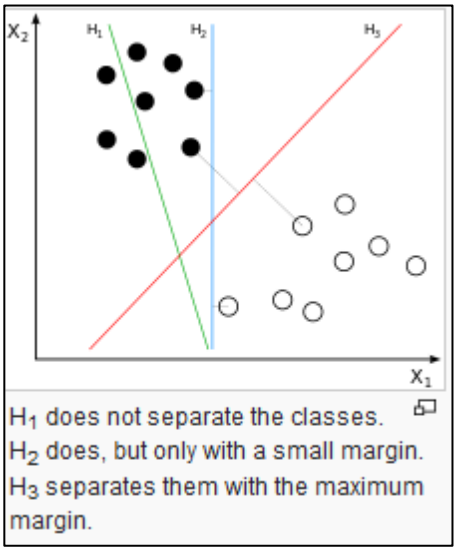
Gradient Boosting, es un algoritmo incisivo, que apura al límite la reducción del error, su aprovechamiento de los árboles, su eficacia y su capacidad de manejo de datos complejos es muy grande (Portela, Curso Machine Learning, 2020).

### 4.4.5. Support Vector Machine

Los modelos de Support Vector Machine se trata de plantear el problema de separación lineal de clases con métodos algebraicos (busca el hiperplano de separación). Se basa en tres ideas importantes.

- ✓ Maximal margin: no solo separa las clases por un hiperplano (función lineal), esto mejora el sesgo como la varianza de los resultados.

**Ilustración 4: Maximal margin (Varnik,1963)**




Del grafico podemos decir que el mejor modelo es el rojo, y aunque el modelo azul aparentemente también es un buen modelo, puede errar en algún punto, el rojo esta mejor preparado pues deja más espacio respecto de las dos clases.

Por tanto, el planteamiento de este nuevo algoritmo geométrico se basa en la separación máxima.

- ✓ Soft margin: según el documento elaborado por Portela (Portela, Tema 5. Support Vector Machines, 2019), la separación perfecta no suele

existir, en ese sentido es necesario permitir observaciones mal clasificadas por los separadores para no incurrir en sobreajustes.

- ✓ Kernel: en algunos casos se ha observado que la separación entre clases no es lineal, por ello, a través del truco kernel, el cual es un concepto que dio gran popularidad y uso a esta técnica.
  Técnica que consiste en trabajar en un espacio de dimensión superior donde sí tenga sentido la separación lineal. Simplemente extrapolar los datos con más dimensiones nos permite encontrar separadores lineales

Parámetro a utilizar en SVM:

1. Parámetro C: aumentar C implica menor sesgo y mayor sobre ajuste. El rango depende de los datos que se esté analizando.
2. La función Kernel: no siempre es necesaria y sus parámetros en cada caso son: RBF (aumentar gamma en la función RBF implica menor sesgo y mayor sobre ajuste), Polinomial (aumentar el grado del polinomio implica menor sesgo y mayor sobre ajuste)

Hay interdependencia entre los parámetros C y los del Kernel (scikit-learn, 2020) (Portela, Tema 5. Support Vector Machines, 2019).

### 4.4.6. Ensamblado

Los métodos de Ensamble (conjunto), consisten en la construcción de predicciones a partir de la combinación de varios modelos (Portela, Ensamblado-Curso de Machine Learning, 2019).

Una primera aproximación a esta idea la puede dar el siguiente ejemplo suponiendo un problema de regresión:

- ✓ Se construye un modelo de redes que da lugar a la predicción y1 (sobre los datos test).
- ✓ Se construye otro modelo con regresión, que da lugar a la predicción y2.
- ✓ Se construye un modelo random forest que da lugar a la predicción y3.
- ✓ Se estudia la performance de y1, y2, y3, y del promedio de ellas, y4. Esta variable y4 es una predicción nueva, que a veces puede funcionar mejor que cualquiera de las predicciones y1, y2, y3.
- ✓ O bien, en lugar de promediar y1, y2, y3, se construye, por ejemplo, un modelo de red neuronal en el que las variables de entrada sean y1, y2, y3. Se obtendría una nueva predicción y4 a partir de este modelo.

Algunas de las ventajas de los métodos de ensamble son:

- ✓ Bastante robustos, unos modelos corrigen a otros.
- ✓ Reducen la varianza del error en general, casi nunca empeoran los modelos.

Desventajas de los métodos de ensamble:

- ✓ Cada modelo tiene sus errores de estimadores de parámetros lo que aumenta aparentemente la complejidad.
- ✓ Excesivas posibilidades que a veces llevan al sobreajuste.
- ✓ Los resultados no son interpretables.

### 4.4.7. Validación de modelo

Para analizar los resultados obtenidos de los modelos, utilizaremos el diagrama de cajas, pues ofrecen información acerca del comportamiento medio y la variabilidad.

Además, una vez identificado el mejor modelo, calcularemos:

- ✓ Área bajo la curva ROC (Receiver Operating Characteristic Curve): representación gráfica de la sensibilidad frente a la especificidad, en el eje x está 1-especificidad, y en el eje y la sensibilidad. Por tanto, cuanto más cerca del valor 1 (área total del cuadrado) esté el AUC mejor.
- ✓ Matriz de confusión: esta herramienta nos permitirá visualizar el desempeño del modelo, describiendo como se distribuyen los valores reales y las predicciones obtenidas por el modelo. Frecuentemente son utilizadas las siguientes medidas:
  - Sensibilidad: tasa de verdaderos positivos, es la proporción de casos positivos que fueron correctamente identificados por el algoritmo.
  - Especificidad: tasa de verdaderos negativos, se trata de los casos negativos que el algoritmo ha clasificado correctamente.

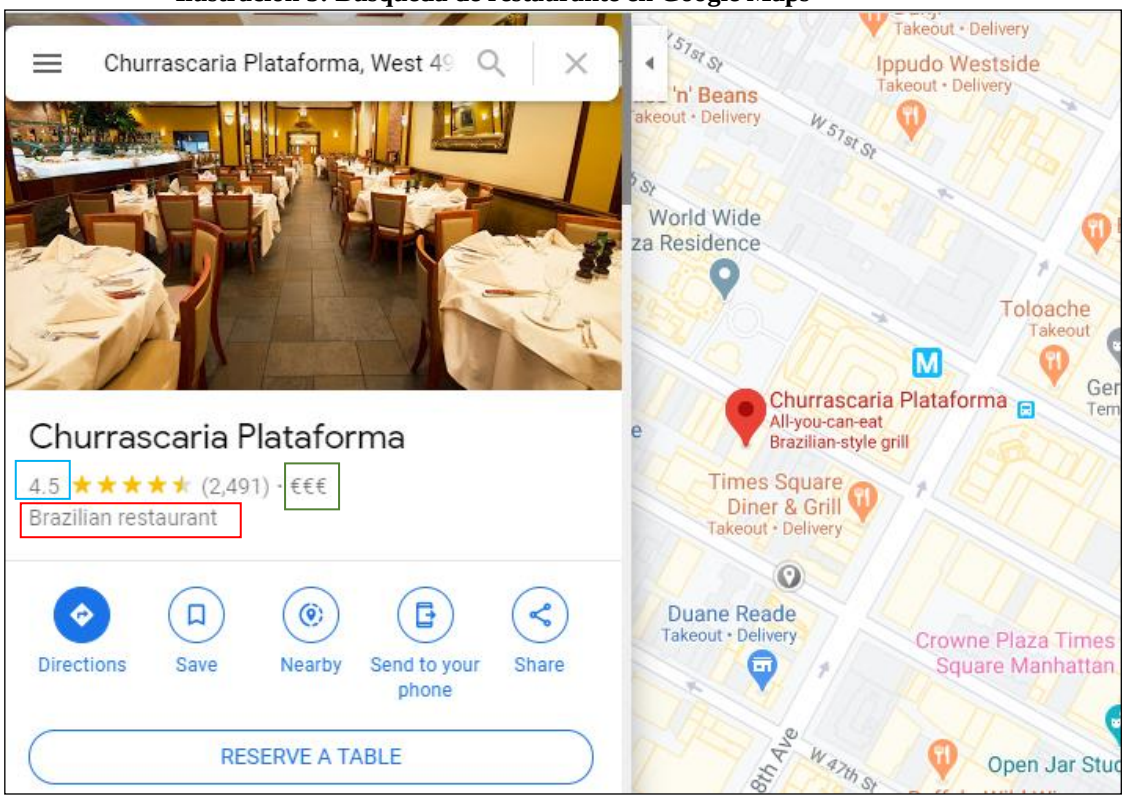
5. Desarrollo del trabajo y principales resultados

5.1. Web Scraping de restaurantes de Nueva York

Como nuestra base de datos cuenta con información sobre el nombre del restaurante decidimos extraer información de Google Maps para sacar la máxima información posible del establecimiento, de esta manera enriquecemos los datos obtenidos de OpenData Nueva York.

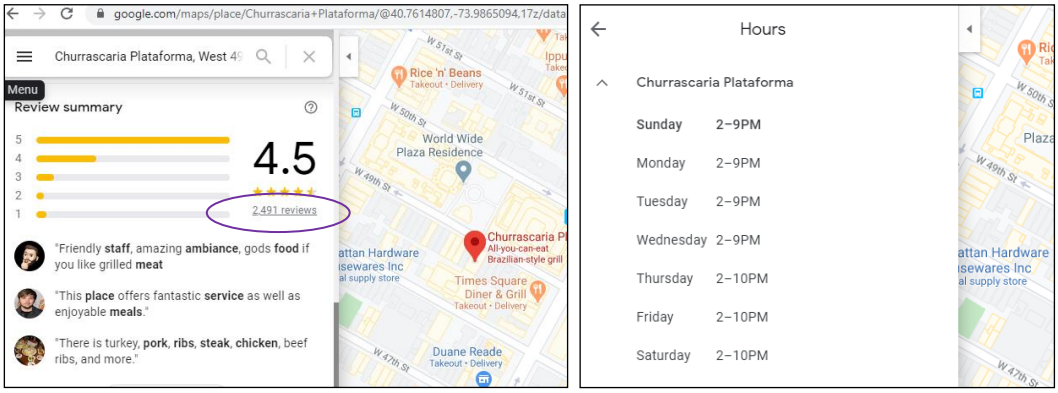
Google Maps cuenta con un sistema de valoración por reseñas, donde el usuario puede escribir su opinión sobre establecimientos visitados. Este sistema de reseñas en Google Maps es muy importante de cara al público tanto para una empresa como para los usuarios que pueden considerar valorar su experiencia con ella, bien sea para recomendarla o para disuadir a otras personas de establecer una relación con la empresa, habitualmente por culpa de un mal servicio o producto.

**Ilustración 5: Búsqueda de restaurante en Google Maps**




Una vez ubicado el restaurante en Google Maps, como se observa en la Ilustración 5, podemos ver la puntuación sobre cinco estrellas (enmarcado de celeste), saber si el sitio es caro o no (enmarcado de verde), tipo de comida (enmarcado de rojo), la descripción y el horario de atención.

**Ilustración 6: Puntuación y horario de atención del restaurante**




En la Ilustración 6 podemos apreciar los comentarios realizados, el número total de comentarios (enmarcado de morado) y el horario de atención.

Por lo tanto, siendo notable, se ha visto conveniente extraer toda esta información, pero ¿Cómo se puede obtener esta información sin tener que almacenarla manualmente restaurante por restaurante?, la respuesta es utilizando la técnica de web scraping.

Esta técnica es la manera más practica para obtener un gran volumen de datos públicos de sitios web. Automatiza la recopilación de datos y convierte los datos extraídos en formatos de su elección como HTML, CSV, Excel, JSON, txt.

La forma principal para realizar web scraping es a través de la programación, en nuestro caso desarrollaremos un código en Python.

Primero empezaremos definiendo los datos que deseamos extraer por cada restaurante:

a) URL: url de cada ubicación de restaurante.
b) Puntuación: el total de estrellas que tiene el restaurante.
c) 1_estrella: número total de puntuación de 1 estrella recibido.
d) 2_estrellas: número total de puntuación de 2 estrellas recibido.
e) 3_estrellas: número total de puntuación de 3 estrellas recibido.
f) 4_estrellas: número total de puntuación de 4 estrellas recibido.
g) 5_estrellas: número total de puntuación de 5 estrellas recibido.
h) Tipo_comida: tipo de comida que ofrece el establecimiento.
i) Precio: esta variable indicará si el restaurante está clasificado como caro o no.
j) Horario: horario de atención.
k) N_dias_aperturado: número de días laborable a la semana.
l) N_comentarios: número de comentarios.

En total se ha definido 12 variables, los cuales se adicionarán a la base de datos.

Debido a que la extracción de información por cada establecimiento se realizará por nombre, latitud y longitud, se identificó que había restaurantes con diferente

ID (CAMIS) pero con el mismo nombre, por tanto, para evitar inconvenientes al momento de realizar el web scraping, se decidió filtrar y solo quedarnos con un único ID (CAMIS) y un único nombre del establecimiento.

Para realizar el web scraping existen diferentes software, plataformas o interfaz como por ejemplo WebHarvy, OutWit Hub, Visual Web Ripper, etc. Además, también hay software que permite a través de un código de escritura o mediante una API personalizar necesidades específicas para extraer datos de cualquier sitio web como son R o Python (Data, 2020).

En nuestro caso utilizaremos Python ya que consideramos lo siguiente:

- Mayor flexibilidad: como sabemos las páginas web, se van actualizando rápidamente, no solo en contenido sino en estructura. Python es un lenguaje de programación fácil de usar ya que es altamente productivo y dinámicamente imputable, por lo que el código se puede ir actualizando sin inconveniente alguno e ir a la misma velocidad de las actualizaciones web.

- Mayor potencia: este software tiene una una gran variedad de bibliotecas, por ejemplo, Beautifulsoup4, Selenium, además de contar con otras librerías para el limpiados de los datos una vez extraída la información como numpy y pandas.

Como ya se mencionó, para el desarrollo del código contamos con varias librerías como Beautiful Soup, requests y Selenium principalmente, estas librerías nos facilitarán la manera de realizar peticiones a la web con una sintaxis no tan complicada, debido a que la estructura HTML de Google Maps es más complejo, no utilizaremos Beautiful Soup ni requests, ya que cuando intentamos extraer la URL de cada establecimiento con estos paquetes no obtenemos información alguna, solo tenemos algunos toakens,y ellos llaman a algunos ajax Request, las respuestas obtenidas están encriptadas, así que no se puede analizar. Por ello utilizaremos la librería Selenium.

También necesitaremos instalar el controlador web del navegador que queremos usar, en nuestro caso Google Chromedriver. Este controlador web es el que ejecuta automáticamente una instancia del navegador, sobre la cual funcionará Selenium.

A manera, para extraer la información de Google maps, se siguió los siguientes pasos:

- Extraer la URL de cada establecimiento, para esto se utilizó el nombre del restaurante, la latitud y longitud.
- Inspeccionar la página.
- Encontrar los datos que se desea extraer.
- Escribir el código.
- Ejecutar el código y extraer los datos.

- Almacenar los datos en formato Excel.

El código desarrollado en Python se puede observar en el Anexo D: Web Scraping, este proceso demoró aproximadamente ocho horas, se extrajo información de 14.952 establecimientos y debido a la cantidad de datos se tuvo que trabajar con subprocesos.

Para evitar problemas con el rendimiento al momento de realizar la consulta con el algoritmo, se crea un grupo de subprocesos, el cual gestiona la ejecución simultanea de una gran cantidad de consultas. En Python se utilizó el módulo multiprocessing.pool para crear este grupo de subprocesos, lo que permitió que el rendimiento computacional en el equipo donde se ejecutó la consulta no se vea afectado.

Una vez obtenido nuestros datos en Excel, retiro a todos los establecimientos sin información, quedando un total de 13.743 registros.

Luego se observó que en la columna Tipo_comida se identificaron establecimientos que estaban registrados como Nigth Club, entretenimiento para adultos, cabaret, Bakery, entre otros rubros diferentes a restaurantes, los cuales no habían sido identificados en la columna de Tipo de Comida de OpenData Nueva York, por tanto, se procedió a retirarlos, quedándonos con 12.343 registros, tamaño final del data set.

Por último, es importante resaltar que la información extraída de Google Maps podría no coincidir necesariamente con la última fecha de inspección de sanidad que se realizó a los restaurantes. Por este motivo en nuestra base de datos generada por OpenData Nueva York después de todos los filtros realizados solo nos quedamos con inspecciones hechas en los últimos años, es decir; desde enero del 2019 a marzo del 2020. Por tanto, para el web scraping se consideró solo a los establecimientos inspeccionados en dicha fecha con la finalidad de poder acercarnos lo máximo posible en el tiempo con la información extraída de Google Maps.

## 5.2. Descripción de base de datos complementada

En este apartado mostraremos tanto el resumen del filtrado de variables, como de los registros.

La Ilustración 7 describe como se ha venido construyendo el set de variables para la base de datos, para una fácil identificación hemos incorporado a las variables descargadas de la página web OpenData Nueva York el prefijo NY_, las variables construidas tendrán el prefijo C_, y las variables obtenidas por el web scraping tendrán el prefijo WS_. Tal como se muestra en la Ilustración 7, la base de datos inicial tenía 20 variables, de las cuales se han excluido 10 y de los 10 restantes se han construido otras 12 variables. Adicional a ello también se han incorporado 12 variables obtenidas por web scraping, dando un total de 34 variables

preseleccionadas para realizar los modelos, de las cuales se han descartado las siguiente 6 en SAS Miner (Tabla 2),

**Ilustración 7: Resumen de variables**

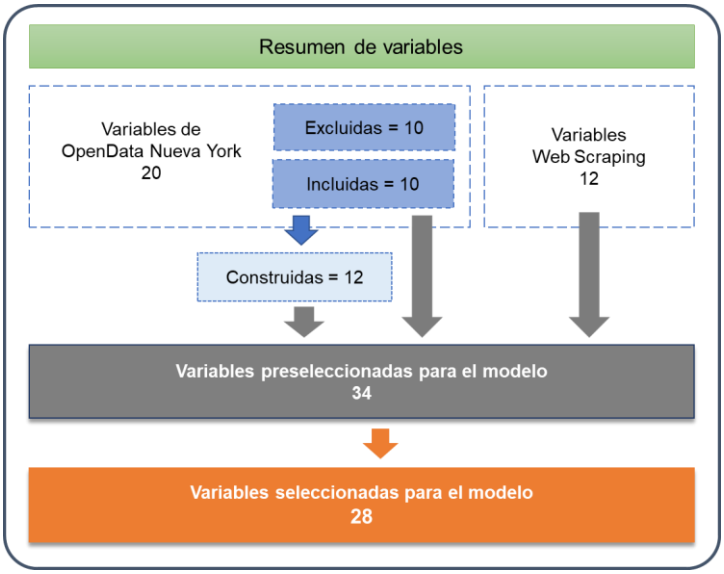



**Tabla 2: Variables excluidas**

| Núm. | Variables excluidas | Razón de rechazo |
|---|---|---|
| 1 | NY_CAMIS | ID de restaurante |
| 2 | NY_DBA | Solo indica el nombre del establecimiento |
| 3 | NY_Grado | Estas variables se utilizaron para la creación de la variable objetivo. |
| 4 | NY_Score | |
| 5 | WS_URL | Solo tiene información de URL por cada establecimiento. |
| 6 | WS_DescripcionComida | Ya se utilizó para retroalimentar la variable NY_Decripcióncomida. |

Quedando un total de 28 variables para realizar los modelos (Tabla 3).

**Tabla 3: Set de variables para realizar modelos**

| N° | Variables |
|---|---|
| 1 | NY_DescripcionComida |
| 2 | NY_FechaInspeccion |
| 3 | NY_Latitud |
| 4 | NY_Longitud |
| 5 | NY_TipoInspeccion |
| 6 | NY_Inspecciones |

| N° | Variable |
|---|---|
| 7 | C_TotalInsp |
| 8 | C_DiaInspecc |
| 9 | C_EstacionInspecc |
| 10 | C_Inspecciones |
| 11 | C_MesInspecc |
| 12 | C_NumRestCiudad |
| 13 | C_NumViolaciones |
| 14 | C_NumViolacionesCriticas_N |
| 15 | C_NumViolacionesCriticas_Y |
| 16 | C_TotalGradoA |
| 17 | C_TotalGradoB |
| 18 | C_TotalGradoC |

| N° | Variable |
|---|---|
| 19 | WS_1Estrella |
| 20 | WS_2Estrellas |
| 21 | WS_3Estrellas |
| 22 | WS_4Estrellas |
| 23 | WS_5Estrellas |
| 24 | WS_HorarioAtencion |
| 25 | WS_NumDiasTrabajo |
| 26 | WS_Precio |
| 27 | WS_PuntuacionRating |
| 28 | WS_TotalComentarios |

Ahora mostraremos el resumen del filtrado de datos que se ha venido realizando en cada una de las etapas (Ilustración 8).

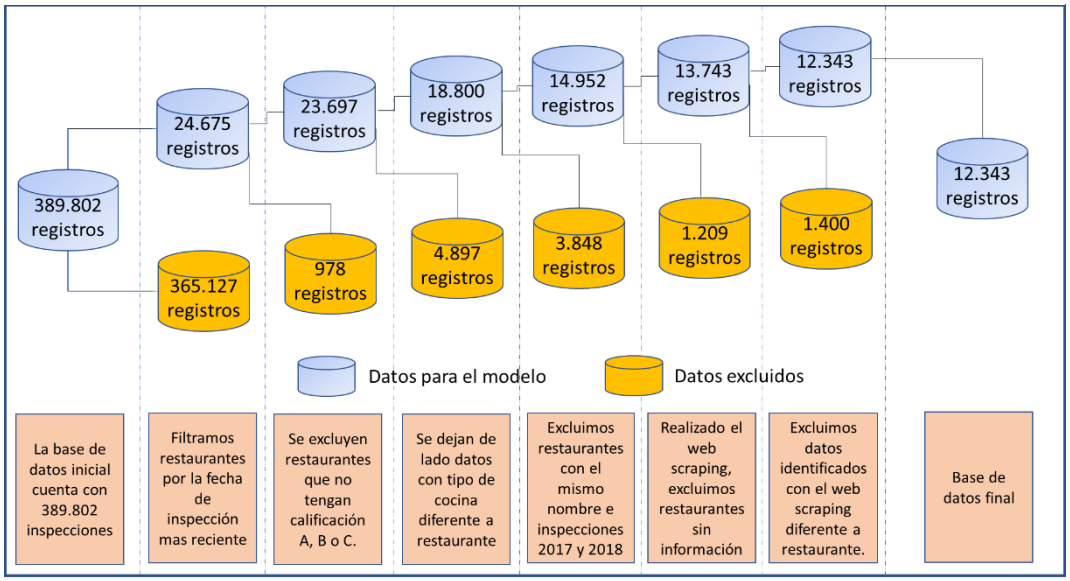
Inicialmente se descargó de la página web de OpenData de Nueva York 389.802 registros, tal como se detalló en el apartado *4.3 Preparación de datos,* nos quedamos con la inspección más reciente por CAMIS (ID del restaurante), excluyendo así 365.127 observaciones. Una vez obtenidos los 23.697 registros creamos la variable objetivo *Inspecciones*, por lo que en la siguiente etapa decidimos excluir 978 registros debido a que no contaban con información para la variable objetivo, quedándonos con 18.800 registros.

En una siguiente etapa se decide dejar de lado a todos aquellos establecimientos que no precisaban ser restaurantes, por lo que se excluyó a 4.897 registros.

Luego debido a que el análisis de web scraping se realizaría considerando el nombre del establecimiento, nos percatamos que si bien nuestra base de datos contaba con un único CAMIS, encontramos que algunos establecimientos tenían varias sedes en una misma ciudad, por lo que registraban el mismo nombre, esto nos generaría inconvenientes al momento de extraer información de Google Maps, por tanto se decidió filtrar y quedarnos solo con establecimientos que registren un único CAMIS y nombre, además para que la información extraída por la web no diste mucho con el tiempo de inspección, retiramos restaurantes inspeccionados en el año 2017 y 2018, quedándonos con 14.952 registros.

Una vez obtenida toda la información de interés de Google Maps, retiramos registros de restaurantes que nuestro código en Python no pudo obtener.
En una última etapa, se procedió a comparar la variable WS_DescripcionComida con la variable NY_DescripcionComida, donde notamos que algunos establecimientos de uno de los niveles de la variable NY_DescripcionComida estaban mal clasificados y no eran necesariamente restaurantes, por lo que se procedió a excluir a los 1.400 registros, quedando un total de 12.342 registros.

**Ilustración 8: Resumen de filtrado de base de datos**



## 5.3. Análisis descriptivo de las variables

En esta sección, seguiremos trabajando en SAS Miner para mostrar la descripción de los estadísticos para las variable cuantitativas y cualitativas.

### 5.3.1. Variables de intervalo:

Principales estadísticos de las variables de intervalo, encontraremos tres variables con valores ausentes y siete variables con gran diferencia entre la media y sus mediana, lo que nos estaría indicando la presencia de valores atípicos, para solucionar ello utilizaremos los siguientes métodos: desviación estándar, desviación absoluta media y percentiles extremos (Ilustración 9).

**Ilustración 9: Estadísticos variable de intervalo**

| Variable ▲ | Ausente | Mediana | Mínimo | Máximo | Media | Desviación estándar | Asimetría | Curtosis |
|---|---|---|---|---|---|---|---|---|
| C_NumRestCiudad | 0 | 6290 | 916 | 9997 | 6932.291 | 2789.066 | -0.35037 | -0.84328 |
| C_NumViolaciones | 0 | 12 | 0 | 96 | 15.15077 | 11.24873 | 1.353925 | 2.432315 |
| C_NumViolacionesCriticas_N | 0 | 5 | 0 | 36 | 5.527262 | 4.083501 | 1.282506 | 2.379829 |
| C_NumViolacionesCriticas_Y | 0 | 7 | 0 | 62 | 8.353885 | 6.981631 | 1.45369 | 2.891621 |
| C_TotalGradoA | 0 | 6 | 0 | 18 | 6.024953 | 2.93527 | 0.217209 | -0.25721 |
| C_TotalGradoB | 0 | 0 | 0 | 21 | 1.114235 | 2.408375 | 2.699233 | 8.648381 |
| C_TotalGradoC | 0 | 0 | 0 | 72 | 3.470874 | 5.744811 | 2.630261 | 11.46622 |
| C_TotalInsp | 0 | 9 | 0 | 91 | 10.72268 | 7.915111 | 1.83026 | 5.679725 |
| NY_Latitud | 9 | 40.73183 | 40.50807 | 40.91282 | 40.72658 | 0.068835 | -0.13506 | 0.174163 |
| NY_Longitud | 9 | -73.9618 | -74.2487 | -73.7009 | -73.9451 | 0.076003 | 0.196799 | 1.448474 |
| WS_1Estrella | 0 | 9 | 0 | 962 | 17.80977 | 37.0082 | 9.931333 | 157.8719 |
| WS_2Estrellas | 0 | 5 | 0 | 774 | 10.69351 | 26.11972 | 12.30108 | 240.839 |
| WS_3Estrellas | 0 | 13 | 0 | 959 | 29.39294 | 57.41376 | 6.735731 | 68.22897 |
| WS_4Estrellas | 0 | 32 | 0 | 991 | 72.22766 | 111.8134 | 3.482174 | 15.84864 |
| WS_5Estrellas | 0 | 89 | 0 | 997 | 152.1392 | 176.0182 | 2.035133 | 4.53663 |
| WS_PuntuacionRating | 0 | 4.3 | 1 | 5 | 4.251892 | 0.388332 | -1.87258 | 9.651434 |
| WS_TotalComentarios | 1 | 162 | 0 | 59575 | 371.8956 | 1257.111 | 29.80253 | 1262.452 |

*Variable NY_Latitud y NY_Longitud*

Observamos que presentan 9 observaciones ausentes, por lo tanto, se ha visto conveniente imputar por el valor de la media de latitud y valor de media de longitud.

*Variable WS_TotalComentarios*

Solo presenta un valor ausente, por lo que se imputa con el valor de su media.

Respecto a las demás variables no se observan valores ausentes, ni valores máximos anómalos.

Ahora estudiaremos los datos atípicos, para ello es necesario ver el histograma de cada variable (Ilustración 10).

Como vemos en la Ilustración 9 y 10, solo la variable C_TotalGradoA es simétrica, por lo que le aplicaremos el método de desviación estándar, las variables C_TotalGradoB y C_TotalGradoC son asimétricas con mediana igual a 0, a estas variables se aplicará el método de percentiles

extremos y a las restantes aplicaremos el método de desviación absoluta media (Calviño, 2019).

**Ilustración 10: Histograma para identificar valores atípicos**

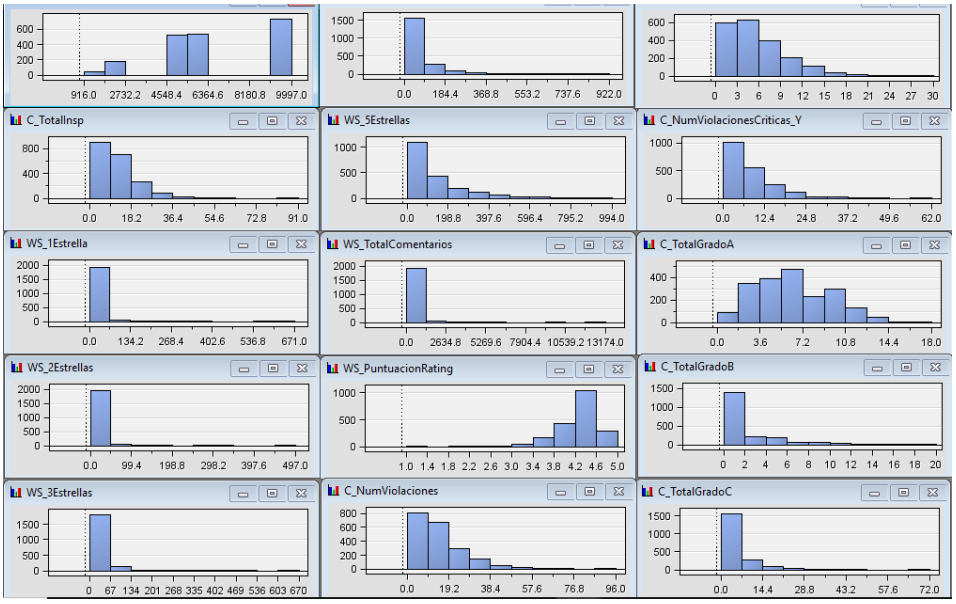



**Ilustración 11: Estadísticos de variable intervalo después de ser depurado**

| Variable ▲ | Ausente | Mediana | Mínimo | Máximo | Media | Desviación estándar | Asimetría | Curtosis |
|---|---|---|---|---|---|---|---|---|
| C_NumRestCiudad | 0 | 6290 | 916 | 9997 | 6932.291 | 2789.066 | -0.35037 | -0.84328 |
| IMP_NY_Latitud | 0 | 40.7317 | 40.50807 | 40.91282 | 40.72658 | 0.06881 | -0.13511 | 0.17681 |
| IMP_NY_Longitud | 0 | -73.9617 | -74.2487 | -73.7009 | -73.9451 | 0.075976 | 0.19687 | 1.450531 |
| IMP_REP_WS_TotalComentarios | 0 | 162 | 0 | 1233 | 286.75 | 321.6021 | 1.666831 | 2.016626 |
| REP_C_NumViolaciones | 0 | 12 | 0 | 66 | 15.13732 | 11.17841 | 1.282699 | 1.854159 |
| REP_C_NumViolacionesCriticas_N | 0 | 5 | 0 | 32 | 5.526938 | 4.08124 | 1.27326 | 2.283721 |
| REP_C_NumViolacionesCriticas_Y | 0 | 7 | 0 | 43 | 8.34716 | 6.942908 | 1.383026 | 2.25599 |
| REP_C_TotalGradoA | 0 | 6 | 0 | 14.83076 | 6.022786 | 2.928024 | 0.19599 | -0.3429 |
| REP_C_TotalGradoB | 0 | 0 | 0 | 13 | 1.10508 | 2.355545 | 2.494288 | 6.544915 |
| REP_C_TotalGradoC | 0 | 0 | 0 | 30 | 3.434659 | 5.513842 | 2.094172 | 4.912641 |
| REP_C_TotalInsp | 0 | 9 | 0 | 45 | 10.69051 | 7.733455 | 1.546094 | 2.904081 |
| REP_WS_1Estrella | 0 | 9 | 0 | 72 | 14.93097 | 17.22035 | 1.91798 | 3.297486 |
| REP_WS_2Estrellas | 0 | 5 | 0 | 41 | 8.530827 | 10.34756 | 1.812124 | 2.685657 |
| REP_WS_4Estrellas | 0 | 32 | 0 | 275 | 63.17362 | 75.30029 | 1.61091 | 1.67928 |
| REP_WS_5Estrellas | 0 | 89 | 0 | 701 | 149.124 | 164.6227 | 1.697211 | 2.475575 |
| REP_WS_PuntuacionRating | 0 | 4.3 | 2.5 | 5 | 4.255181 | 0.368198 | -1.13261 | 2.974556 |
| WS_3Estrellas | 0 | 13 | 0 | 959 | 29.39294 | 57.41376 | 6.735731 | 68.22897 |

En la Ilustración 11 ya no se observa valores ausentes tanto en la variable NY_Latitud, NY_Longitud y WS_TotalComentarios.

### 5.3.2. Variables nominales

Ahora analizaremos las variables nominales, para ello mostraremos un resumen de sus estadísticos. (Ilustración 12)

**Ilustración 12: Estadísticos variables nominales**

| Etiqueta | Tipo | Número de niveles | Ausente |
|---|---|---|---|
| C_Ciudad | C | 5 | 0 |
| C_DiaInspecc | C | 26 | 0 |
| C_EstacionInspecc | C | 4 | 0 |
| C_Inspecciones | C | 2 | 0 |
| C_MesInspecc | C | 12 | 0 |
| NY_DescripcionComida | C | 26 | 0 |
| NY_FechaInspeccion | C | 26 | 0 |
| NY_TipoInspeccion | C | 4 | 0 |
| WS_HorarioAtencion | C | 5 | 1020 |
| WS_NumDiasTrabajo | C | 8 | 0 |
| WS_Precio | C | 4 | 4668 |

Encontramos 3 variables que cuentan con más de 26 niveles (recordemos que el Miner asigna como valor el número 26 cuando identifica muchos niveles en una variable), así como dos variables que presentan valores ausentes, ahora realizaremos un análisis descriptivo por cada variable.

*Variable C_Ciudad:* en la tabla 4 notamos que la mayor cantidad de restaurantes se ubica en la ciudad de Manhattan, seguido de Brooklin y Queens, por otro lado. la ciudad con el menor número de restaurantes es Staten Island.

**Tabla 4: Variable C_Ciudad**

| Nivel | Frecuencia | Porcentaje |
|---|---|---|
| Manhattan | 4,814 | 39% |
| Brooklyn | 3,235 | 26% |
| Queens | 2,719 | 22% |
| Bronx | 1,090 | 9% |
| Staten Island | 485 | 4% |
| Total | 12,343 | 100% |

*Variable C_DiaInspecc:* debido a que esta variable cuenta con 26 niveles y observando que la distribución según el porcentaje está equilibrada, se ha decidido agrupar en solo dos nuevos niveles: primera_quincena y segunda_quincena (Tabla 5).

*Variable C_EstacionInspecc:* recordemos que esta variable fue creada a partir de la variable fecha de inspección, en donde, notamos que hay un equilibrio porcentual entre cada estación del año y el número de inspecciones (Tabla 6).

**Tabla 5: Variable C_DiaInspecc**

| Nivel | Frecuencia | Porcentaje | Nueva | Porcentaje |
|---|---|---|---|---|
| 1 | 279 | 2% | primera_quincena | 50.60% |
| 2 | 360 | 3% | | |
| 3 | 510 | 4% | | |
| . | . | . | | |
| . | . | . | | |
| . | . | . | | |
| 14 | 360 | 3% | | |
| 15 | 351 | 3% | | |
| 16 | 475 | 4% | segunda_quincena | 49.40% |
| . | . | . | | |
| . | . | . | | |
| . | . | . | | |
| 29 | 346 | 3% | | |
| 30 | 428 | 3% | | |
| 31 | 182 | 1% | | |
| Total | 12,343 | 100% | | |

**Tabla 6: Variable C_EstacionInspecc**

| Nivel | Frecuencia | Porcentaje |
|---|---|---|
| invierno | 4,175 | 34% |
| otoño | 3,545 | 29% |
| verano | 2,953 | 24% |
| primavera | 1,670 | 14% |
| Total | 12,343 | 100% |

*Variable C_Inspecciones:* objetivo, el cual indica si el restaurante paso o no la inspección, como vemos, nuestros datos están compuestos por un 93% de restaurantes que si pasaron la inspección (Tabla 7).

**Tabla 7: Variable C_Inspecciones**

| Nivel | Frecuencia | Porcentaje |
|---|---|---|
| P | 11,432 | 93% |
| NP | 911 | 7% |
| Total | 12,343 | 100% |

*Variable C_MesInspecc:* esta variable indica el número de inspecciones realizados por cada mes, claramente vemos que los meses con más inspecciones son enero, febrero, marzo y octubre (Tabla 8).

**Tabla 8: Variable C_MesInspecc.**

| Nivel | Frecuencia | Porcentaje |
|---|---|---|
| 1 | 1,412 | 11% |
| 2 | 1,487 | 12% |
| 3 | 1,393 | 11% |
| 4 | 831 | 7% |
| 5 | 838 | 7% |
| 6 | 710 | 6% |
| 7 | 633 | 5% |
| 8 | 707 | 6% |
| 9 | 924 | 7% |
| 10 | 1,201 | 10% |
| 11 | 1,149 | 9% |
| 12 | 1,058 | 9% |
| Total | 12,343 | 100% |

*Variable NY_DescripcionComida:* esta variable tuvimos que reagruparla y unir el nivel de África con otros, debido a la poca frecuencia de restaurantes en ese nivel (Tabla 9).

**Tabla 9: Variable NY_DescripcionComida**

| Nivel | Frecuencia | Porcentaje | Nueva |
|---|---|---|---|
| America_N | 3,765 | 31% | 3,765 |
| Europa | 3,372 | 27% | 3,372 |
| Asia | 3,317 | 27% | 3,317 |
| America_S | 645 | 5% | 645 |
| Africa | 175 | 1% | 1,244 |
| Otros | 1,069 | 9% | |
| Total | 12,343 | 91% | |

*Variable NY_FechaInspeccion:* Como ya se había comentado, se ha realizado un filtro de tal manera que nuestra base de datos cuenta solo con un único registro de restaurantes inspeccionados desde el 01/02/2019 al 03/16/2020.

*Variable NY_TipoInspeccion:* en esta variable se observa que las inspecciones de tipo Cycle Inspection representan el 89%, mientras que las de tipo Pre-permit el 11% (Tabla 10).

**Tabla 10: Variable NY_TipoInspeccion**

| Nivel | Frecuencia | Porcentaje |
|---|---|---|
| Cycle Inspection / Initial Inspection | 7,294 | 59% |
| Cycle Inspection / Re-inspection | 3,755 | 30% |
| Pre-permit (Operational) / Initial Inspection | 710 | 6% |
| Pre-permit (Operational) / Re-inspection | 584 | 5% |
| Total | 12,343 | 100% |

*Variable WS_HorarioAtencion:* En la tabla 11, se observa que hay 1.020 valores perdidos, lo cual representa más del 5 % del total de observaciones, por lo que creamos un nuevo nivel "No_Consta", además unimos los niveles Mañana y Mañana_Tarde por tener pocas observaciones.

**Tabla 11: Variable WS_HorarioAtencion:**

| Nivel | Frecuencia | Porcentaje | Nueva |
|---|---|---|---|
| Mañana | 19 | 0.2% | 3,807 |
| Mañana_Tarde | 3,788 | 30.7% | |
| Tarde | 4,934 | 40.0% | 4,934 |
| Tarde_Noche | 1,682 | 13.6% | 1,682 |
| Todo_dia | 900 | 7.3% | 900 |
| No_Consta | 1,020 | 8.3% | 1,020 |
| Total | 12,343 | 100% | |

*Variable WS_NumDiasTrabajo:* para esta variable también tuvimos que crear el nivel "No_Consta", ya que los valores ausentes superan el 5% con respecto al total de observaciones, además los niveles 1,2,3,,4 y 5 los unimos en un solo nivel llamado 1_5 (Tabla 12).

**Tabla 12: Variable WS_NumDiasTrabajo**

| Nivel | Frecuencia | Porcentaje | Nueva | Frecuencia |
|---|---|---|---|---|
| 1 | 72 | 1% | 1_5 | 843 |
| 2 | 21 | 0% | | |
| 3 | 48 | 0% | | |
| 4 | 79 | 1% | | |
| 5 | 623 | 5% | | |
| 6 | 1,744 | 14% | 6 | 1,744 |
| 7 | 8,736 | 71% | 7 | 8,736 |
| No_Consta | 1,020 | 8% | No_Consta | 1,020 |
| Total | 12,343 | 100% | | |

*Variable WS_Precio: e*sta variable presenta más del 5% de valores ausentes respecto al total de las observaciones (Tabla 13) por lo que creamos un nuevo nivel "No_Consta".

**Tabla 13: Variable WS_Precio**

| Nivel | Frecuencia | Porcentaje | Nueva |
|---|---|---|---|
| No_Consta | 4,668 | 38% | 4,668 |
| PocoCostoso | 4,215 | 34% | 4,215 |
| NadaCostoso | 2,811 | 23% | 2,811 |
| Costoso | 512 | 4% | 649 |
| MuyCostoso | 137 | 1% | |
| Total | 12,343 | 100% | |

Ahora mostraremos los estadísticos de las variables cualitativas después de la depuración realizada en esta sección.

**Ilustración 13: Estadísticos de variables cualitativas después de ser depuradas**

| Variable | Tipo | Número de niveles | Ausente |
|---|---|---|---|
| C_Ciudad | C | 5 | 0 |
| C_EstacionInspecc | C | 4 | 0 |
| C_Inspecciones | C | 2 | 0 |
| C_MesInspecc | C | 12 | 0 |
| NY_TipoInspeccion | C | 4 | 0 |
| REP_C_DiaInspecc | C | 2 | 0 |
| REP_REP_NY_DescripcionComida | C | 5 | 0 |
| REP_REP_WS_Precio | C | 4 | 0 |
| REP_WS_HorarioAtencion | C | 5 | 0 |
| REP_WS_NumDiasTrabajo | C | 4 | 0 |

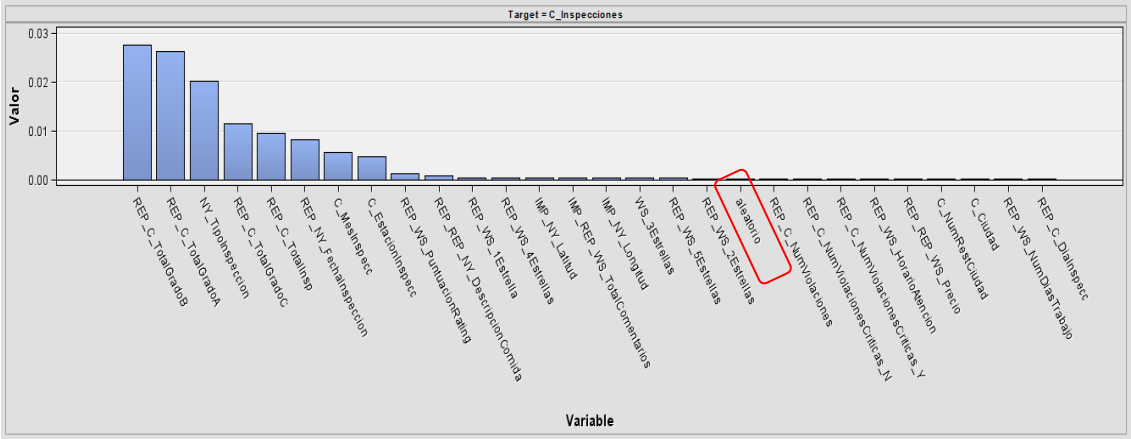
Como se observa en la Ilustración 13, las variables ya no presentan valores ausentes, ya que al ser mayores al 5% de total de las observaciones sea decidido consideran como un nivel más. Así mismo en algunas variables con muchos niveles se han reducido, ya que no tenían mucha representatividad.

## 5.4. Relación entre variables

En esta sección identificaremos que variables van a ser útiles para la predicción, para ello es necesario recordar lo siguiente: dos variables están relacionadas si, al conocer el valor de una de ellas para cierta observación, podemos sacar conclusiones sobre el valor de la otra variable sobre la misma observación (Calviño, 2019).

Ahora crearemos en SAS Miner una variable que tome valores aleatorios, de tal manera que servirá como una referencia al momento de visualizar el grafico de valor de variables. Por tanto, aquellas variables que se encuentren por debajo de la variable aleatoria, serán consideradas variables nada útiles.

**Ilustración 14 Valor de las variables**



En la ilustración 14 observamos que las variables *C_TotalGradoB* y *C_TotalGradoA* son las más importantes, esta relación es muy coherente ya que estas variables recogen información sobre el número de veces que el restaurante obtuvo una calificación de grado A o grado B en cada una de las inspecciones que se le realizó.

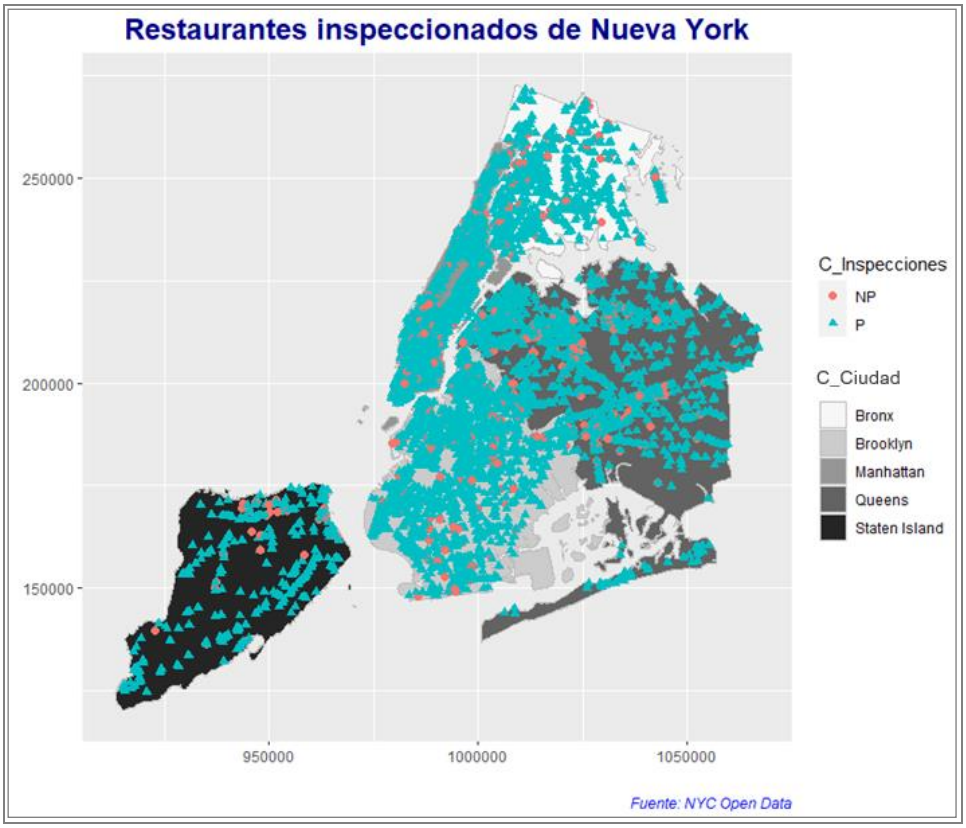
Por otro lado, observamos que las variables que se ubican por debajo de la variable aleatoria: *C_NumViolaciones, C_NumViolacionesCriticas_N, C_NumViolacionesCriticas_Y, WS_Horarioatencion, WS_Precio, C_NumRestCiudad, C_Ciudad, WS_NumDiastrabajo* y *C_DiaInspecc* no serán de utilidad para el modelo.

## 5.5. Análisis Geovisual

Ahora con la ayuda del software R, representaremos geográficamente a los restaurantes de la base de datos (véase el Anexo B: Elaboración de Mapas).

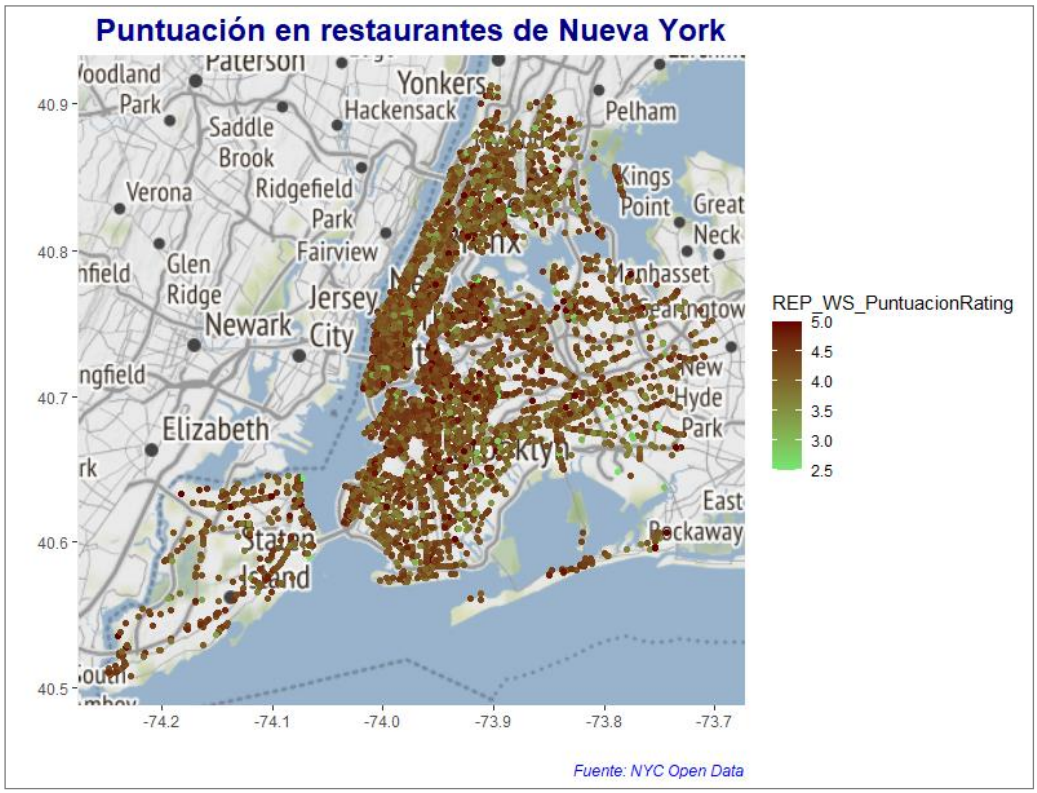
En el primer mapa (Ilustración 15) se ha plasmado la distribución de restaurantes por cada ciudad, además podemos apreciar que los triángulos de color turquesa son aquellos restaurantes que pasaron las inspecciones y los restaurantes que no pasaron están representados por puntos rojos. Además, no observamos ningún patrón geográfico (como ya lo habíamos notado anteriormente en la Ilustración 14, ya que la variable C_Ciudad ha quedado por debajo de la aleatoria).

**Ilustración 15: Mapa de restaurantes inspeccionados según Ciudad**



En la ilustración 16, veremos la puntuación asignada por parte de los consumidores a restaurante mediante Google Maps. La puntuación va desde 0 hasta 5, por lo que notamos que en la mayoría de los casos tienen una puntuación alta, por otro lado, en la ciudad de Staten Island, no hay muchos restaurantes con calificación.

**Ilustración 16: Restaurantes con puntuación de Google Maps**




## 5.6. Modelización

### 5.6.1. Selección de variables

Antes de comenzar aplicando los modelos de machine learning, realizaremos una selección de variables, para ello compilaremos la macro interactodolog (Anexo E: Macro de selección de variables) que a partir del remuestreo repetido presenta los mejores modelos seleccionados por el método Stepwise.

Cabe mencionar que la macro interactodolog calcula un listado de interacciones entre variables categóricas y continuas hasta de orden 2. Además de ello, la macro permite personalizar el número de interacciones, es decir, si asignamos interac=1, obtendremos interacciones de orden 2 y en caso de asignar interac=0, obtendremos la relación de variables ordenados de manera ascendente de AIC (cuanto más pequeño es mejor), lo cual, nos permitirá visualizar la importancia de las mismas.

Recordemos que todas las variables son originales, no se ha realizado ninguna transformación hasta el momento Además en la sección *5.4 Relación entre*

*variables* ya habíamos identificados con el Miner que, de las 28 variables, 9 serían rechazadas por ser de poca importancia quedando 18 variables.
Ahora con la macro ya mencionada y con interacción=0, se evaluó las variables de mayor importancia, del cual se obtuvo que las 10 primeras variables son de importancia para la variable objetivo (Tabla 14).

**Tabla 14: Selección de variables**

| N° | Variables | ChiSq | Prob Chisq | AIC | percentconcord |
|---|---|---|---|---|---|
| 1 | NY_TipoInspeccion | 1869.64 | <.0001 | 4076.09 | 73.50 |
| 2 | REP_C_TotalGradoB | 1369.57 | <.0001 | 4572.16 | 78.80 |
| 3 | REP_C_TotalGradoA | 672.70 | <.0001 | 5269.03 | 70.10 |
| 4 | REP_C_TotalGradoC | 661.40 | <.0001 | 5280.33 | 67.40 |
| 5 | REP_C_TotalInsp | 594.05 | <.0001 | 5347.68 | 73.10 |
| 6 | C_MesInspecc | 523.05 | <.0001 | 5438.68 | 67.80 |
| 7 | C_EstacionInspecc | 393.62 | <.0001 | 5552.11 | 55.10 |
| 8 | REP_REP_NY_DescripcionComida | 54.42 | <.0001 | 5893.31 | 45.30 |
| 9 | REP_WS_PuntuacionRating | 40.44 | <.0001 | 5901.29 | 52.90 |
| 10 | WS_3Estrella | 40.44 | <.0001 | 5901.29 | 52.90 |
| 11 | REP_NY_FechaInspeccion | 14.46 | 0.0002 | 5925.74 | 52.70 |
| 12 | REP_WS_1Estrella | 12.85 | 0.0003 | 5928.88 | 52.60 |
| 13 | IMP_NY_Latitud | 7.68 | 0.0056 | 5934.05 | 52.10 |
| 14 | REP_WS_2Estrellas | 1.59 | 0.2068 | 5940.14 | 48.70 |
| 15 | IMP_NY_Longitud | 1.16 | 0.2825 | 5940.57 | 51.40 |
| 16 | IMP_REP_WS_TotalComentarios | 1.00 | 0.3172 | 5940.73 | 50.00 |
| 17 | REP_WS_4Estrellas | 0.69 | 0.4071 | 5941.04 | 49.50 |
| 18 | REP_WS_5Estrellas | 0.06 | 0.8118 | 5941.67 | 49.70 |

Luego se vuelve a ejecutar macro **interacttodolog** pero con interacción igual a 1 para las 10 variables seleccionadas en el paso anterior, y obtuve un total de 40 variables entre variables con interacciones (30) y originales (10) (Tabla 15).

Luego con estas 40 variables se ejecutó la macro randomseleclog, el cual realiza un método stepwise repetidas veces con diferentes archivos train, del que se dos interacciones significativas. Por lo que nos quedaríamos con 10 variables originales y 2 variable con interacciones (Tabla 16).

**Tabla 15: Principales variables con interacción**

| Obs | Efecto | Count | Percent |
|---|---|---|---|
| 1 | NY_TipoInspeccion*REP_C_TotalGradoB | 11 | 50% |
| 2 | NY_TipoInspeccion*Rep_WS_PuntuacionRating | 11 | 50% |

**Tabla 16: Variables con interacción**

| N° | Variables | ChiSq | Prob Chisq | AIC | percentcon |
|---|---|---|---|---|---|
| 1 | NY_TipoInspeccion*REP_C_TotalGradoB | 2311.81 | <.0001 | 3633.92 | 84.9 |
| 2 | NY_TipoInspeccion*REP_C_TotalInsp | 1940.72 | <.0001 | 4005.01 | 90 |
| 3 | NY_TipoInspeccion | 1869.64 | <.0001 | 4076.09 | 73.5 |
| 4 | NY_TipoInspeccion*REP_WS_PuntuacionRating | 1775.43 | <.0001 | 4170.3 | 82.3 |
| 5 | NY_TipoInspeccion*WS_3Estrella | 1775.43 | <.0001 | 4170.3 | 82.3 |
| 6 | C_MesInspecc*REP_C_TotalGradoB | 1414.21 | <.0001 | 4547.52 | 81 |
| 7 | C_EstacionInspecc*REP_C_TotalGradoB | 1382.70 | <.0001 | 4563.03 | 80.2 |
| 8 | REP_C_TotalGradoB | 1369.57 | <.0001 | 4572.16 | 78.8 |
| 9 | NY_TipoInspeccion*REP_C_TotalGradoC | 1276.67 | <.0001 | 4669.06 | 77 |
| 10 | NY_TipoInspeccion*REP_C_TotalGradoA | 1207.71 | <.0001 | 4738.02 | 68.3 |
| 11 | C_EstacionInspecc*NY_TipoInspeccion | 917.54 | <.0001 | 5040.19 | 75.9 |
| 12 | C_MesInspecc*REP_C_TotalInsp | 837.67 | <.0001 | 5124.06 | 79.9 |
| 13 | C_MesInspecc*NY_TipoInspeccion | 852.48 | <.0001 | 5153.25 | 79.3 |
| 14 | C_EstacionInspecc*REP_C_TotalInsp | 732.62 | <.0001 | 5213.11 | 78.3 |
| 15 | C_MesInspecc*REP_C_TotalGradoC | 743.24 | <.0001 | 5218.49 | 70.6 |
| 16 | REP_C_TotalGradoA | 672.70 | <.0001 | 5269.03 | 70.1 |
| 17 | C_EstacionInspecc*REP_C_TotalGradoC | 675.73 | <.0001 | 5270.01 | 69.3 |
| 18 | REP_C_TotalGradoC | 661.40 | <.0001 | 5280.33 | 67.4 |
| 19 | REP_C_TotalInsp | 594.05 | <.0001 | 5347.68 | 73.1 |
| 20 | C_MesInspecc | 523.05 | <.0001 | 5438.68 | 67.8 |
| 21 | C_EstacionInspecc*C_MesInspecc | 530.71 | <.0001 | 5439.02 | 68.3 |
| 22 | C_MesInspecc*REP_WS_PuntuacionRating | 499.69 | <.0001 | 5462.04 | 71.2 |
| 23 | C_MesInspecc*WS_3Estrella | 499.69 | <.0001 | 5462.04 | 71.2 |
| 24 | NY_TipoInspeccion*REP_REP_NY_DescripcionComida | 482.31 | <.0001 | 5481.42 | 72.2 |
| 25 | REP_REP_NY_DescripcionComida*REP_C_TotalGradoB | 460.84 | <.0001 | 5486.89 | 74.1 |
| 26 | C_EstacionInspecc | 393.62 | <.0001 | 5552.11 | 55.1 |
| 27 | C_EstacionInspecc*REP_WS_PuntuacionRating | 373.71 | <.0001 | 5572.02 | 67.2 |
| 28 | C_EstacionInspecc*WS_3Estrella | 373.71 | <.0001 | 5572.02 | 67.2 |
| 29 | C_MesInspecc*REP_C_TotalGradoA | 186.31 | <.0001 | 5775.42 | 63.6 |
| 30 | REP_REP_NY_DescripcionComida*REP_C_TotalGradoC | 172.19 | <.0001 | 5775.54 | 63.5 |
| 31 | C_EstacionInspecc*REP_C_TotalGradoA | 127.71 | <.0001 | 5818.02 | 68.3 |
| 32 | REP_REP_NY_DescripcionComida*REP_C_TotalGradoA | 107.73 | <.0001 | 5840 | 67.3 |
| 33 | C_MesInspecc*REP_REP_NY_DescripcionComida | 173.30 | <.0001 | 5854.43 | 66.2 |
| 34 | REP_REP_NY_DescripcionComida*REP_C_TotalInsp | 79.00 | <.0001 | 5868.73 | 61.3 |
| 35 | C_EstacionInspecc*REP_REP_NY_DescripcionComida | 92.29 | <.0001 | 5871.44 | 61.7 |
| 36 | REP_REP_NY_DescripcionComida | 54.42 | <.0001 | 5893.31 | 45.3 |
| 37 | REP_REP_NY_DescripcionComida*REP_WS_Puntuacion | 54.33 | <.0001 | 5893.4 | 56.6 |
| 38 | REP_REP_NY_DescripcionComida*WS_3Estrella | 54.33 | <.0001 | 5893.4 | 56.6 |
| 39 | REP_WS_PuntuacionRating | 40.44 | <.0001 | 5901.29 | 52.9 |
| 40 | WS_3Estrella | 40.44 | <.0001 | 5901.29 | 52.9 |

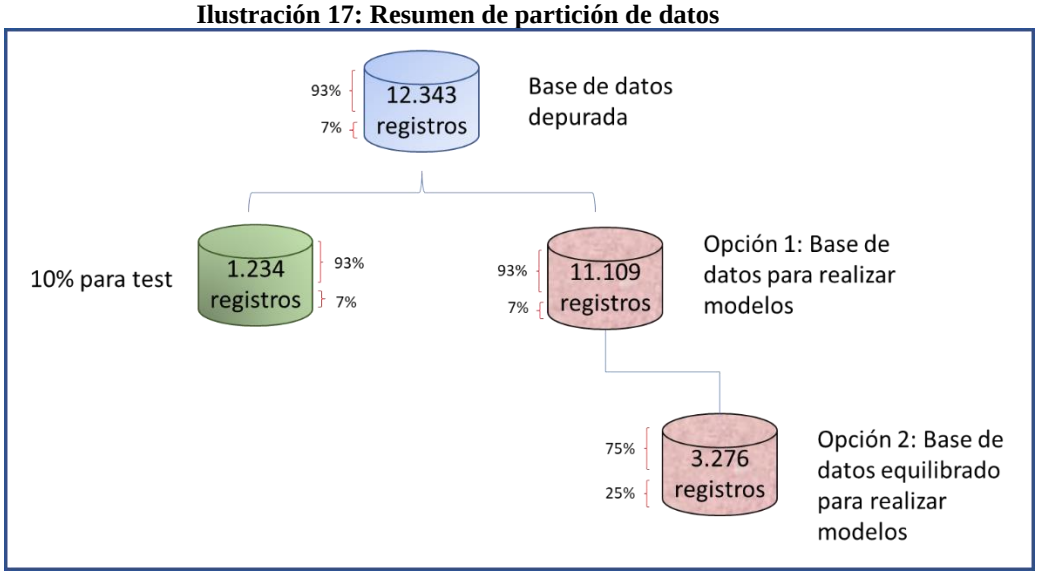
### 5.6.2. Partición de datos y undersampling

Antes de aplicar las diferentes técnicas de machine learning, hemos separado un 10% de nuestros datos (datos test) para aplicarlo en la fase de evaluación en el modelo seleccionado. Por otro lado, como se mencionó anteriormente, nuestro conjunto de datos no se encuentra equilibrado respecto a la variable objetivo, por ello hemos considerado realizar los modelos tanto con el conjunto de datos total como con un conjunto de datos que se encuentren equilibrados, ello nos permitirá observar si realmente conviene trabajar con nuestro conjunto de datos equilibrados o con el total de datos.

Para la obtención de datos equilibrados utilizamos la técnica de undersampling en el sotfware R (Anexo C: Undersampling), esta técnica consiste en eliminar aleatoriamente registros de la clase mayoritaria, quedándonos de esta manera

una segunda base de datos con 3.276 registros de los cuales el 25% está conformado por restaurantes que no pasaron la inspección y el 75% por restaurantes que sí pasaron la inspección, el resumen se puede visualizar en la Ilustración 17.

**Ilustración 17: Resumen de partición de datos**




### 5.6.3. Aplicación de Regresión Logística

Ahora empezaremos a realizar modelos de regresión logística binaria con ambos conjuntos de datos (total de datos y datos equilibrados).

Con las variables identificadas como las más importantes utilizaremos la macro cruzada logística en SAS 9.4 (Anexo F: Macro cruzada logística), para realizar los modelos de regresión logística con validación cruzada repetida y comparar los resultados a través de la tasa de fallo.

Como se observa en la Tabla 17, hemos decidido probar modelos con dos semillas finales diferentes para ambos conjuntos de datos, además de ejecutar modelos sin interacciones y con interacciones.

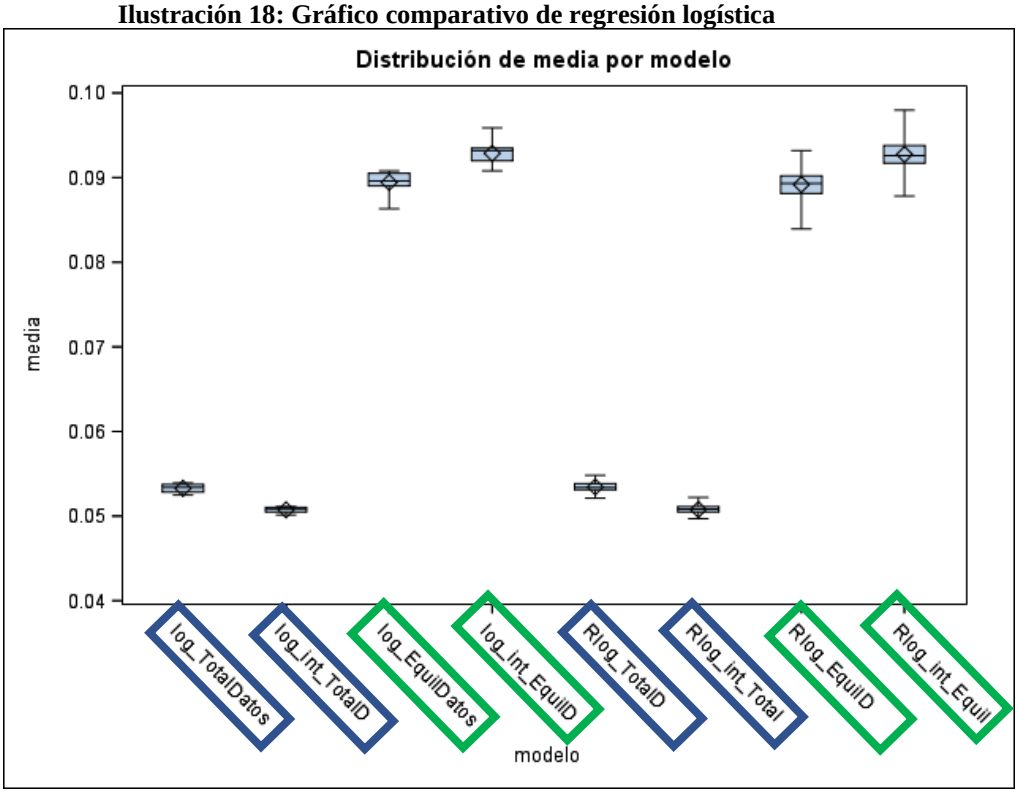
Respecto a las variables utilizadas en los modelos, cabe mencionar que inicialmente se probaron con todas las variables que quedaron por encima de la variable aleatoria (Ilustración 14), de ello se obtuvo como resultado que no todas variables incluidas nos aportan información para el modelo, lo cual ya lo habíamos identificado anteriormente al utilizar la macro **interacttodolog** (Tabla 14), por ello, se decidió que en adelante solo incluiremos el set de variables obtenidos por la macro en mención (10 variables).

Del grafico comparativo de boxplot (Ilustración 18), podemos observar que los modelos realizados en el conjunto de datos equilibrados (color azul) presentan la mayor tasa de error. Además, los modelos con interacciones (color verde) y con semilla de inicio 12345 y semilla final 12456 son más estables ya que la tasa de error se sitúa alrededor del 5.1%.

**Tabla 17: Modelos de Regresión Logística**

| Nombre modelo | Descripción modelo | Semilla inicio | Semilla final |
|---|---|---|---|
| log_totalDatos | Datos completos- var. seleccionadas. | 12345 | 12355 |
| log_Int_TotalD | Datos completos- var. seleccionadas. - var. interacción. | 12345 | 12355 |
| log_EquiD | Datos equilibrados- var. seleccionadas. | 12345 | 12355 |
| log_int_EquiD | Datos equilibrados- var. seleccionadas. - var. interacción. | 12345 | 12355 |
| Rlog_totalDatos | Datos completos- var. seleccionadas. | 12345 | 12456 |
| Rlog_Int_TotalD | Datos completos- var. seleccionadas. - var. interacción. | 12345 | 12456 |
| Rlog_EquiD | Datos equilibrados- variables seleccionadas. | 12345 | 12456 |
| Rlog_int_EquiD | Datos equilibrados- variables seleccionadas y variables de interacción. | 12345 | 12456 |

**Ilustración 18: Gráfico comparativo de regresión logística**




De la tabla 18, podemos observar el aporte de información de cada una de las variables del modelo ganador. Las variables que tienen el valor más alto de Chi-cuadrado de Wald son las de mayor importancia para el modelo, por lo

que la interacción REP_C_TotalInsp*NY_TipoInspeccion y las variables C_MesInspecc y REP_C_TotalGradoA serían las que aporten más al modelo.

**Tabla 18: Análisis de efecto del modelo ganador**

| Efecto | DF | Chi-cuadrado de Wald | Pr > ChiSq |
|---|---|---|---|
| C_EstacionInspecc | 3 | 2.956 | 0.399 |
| C_MesInspecc | 11 | 23.275 | 0.016 |
| NY_TipoInspeccion | 3 | 0.553 | 0.907 |
| REP_REP_NY_Descripci | 4 | 10.846 | 0.028 |
| REP_C_TotalGradoB | 1 | 0.143 | 0.706 |
| REP_C_TotalGradoC | 1 | 4.277 | 0.039 |
| REP_C_TotalGradoA | 1 | 17.889 | <.0001 |
| REP_C_TotalInsp | 1 | 1.633 | 0.201 |
| REP_WS_PuntuacionRating | 1 | 0.000 | 0.984 |
| WS_3Estrellas | 1 | 1.475 | 0.225 |
| REP_C_TotalInsp*NY_ TipoInspeccion | 2 | 31.462 | <.0001 |
| REP_WS_PuntuacionRating *NY_TipoInspeccion | 3 | 0.018 | 0.999 |

Ahora interpretaremos algunos coeficientes mostrados en la Tabla 19. El logaritmo de los odds de que un restaurante no pase una inspección está positivamente relacionado con la puntuación obtenida en la variable Rep_C_TotalGradoC (coeficiente 0.28). Esto significa que, por cada unidad que se incremente la variable Rep_C_TotalGradoC ($e^{0.28} = 1.32$), los odds de que un restaurante no pase la inspección se incrementa en promedio 1.32 unidades. Recordemos que la variable Rep_C_TotalGradoC indica el número de veces que el establecimiento ha sido clasificado en con esa categoría en inspecciones pasadas.

Así mismo, por cada unidad que se incremente la variable Rep_C_TotalGradoB ($e^{-0.14} = 0.869$), los odds de que un restaurante no pase la inspección se incrementa en promedio 0.86 unidades.

Y respecto a la variable C_EstaciónInspección, las posibilidades se multiplican por 1.63 ($e^{0.49}$) de que el restaurante no pase la inspección en el invierno en comparación con la estación de verano.

**Tabla 19: Análisis de Máximo Likelihood Estimates**

| Parámetros | Niveles | DF | Estimate |
|---|---|---|---|
| Intercept | | 1 | 5.05 |
| REP_C_TotalGradoA | | 1 | -0.86 |
| REP_C_TotalGradoB | | 1 | -0.14 |
| REP_C_TotalGradoC | | 1 | 0.28 |
| REP_C_TotalInsp | | 1 | 0.08 |
| REP_WS_PuntuacionRat | | 1 | 0.46 |
| WS_3Estrellas | | 1 | 0.00 |
| REP_C_Tot*NY_TipoIns | Cycle Inspection/InitialInspection | 1 | 0.71 |
| REP_C_Tot*NY_TipoIns | Cycle Inspection/Re-inspection | 1 | 0.51 |
| REP_C_Tot*NY_TipoIns | Pre-permit (Operational) /InitialInspection | 0 | 0.00 |
| REP_WS_Pu*NY_TipoIns | Cycle Inspection/InitialInspection | 1 | 0.14 |
| REP_WS_Pu*NY_TipoIns | Cycle Inspection/Re-inspection | 1 | -0.08 |
| REP_WS_Pu*NY_TipoIns | Pre-permit (Operational) /InitialInspection | 1 | -0.04 |
| C_EstacionInspecc | invierno | 1 | 0.49 |
| C_EstacionInspecc | otoño | 1 | 0.12 |
| C_EstacionInspecc | primavera | 1 | -0.85 |
| C_MesInspecc | 1 | 1 | -0.11 |
| C_MesInspecc | 2 | 1 | 0.37 |
| C_MesInspecc | 3 | 1 | 1.70 |
| C_MesInspecc | 4 | 1 | 0.03 |
| C_MesInspecc | 5 | 1 | 0.24 |
| C_MesInspecc | 6 | 1 | -0.45 |
| C_MesInspecc | 7 | 1 | -0.54 |
| C_MesInspecc | 8 | 1 | -0.28 |
| C_MesInspecc | 9 | 1 | -0.06 |
| C_MesInspecc | 10 | 1 | -0.40 |
| C_MesInspecc | 11 | 1 | -0.29 |
| NY_TipoInspeccion | Cycle Inspection/InitialInspection | 1 | 5.30 |
| NY_TipoInspeccion | Cycle Inspection/Re-inspection | 1 | -6.25 |
| NY_TipoInspeccion | Pre-permit (Operational) /InitialInspection | 1 | 5.87 |
| REP_REP_NY_Descripci | America_N | 1 | 0.10 |
| REP_REP_NY_Descripci | America_S | 1 | -0.16 |
| REP_REP_NY_Descripci | Asia | 1 | 0.28 |
| REP_REP_NY_Descripci | Europa | 1 | -0.04 |

### 5.6.4. Aplicación de Redes Neuronales

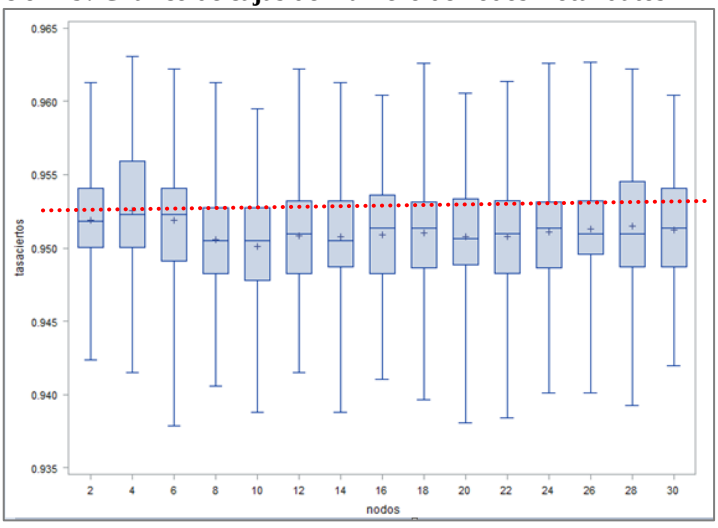
Ahora realizaremos modelos de redes neuronales en el software SAS 9.4, estos modelos se ejecutarán con las mismas variables que se usaron para los modelos de regresión logística a excepción de las interacciones.

Debido a que los modelos de redes neuronales son modelos paramétricos es necesario tunearlos, para ello evaluaremos lo siguiente:

- Numero de nodos
- Algoritmo de optimización más adecuado
- Función de activación (tangente (TAN), tangente hiperbólica (TANH), lineal (LIN), seno (SEN) y arco tangente (ARC))
- Early Stopping

Como ya lo mencionamos, el primer paso es investigar el número de nodos, por tanto, utilizaremos la macro variar (Anexo G: Macro variar) para obtener un gráfico de cajas con diferente número de nodos (variando de 2 en dos), estas redes se comparan a través de validación cruzada repetida. Para el número de nodos tendremos en cuenta que en una red el número de parámetros es igual a h(k+1) + h+1, donde h es el número de nodos ocultos y k es el número de variables independientes, se recomienda como mínimo 30 registros por cada parámetro para evitar sobreajustes en el modelo. Como contamos con 11,109 registros en nuestros datos totales y 10 variables independientes, por tanto, probaremos desde 2 nodos (444 registros) hasta 30 nodos (30 registros).

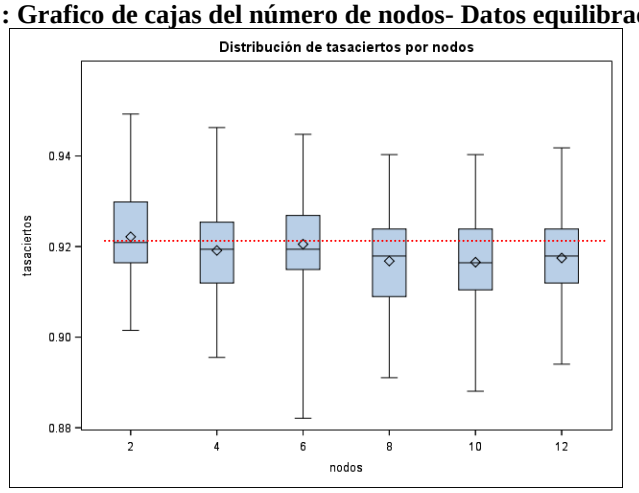
**Ilustración 19: Grafico de cajas del número de nodos-Total datos**




Como venimos trabajando con dos conjuntos de datos, primero hemos ejecutado la macro para el total de datos obteniendo el grafico de cajas según número de nodos (Ilustración 19), donde se observa que las redes con mayor tasa de acierto se alcanzan con 6, 4 y hasta con 2 nodos.

Ahora nos fijaremos en el número de nodos ideal para el conjunto de datos equilibrado, en este caso al tener menor cantidad de observaciones, probaremos desde 2 a 12 nodos (Ilustración 20).

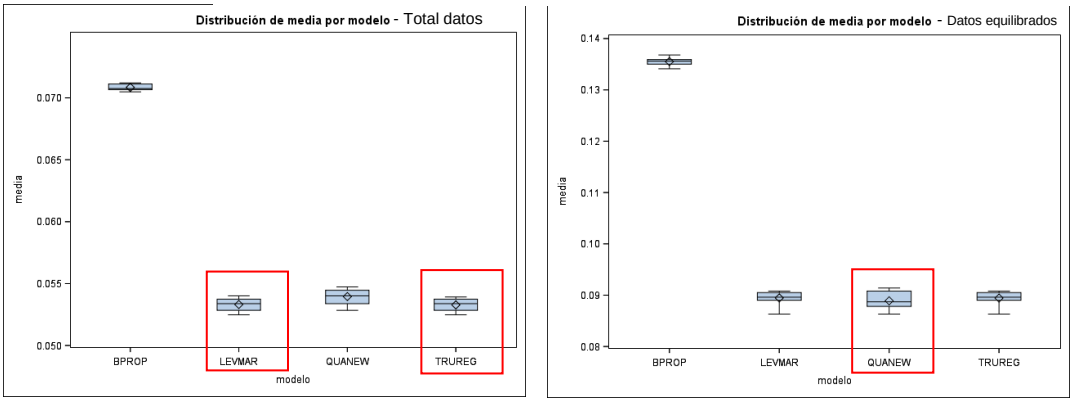
**Ilustración 20: Grafico de cajas del número de nodos- Datos equilibrado**




En este caso, vemos que la mejor red se alcanza con solo 2 nodos, esto tiene mucho sentido, ya que en este conjunto de datos solo contamos con 3.348 observaciones y al trabajar con un numero de nodo mayor podríamos sobre ajustar el modelo.

Respecto al algoritmo de optimización empezaremos evaluando para el primer conjunto de datos (total de datos), para ello utilizaremos la macro algovalcruza (Anexo H: Macro algoritmo de optimización). Primero ejecutaremos la macro para 6 nodos, luego para 4 nodos y por último para 2 nodos, en los tres casos nuestro gráfico de cajas a coincidido en valor, por lo que solo colocaremos uno de ellos (Ilustración 21-imagen izquierda), de los cuatro algoritmos Backpropagation (Bprop), Levenberg-Marquardt (Levmar), Quasi-Newton (Quanew) y Trust región (Trureg) observamos que la mejor red se alcanza con el algoritmo Levmar y Trureg. Por otro lado, en la Ilustración 21-grafico derecho hemos ejecutamos la macro con el conjunto de datos equilibrado para 2 nodos y observamos que la mejor red se da con el algoritmo Quanew.

**Ilustración 21:Grafico de cajas de algoritmo de optimización-**




En este siguiente paso se evaluó con la %macro activalcruza (Anexo I: Macro función de activación) la función de activación con validación cruzada y el early stopping con la macro %redneuronalbinaria (Anexo J: Early stopping), con esta

técnica se dividen los datos en entrenamiento y validación y se detiene el proceso de estimación cuando el error en los datos de validación comienza a incrementarse.

Tras la ejecución de las macros mencionadas anteriormente para la identificación del número de nodos, algoritmo de optimización, función de activación y early, se obtuvo las siguientes combinaciones para la construcción de las redes neuronales (Tabla 20).

**Tabla 20: Parámetros para la construcción de redes neuronales**

| Conjunto de datos | Numero de nodos | Algoritmo de optimización | Función de activación | Early |
|---|---|---|---|---|
| Total de datos | 6 | Levmar | Sof | 16 |
| | 6 | Trureg | Sof | 22 |
| | 4 | Levmar | Sof | 18 |
| | 4 | Trureg | Sin | 19 |
| | 2 | Levmar | Arc | 30 |
| | 2 | Trureg | Sin | 13 |
| Datos equilibrados | 2 | Quanew | Arc | 25 |

Una vez identificado estos principales parámetros para la construcción de las redes neuronales en ambos conjuntos de datos, procedimos a ejecutar la macro %cruzadabinarianeural (Anexo K: Redes neuronales) con las siguientes combinaciones (Tabla 21).

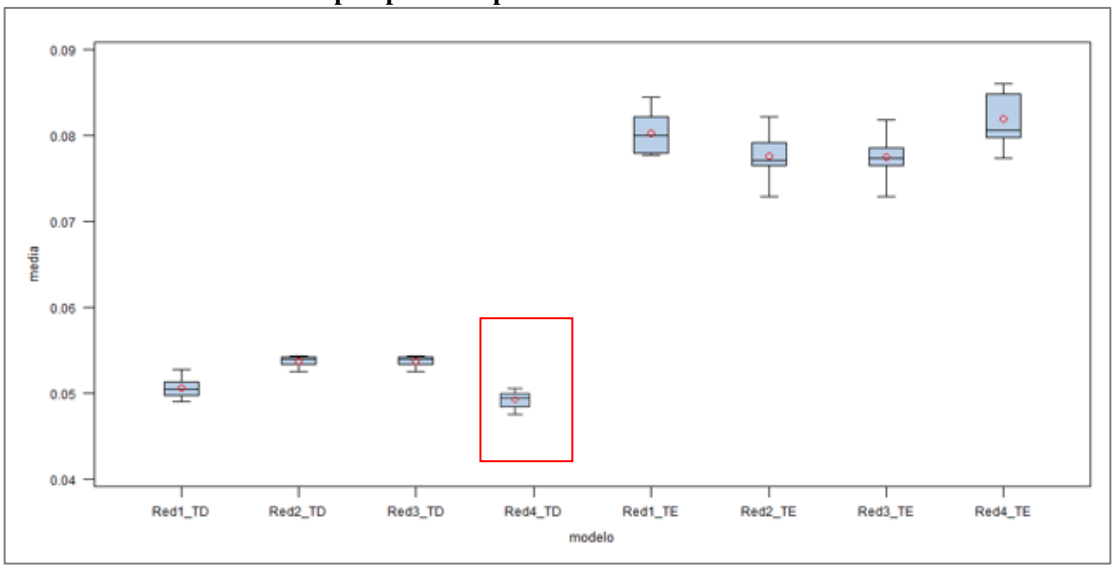
**Tabla 21: Modelos de redes neuronales para ambos conjuntos de datos**

| Modelo | Conjunto de datos | Numero de nodos | Algoritmo de optimización | Función de activación | Early |
|---|---|---|---|---|---|
| Red1_TD | Total de datos | 6 | Levmar | Sof | 16 |
| Red2_TD | | 6 | Trureg | Sof | 22 |
| Red3_TD | | 4 | Levmar | Sof | 18 |
| Red4_TD | | 4 | Trureg | Sin | 19 |
| Red1_TE | Datos equilibrados | 2 | Trureg | Sin | 13 |
| Red2_TE | | 2 | Levmar | Arc | 30 |
| Red3_TE | | 2 | Quanew | Arc | 25 |
| Red4_TE | | 2 | Quanew | Arc | - |

En la Ilustración 22, mostramos un gráfico de boxplot comparando los modelos de redes neuronales en ambos conjuntos de datos, y observamos que los modelos realizados para el total de los datos presentan menor tasa de error en comparación con los modelos realizados con el conjunto de datos equilibrados. Además, cabe mencionar que la tasa de error es similar a la de Regresión Logística.

De todos los modelos ejecutados con el total de datos, se observa que la red neuronal con 4 nodos, algoritmo de optimización=Trureg, función de activación=Sin y early=19 es el mejor.

**Ilustración 22: Grafico de boxplot para comparar modelos de Redes neuronales**




### 5.6.5. Aplicación de Random forest y Bagging

A continuación, construiremos modelos de Random forest.
Como se mencionó anteriormente esta técnica está basada en árboles, incorpora aleatoriamente las variables a utilizar para segmentar cada nodo del árbol.

Ejecutaremos la macro %*cruzadarandomforestbin* (Anexo L: Random forest) para ambos conjuntos de datos con las 10 variables seleccionadas de la Tabla 14.

La macro %*cruzadarandomforestbin* realiza validación cruzada repetida para la variable dependiente binaria.

Los parámetros que dejaremos fijo será el número máximo de divisiones del nodo que será 2, ya que solo de busca construir arboles binarios, luego elegiremos una profundidad igual a 10, con los demás parámetros realizaremos diferentes combinaciones para ambos conjuntos de datos, tal como mostraremos en la Tabla 22.

Luego con los modelos descritos en la Tabla 22 se realizó la validación cruzada obteniendo el grafico de boxplot (Ilustración 23).

En este grafico (Ilustración 23) se muestran los mejores modelos obtenidos de Random forest y bagging tanto para el total de datos como para los datos equilibrados. Vemos que, para el total de datos, el mejor modelo es rf1_TD y para los datos equlibrados el mejor modelo es rf4_TE, ya que representan un equilibrio entre tasa de fallo y variabilidad. Y entre ambos modelos elegidos

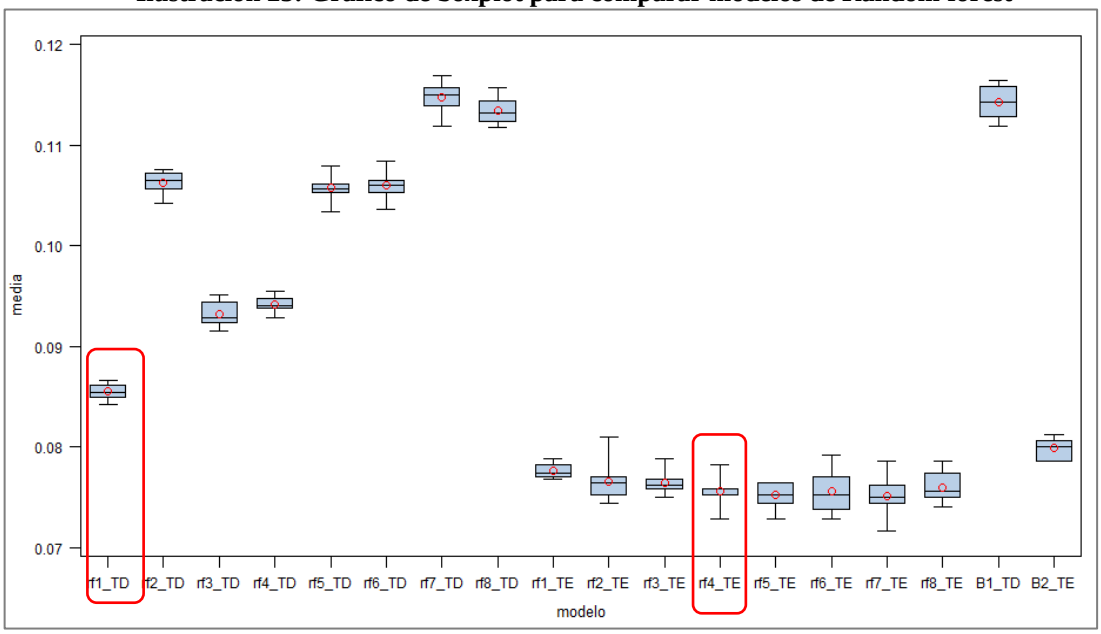
podemos decir que el modelo con menor tasa de fallo se da para el conjunto de datos equilibrados.

En general estos modelos se encuentran son peores que los modelos de regresión logística.

**Tabla 22: Resumen de parámetros para Random forest**

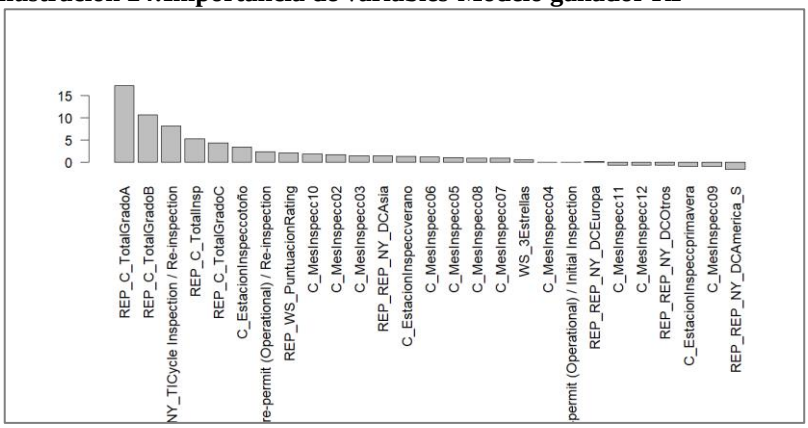
| Modelo | Conjunto de datos | Máx tree | Var | Porcent bag | Tamaño hoja | P valor |
|---|---|---|---|---|---|---|
| rf1_TD | Total de datos | 700 | 3 | 0.70 | 20 | 0.05 |
| rf2_TD | | 200 | 5 | 0.70 | 15 | 0.10 |
| rf3_TD | | 100 | 3 | 0.70 | 15 | 0.20 |
| rf4_TD | | 200 | 3 | 0.75 | 15 | 0.20 |
| rf5_TD | | 400 | 4 | 0.70 | 10 | 0.20 |
| rf6_TD | | 850 | 5 | 0.80 | 5 | 0.05 |
| rf7_TD | | 850 | 12 | 0.70 | 5 | 0.20 |
| rf8_TD | | 900 | 13 | 0.60 | 15 | 0.20 |
| rf1_TE | Datos equilibrados | 700 | 3 | 0.70 | 20 | 0.05 |
| rf2_TE | | 200 | 5 | 0.70 | 15 | 0.10 |
| rf3_TE | | 100 | 3 | 0.70 | 15 | 0.20 |
| rf4_TE | | 200 | 3 | 0.75 | 15 | 0.20 |
| rf5_TE | | 400 | 4 | 0.70 | 10 | 0.20 |
| rf6_TE | | 850 | 5 | 0.80 | 5 | 0.05 |
| rf7_TE | | 850 | 12 | 0.70 | 5 | 0.20 |
| rf8_TE | | 900 | 13 | 0.60 | 15 | 0.20 |

**Ilustración 23: Grafico de boxplot para comparar modelos de Random forest**



Respecto a la importancia de variables para el modelo más estable en el conjunto de datos equilibrados (Ilustración 24), notamos que las variables Rep_C_TotalGradoA, Rep_C_TotalGradoB y NY_TipoInspección_ Cycle Inspection / Re-inspection serían las que mayor información brinden al modelo.

**Ilustración 24:Importancia de variables-Modelo ganador RF**




### 5.6.6. Aplicación de Gradient Boosting

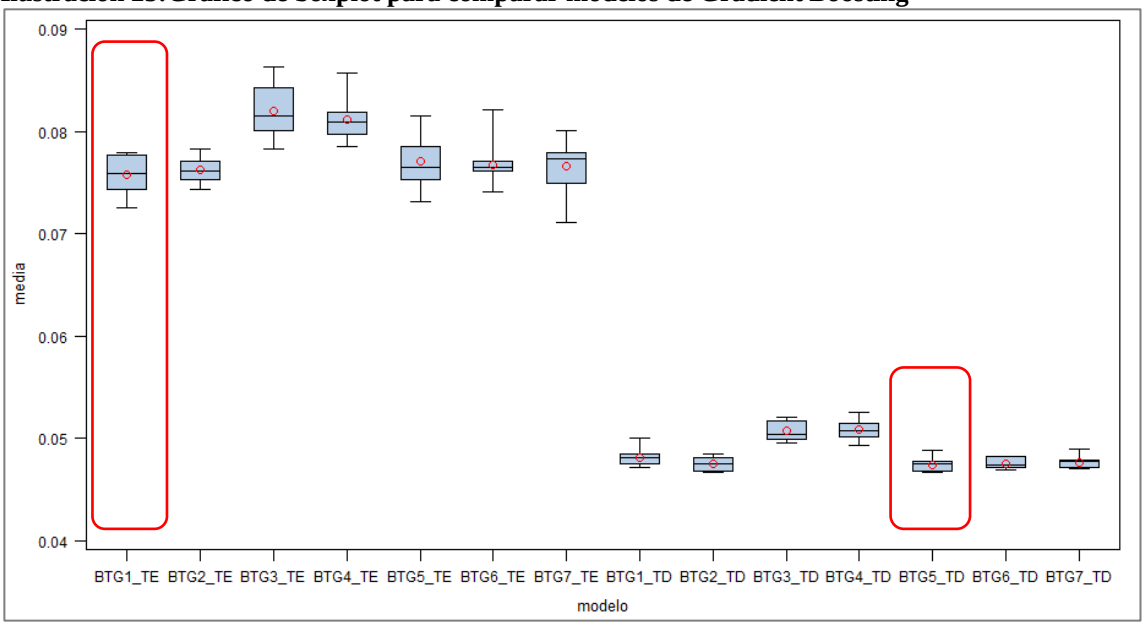
Ahora analizaremos modelos de Gradient Boosting con la macro %cruzadatreeboostbin (Anexo LL: Gradient boosting), este algoritmo consiste en construir un bosque modificando predicciones iniciales e ir minimizando los residuos en la dirección de decrecimiento.
Para este modelo serán incluirán todas las variables tratadas ya que el modelo hace su propia selección de variables. Los modelos construidos se mostrarán en la Tabla 23.

**Tabla 23: Resumen de parámetros para modelos de Gradient Boosting**

| Modelo | Conjunto de datos | Tamaño mín. de la hoja | Mín. núm. de observaciones para div un nodo | Parámetro de regularización |
|---|---|---|---|---|
| BTG1_TD | Total de datos | 5 | 50 | 0.10 |
| BTG2_TD | | 5 | 20 | 0.03 |
| BTG3_TD | | 5 | 20 | 0.20 |
| BTG4_TD | | 15 | 20 | 0.20 |
| BTG5_TD | | 15 | 20 | 0.10 |
| BTG6_TD | | 5 | 20 | 0.10 |
| BTG7_TD | | 25 | 15 | 0.10 |
| BTG1_TE | Datos equilibrados | 5 | 50 | 0.10 |
| BTG2_TE | | 5 | 20 | 0.03 |
| BTG3_TE | | 5 | 20 | 0.20 |
| BTG4_TE | | 15 | 20 | 0.20 |
| BTG5_TE | | 15 | 20 | 0.10 |
| BTG6_TE | | 5 | 20 | 0.10 |
| BTG7_TE | | 25 | 15 | 0.10 |

Ahora mostraremos los grafico de box plot comparativo para ambos conjuntos de datos (Ilustración 25):

**Ilustración 25:Grafico de boxplot para comparar modelos de Gradient Boosting**




En este grafico box-plot observamos que el modelo con menor tasa de fallo para el conjunto de datos equilibrados es el BTG1_TE y para el conjunto total de datos es BTG5_TD. Por otro lado, debemos mencionar que hasta el momento con este algoritmo se obtiene tasas de fallos inferiores a las de la Regresión Logística.

### 5.6.7. Aplicación de Support Vector Machine

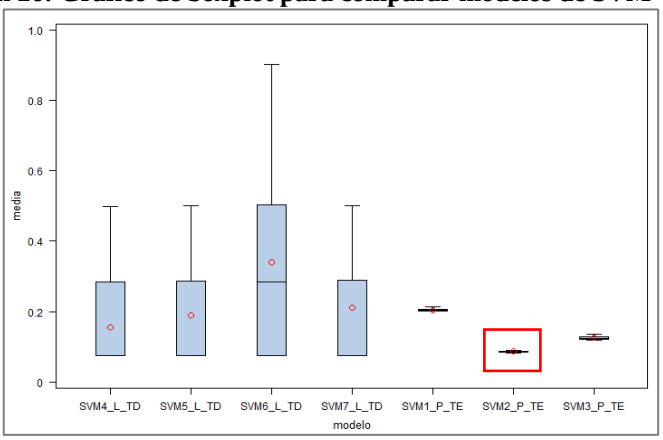
Ahora ejecutaremos la macro %cruzadaSVMbin (Anexo M: Sopport vector machine) con el set de las 10 variables seleccionadas de la tabla 14. Este algoritmo construye un modelo que separa los puntos nuevos (cuya clase desconocemos) en una categoría u otra. Como comentamos anteriormente, existen diferentes tipos de funciones kernel para llevar a cabo la separación, pero en nuestro caso como tenemos dos conjuntos de datos diferentes, se observó que, para el total de datos la única que converge y nos permite llevar a cabo la separación en SAS es SVM lineal, y para el conjunto de datos equilibrado es SVM polymon, por lo que se desarrollaron las siguientes combinaciones:

- ✓ Total de datos:
  - SVM4_L_TD: kernel=linear, c=15.
  - SVM5_L_TD: kernel=linear, c=25.
  - SVM6_L_TD: kernel=linear, c=20.
  - SVM7_L_TD: kernel=linear, c=30.
- ✓ Datos Equilibrados:

- SVM1_P_TE: kernel=Polynom, k_par=3, c=10.
- SVM2_P_TE: kernel=Polynom, k_par=2, c=25.
- SVM3_P_TE: kernel=Polynom, k_par=3, c=0.5.

En la Ilustración 26 se muestran los gráficos de boxplot de los modelos más estables de SVM en ambos conjuntos de datos. De los cuales, los modelos para el total de datos muestran una gran varianza y sesgo en comparación con los modelos realizados en los datos equilibrados. Por tanto, para este algoritmo diremos que el modelo más estable es SVM2_P_TE con una tasa de error menor de 0.2.

**Ilustración 26: Grafico de boxplot para comparar modelos de SVM**

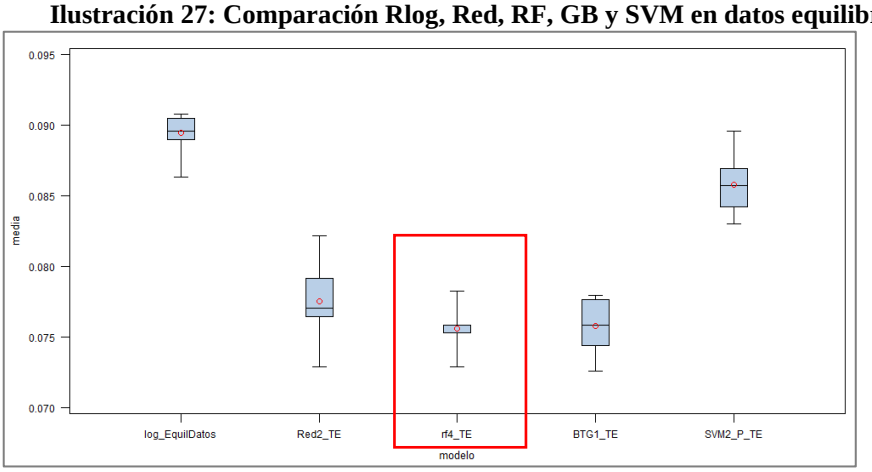



### 5.6.8. Comparar modelos

Una vez tuneados y estimados cinco algoritmos de Machine Learning (regresión logística, red neuronal, random forest, gradient boosting y SVM) elegimos los mejores modelos de cada algoritmo y los comparamos con validación cruzada repetida de 4 grupos y 10 semillas en ambos conjuntos de datos.

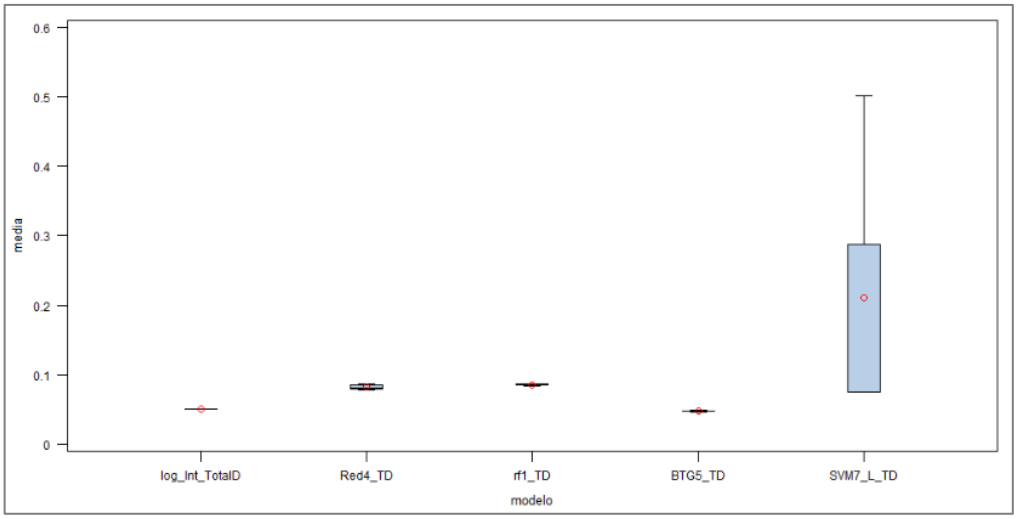
Empezaremos con el conjunto de datos equilibrados, compararemos el mejor modelo obtenido en cada uno de los algoritmos utilizados en este trabajo. Por lo tanto, en el gráfico comparativo de box-plot (Ilustración 27), se observa que el modelo con menor tasa de fallo y buena varianza es random forest.

**Ilustración 27: Comparación Rlog, Red, RF, GB y SVM en datos equilibrados**



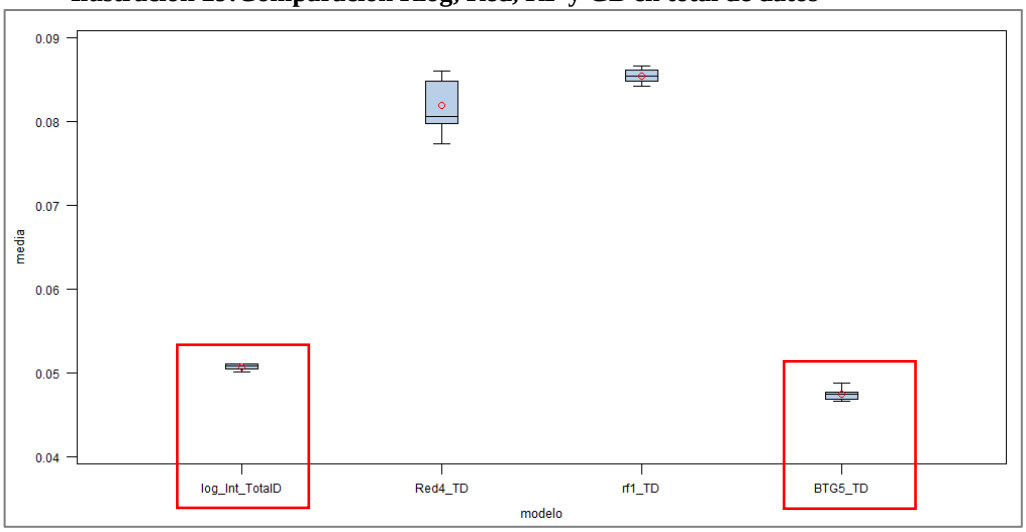
Respecto a los modelos obtenidos con el total de datos, el modelo SVM tiene gran varianza por lo que no permite tener una buena visibilidad de los otros cuatro modelos (Ilustración 28).

**Ilustración 28: Comparación Rlog, Red, RF, GB y SVM en total de datos**



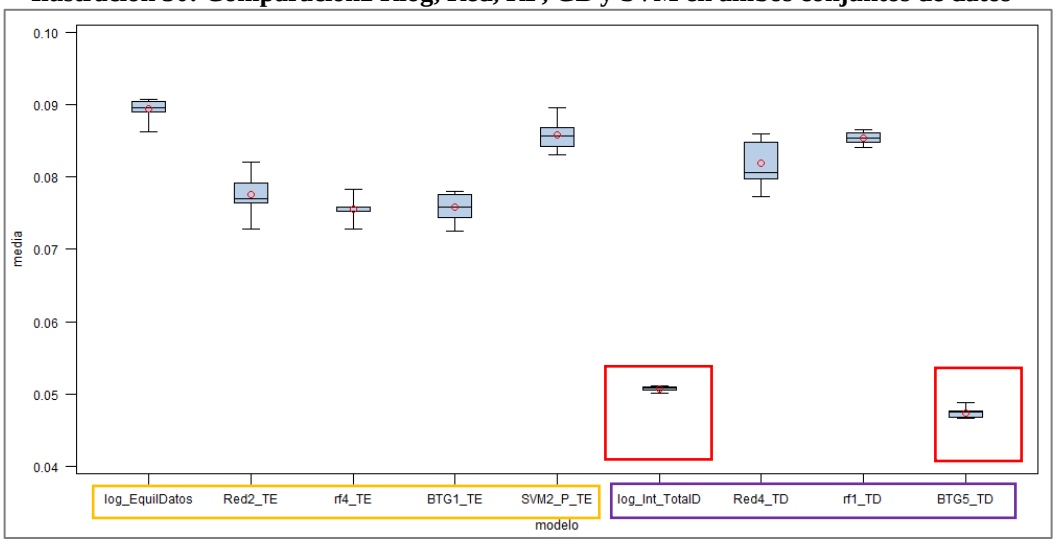

Por tanto, seleccionamos solo los cuatro primeros para compararlos (regresión logística, red neural, random forest y gradient boosting) de los cuales los modelos con menor valor en tasa de fallo y buena varianza son gradient boosting y regresión logística con interacciones (Ilustración 29).

**Ilustración 29:Comparación Rlog, Red, RF y GB en total de datos**




En este siguiente gráfico (Ilustración 30) compararemos todos los modelos más estables obtenidos en ambos conjuntos de datos a excepción del SVM7_L_TD ya que, al presentar una varianza muy amplia, no nos permite visualizar los demás modelos. Así mismo, para los modelos en el recuadro amarillo se utilizaron los datos equilibrados y los del recuadro morado el total de los datos. En donde se aprecia que para el conjunto de datos equilibrados el modelo más estable con menor tasa de fallo y menor varianza se obtuvo con random forest y para el total de datos el modelo de regresión logística.

**Ilustración 30: Comparación2 Rlog, Red, RF, GB y SVM en ambos conjuntos de datos**




### 5.6.9. Aplicación de Ensamblado

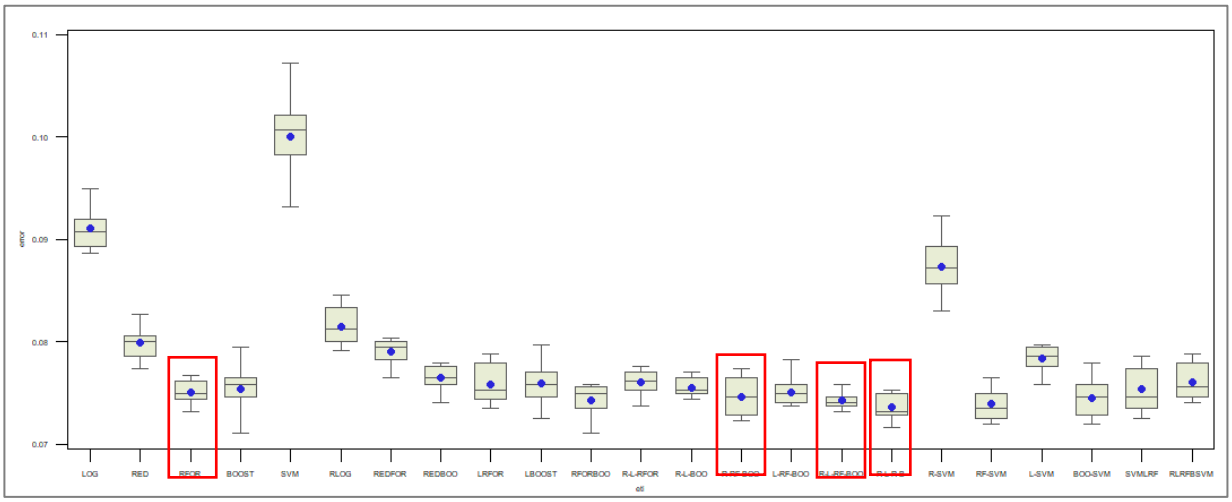
Para concluir la fase de modelizado, realizaremos el proceso de ensamblado, el cual combina varias técnicas de machine learning. En nuestro caso utilizaremos las técnicas de regresión logística, red neuronal, random forest, gradient boosting y SVM.

Para ello usaremos la técnica Stacking con la %macro cruzadastackcon (Anexo N: Ensamblado), el cual consiste en promediar las predicciones de los diferentes modelos utilizados, es decir, a partir de los mejores modelos calculados con las técnicas mencionadas en el párrafo anterior, la variable respuesta será un promedio de ellas. Los modelos a combinar son los mejores modelos obtenidos por cada técnica.

Empezaremos evaluando el ensamblado en datos equilibrados (Ilustración 31), en donde observamos que los mejores modelos son:

- R-L-RF-Boo (Regresión logística-Red-Random Forest-Gradient Boosting).
- RL-RF-Boo (Regresión logística-Random Forest- Gradient Boosting).
- R-RF-Boo (Red- Random Forest- Gradient Boosting).
- RFor (Random Forest).

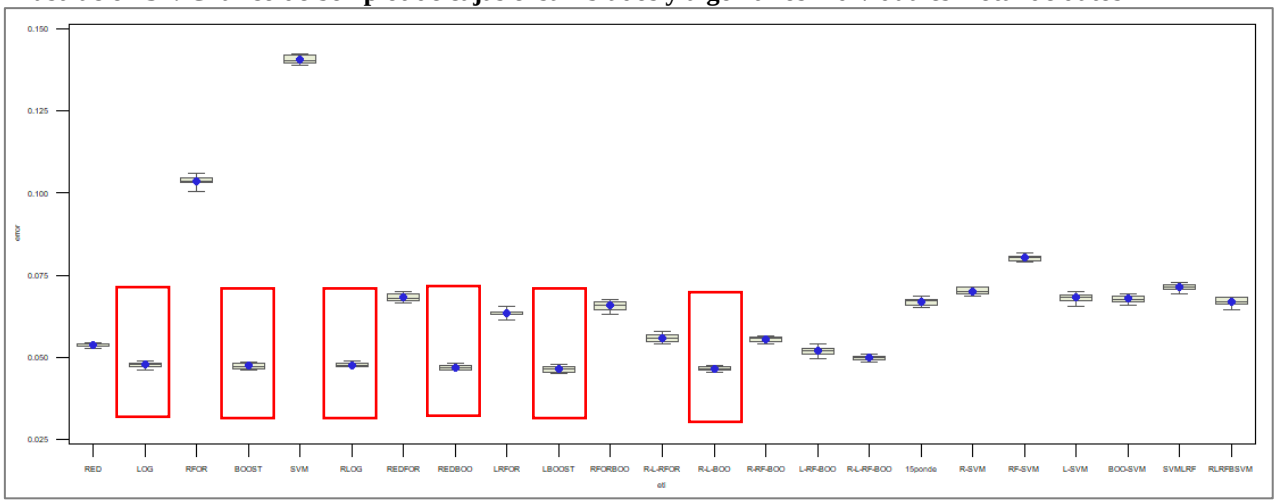
**Ilustración 31: Grafico de box plot de cajas ensamblados y algoritmos individuales-Datos equilibrados**




Ahora evaluaremos el grafico de box plot para el total de datos (Ilustración 32). Del cual, podemos decir que los mejores modelos son los siguientes:

- Log (Regresión Logística)
- Boost (Gradient Boosting)
- RLog (Red-Regresión logística)
- RedBoo (Red-Gradient Boosting)
- LBoost (Regresión logística-Gradient Boosting)
- R-L-Boo (Red-Regresión logística-Gradient Boosting)

**Ilustración 32: Grafico de box plot de cajas ensamblados y algoritmos individuales-Total de datos**



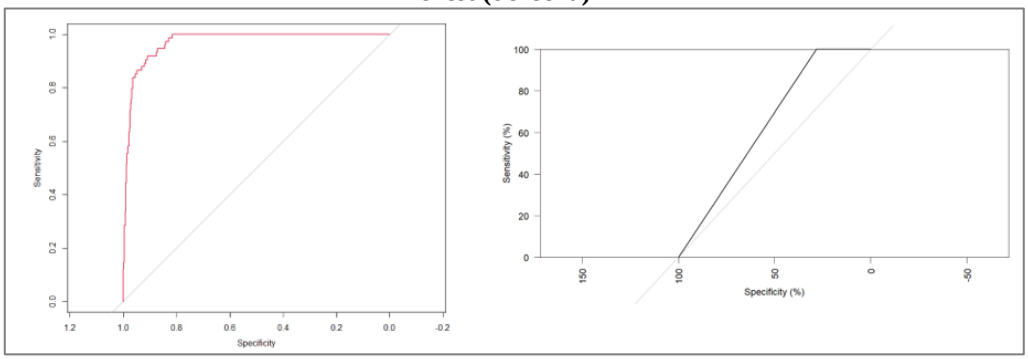

5.7. Selección de modelo:

Ya que en la sección anterior pudimos observar que los modelos más estables para el conjunto de dato equilibrados y el total de datos son random forest y regresión logística respectivamente, ahora con los datos test que inicialmente reservamos lo utilizaremos para verificar la calidad de estos modelos seleccionados.

**Tabla 24: Validación de modelos**

| | Total de Datos | Datos Equilibrados |
|---|---|---|
| | Regresión Logística | Random Forest |
| Curva ROC | 0.9722 | 0.6415 |
| Especificidad | 0.9871 | 0.6627 |
| Sensibilidad | 0.4865 | 0.2898 |

**Ilustración 33: Curva ROC para modelos de Regresión Logística (izquierda) y Random Forest (derecha)**




En la tabla 24 hemos comparado la bondad de ajuste de los 2 modelos ganadores para ambos conjuntos de datos. En donde se observa que el modelo de regresión logística presenta mayor valor en especificidad, sensibilidad y en curva ROC. Esto último también se puede observar en la Ilustración 33, en donde se representa gráficamente que la regresión logística muestra una mayor área bajo la curva. Por tanto, podemos afirmar que al trabajar con el conjunto total de datos nos brindó un modelo más estable.

5.8. Probabilidades obtenidas:

Como último paso, mostraremos el resultado de la ordenación jerárquica de los restaurantes que requieren inspección pronta, para no entrar en detalles del nombre del establecimiento, se colocó en la primera columna el código de identificación del restaurante y en la segunda columna la ciudad al cual pertenece.

Esta herramienta facilitará al Departamento de Sanidad de Nueva York establecer un orden prioritario en función a aquellos restaurantes que tienen una probabilidad alta de no pasar la inspección, además como se mención en la introducción de este trabajo, el Departamento no cuenta con la cantidad suficiente de inspecciones, por lo que esta herramienta ayudaría a optimizar los recursos del personal disponible (Tabla 25).

**Tabla 25: Probabilidades de restaurantes**

| NY_CAMIS | C_Ciudad | Estimated Probability |
|---|---|---|
| 50094332 | Queens | 0.999990 |
| 50098020 | Brooklyn | 0.999970 |
| 50094831 | Bronx | 0.999970 |
| 50091283 | Queens | 0.999950 |
| 50080122 | Queens | 0.999910 |
| 50086002 | Manhattan | 0.999830 |
| 50065637 | Bronx | 0.999820 |
| 50091708 | Queens | 0.999790 |
| 50094900 | Manhattan | 0.999780 |
| 50092539 | Brooklyn | 0.999760 |
| 50093763 | Manhattan | 0.999320 |
| 50082467 | Brooklyn | 0.999120 |
| 50071927 | Manhattan | 0.999070 |
| 50081358 | Staten Island | 0.999070 |
| 50100933 | Queens | 0.999000 |
| 50098848 | Queens | 0.998880 |
| 50070859 | Manhattan | 0.998580 |
| 50096032 | Manhattan | 0.998490 |
| 50098085 | Queens | 0.998400 |
| 50086380 | Queens | 0.998210 |
| 50086418 | Queens | 0.998050 |
| 50094332 | Queens | 0.999990 |
| 50098020 | Brooklyn | 0.999970 |
| 50094831 | Bronx | 0.999970 |
| 50091283 | Queens | 0.999950 |
| . . . | . . . | . . . |

6. Conclusiones y trabajos futuros:

   Conclusiones:

   - En este trabajo hemos identificado los principales factores que pueden influir para que un restaurante pase la inspección de sanidad o no. Para esto se han utilizado cinco algoritmos de Machine Learning, además las estimaciones se han realizado en dos conjuntos de datos:

     - ✓ Datos totales: variable en estudio está conformada por el 93% de restaurantes que pasaron la inspección y el 7% por restaurantes que no pasaron la inspección.
     - ✓ Datos equilibrados: variable en estudio está conformada por el 25% de restaurantes que no pasaron la inspección y el 75% por restaurantes que si pasaron la inspección.

   - En la depuración de datos, se creó 12 nuevas variables a partir de los campos obtenidos de OpenData New York, además se adicionó 12 variables con la técnica de Web Scraping.

   - Con el set de variables implementado, se procedió a analizar cada una de ellas para poder aplicar técnicas de imputación de datos y agrupación de niveles con poca representatividad en variables categóricas.

   - Respecto al análisis geovisual realizado, nos ayudó a ver de manera geográfica que la distribución del total de restaurantes en cada una de las ciudades de Nueva Yotk no influye en el resultado de inspección. Además, también pudimos observar la puntuación que obtienen los restaurantes en Google Maps según cada ciudad.

   - Notamos que en su mayoría los modelos trabajados con el total de datos presentan menor tasa de fallo. Obteniendo, así como mejores modelos Gradient Boosting y Regresión Logística. Pero se eligió el Modelo de Regresión Logística, ya que es un modelo más clásico, además nos permite tener una mayor comprensión de los parámetros. Además, podemos decir que, para obtener modelos con menor tasa de fallo en nuestro conjunto de datos, no es necesario equilibrar la variable objetivo.

   - Las cuatro principales variables de aporte al modelo son: la interacción entre la variable REP_C_TotalInsp y NY_ TipoInspeccion sería la principal, seguidas de C_MesInspec, Rep_C_TotalGradoA y Rep_Rep_NYDecri.

   - Se proporcionó un listado de restaurantes con sus probabilidades a posteriori, lo cual serviría como una herramienta que permitirá establecer un orden para priorizar los restaurantes que requieran una inspección inmediata.

Trabajos futuros:

- Se podría incorporar más variables de diferente naturaleza como:
    - ✓ Variables del manejo interno del restaurante (número de empleados, número de empleados de sexo femenino, número de empleados de sexo masculino, edad promedio de los empleados, monto promedio de venta mensual, mes con mayor demanda, etc).
    - ✓ Variable respecto al tipo de infracción (si bien esta variable inicialmente se encontraba en nuestra base de datos, pues debido a que no se cuenta con la información necesaria en la página web de Nueva York no se pudo categorizar sus múltiples niveles, por lo que se considera que con una información más a detalle sobre las infracciones se podría sacar mayor provecho.
- En este trabajo se ha considerado por defecto un punto de corte de 0,5. Sin embargo para un próximo trabajo se podría entrar al detalle y evaluar diferentes puntos de corte para categorizar las probabilidades y definir acciones en función a cada categoría.
- Se podría implementar un mecanismo de seguimiento para el modelo, el cual permitirá observar su comportamiento al ir incorporando a nuevos restaurantes.
- Otro trabajo interesante sería realizar text mining con los comentarios extraídos de google maps por cada restaurante.
- Implementación de un dashboard para la presentación de la herramienta que permite establecer el orden de inspección, además de incorporar los indicadores de mayor importancia.

7. Bibliografía


- Bloomberg, M. R. (2016). *A Guide for Food Service Operators.* Nueva York.
- Calviño, A. (2019). *Breve guía para el tratamiento de datos atípicos y faltantes.* Madrid.
- Chicago, D. d. (2017). *Food Inspection Failires Chicago*. Obtenido de https://chicago.github.io/food-inspections-evaluation/
- Data, O. (10 de Febrero de 2020). *Web Scraping Python: Guía Paso a Paso*. Obtenido de https://www.octoparse.es/blog/web-scraping-con-python
- Gareth, Witten, Hastie and Tibshirani. (2009). An Introduction to Statistical Learning.
- Michael R. Bloomberg ,Thomas Farley. (2012). *How we Score And Grade.* Ciudad de Nueva York.
- Michael,Farly. (2016). *Guía para operadores de servicios de alimentos.* Nueva York.
- Portela. (2019). Ensamblado-Curso de Machine Learning. Madrid, España.
- Portela. (2019). *Tema 5. Support Vector Machines.* Madrid.
- Portela. (2020). Curso Machine Learning. Madrid.
- Ramírez, P. (2019). *Minería de datos para la mejora de la gestión de las inspecciones de sanidad en restaurantes de la ciudad de Chicago.* Madrid.
- SAS, I. (2000). *The Neural Procedure.* NC, USA.
- scikit-learn. (2020). *RBF SVM parameters*. Obtenido de scikit-learn 0.23.2: https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html
- Smolyakov, V. (2017). Ensemble Learning to Improve Machine Learning Results.
- Walter, M. (2013). *Prediction of NYC Restaurant Health Inspection Results.* Ciudad de Nueva York.
- Yuniesky González Muñoz,Carolina Esthela Palomino Camargo. (2012). Acciones para la gestión de la calidad sanitaria e inocuidad de los alimentos en un restaurante con servicio bufet. *Revista Gerencial Politica y Salud, Bogota*, 123-140.

# 8. Anexos

## 8.1 Software R:

### Anexo A: Modificación de la base de datos

```
if(!require(readxl)){install.packages("readxl")}
if(!require(xlsx)){install.packages("xlsx")}
if(!require(urltools)){install.packages("urltools")}
if(!require(rvest)){install.packages("rvest")}
library(readxl)
library(xlsx)
library(urltools)
library(rvest)
library(tidyverse)

# código open data NY, le adicioné algunas variables mas
# Download DOHMH NYC Restaurant Inspection Results data set and save
as CSV file: NYC_Insp_Results.csv
Open_Data_Sample <- read_csv("C:/Users/Ruth/OneDrive/Cuso de Master
II ciclo/TFM/DOHMH_NY_Restaurant_Inspection_Results_TrabajadoRuth-
DESKTOP-24NH9GN.csv",
                        col_types  =  cols(ZIPCODE = col_character())
)
View(Open_Data_Sample)
#Filter on inspection type, score, grade
Inspections <- Open_Data_Sample %>%
  filter((`INSPECTION TYPE` %in%
            c('Cycle Inspection / Re-inspection'
              ,'Pre-permit (Operational) / Re-inspection')
          |(`INSPECTION TYPE` %in%
              c('Cycle Inspection / Initial Inspection'
                ,'Pre-permit (Operational) / Initial Inspection'))
          & SCORE <= 13)
         | (`INSPECTION TYPE` %in%
              c('Pre-permit (Operational) / Reopening Inspection'
                ,'Cycle Inspection / Reopening Inspection'))
         & GRADE %in% c('A', 'B', 'C', 'P', 'Z')) %>%

  select(CAMIS,`INSPECTION DATE`)

#Select distinct inspections
Inspections_Distinct <- distinct(Inspections)

#Select most recent inspection date
MostRecentInsp <- Inspections_Distinct %>%
  group_by(CAMIS) %>%
  slice(which.max(as.Date(`INSPECTION DATE`,'%m/%d/%Y')))

#Join most recent inspection with original dataset
inner_join(Open_Data_Sample,MostRecentInsp, by =
"CAMIS","INSPECTION DATE")

#Select restaurant inspection data based on most recent inspection date
Final <- Open_Data_Sample %>% inner_join(MostRecentInsp) %>%
```

```
  filter((`INSPECTION TYPE` %in%
            c('Cycle Inspection / Re-inspection'
              ,'Pre-permit (Operational) / Re-inspection'
              , 'Pre-permit (Operational) / Reopening Inspection'
              ,'Cycle Inspection / Reopening Inspection')
          |(`INSPECTION TYPE` %in%
              c('Cycle Inspection / Initial Inspection'
                ,'Pre-permit (Operational) / Initial Inspection'))
          & SCORE <= 13)) %>%

  select(CAMIS,DBA,BORO,`CUISINE DESCRIPTION`,`INSPECTION
DATE`,GRADE,`INSPECTION TYPE`,SCORE,Latitude,Longitude)

#Select distinct restaurant inspection data
Final <- distinct(Final)
View(Final)

#exportar en excel
write.xlsx(Final, "C:/Users/Ruth/OneDrive/Cuso de Master II
ciclo/TFM/final.xlsx")

#Ya incorporado más variables a mi base de datos final, ahora lo cargo
Data_Restaurant <- read_excel("C:/Users/Ruth/OneDrive/Cuso de Master II
ciclo/TFM/final.xlsx")

#ahora me fijo la información en cada variable:
summary(Data_Restaurant)
str(Data_Restaurant)

# Me creo una nueva variable "Y" que recoja solo los restaurantes con
Grado: A=P y B,C=NP
Data_Restaurant$Inspecc<-with(Data_Restaurant,ifelse(GRADE %in%
c("B","C"),"NP", ifelse(GRADE %in% c("A"),"P",NA)))

#elimino Columnas
Data_Restaurant$ID_join<-NULL
Data_Restaurant$Pass<-NULL

#Vemos el reparto de P y NP
prop.table(table(Data_Restaurant$Inspecc))

#elimino filas vacias de la var: Inspecc
Data_Restaurant1 <- Data_Restaurant[!is.na(Data_Restaurant$Inspecc),]

#me fijo los niveles de la variable: CUISINE DESCRIPTION
grupos <- group_by(Data_Restaurant1, `CUISINE DESCRIPTION`)
summarise(grupos,
          num = n()
)

#elimino establecimientos y solo me quedo con restaurantes (variable:
Cuisine Descripction):
DESCRIPTION`!="Bakery",] #cuando es solo un nivel
```

```
Data_Restaurant2<-Data_Restaurant1[!(Data_Restaurant1$`CUISINE
DESCRIPTION` %in% c("Bakery","Bagels/Pretzels","Bottled beverages,
including water, sodas, juices, etc.","Café/Coffee/Tea",

"Donuts","Fruits/Vegetables","Hotdogs","Hotdogs/Pretzels","Ice Cream,
Gelato, Yogurt, Ices",
                                                    "Juice, Smoothies, Fruit
Salads","Not Listed/Not Applicable","Nuts/Confectionary","Other",

"Pancakes/Waffles","Salads","Sandwiches","Sandwiches/Salads/Mixed
Buffet","Soups & Sandwiches")),]

#exportar el excel con la columna CountRest creada:
Data_Restaurant2 <- read_excel("C:/Users/Ruth/OneDrive/Cuso de Master
II ciclo/TFM/Data_Restaurant2.xlsx")

#todas las filas con NRest >1, serán eliminados para evitar equivocarnos al
momento de elejir uno u otro
#ahora elimínanos las filas con CountRest>1, d etal forma que solo nos
quedamos con un solo nombre del restaurante
Data_Restaurant3<-Data_Restaurant2[Data_Restaurant2$CountRest<=1,]

#Vemos el reparto de P y NP
prop.table(table(Data_Restaurant3$Inspecc))

#ahora exportamos el excel para hacer el web scrapin con Phyton:
write.xlsx(Data_Restaurant3, "C:/Users/Ruth/OneDrive/Cuso de Master II
ciclo/TFM/Data_Restaurant3.xlsx")

#en excel junto el archivo "Data_Restaurant3" con los resultados de web
scrapping y le llamo "Data_Restaurant4"

#en Excel del archivo "Data_Restaurant4", elimino las filas con (Score
rating=0 y #N/D) y creo la nueva base de datos: Data_Restaurant5

#leo Data_Restaurant5_web
Data_Restaurant5_Web <- read_excel("C:/Users/Ruth/OneDrive/Cuso de
Master II ciclo/TFM/Data_Restaurant5_Web.xlsx")
#elimino niveles de la columna "Category", creada por el web scrapping
Data_Restaurant6_Web<-
Data_Restaurant5_Web[!(Data_Restaurant5_Web$Category %in%
c("Airline","Adult entertainment club","Alcoholism treatment program",
                                                    "Amusement
park","Antique store","Apartment building","Apartment complex","Art center",
                                                    "Art gallery","Art
school","Arts organization","ATM","Baby store","Bagel
shop","Bakery","Bank","Bar","Blues club",
                                                    "Business
school","Cabaret club","Cake shop","Casino","Cafeteria","Chocolate
shop","Cigar shop","Clothing store","coffee shop","Collage","Comedy
club","Engineering school",
                                                    "Dental
school","Dentist","Doctor","Doctor shop","Donut shop","Night club","Taxi
service","University","Zoo","Cabine store","Ice cream shop")),]
```

```
# Me quedo con "Total: 1 star" diferente a 0, ya que serían restaurantes sin ningun comentario, por error del webs scrapping
Data_Restaurant6_Web<-
Data_Restaurant6_Web[Data_Restaurant6_Web$`Total: 1 star`!= 0,]

prop.table(table(Data_Restaurant6_Web$Inspecc))

#quito las inspecciones del 2017 y 2018
Data_Restaurant6_Web<-
Data_Restaurant6_Web[Data_Restaurant6_Web$Inspecciones!= 2017,]
Data_Restaurant6_Web<-
Data_Restaurant6_Web[Data_Restaurant6_Web$Inspecciones!= 2018,]

#exporto el excel
write.xlsx(Data_Restaurant6_Web, "C:/Users/Ruth/OneDrive/Cuso de Master II ciclo/TFM/Data_Restaurant6_web.xlsx")
```

## Anexo B: Elaboración de Mapas

```
library(viridis)
library(ggplot2)
library(readxl)
library(ggmap)
library(rgdal)
library(sf)

rest<- read_excel("C:/Users/Ruth/OneDrive/Cuso de Master II ciclo/TFM/tfm/Restaurantes_Modelo.xlsx")

# Mapa de las ciudades de NY con información de mis datos:
mapdata<-rest
head(mapdata)

#Mapa de Nueva York
counties<-readOGR("nybb.shp",layer="nybb") #copiar los archivos:nybb.dbf, nybb.shp, nybb.shx, nybb.shp.xml y nybb.prj (http://zevross.com/blog/2014/07/16/mapping-in-r-using-the-ggplot2-package/)
head(counties@data)
head(counties@data$BoroName)
class(counties)

# Junto la información del mapa de Nueva York con la información de mis base de datos
ggplot()+geom_polygon(data=counties, aes(x=long, y=lat, group=group)) +
  geom_point(data=mapdata, aes(x=IMP_NY_Longitud,
y=IMP_NY_Latitud), color="blue")

#sistema de proyección / coordenadas existente para las capas
proj4string(counties)

#estandarizar escalas
class(mapdata)
coordinates(mapdata)<-~IMP_NY_Longitud+IMP_NY_Latitud
class(mapdata)
```

```
# does it have a projection/coordinate system assigned?
proj4string(mapdata)

# tell R what the coordinate system is
proj4string(mapdata)<-CRS("+proj=longlat +datum=NAD83")

# assign the projection from counties
mapdata<-spTransform(mapdata, CRS(proj4string(counties)))

# double check that they match
identical(proj4string(mapdata),proj4string(counties))

#mapas fusionados:
mapdata<-data.frame(mapdata)

# we're not dealing with lat/long but with x/y
# this is not necessary but for clarity change variable names
names(mapdata)[names(mapdata)=="IMP_NY_Longitud"]<-"x"
names(mapdata)[names(mapdata)=="IMP_NY_Latitud"]<-"y"

#instalo otros paquetes
library(wesanderson)
library(rgeos)

shp<-readOGR("nybb.shp",layer="nybb")
summary(shp@data)
map <- ggplot() + geom_polygon(data = shp, aes(x = long, y = lat, group =
group), colour = "black", fill = NA)
map + theme_void()
shp_df <- broom::tidy(shp,region = "BoroName")
lapply(shp_df, class)
head(shp_df)
map<- ggplot() +geom_path(data = shp, aes(x = long, y = lat, group =
group),color = 'gray40')
#asignar nombre a las ciudades a map
cnames <- aggregate(cbind(long, lat) ~ id, data=shp_df, FUN=mean)

library(RColorBrewer)
#mapa_1 nombres de ciudades en leyenda:
ggplot() + geom_polygon(data = shp_df, aes(x = long, y = lat, group =
group, fill = id),colour="grey")+
  scale_fill_brewer(palette = "Greys")+ #color del fondo de ciudades
  geom_point(data=mapdata, aes(x=x, y=y,colour =
C_Inspecciones,shape=C_Inspecciones),alpha=1, size=2)+
  labs(x="", y="", title="Restaurantes inspeccionados de Nueva
York",caption = "Fuente: NYC Open Data")+
  theme(plot.title = element_text(color = "darkblue", size = 18, face =
"bold"),plot.subtitle = element_text(color = "blue",size = 15),plot.caption =
element_text(color = "blue", face = "italic"))+
  theme(plot.title = element_text(hjust = 0.5),plot.subtitle =
element_text(hjust = 0.5))

#mapa_2: Total de inspecciones
```

```
ggmap(rest_map)+geom_point(aes(IMP_NY_Longitud, IMP_NY_Latitud
,color=REP_C_TotalInsp),data=rest,fill="darkslategrey")+
  labs(x="", y="", title="Incidencia de inspecciones en restaurantes de Nueva
York",caption = "Fuente: NYC Open Data")+
  theme(plot.title = element_text(color = "darkblue", size = 18, face =
"bold"),plot.subtitle = element_text(color = "blue",size = 15),plot.caption =
element_text(color = "blue", face = "italic"))+
  theme(plot.title = element_text(hjust = 0.5),plot.subtitle =
element_text(hjust = 0.5))+
  scale_colour_gradientn( colours=c( "#76e872","#660000"))

#Map_3: Puntuación obtenido en Google Maps
ggmap(rest_map)+geom_point(aes(IMP_NY_Longitud, IMP_NY_Latitud
,color=REP_WS_PuntuacionRating),data=rest,fill="darkslategrey")+
  labs(x="", y="", title="Puntuación en restaurantes de Nueva York",caption =
"Fuente: NYC Open Data")+
  theme(plot.title = element_text(color = "darkblue", size = 18, face =
"bold"),plot.subtitle = element_text(color = "blue",size = 15),plot.caption =
element_text(color = "blue", face = "italic"))+
  theme(plot.title = element_text(hjust = 0.5),plot.subtitle =
element_text(hjust = 0.5))+
  scale_colour_gradientn( colours=c( "#76e872","#660000"))
```

## Anexo C: Undersampling

```
library(dplyr)
library(sampling)
library(haven)
library(readxl)
library(sas7bdat)

#Cargando las datas para el análisis
restaurantes = read.sas7bdat("C:/Users/Ruth/OneDrive/Cuso de Master II
ciclo/TFM/tfm/Restaurantes_Modelo.sas7bdat")

#Separo una muestra aleatoria del 10% para test
12343*(10/100) #este cálculo es para mis datos test
test_restaurantes <- sample_n(restaurantes, size= 1234)

#generar una data, quitando los datos del test_restaurantes
DatosT_restaurante<-
anti_join(restaurantes,test_restaurantes,by="NY_CAMIS")

#undersampling
library(unbalanced)
DatosT_restaurante$C_Inspecciones1[DatosT_restaurante$C_Inspeccione
s=="NP"]<-1 #se da 1 a la clase minoritaria
DatosT_restaurante$C_Inspecciones1[DatosT_restaurante$C_Inspeccione
s =="P"]<-0

n<-ncol(DatosT_restaurante)
output<-DatosT_restaurante$C_Inspecciones1
input<-DatosT_restaurante[ ,-n]
data<-ubUnder(X=input, Y= output, perc = 25, method = "percPos")
DatosE_restaurante<-cbind(data$X, data$Y)
```

## 8.2 Software Python:

### Anexo D: Web Scraping

```
import os
import sys
import time
import pandas as pd
from tqdm import tqdm
from selenium import webdriver
from selenium.webdriver.chrome.options import Options

#inicializamos el controlador web con la configuración ingles del navegador
chrome_options = Options()
chrome_options.add_argument('--headless')
chrome_options.add_argument('log-level=3')

weekdays = ['Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday',
'Sunday', 'Monday']

#información a extraer de cada restaurant de google maps
def get_restaturant(driver, record):

    data = {
        'NameRestaurant': record['NameRestaurant'],
        'ID': record['ID'],
        'CityRestaurant': record['CityRestaurant'],
        'Latitude': record['Latitude'],
        'Longitude': record['Longitude'],
        'URL': '',
        'Score rating': '',
        'Total: 1 star': '',
        'Total: 2 star': '',
        'Total: 3 star': '',
        'Total: 4 star': '',
        'Total: 5 star': '',
        'Category': '',
        'Price': '',
        'Hours': '',
        'Number of days': '',
        'Total reviews': '',
        }

    try:
        name = record['NameRestaurant'].replace('#', 'No.') #reemplazo el “#”
por el No

        url = 'https://www.google.com/maps/search/{}/@{},{},20z?hl=en'.format(
```

```
            name, record['Latitude'], record['Longitude'])  #concateno el nombre,
lat y log en la URL de búsqueda
        driver.get(url)
        time.sleep(5)

        if driver.current_url.find('/data=') == -1:
            first_result = driver.find_element_by_css_selector(
                '.section-layout.section-scrollbox .section-result')
            if first_result == None:
                return False

            first_result.click()
            time.sleep(3)

        #aqui asigno el URL por cada restaurant
        data['URL'] = driver.current_url

        #extrae el precio
        try:
            price_wrapper = driver.find_element_by_css_selector(
                '.section-rating-term [aria-label*="Price: "]')
            data['Price'] = price_wrapper.get_attribute(
                'aria-label').replace('Price: ', '').strip()
        except:
            pass

        #extraemos el tipo de restaurante
        try:
            category_wrapper = driver.find_element_by_css_selector(
                '.section-rating-term [jsaction="pane.rating.category"]')
            data['Category'] = category_wrapper.text.strip()
        except:
            pass

        #extraemos el horario de atención
        try:
            rest_hour_wrapper = driver.find_element_by_css_selector(
                '.section-open-hours-container')
            _hours = rest_hour_wrapper.get_attribute(
                'aria-label').replace('Hide open hours for the week', '').strip()
            data['Hours'] = _hours
            days = 0
            hours = _hours.split('; ')
            for hour in hours:
                for weekday in weekdays:
                    if hour.find(weekday) != -1:
                        if hour.find('Close') == -1:
                            days = days + 1
            data['Number of days'] = days
        except:
            pass

        #extraemos la puntuación
        try:
            rest_rating_wrapper = driver.find_element_by_css_selector(
```

```
            '.section-hero-header-title-description-container .section-star-
display')
            data['Score rating'] = rest_rating_wrapper.text.strip()
        except:
            pass

        #extraemos el número total de comentarios
        try:
            rest_reviews_wrapper = driver.find_element_by_css_selector(
                '.section-rating-term-list .widget-pane-link')
            data['Total reviews'] = rest_reviews_wrapper.text.strip().replace(
                '(', '').replace(')', '')
        except:
            pass

    #extraemos información del número de estrellas:
        reviews = []
        for i in range(1, 6): #para 1,2,3,4 y 5 estrellas
            try:
                rest_star_wrapper = driver.find_element_by_css_selector(
                    'tr[aria-label^="{} stars"]'.format(i)
                )
                reviews_label = rest_star_wrapper.get_attribute('aria-label')
                _reviews = reviews_label.split(',')[1].strip()

                data['Total: {} star'.format(i)] = _reviews
            except:
                pass

#Al hacer el web scrapping en google maps no me bloqueó, pero de todos
modos decidí usar modo:hilos.
def get_results(records):
    driver = webdriver.Chrome(options=chrome_options)
    idx = 0
    results = [ ]  #declaro a la variable resultados como matriz
    for record in records:
        if record == None:
            continue
        # En esta parte es solo para mostrar el progreso
        idx = idx + 1
        if idx % 30 == 0:
            print('{}'.format(idx))
        result = get_restaturant(driver, record)
        results.append(result)
    try:
        driver.quit()
    except:
        pass
    return results

#Lectura de excel y adicionar más columnas con información
if __name__ == "__main__":

# leemos nuestro archivo en Excel y agregamos las nuevas variables
```

```
    df = pd.read_excel('C:/Users/Ruth/Desktop/Restaurant_ NewYork.xlsx',
columns=[
        'NameRestaurant', 'ID', 'CityRestaurant', 'Latitude', 'Longitude'])
    records = []

    #adicionar nuevas columnas al excel
    for index, row in df.iterrows():
        _name = row['NameRestaurant']
        _latitude = row['Latitude']
        _longitude = row['Longitude']
        _city = row['CityRestaurant']
        _id = row['ID']
        records.append({
            'NameRestaurant': row['NameRestaurant'],
            'ID': row['ID'],
            'CityRestaurant': row['CityRestaurant'],
            'Latitude': row['Latitude'],
            'Longitude': row['Longitude'],
        })

    # Llenamos registros faltantes
    for i in range(threads - len(records) % threads):
        records.append(None)

    _records = []
    record_list = map(list, zip(*zip(*[iter(records)] * threads)))
    for sub_records in record_list:
        _records.append(sub_records)

    # obtenemos hilos
    thread_results = []
    with Pool(threads) as p:
        thread_results = p.map(get_results, _records)

    total_results = []
    for thread_result in thread_results:
        for _result in thread_result:
            total_results.append(_result)

    df2 = pd.DataFrame(total_results)
    df2.to_csv('resultsSample.csv', encoding='utf-8', index=False)
```

## 8.3 Software SAS: Modelización de la base de datos

### Anexo E: Macro de selección de variables

```
proc import datafile='C:\Users\Ruth\OneDrive\Cuso de Master II
ciclo\TFM\modelos\base de datos\DatosE_restaurante.xlsx'
 DBMS=EXCELCS
 OUT=REST_EQUILIBRADO REPLACE;RUN;
PROC PRINT DATA=REST_EQUILIBRADO; RUN;

 proc import datafile='C:\Users\Ruth\OneDrive\Cuso de Master II
ciclo\TFM\modelos\base de datos\DatosT_restaurante.xlsx'
```

```
 DBMS=EXCELCS
 OUT=REST_TOTAL REPLACE;RUN;
 PROC PRINT DATA=REST_TOTAL; RUN;

/*EVALUAR CUALES SON LAS MEJORES VARIABLES, PERO SIN
INTERACCIONES*/
%interacttodolog(archivo=WORK.REST_TOTAL,vardep=C_Inspecciones1
,
listclass=C_Ciudad C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_C_DiaInspecc REP_NY_FechaInspeccion
REP_REP_NY_DescripcionComida REP_REP_WS_Precio
REP_WS_HorarioAtencion REP_WS_NumDiasTrabajo,
listconti=C_NumRestCiudad IMP_NY_Latitud IMP_NY_Longitud
IMP_REP_WS_TotalComentarios REP_C_NumViolaciones
REP_C_NumViolacionesCriticas_N REP_C_NumViolacionesCriticas_Y
REP_C_TotalGradoA REP_C_TotalGradoB REP_C_TotalGradoC
REP_C_TotalInsp REP_WS_1Estrella REP_WS_2Estrellas
REP_WS_4Estrellas REP_WS_5Estrellas REP_WS_PuntuacionRating
WS_3Estrella,
interac=0,
directorio=C:\Users\Ruth\Documents\basura sas);

/*de las selecionadas en el punto anterior, colocar aqui para evaluar la
interacción de var*/
%interacttodolog(archivo=WORK.REST_TOTAL,vardep=C_Inspecciones1
,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_NY_FechaInspeccion REP_REP_NY_DescripcionComida,
listconti=REP_C_TotalGradoA REP_C_TotalGradoB REP_C_TotalGradoC
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrella,
interac=1,
directorio=C:\Users\Ruth\Documents\basura sas);

/*seleccion de var.*/
%randomselectlog(data=WORK.REST_TOTAL,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_NY_FechaInspeccion REP_REP_NY_DescripcionComida,
vardepen=C_Inspecciones1,
modelo=NY_TipoInspeccion*REP_C_TotalGradoB
NY_TipoInspeccion*REP_C_TotalInsp NY_TipoInspeccion
NY_TipoInspeccion*REP_WS_PuntuacionRating
NY_TipoInspeccion*WS_3Estrella C_MesInspecc*REP_C_TotalGradoB
C_EstacionInspecc*REP_C_TotalGradoB REP_C_TotalGradoB
NY_TipoInspeccion*REP_C_TotalGradoC
NY_TipoInspeccion*REP_C_TotalGradoA
C_EstacionInspecc*NY_TipoInspeccion C_MesInspecc*REP_C_TotalInsp
C_MesInspecc*NY_TipoInspeccion C_EstacionInspecc*REP_C_TotalInsp
C_MesInspecc*REP_C_TotalGradoC REP_C_TotalGradoA
C_EstacionInspecc*REP_C_TotalGradoC REP_C_TotalGradoC
REP_C_TotalInsp C_MesInspecc C_EstacionInspecc*C_MesInspecc
C_MesInspecc*REP_WS_PuntuacionRating C_MesInspecc*WS_3Estrella
NY_TipoInspeccion*REP_REP_NY_DescripcionComida
REP_REP_NY_DescripcionComida*REP_C_TotalGradoB
C_EstacionInspecc C_EstacionInspecc*REP_WS_PuntuacionRating
C_EstacionInspecc*WS_3Estrella C_MesInspecc*REP_C_TotalGradoA
```

```
REP_REP_NY_DescripcionComida*REP_C_TotalGradoC
C_EstacionInspecc*REP_C_TotalGradoA
REP_REP_NY_DescripcionComida*REP_C_TotalGradoA
C_MesInspecc*REP_REP_NY_DescripcionComida
REP_REP_NY_DescripcionComida*REP_C_TotalInsp
C_EstacionInspecc*REP_REP_NY_DescripcionComida
REP_REP_NY_DescripcionComida
REP_REP_NY_DescripcionComida*REP_WS_PuntuacionRating
REP_REP_NY_DescripcionComida*WS_3Estrella
REP_WS_PuntuacionRating
REP_REP_NY_DescripcionComida*REP_C_TotalGradoC
C_EstacionInspecc*REP_C_TotalGradoA
REP_REP_NY_DescripcionComida*REP_C_TotalGradoA
C_MesInspecc*REP_REP_NY_DescripcionComida
REP_REP_NY_DescripcionComida*REP_C_TotalInsp
C_EstacionInspecc*REP_REP_NY_DescripcionComida
REP_REP_NY_DescripcionComida
REP_REP_NY_DescripcionComida*REP_WS_PuntuacionRating
REP_REP_NY_DescripcionComida*WS_3Estrella
REP_WS_PuntuacionRating WS_3Estrella,
sinicio=12345,sfinal=12355,fracciontrain=0.7,
        directorio=C:\Users\Ruth\Documents\basura sas);
```

## Anexo F: Macro cruzada logística

```
/*Regresión Logisitca-Semilla inicio:12345 y semilla final=12355 */

%cruzadalogistica
(archivo=REST_TOTAL,vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355);
data final1;set final;modelo='Modelo 1.1';

%cruzadalogistica
(archivo=REST_TOTAL,vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
NY_TipoInspeccion*REP_C_TotalGradoB
NY_TipoInspeccion*Rep_WS_PuntuacionRating,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355);
data final2;set final;modelo='Modelo 1.2';

%cruzadalogistica
(archivo=REST_EQUILIBRADO,vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355);
data final3;set final;modelo='log_EquilD';
```

```
%cruzadalogistica
(archivo=REST_EQUILIBRADO,vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
NY_TipoInspeccion*REP_C_TotalGradoB
NY_TipoInspeccion*Rep_WS_PuntuacionRating,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355);
data final4;set final;modelo='log_int_EquilD';

data union;set final1 final2 final3 final4;
proc boxplot data=union;plot media*modelo;run;

/*Regresión Logisitca-Semilla inicio:12345 y semilla final=12456*/

%cruzadalogistica
(archivo=REST_TOTAL,vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12456);
data final5;set final;modelo='Rlog_TotalD';

%cruzadalogistica
(archivo=REST_TOTAL,vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
NY_TipoInspeccion*REP_C_TotalGradoB
NY_TipoInspeccion*Rep_WS_PuntuacionRating,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12456);
data final6;set final;modelo='Rlog_int_TotalD';

%cruzadalogistica
(archivo=REST_EQUILIBRADO,vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12456);
data final7;set final;modelo='Rlog_EquilD';

%cruzadalogistica
(archivo=REST_EQUILIBRADO,vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
NY_TipoInspeccion*REP_C_TotalGradoB
NY_TipoInspeccion*Rep_WS_PuntuacionRating,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12456);
```

```
data final8;set final;modelo='Rlog_int_EquilD';

data union;set final1 final2 final3 final4 final5 final6 final7 final8;
proc boxplot data=union;plot media*modelo;run;

data union;set final1 final2 final5 final6;
proc boxplot data=union;plot media*modelo;run;
```

## Anexo G: Macro variar

```
/* MACRO VARIAR PARA VER EL NUM DE NODOS*/
/* Todas los datos*/
%macro variar(seminicio=,semifin=,inicionodos=,finalnodos=,increnodos=);
title '';
data union;run;
%do semilla=&seminicio %to &semifin;
%do nodos=&inicionodos %to &finalnodos %by &increnodos;
  %neuralbinariabasica(archivo=REST_TOTAL,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
vardep=C_Inspecciones1,nodos=&nodos,corte=50,semilla=&semilla,porce
n=0.80,algo=BPROP);
data estadisticos;set estadisticos;nodos=&nodos;semilla=&semilla;run;
data union;set union estadisticos;run;
%end;
%end;
proc sort data=union;by nodos;run;
proc boxplot data=union;plot (tasaciertos)*nodos;run;
%mend;
%variar(seminicio=12345,semifin=12455,inicionodos=2,finalnodos=12,incr
enodos=2);

/* MACRO VARIAR PARA VER EL NUM DE NODOS*/
/* Datos equilibrados*/
%macro variar(seminicio=,semifin=,inicionodos=,finalnodos=,increnodos=);
title '';
data union;run;
%do semilla=&seminicio %to &semifin;
%do nodos=&inicionodos %to &finalnodos %by &increnodos;
  %neuralbinariabasica(archivo=REST_EQUILIBRADO,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
vardep=C_Inspecciones1,nodos=&nodos,corte=50,semilla=&semilla,porce
n=0.80,algo=BPROP);
data estadisticos;set estadisticos;nodos=&nodos;semilla=&semilla;run;
data union;set union estadisticos;run;
%end;
%end;
proc sort data=union;by nodos;run;
proc boxplot data=union;plot (tasaciertos)*nodos;run;
%mend;
```

```
%variar(seminicio=12345,semifin=12455,inicionodos=2,finalnodos=12,incr
enodos=2);
```

## Anexo H: Macro algoritmo de optimización

```
/* MACRO algoritmo de optimización*/
/* para 2 nodos - total de datos*/
%macro algovalcruza;
%let lista='BPROP LEVMAR QUANEW TRUREG';
%let nume=4; %do i=1 %to &nume; data _null_; meto=scanq(&lista,&i); call
symput('meto',left(meto)); run;
%cruzadabinarianeural (archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12356,nodos=2,algo=&meto,objetivo=tasaf
allos,early=, acti= ); /* aqui cambiar el num de nodos*/
data final&i;set final;modelo="&meto";put modelo=;run; %end; data
union;set %do i=1 %to &nume; final&i %end; %mend;
%algovalcruza; data union1;
set final10 final11 final12 final13;
run;
proc boxplot data=union; plot media*modelo;
run;

/* para 2 nodos-datos equilibrados */
%macro algovalcruza;
%let lista='BPROP LEVMAR QUANEW TRUREG';
%let nume=4; %do i=1 %to &nume; data _null_; meto=scanq(&lista,&i); call
symput('meto',left(meto)); run;
%cruzadabinarianeural (archivo=REST_EQUILIBRADO,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12356,nodos=2,algo=&meto,objetivo=tasaf
allos,early=, acti= );
data final&i;set final;modelo="&meto";put modelo=;run; %end; data
union;set %do i=1 %to &nume; final&i %end; %mend;
%algovalcruza; data union1;
set final14 final15 final16 final17;
run;
proc boxplot data=union; plot media*modelo;
run;
```

## Anexo I: Macro función de activación

```
/*FUNCION DE ACTIVACION CON VALIDACION CRUZADA*/
/*nodo 2 LEVMAR*/
%macro activalcruza;
%let lista='TANH ARC SIN LOGISTIC SOF LOG';
```

```
%let nume=6; %do i=1 %to &nume; data _null_;activa=scanq(&lista,&i);call
symput('activa',left(activa));run;
%cruzadabinarianeural(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,nodos=2,algo=LEVMAR mom=0.8
learn=0.2,acti=&activa,objetivo=tasafallos);
data final&i;set final;modelo="&activa";put modelo=;run; %end; data
union;set %do i=1 %to &nume; final&i
%end;
%mend;
%activalcruza;
data union;
set final18 final19 final20 final21 final22 final23; run;
proc boxplot data=union; plot media*modelo;run;


/*nodo 2 TRUREG*/
%macro activalcruza;
%let lista='TANH ARC SIN LOGISTIC SOF LOG';
%let nume=6; %do i=1 %to &nume; data _null_;activa=scanq(&lista,&i);call
symput('activa',left(activa));run;
%cruzadabinarianeural(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,nodos=2,algo=TRUREG mom=0.8
learn=0.2,acti=&activa,objetivo=tasafallos);
data final&i;set final;modelo="&activa";put modelo=;run; %end; data
union;set %do i=1 %to &nume; final&i
%end;
%mend;
%activalcruza;
data union;
set final25 final26 final27 final28 final29 final30 final31; run;
proc boxplot data=union; plot media*modelo;run;

/*nodo 4 LEVMAR*/
%macro activalcruza;
%let lista='TANH ARC SIN LOGISTIC SOF LOG';
%let nume=6; %do i=1 %to &nume; data _null_;activa=scanq(&lista,&i);call
symput('activa',left(activa));run;
%cruzadabinarianeural(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,nodos=4,algo=LEVMAR mom=0.8
learn=0.2,acti=&activa,objetivo=tasafallos);
```

```
data final&i;set final;modelo="&activa";put modelo=;run; %end; data
union;set %do i=1 %to &nume; final&i
%end;
%mend;
%activalcruza;
data union;
set final32 final33 final34 final35 final36 final37 final38; run;
proc boxplot data=union; plot media*modelo;run;

/*nodo 6 LEVMAR*/
%macro activalcruza;
%let lista='TANH ARC SIN LOGISTIC SOF LOG';
%let nume=6; %do i=1 %to &nume; data _null_;activa=scanq(&lista,&i);call
symput('activa',left(activa));run;
%cruzadabinarianeural(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,nodos=6,algo=LEVMAR mom=0.8
learn=0.2,acti=&activa,objetivo=tasafallos);
data final&i;set final;modelo="&activa";put modelo=;run; %end; data
union;set %do i=1 %to &nume; final&i
%end;
%mend;
%activalcruza;
data union;
set final39 final40 final41 final42 final43 final44; run;
proc boxplot data=union; plot media*modelo;run;

/*nodo 6 TRUREG*/
%macro activalcruza;
%let lista='TANH ARC SIN LOGISTIC SOF LOG';
%let nume=6; %do i=1 %to &nume; data _null_;activa=scanq(&lista,&i);call
symput('activa',left(activa));run;
%cruzadabinarianeural(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,nodos=6,algo=TRUREG mom=0.8
learn=0.2,acti=&activa,objetivo=tasafallos);
data final&i;set final;modelo="&activa";put modelo=;run; %end; data
union;set %do i=1 %to &nume; final&i
%end;
%mend;
%activalcruza;
data union;
set final46 final47 final48 final49 final50 final51; run;
proc boxplot data=union; plot media*modelo;run;

/*nodo 2 Quanew-Datos eqiulibrados*/
%macro activalcruza;
```

```
%let lista='TANH ARC SIN LOGISTIC SOF LOG';
%let nume=6; %do i=1 %to &nume; data _null_;activa=scanq(&lista,&i);call
symput('activa',left(activa));run;
%cruzadabinarianeural(archivo=REST_EQUILIBRADO,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,nodos=2,algo=Quanew mom=0.8
learn=0.2,acti=&activa,objetivo=tasafallos);
data final&i;set final;modelo="&activa";put modelo=;run; %end; data
union;set %do i=1 %to &nume; final&i
%end;
%mend;
%activalcruza;
data union;
set final46 final47 final48 final49 final50 final51; run;
proc boxplot data=union; plot media*modelo;run;
```

## Anexo J: Macro early stopping

```
/*early*/
%redneuronalbinaria(archivo=REST_TOTAL,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
vardep=C_Inspecciones1,porcen=0.80,semilla=12345,ocultos=6,meto=LEV
MAR,acti=sof);

%redneuronalbinaria(archivo=REST_TOTAL,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
vardep=C_Inspecciones1,porcen=0.80,semilla=12345,ocultos=6,meto=TR
UREG,acti=SIN);

%redneuronalbinaria(archivo=REST_TOTAL,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
vardep=C_Inspecciones1,porcen=0.80,semilla=12345,ocultos=6,meto=TR
UREG,acti=Sof);

%redneuronalbinaria(archivo=REST_TOTAL,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
vardep=C_Inspecciones1,porcen=0.80,semilla=12345,ocultos=4,meto=LEV
MAR,acti=sof);
```

```
%redneuronalbinaria(archivo=REST_TOTAL,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
vardep=C_Inspecciones1,porcen=0.80,semilla=12345,ocultos=4,meto=TR
UREG,acti=SOF);

%redneuronalbinaria(archivo=REST_TOTAL,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
vardep=C_Inspecciones1,porcen=0.80,semilla=12345,ocultos=4,meto=TR
UREG,acti=SIN);

%redneuronalbinaria(archivo=REST_EQUILIBRADO,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
vardep=C_Inspecciones1,porcen=0.80,semilla=12345,ocultos=2,meto=LEV
MAR,acti=arc);

%redneuronalbinaria(archivo=REST_EQUILIBRADO,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
vardep=C_Inspecciones1,porcen=0.80,semilla=12345,ocultos=2,meto=TR
UREG,acti=sin);
```

Anexo K: Redes neuronales

```
/*CRUZADABINARIANEURAL-TOTAL DATOS */
%cruzadabinarianeural(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,
sfinal=12355,nodos=6,algo=LEVMAR,acti=sof,objetivo=tasafallos,early=16);
data final46; set final; modelo='Red1_TD'; run;

%cruzadabinarianeural(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,
sfinal=12355,nodos=4,algo=levmar,acti=sof,objetivo=tasafallos,early=18);
data final48; set final; modelo='Red3_TD'; run;
```

```
%cruzadabinarianeural(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,
sfinal=12355,nodos=6,algo=TRUREG,acti=sof,objetivo=tasafallos,early=22);
data final47; set final; modelo='Red2_TD'; run;

%cruzadabinarianeural(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,
sfinal=12355,nodos=4,algo=TRUREG,acti=sof,objetivo=tasafallos,early=19);
data final49; set final; modelo='Red4_TD'; run;

data union_redes;
set final46-final49; run;
proc boxplot data=union_redes;
plot media*modelo; run;

/*CRUZADABINARIANEURAL-DATOS EQUILIBRADOS */
%cruzadabinarianeural(archivo=REST_EQUILIBRADO,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,
sfinal=12355,nodos=2,algo=LEVMAR,acti=ARC,objetivo=tasafallos,early=30
);
data final51; set final; modelo='Red2_TE'; run;

%cruzadabinarianeural(archivo=REST_EQUILIBRADO,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,
sfinal=12355,nodos=2,algo=TRUREG,acti=SIN,objetivo=tasafallos,early=13)
;
data final50; set final; modelo='Red1_TE'; run;
```

```
%cruzadabinarianeural(archivo=REST_EQUILIBRADO,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,
sfinal=12355,nodos=2,algo=Quanew,acti=ARC,objetivo=tasafallos,early=25);
data final52; set final; modelo='Red3_TE'; run;

%cruzadabinarianeural(archivo=REST_EQUILIBRADO,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,
sfinal=12355,nodos=2,algo=Quanew,acti=ARC,objetivo=tasafallos);
data final53; set final; modelo='Red4_TE'; run;

data union_redes;
set final46-final53; run;
proc boxplot data=union_redes;
plot media*modelo; run;
```

## Anexo L: Random forest

```
/*RANDOM FOREST – TOTAL DATOS:

%cruzadarandomforestbin(archivo=REST_TOTAL,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
maxtrees=700,variables=3,porcenbag=0.70,maxbranch=2,tamhoja=20,max
depth=10,pvalor=0.05,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data rf1; set final; modelo='rf1_TD'; run;

%cruzadarandomforestbin(archivo=REST_TOTAL,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
```

```
maxtrees=200,variables=5,porcenbag=0.70,maxbranch=2,tamhoja=15,max
depth=10,pvalor=0.1,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data rf2; set final; modelo='rf2_TD'; run;

%cruzadarandomforestbin(archivo=REST_TOTAL,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
maxtrees=100,variables=3,porcenbag=0.70,maxbranch=2,tamhoja=15,max
depth=10,pvalor=0.2,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data rf3; set final; modelo='rf3_TD'; run;

%cruzadarandomforestbin(archivo=REST_TOTAL,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
maxtrees=200,variables=3,porcenbag=0.75,maxbranch=2,tamhoja=15,max
depth=10,pvalor=0.2,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data rf4; set final; modelo='rf4_TD'; run;

%cruzadarandomforestbin(archivo=REST_TOTAL,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
maxtrees=400,variables=4,porcenbag=0.70,maxbranch=2,tamhoja=10,max
depth=10,pvalor=0.2,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data rf5; set final; modelo='rf5_TD'; run;

%cruzadarandomforestbin(archivo=REST_TOTAL,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
maxtrees=850,variables=5,porcenbag=0.80,maxbranch=2,tamhoja=5,maxd
epth=10,pvalor=0.05,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
```

```
data rf6; set final; modelo='rf6_TD'; run;

%cruzadarandomforestbin(archivo=REST_TOTAL,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
maxtrees=850,variables=12,porcenbag=0.70,maxbranch=2,tamhoja=5,max
depth=10,pvalor=0.2,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data rf7; set final; modelo='rf7_TD'; run;

%cruzadarandomforestbin(archivo=REST_TOTAL,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
maxtrees=900,variables=13,porcenbag=0.60,maxbranch=2,tamhoja=15,ma
xdepth=10,pvalor=0.2,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data rf8; set final; modelo='rf8_TD'; run;

data union_randomForest;
set rf1-rf8; run;
proc boxplot data=union_randomForest;
plot media*modelo;run;

/*RANDOM FOREST-DATOS EQUILIBRADO*/

%cruzadarandomforestbin(archivo=REST_EQUILIBRADO,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
maxtrees=700,variables=3,porcenbag=0.70,maxbranch=2,tamhoja=20,max
depth=10,pvalor=0.05,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data rf9; set final; modelo='rf1_TE'; run;

%cruzadarandomforestbin(archivo=REST_EQUILIBRADO,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
```

```
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
maxtrees=200,variables=5,porcenbag=0.70,maxbranch=2,tamhoja=15,max
depth=10,pvalor=0.1,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data rf10; set final; modelo='rf2_TE'; run;

%cruzadarandomforestbin(archivo=REST_EQUILIBRADO,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
maxtrees=100,variables=3,porcenbag=0.70,maxbranch=2,tamhoja=15,max
depth=10,pvalor=0.2,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data rf11; set final; modelo='rf3_TE'; run;

%cruzadarandomforestbin(archivo=REST_EQUILIBRADO,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
maxtrees=200,variables=3,porcenbag=0.75,maxbranch=2,tamhoja=15,max
depth=10,pvalor=0.2,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data rf12; set final; modelo='rf4_TE'; run;

%cruzadarandomforestbin(archivo=REST_EQUILIBRADO,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
maxtrees=400,variables=4,porcenbag=0.70,maxbranch=2,tamhoja=10,max
depth=10,pvalor=0.2,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data rf13; set final; modelo='rf5_TE'; run;

%cruzadarandomforestbin(archivo=REST_EQUILIBRADO,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
```

```
maxtrees=850,variables=5,porcenbag=0.80,maxbranch=2,tamhoja=5,maxd
epth=10,pvalor=0.05,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data rf14; set final; modelo='rf6_TE'; run;

%cruzadarandomforestbin(archivo=REST_EQUILIBRADO,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
maxtrees=850,variables=3,porcenbag=0.70,maxbranch=2,tamhoja=5,maxd
epth=10,pvalor=0.2,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data rf15; set final; modelo='rf7_TE'; run;

%cruzadarandomforestbin(archivo=REST_EQUILIBRADO,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
maxtrees=900,variables=6,porcenbag=0.60,maxbranch=2,tamhoja=15,max
depth=10,pvalor=0.2,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data rf16; set final; modelo='rf8_TE'; run;

data union_randomForest;
set rf1-rf16 B1 B2; run;
proc boxplot data=union_randomForest;
plot media*modelo;run;


/*BAGGING*/
%cruzadarandomforestbin(archivo=REST_TOTAL,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
maxtrees=100,variables=15,porcenbag=0.70,maxbranch=2,tamhoja=15,ma
xdepth=10,pvalor=0.05,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data B1; set final; modelo='B1_TD'; run;

%cruzadarandomforestbin(archivo=REST_EQUILIBRADO,
vardep=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
```

```
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas
REP_WS_4Estrellas REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
maxtrees=100,variables=15,porcenbag=0.70,maxbranch=2,tamhoja=15,ma
xdepth=10,pvalor=0.2,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);
data B2; set final; modelo='B2_TE'; run;

data union_Bagging;
set B1 B2; run;
proc boxplot data=union_Bagging;
plot media*modelo;run;
```

## Anexo LL: Gradient boosting

```
%cruzadatreeboostbin(archivo=REST_EQUILIBRADO,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4, sinicio=12345, sfinal=12355,
leafsize=5,iteraciones=20,shrink=0.1,
maxbranch=2,maxdepth=4,mincatsize=15,minobs=50,objetivo=tasafallos);
data finalBTG1;set final;modelo='BTG1_TE';RUN;

%cruzadatreeboostbin(archivo=REST_EQUILIBRADO,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4, sinicio=12345, sfinal=12355,
leafsize=5,iteraciones=80,shrink=0.03,
maxbranch=2,maxdepth=4,mincatsize=15,minobs=20,objetivo=tasafallos);
data finalBTG2;set final;modelo='BTG2_TE';RUN;

%cruzadatreeboostbin(archivo=REST_EQUILIBRADO,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4, sinicio=12345, sfinal=12355,
leafsize=5,iteraciones=80,shrink=0.20,
maxbranch=2,maxdepth=4,mincatsize=15,minobs=20,objetivo=tasafallos);
data finalBTG3;set final;modelo='BTG3_TE';RUN;
```

```
%cruzadatreeboostbin(archivo=REST_EQUILIBRADO,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4, sinicio=12345, sfinal=12355,
leafsize=15,iteraciones=80,shrink=0.2,
maxbranch=2,maxdepth=4,mincatsize=15,minobs=20,objetivo=tasafallos);
data finalBTG4;set final;modelo='BTG4_TE';RUN;

%cruzadatreeboostbin(archivo=REST_EQUILIBRADO,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4, sinicio=12345, sfinal=12355,
leafsize=15,iteraciones=80,shrink=0.1,
maxbranch=2,maxdepth=4,mincatsize=15,minobs=20,objetivo=tasafallos);
data finalBTG5;set final;modelo='BTG5_TE';RUN;

%cruzadatreeboostbin(archivo=REST_EQUILIBRADO,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4, sinicio=12345, sfinal=12355,
leafsize=5,iteraciones=80,shrink=0.10,
maxbranch=2,maxdepth=4,mincatsize=15,minobs=20,objetivo=tasafallos);
data finalBTG6;set final;modelo='BTG6_TE';RUN;

%cruzadatreeboostbin(archivo=REST_EQUILIBRADO,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4, sinicio=12345, sfinal=12355,
leafsize=25,iteraciones=80,shrink=0.1,
maxbranch=2,maxdepth=4,mincatsize=15,minobs=60,objetivo=tasafallos);
data finalBTG7;set final;modelo='BTG7_TE';RUN;

data union_GRADIENT;
set finalBTG1-finalBTG7; run;
proc boxplot data=union_GRADIENT;
```

```
plot media*modelo;run;


/*MACRO DE GRADIENT BOOSTING -Datos equilibrados*/

%cruzadatreeboostbin(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4, sinicio=12345, sfinal=12355,
leafsize=5,iteraciones=20,shrink=0.1,
maxbranch=2,maxdepth=4,mincatsize=15,minobs=50,objetivo=tasafallos);
data finalBTG8;set final;modelo='BTG1_TD';RUN;

%cruzadatreeboostbin(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4, sinicio=12345, sfinal=12355,
leafsize=5,iteraciones=80,shrink=0.03,
maxbranch=2,maxdepth=4,mincatsize=15,minobs=20,objetivo=tasafallos);
data finalBTG9;set final;modelo='BTG2_TD';RUN;

%cruzadatreeboostbin(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4, sinicio=12345, sfinal=12355,
leafsize=5,iteraciones=80,shrink=0.20,
maxbranch=2,maxdepth=4,mincatsize=15,minobs=20,objetivo=tasafallos);
data finalBTG10;set final;modelo='BTG3_TD';RUN;

%cruzadatreeboostbin(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4, sinicio=12345, sfinal=12355,
leafsize=15,iteraciones=80,shrink=0.2,
maxbranch=2,maxdepth=4,mincatsize=15,minobs=20,objetivo=tasafallos);
```

```
data finalBTG11;set final;modelo='BTG4_TD';RUN;

%cruzadatreeboostbin(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4, sinicio=12345, sfinal=12355,
leafsize=15,iteraciones=80,shrink=0.1,
maxbranch=2,maxdepth=4,mincatsize=15,minobs=20,objetivo=tasafallos);
data finalBTG12;set final;modelo='BTG5_TD';RUN;

%cruzadatreeboostbin(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4, sinicio=12345, sfinal=12355,
leafsize=5,iteraciones=80,shrink=0.10,
maxbranch=2,maxdepth=4,mincatsize=15,minobs=20,objetivo=tasafallos);
data finalBTG13;set final;modelo='BTG6_TD';RUN;

%cruzadatreeboostbin(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
conti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
categor=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4, sinicio=12345, sfinal=12355,
leafsize=25,iteraciones=80,shrink=0.1,
maxbranch=2,maxdepth=4,mincatsize=15,minobs=60,objetivo=tasafallos);
data finalBTG14;set final;modelo='BTG7_TD';RUN;

data union_GRADIENT;
set finalBTG1-finalBTG14; run;
proc boxplot data=union_GRADIENT;
plot media*modelo;run;
```

## Anexo M: Support vector machines

```
/*Support Vector Machines-todos los datos*/

/*POLYNOM*/
%cruzadaSVMbin
(archivo=REST_TOTAL,vardepen=C_Inspecciones1,
```

```
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12358,kernel=Polynom k_par=3,c=10);
data finalSVM1;set final;modelo='SVM1_P_TD';

%cruzadaSVMbin
(archivo=REST_TOTAL,vardepen=C_Inspecciones1,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,kernel=polynom k_par=2,c=0.5);
data finalSVM2;set final;modelo='SVM2_P_TD';

%cruzadaSVMbin
(archivo=REST_TOTAL,vardepen=C_Inspecciones1,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,kernel=polynom k_par=3,c=0.001);
data finalSVM3;set final;modelo='SVM3_P_TD';

/*LINEAL*/
%cruzadaSVMbin
(archivo=REST_TOTAL,vardepen=C_Inspecciones1,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,kernel=linear,c=15);
data finalSVM4;set final;modelo='SVM4_L_TD';

%cruzadaSVMbin
(archivo=REST_TOTAL,vardepen=C_Inspecciones1,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,kernel=linear,c=25);
data finalSVM5;set final;modelo='SVM5_L_TD';

%cruzadaSVMbin
(archivo=REST_TOTAL,vardepen=C_Inspecciones1,
```

```
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,kernel=linear,c=20);
data finalSVM6;set final;modelo='SVM6_L_TD';

%cruzadaSVMbin
(archivo=REST_TOTAL,vardepen=C_Inspecciones1,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,kernel=linear,c=30);
data finalSVM7;set final;modelo='SVM7_L_TD';

data union_SVM;
set finalSVM1-finalSVM7; run;
proc boxplot data=union_SVM;
plot media*modelo;run;

data union_SVM;
set finalSVM4 finalSVM5 finalSVM6 finalSVM7 finalSVM10 finalSVM11
finalSVM12; run;
proc boxplot data=union_SVM;
plot media*modelo;run;

/*Support Vector Machines-Datos equilibrados*/

/*POLYNOM*/
%cruzadaSVMbin
(archivo=REST_EQUILIBRADO,vardepen=C_Inspecciones1,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,kernel=Polynom k_par=3,c=10);
data finalSVM10;set final;modelo='SVM1_P_TE';

%cruzadaSVMbin
(archivo=REST_EQUILIBRADO,vardepen=C_Inspecciones1,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,kernel=polynom k_par=2,c=25);
data finalSVM11;set final;modelo='SVM2_P_TE';
```

```
%cruzadaSVMbin
(archivo=REST_EQUILIBRADO,vardepen=C_Inspecciones1,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,kernel=polynom k_par=3,c=0.5);
data finalSVM12;set final;modelo='SVM3_P_TE';

/*LINEAL*/
%cruzadaSVMbin
(archivo=REST_EQUILIBRADO,vardepen=C_Inspecciones1,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,kernel=linear,c=15);
data finalSVM15;set final;modelo='SVM6_L_TE';

%cruzadaSVMbin
(archivo=REST_EQUILIBRADO,vardepen=C_Inspecciones1,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,kernel=linear,c=25);
data finalSVM16;set final;modelo='SVM7_L_TE';

%cruzadaSVMbin
(archivo=REST_EQUILIBRADO,vardepen=C_Inspecciones1,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,sinicio=12345,sfinal=12355,kernel=linear,c=20);
data finalSVM17;set final;modelo='SVM8_L_TE';

data union_SVM;
set finalSVM10 finalSVM11 finalSVM12 finalSVM15 finalSVM16 finalSVM17;
run;
proc boxplot data=union_SVM;
plot media*modelo;run;
```

## Anexo N: Ensamblado

```
/*MACRO ENSAMBLADO*/
```

```
%macro cruzadastackcon
(archivo=,vardepen=,listclass=,listconti=,ngrupos=,seminicio=,semifinal=,
nodos=10,algo=levmar,rediter=100,/*red*/
maxtrees=,vars_to_try=,trainfraction=,leafsize=,maxdepth=,/*random forest
*/
bleafsize=,iterations=,bmaxbranch=,bmaxdepth=,shrinkage=,/* g boosting*/
kernel=lineal,c=10,degree=2,k_par=0.6 /*SVM*/);

data final;run;
*proc printto print='c:\ca.txt' log='c:\loga.txt';run;
%do semilla=&seminicio %to &semifinal;/*<<<<<******AQUI SE PUEDEN
CAMBIAR LAS SEMILLAS */
data dos;set &archivo;u=ranuni(&semilla);
proc sort data=dos;by u;run;

data dos (drop=nume);
retain grupo 1;
set dos nobs=nume;
if _n_>grupo*nume/&ngrupos then grupo=grupo+1;
run;

data fantasma;run;
data unionsalfin;run;
data unifin;run;

%do exclu=1 %to &ngrupos;

data tres;set dos;semilla=&semilla;if grupo ne &exclu then
vardep=&vardepen*1;run;

/*************************************************************/
/* LOGISTICA */
proc logistic data=tres noprint;/*<<<<<******SE PUEDE QUITAR EL
NOPRINT */
class &listclass;
model vardep=&listconti &listclass;
score out=saco;
;run;
/*************************************************************/
data sal1 (drop=p_1);set saco;predi1=p_1;run;
/*************************************************************/
/*RED */
PROC DMDB DATA=tres dmdbcat=catatres;
target vardep ;
var &listconti;
class vardep &listclass;
;run;

proc neural data=tres dmdbcat=catatres ;
input &listconti;
input &listclass /level=nominal;
target vardep/ id=o level=nominal;
hidden &nodos/ id=h act=sof;
netoptions randist=normal ranscale=0.15 random=15459;
prelim 15 preiter=10 ;
```

```
train maxiter=&rediter technique=&algo;
score data=tres out=salred;
run;

data sal2 (keep=&vardepen predi2 grupo vardep semilla);set
salred;predi2=p_vardep1;run;

/*************************************************************/
/*RANDOM FOREST*/
/*************************************************************/

proc hpforest data=tres
maxtrees=&maxtrees vars_to_try=&vars_to_try trainfraction=&trainfraction
leafsize=&leafsize maxdepth=&maxdepth
exhaustive=5000
missing=useinsearch ;
target vardep/level=nominal;
input &listconti/level=interval;
input &listclass/level=nominal;
score out=salo;
run;

data sal3 (keep=&vardepen predi3 grupo vardep);set
salo;predi3=p_vardep1;run;

/*************************************************************/
/*GRADIENT BOOSTING */
/*************************************************************/

  proc treeboost data=tres
  exhaustive=1000 intervaldecimals=max
  leafsize=&bleafsize iterations=&iterations maxbranch=&bmaxbranch
  maxdepth=&bmaxdepth mincatsize=15 missing=useinsearch
shrinkage=&shrinkage
  splitsize=50;
  input &listclass/level=nominal;
  input &listconti/level=interval;
  target vardep /level=binary;
  subseries largest;
  score out=salboost;
run;
data sal4 (keep=&vardepen predi4 grupo vardep);set
salboost;predi4=p_vardep1;run;

/*************************************************************/
/* SVM */
/*************************************************************/
data tres ;set dos;if grupo ne &exclu then vardep=&vardepen;run;

/*************************************************************/
/* SVM */
/*************************************************************/
%if &kernel=lineal %then %do;
proc hpsvm data=tres;
input &listclass/ level=nominal;
```

```
input &listconti / level=interval;
target vardep;
penalty C=&c;
output out=sal5;
run;

data sal5;merge tres sal5;run;

data sal5(keep=predi5);set sal5;
predi5=p_vardep1;
run;

%end;

%else %if &kernel=polynom %then %do;

proc hpsvm data=tres;
input &listclass/ level=nominal;
input &listconti / level=interval;
target vardep;
kernel polynom /degree=&degree;
penalty C=&c;
output out=sal5;
run;

data sal5;merge tres sal5;run;

data sal5(keep=predi5);set sal5;
predi5=p_vardep1;
run;

%end;

%else %if &kernel=RBF %then %do;

proc hpsvm data=tres method=activeset;
input &listclass/ level=nominal;
input &listconti / level=interval;
target vardep;
kernel RBF /k_par=&k_par;
penalty C=&c;
output out=sal5;
run;

data sal5;merge tres sal5;run;

data sal5(keep=predi5);set sal5;
predi5=p_vardep1;
run;

%end;


/* PRUEBAS CON STACKING */
data unionsal (drop=ygorro);merge sal1 sal2 sal3 sal4 sal5;
```

```
predi6=(predi1+predi2)/2; /* RED -LOG */
predi7=(predi1+predi3)/2;/* RED -RFOR */
predi8=(predi1+predi4)/2;/* RED -BOOST*/
predi9=(predi2+predi3)/2;/* LOG-RFOR */
predi10=(predi2+predi4)/2;/* LOG-BOOST */
predi11=(predi3+predi4)/2;/* RFOR-BOOST */
predi12=(predi1+predi2+predi3)/3;/* RED -LOG-RFOR */
predi13=(predi1+predi2+predi4)/3;/* RED -LOG-BOOST*/
predi14=(predi1+predi3+predi4)/3;/* RED -RFOR-BOOST*/
predi15=(predi2+predi3+predi4)/3;/* LOG-RFOR-BOOST*/
predi16=(predi1+predi2+predi3+predi4)/4;/* RED-LOG-RFOR-BOOST*/
predi17=(predi1*0.2+predi2*0.1+predi3*0.5+predi4*0.2);/* RED-LOG-RFOR-
BOOST ponderado*/
predi18=(predi1+predi5)/2; /* RED -SVM */
predi19=(predi3+predi5)/2; /* RFOR -SVM */
predi20=(predi2+predi5)/2; /* LOG-SVM */
predi21=(predi4+predi5)/2; /* BOOST-SVM */
predi22=(predi5+predi2+predi3)/3;/* SVM-LOG-RFOR */
predi23=(predi1+predi2+predi3+predi4+predi5)/4;/* RED-LOG-RFOR-
BOOST-SVM*/
run;


data salfin (keep=&vardepen vardep predi1-predi23 grupo);set unionsal;if
grupo=&exclu then output;run;

data unionsalfin;set unionsalfin salfin;run;

data salbis (drop=i);
array predi{23};
array pre{23};
set salfin;
do i=1 to 23;
if predi{i}>0.5 then pre{i}=1;/* se puede cambiar la proporci_n */
if predi{i}<=0.5 then pre{i}=0;
end;
run;
data salbos;run;
%do j=1 %to 23;
proc freq data=salbis noprint;tables pre&j*&vardepen /out=salconfu;run;
data confu&j (keep=tasa&j);retain buenos 0 malos 0;set salconfu
nobs=nume;
if &vardepen=pre&j then buenos=buenos+count;
if &vardepen ne pre&j then malos=malos+count;
if _n_=nume then do;tasa&j=malos/(malos+buenos);output;end;
run;
data salbos;merge salbos confu&j;run;
;
%end;

data fantasma;set fantasma salbos;run;
%end;

/* FIN GRUPOS */
```

```
proc means data=fantasma noprint;var tasa1-tasa23;
output out=mediaresi mean=ase1-ase23 ;
run;
data mediaresi;set mediaresi;semilla=&semilla;run;
data final (keep=ase1-ase23 semilla);set final mediaresi;if ASE1=. then
delete;run;

data unifin;set unifin unionsalfin;run;

%end;
proc printto; run;
proc print data=final;run;
%mend;

/*ENSAMBLADO DATOS-DATOS EQUILIBRADOS*/

%cruzadastackcon(archivo=REST_EQUILIBRADO,
vardepen=C_Inspecciones1,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,seminicio=12345,semifinal=12355,
nodos=2,algo=LEVMAR,rediter=10,/*parametros red (acti=arc,sof,etc se
define en la macro) */
maxtrees=200,vars_to_try=3,trainfraction=0.75,leafsize=15,maxdepth=10,/*r
andom forest */
bleafsize=5, iterations=20, bmaxbranch=2, bmaxdepth=4, shrinkage=0.1,/* g
boosting*/
kernel=polynom,degree=2,c=25); /* SVM */

/* CORRELACIONES ENTRE PREDICCIONES PUNTUALES ULTIMA
SEMILLA Y GRUPO*/
proc corr data=salfin;var predi1-predi5;run;

/*PREPARACION GRAFICO Y ETIQUETAS */

data cajas;
array ase{23};
set final;
do i=1 to 23;
modelo=i;
error=ase{i};
output;
end;
run;

/* EN ESTAS OPCIONES SE CAMBIA LA LETRA Y LA ALTURA DEL
TEXTO EN LOS EJES CON HTEXT.
options font="Courier New" bold 8;
run;goptions htext=8pt;
*/
```

```
proc sort data=cajas;by modelo;
data eti;length eti $ 13;
input modelo eti $;
cards;
1 LOG
2 RED
3 RFOR
4 BOOST
5 SVM
6 RLOG
7 REDFOR
8 REDBOO
9 LRFOR
10 LBOOST
11 RFORBOO
12 R-L-RFOR
13 R-L-BOO
14 R-RF-BOO
15 L-RF-BOO
16 R-L-RF-BOO
17 R-L-RF-B-ponderado
18 R-SVM
19 RF-SVM
20 L-SVM
21 BOO-SVM
22 SVMLRF
23 RLRFBSVM
;
data cajas2;merge cajas eti;by modelo;
title1
h=2 box=1 j=c c=red 'INCIDENT' j=c ;

options font="Courier New" bold 8;
run;goptions htext=5pt;

ods graphics off;

proc boxplot data=cajas2;plot error*ETI /
cboxes       = dagr
cboxfill     = ywh;
/* vaxis=0.20 to 0.35 by 0.01 */
;run;

/*data discoc.final1;set final;run;
data discoc.stack1;set cajas2;run;*/
goptions reset=all;

/*libname onedrive 'C:\Users\Ruth\OneDrive\Cuso de Master II ciclo\Machine
Learning\Practica2';
data incident_final; set onedrive.incident_final; run;
proc print data=incident_final; run;  */

data unifin ;set unifin;if Y=. then delete; /*Y es la var dependiente*/
RED=predi1;
```

```
LOG=predi2;
RFOR=predi3;
BOOST=predi4;
ENSAMBLADO=predi15;
run;

symbol v=dot;
axis1 order=0 to 1;
proc gplot data=unifin;
plot RED*LOG=Y RED*RFOR=Y RED*BOOST=Y LOG*RFOR=Y
LOG*BOOST=Y RFOR*BOOST=Y /*Y es la var dependiente*/
RED*ENSAMBLADO=Y
LOG*ENSAMBLADO=Y
RFOR*ENSAMBLADO=Y
BOOST*ENSAMBLADO=Y
/
vaxis=axis1 haxis=axis1 href=0.5 vref=0.5;
run;

/* SOLO PARA COMPROBAR CORRELACIONES PRINCIPALES
ALGORITMOS */
/* CORRELACIONES ENTRE PREDICCIONES PUNTUALES ULTIMA
SEMILLA Y GRUPO*/
proc corr data=salfin;var predi1-predi5;run;

proc print data=salfin;run;

/*ENSAMBLADO DATOS-TOTAL DE DATOS*/

%cruzadastackcon(archivo=REST_TOTAL,
vardepen=C_Inspecciones1,
listconti=REP_C_TotalGradoB REP_C_TotalGradoC REP_C_TotalGradoA
REP_C_TotalInsp REP_WS_PuntuacionRating WS_3Estrellas
IMP_REP_WS_TotalComentarios REP_WS_5Estrellas REP_WS_4Estrellas
REP_WS_2Estrellas REP_WS_1Estrella,
listclass=C_EstacionInspecc C_MesInspecc NY_TipoInspeccion
REP_REP_NY_DescripcionComida,
ngrupos=4,seminicio=12345,semifinal=12355,
nodos=4,algo=TRUREG,rediter=19,/*parametros red (acti=arc,sof,etc se
define en la macro) */
maxtrees=700,vars_to_try=3,trainfraction=0.70,leafsize=20,maxdepth=10,/*r
andom forest */
bleafsize=15, iterations=80, bmaxbranch=2, bmaxdepth=4, shrinkage=0.1,/*
g boosting*/
kernel=linear,c=30); /* SVM */

/* CORRELACIONES ENTRE PREDICCIONES PUNTUALES ULTIMA
SEMILLA Y GRUPO*/
proc corr data=salfin;var predi1-predi5;run;

/*PREPARACION GRAFICO Y ETIQUETAS */

data cajas;
array ase{23};
set final;
```

```
do i=1 to 23;
modelo=i;
error=ase{i};
output;
end;
run;

/* EN ESTAS OPCIONES SE CAMBIA LA LETRA Y LA ALTURA DEL
TEXTO EN LOS EJES CON HTEXT.
options font="Courier New" bold 8;
run;goptions htext=8pt;
*/

proc sort data=cajas;by modelo;
data eti;length eti $ 13;
input modelo eti $;
cards;
1 LOG
2 RED
3 RFOR
4 BOOST
5 SVM
6 RLOG
7 REDFOR
8 REDBOO
9 LRFOR
10 LBOOST
11 RFORBOO
12 R-L-RFOR
13 R-L-BOO
14 R-RF-BOO
15 L-RF-BOO
16 R-L-RF-BOO
17 R-L-RF-B-ponderado
18 R-SVM
19 RF-SVM
20 L-SVM
21 BOO-SVM
22 SVMLRF
23 RLRFBSVM
;
data cajas2;merge cajas eti;by modelo;
title1
h=2 box=1 j=c c=red 'Restaurantes-Datos equilibrados' j=c ;

options font="Courier New" bold 8;
run;goptions htext=5pt;

ods graphics off;

proc boxplot data=cajas2;plot error*ETI /
cboxes       = dagr
cboxfill     = ywh;
/* vaxis=0.20 to 0.35 by 0.01 */
;run;
```

```
/*data discoc.final1;set final;run;
data discoc.stack1;set cajas2;run;*/
goptions reset=all;

data unifin ;set unifin;if Y=. then delete; /*Y es la var dependiente*/
RED=predi1;
LOG=predi2;
RFOR=predi3;
BOOST=predi4;
ENSAMBLADO=predi15;
run;

symbol v=dot;
axis1 order=0 to 1;
proc gplot data=unifin;
plot RED*LOG=Y RED*RFOR=Y RED*BOOST=Y LOG*RFOR=Y
LOG*BOOST=Y RFOR*BOOST=Y /*Y es la var dependiente*/
RED*ENSAMBLADO=Y
LOG*ENSAMBLADO=Y
RFOR*ENSAMBLADO=Y
BOOST*ENSAMBLADO=Y
/
vaxis=axis1 haxis=axis1 href=0.5 vref=0.5;
run;
```