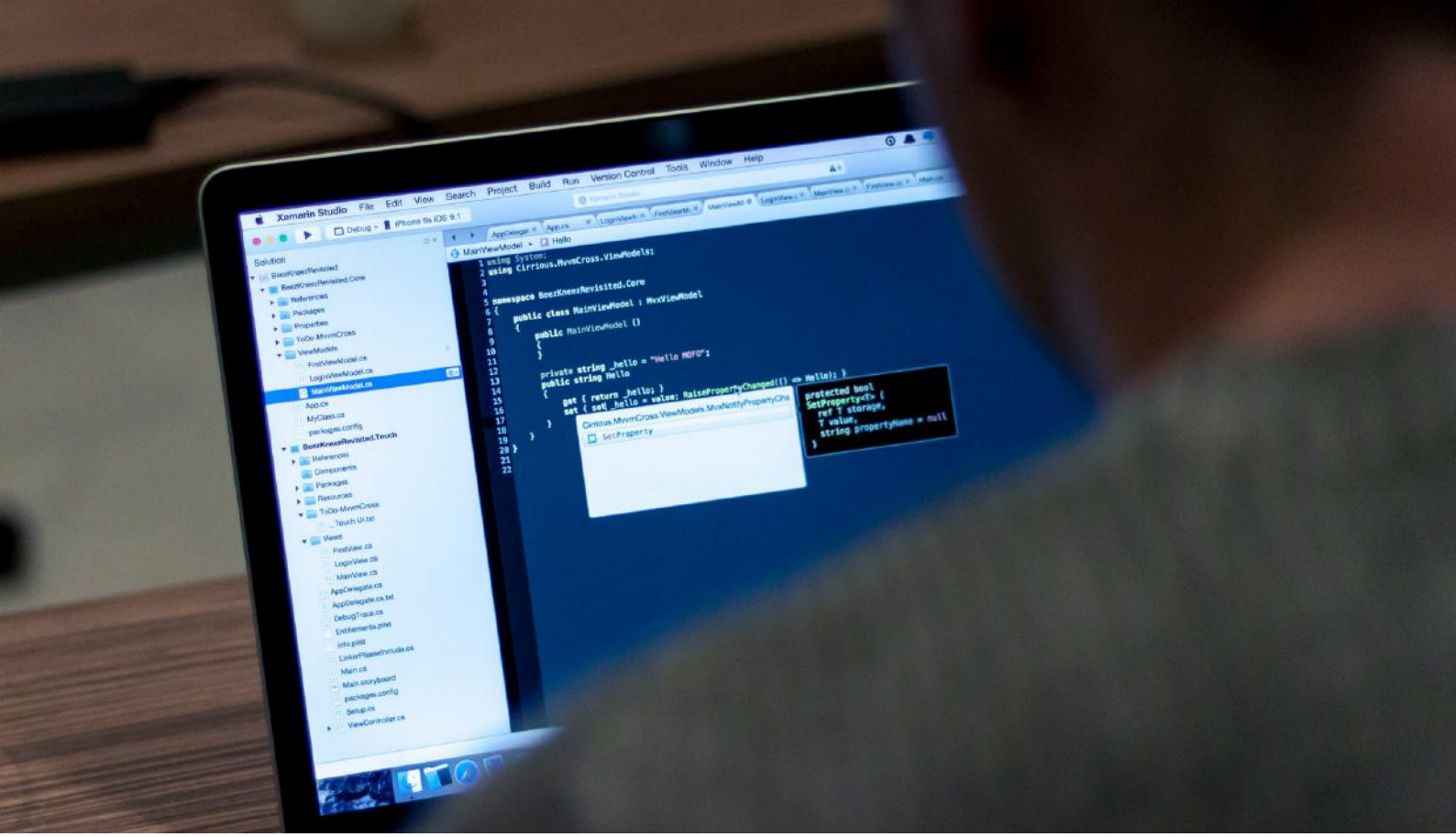


Computación estadística: Uso del lenguaje r básico para auditoría basada en evidencia estadística

Eduardo Leyton Guerrero, MBA en Dirección Informática IDE-CESEM, Madrid, España, director de Carrera Contador Público Auditor, Facultad de Economía y Negocios, UAH.

Introducción

La auditoría, cuya labor es el examen sistemático y crítico de la información financiera y operativa, ha ido ampliando sus fronteras gracias al desarrollo de nuevas tecnologías. En esta evolución, la computación estadística se convierte en un aliado esencial, al combinar métodos estadísticos con sistemas informáticos de alto rendimiento que permiten recolectar, procesar y modelar grandes volúmenes intangibles digitales. Estas capacidades potencian la detección de anomalías y patrones de fraude en entornos contables complejos, facilitando decisiones estratégicas más precisas. No obstante, el acceso a herramientas computacionales avanzadas como SPSS, MiniTab, MatLab, Stata o SAS puede verse limitado por sus licencias privativas. Frente a este escenario, se han consolidado otras opciones aplicables a la auditoría, como las CAAT (Técnicas de Auditoría Asistidas por Computador), con softwares comerciales como ACL, IDEA o Tableau, y también el siempre vigente Excel (Navarrete, 2019; Westland, 2020). Este último ha evolucionado de ser una simple planilla electrónica a incorporar funcionalidades como Power Pivot, Power Query y su extensión en la plataforma Power BI, fortaleciendo la analítica de datos. Sin embargo, estas soluciones no cubren todas las necesidades de modelización avanzada. Es en este punto donde emergen lenguajes de programación de código abierto como R, especializado en computación estadística; Python, con funcionalidades múltiples; y otros más recientes como Julia y Ruby que, entre otras características, son capaces de entrenar algoritmos mediante redes neuronales artificiales para abordar problemas en contextos de alta incertidumbre. Estas herramientas también permiten programar complejos modelos de Minería de Datos, Minería de Texto y Minería de Sentimientos, entre otros enfoques basados en machine learning y deep learning, con gran potencial de aplicación en el ejercicio profesional del auditor (Vargas, 2025). Complementariamente, en la auditoria de vanguardia, hoy, es posible advertir una propensión o señales de potenciales dolo mediante el análisis de textos a notas a los estados financieros, memorias; y, por otro lado, analizar características biométricas de los gestores de la organización, o responsables de procesos de negocios, mediante la descomposición de fotografías de sus rostros. De lo anterior se desprende que el ejercicio de la auditoria, hoy más que nunca, cuenta con herramientas, en todos sus niveles, que le permiten comprender, modelizar y predecir potenciales irregularidades en transacciones contables y comerciales. La cuestión es si lo auditores están capacitados para usarlas, especialmente aquellas basadas en computación estadística. No podemos desconocer que ellas demandan, en algunos casos, sólidos conocimientos de estadística para interpretar correctamente las salidas de sus algoritmos. Ello nos hace meditar en la formación universitaria de la profesión de Contador Público Auditor, en sus perfiles de egresos, entre otras exigencias académicas. En lo particular el lenguaje R, como foco central de análisis y desarrollo de esta investigación, tiene como característica principal ser de código abierto, orientado al objeto, interpretado y apoyado por una potente comunidad de cientos de miles de expertos en estadísticas y en programación computacional, la gran mayoría de ellos con grado académico superior; su comunidad se encuentra en Cran.r-Proyect (https://cran.r-project.org/), que sostienen los distintos paquetes o librerías (conjunto de funciones y argumentos) entre otras múltiples particularidades ya comentadas en párrafos anteriores (Pérez, 2020). Se suma, además, el entorno de desarrollo integrado (IDE) Rstudio que actúa como interfaz gráfica no solamente para el lenguaje R, sino también para otros lenguajes computacionales (Redondo, 2024).

Metodología

Este trabajo se desarrolló bajo un enfoque exploratorio-aplicado, orientado a examinar el potencial de R como herramienta de apoyo en el ámbito de la auditoría financiera y operativa. Durante el periodo noviembre del 2024 a enero del 2025, se ejecutaron diversos paquetes del entorno R (ver Tabla 1) con el propósito de validar su aplicabilidad en tareas de análisis contable, detección de anomalías y auditoría de transacciones masivas. El proceso se apoyó en conjuntos de datos simulados y reales, recopilados desde fuentes internas y externas, que permitieron replicar situaciones habituales en auditorías financieras. Además de la aplicación práctica de los paquetes, se realizó una revisión sistemática de literatura académica y técnica relacionada con el uso de software estadístico en auditoría, lo que proporcionó una base teórica para contextualizar los hallazgos. La triangulación entre evidencia empírica y revisión bibliográfica permitió contrastar las funcionalidades de R con otras soluciones comerciales disponibles, ya mencionadas anteriormente, destacando ventajas y limitaciones del lenguaje en escenarios reales de auditoría.

A continuación, se detallan algunos de los principales paquetes empleados, junto con su propósito metodológico:

Tabla 1. Paquetes de R Usados

Nombre paquete	Función
dplyr	Potente depurador de datos (imprescindible)
benford.analysis	análisis de transacciones irregulares
psych	Análisis estadístico
estadistica	Análisis estadístico en español
skimr	Exploración de dataframe
Dlookr	Análisis exploratorio de datos (EDA) y diagnóstico de datos
DataExplorer	Exploración de datos
Hmisc	Potente análisis estadístico
tseries	Análisis de series de tiempo y predicción de transacciones.
Summarytools	Exploración y resumen de datos
DescTools	Estadística descriptiva y exploración de datos
ineq	Índice de GINI, curva de Lorentz, detección de valores irregulares.
fdth	Tabla de frecuencias

Resultados

Análisis de Estructuras

Una vez importado el o los datasets de interés para el auditor, usando los paquetes que correspondan tales como: readr, readxl, openxlsx, haven, foreign, vroom o data.table, entre otros, es imperativo estudiar su diccionario de datos y su estructura para su comprensión. Este análisis es recomendable efectuarlo con las siguientes funciones, entre otras: str(df), structure(df) y glimpse(df.). Las salidas nos entregarán el tipo de vector (columnas del dataframe) si es numérico, fecha, texto, lógico, etc., que componen al df para definir la estrategia de auditoría de análisis o analítica de datos, según el objetivo de estudio. Función aplicada es:

```
str(iris)
'data.frame':        150 obs. of  5 variables:
 $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species   : Factor w/ 3 levels "setosa","versicolor",..:
```

Adicionalmente, es posible destacar las funciones DataExplorer del paquete del mismo nombre, que resume, en formato html, toda la información en una sola salida de la estructura del df. La función es: DataExplorer::create_report(df).Asimismo, la función GWalkR, una función navegable y de múltiple aplicabilidad, nos proyecta en forma dinámica la estructura del df. Su función es: GWalkR::gwalkr(df). Por último, la función diagnose(df), presenta una rápida visión de la estructura del df.

```
diagnose(iris)
# A tibble: 5 × 6
  variables    types   missing_count missing_percent unique_count unique_rate
  <chr>        <chr>           <int>           <dbl>        <int>       <dbl>
1 Sepal.Length numeric             0               0           35       0.233
2 Sepal.Width  numeric             0               0           23       0.153
3 Petal.Length numeric             0               0           43       0.287
4 Petal.Width  numeric             0               0           22       0.147
5 Species      factor              0               0            3       0.02
```

Análisis Estadístico Descriptivo

Un examen estadístico descriptivo rápido e inicial nos entregará la función summary(df) que nos proyectará, por cada vector, el valor mínimo, el primer cuartil, la mediana o el segundo cuartil, el promedio, el tercer cuartil y el máximo. La función aplicada es:

```
summary(iris)
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   setosa     :50
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50
 Median :5.800   Median :3.000   Median :4.350   Median :1.300   virginica :50
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
```

Una segunda e interesante función describe(df) nos entregará el número de observaciones del df, valores NA (valores irreconocibles del vector), promedio, desviación estándar, error estándar de la media, recorrido intercuartílico, skewness o asimetría de un vector del df o de la distribución bajo estudio, kurtosis o forma de una distribución del vector, valor mínimo y cuartiles. Su función es: describe(iris).

Figura 1: Función aplicada

```
> describe(iris)
# A tibble: 4 × 26
  described_variables     n    na  mean    sd se_mean   IQR skewness kurtosis   p00   p01   p05   p10   p20   p25   p30
  <chr>               <int> <int> <dbl> <dbl>   <dbl> <dbl>    <dbl>    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1 Sepal.Length          150     0  5.84 0.828  0.0676   1.3    0.315   -0.552   4.3   4.4   4.6   4.8   5     5.1   5.27
2 Sepal.Width           150     0  3.06 0.436  0.0356   0.5    0.319    0.228   2     2.2   2.34  2.5   2.7   2.8   2.8
3 Petal.Length          150     0  3.76 1.77   0.144    3.5   -0.275   -1.40    1     1.15  1.3   1.4   1.5   1.6   1.7
4 Petal.Width           150     0  1.20 0.762  0.0622   1.5   -0.103   -1.34    0.1   0.1   0.2   0.2   0.2   0.3   0.4
# i 10 more variables: p40 <dbl>, p50 <dbl>, p60 <dbl>, p70 <dbl>, p75 <dbl>, p80 <dbl>, p90 <dbl>, p95 <dbl>, p99 <dbl>,
#   p100 <dbl>
```

Complementariamente, la detallada y agrupada salida de la función (describe.by(df, group= df$vector) aparece como una de las favoritas del auditor al necesitar una estadística más afinada por un grupo de característica de un vector. Su función es: psych::describeBy(df_iris,group = df_iris$Species). Su función es:

```
Descriptive statistics by group
group: setosa
             vars  n mean   sd median trimmed  mad min max range skew kurtosis   se
Sepal.Length    1 50 5.01 0.35    5.0    5.00 0.30 4.3 5.8   1.5 0.11    -0.45 0.05
Sepal.Width     2 50 3.43 0.38    3.4    3.42 0.37 2.3 4.4   2.1 0.04     0.60 0.05
Petal.Length    3 50 1.46 0.17    1.5    1.46 0.15 1.0 1.9   0.9 0.10     0.65 0.02
Petal.Width     4 50 0.25 0.11    0.2    0.24 0.00 0.1 0.6   0.5 1.18     1.26 0.01
Species         5 50 1.00 0.00    1.0    1.00 0.00 1.0 1.0   0.0  NaN      NaN 0.00
```

Funciones adicionales para considerar, que son una combinación de estadística y exploración de datos, son las dfSummary(df), ctable(df$vector1,df$vector2), stby() de estadística cruzada por 2 vectores, freq(), fre() para tablas de frecuencias, funciones de la familia apply, entre otras que el auditor puede explorar en las distintas opciones de búsqueda tanto en Rstudio como fuera de ella.

Búsqueda de Duplicados/Faltantes y secuencia

Para aquellos auditores cuyas pruebas esenciales de análisis de documento que exigen numeración consecutiva, las funciones que se muestran a continuación aparecen como significativas como paquete janitor::get_dupes(df,nro_factura), para faltantes setdiff(df, facturas), fuera de secuenciafacturas[which(diff(facturas) <= 0) + 1], detectar grandes saltos diff(facturas), no correlativos facturas[which(saltos > 1) + 1] entre otras funciones indirectas. Adicionalmente, los paquetes: audit y assertive, son una importante alternativa.

Ley de Benford o Ley del Primer Digito.

Una de las funciones más importante para detectar valores anómalos, es la Ley de Benford que aparece como insustituible a la hora de estudiar registros de transacciones comerciales o de otra naturaleza. Esta ley se basa en la distribución de los primeros dígitos significativos de un conjunto de datos numéricos, los cuales pueden estar asociados a montos monetarios u otras magnitudes. Esta ley es válida cuando los datos no han sido generados de forma aleatoria, artificial o mediante fórmulas, y cuando el conjunto de datos contiene una cantidad suficiente de observaciones —generalmente se recomienda sobre 500 registros— para que se observe confiablemente el patrón probabilístico esperado. La función Benford establece que, en muchos conjuntos de datos reales, los números que comienzan con dígitos bajos (como 1, 2 o 3) aparecen con mayor frecuencia que los que comienzan con dígitos altos (como 8 o 9). Esta distribución estadística, erróneamente llamada ley, es aceptada internacionalmente como medio de prueba y es utilizada como una irremplazable herramienta de búsqueda de patrones anómalos, incluso se ha utilizado exitosamente en procesos eleccionarios.

(Nigrini, 2020): $P(d) = \log_{10}\left(1 + \frac{1}{d}\right)$

$P(d) = \log_{10}(1 + \frac{1}{d})$

En lo particular, para este caso se aplicó un dataframe[1](df) real del libro de ventas, con 5781 facturas, de una importante compañía de fábrica de chocolates en Chile, la que fue analizada por quien suscribe en una auditoria de rutina. La estructura final del df(), que originalmente contenía de 47 vectores o columnas, una vez depurado con el paquete dplyr, se sintetizó en 6 vectores o columnas: n_doc, fecha_transaccion, rut_emisor, cliente_rut, identificación_vendedor y Valor_Neto. La identificación de Rut y nombres fueron alterados para mantener la confidencialidad del caso. Una vez importado el libro de ventas, efectuado el análisis de estructura, estadístico descriptivo y tablas de frecuencia, es decir comprendido el contenido de la data, se procedió a la carga y activación del paquete para posteriormente aplicar dplyr para la detección e individualización de las transacciones sospechosas.

[1] Estructura de datos de carácter matricial propia del lenguaje R. Es similar a una hoja de Excel.

Simplificando la secuencia de funciones iniciales, script o algoritmos, por la extensión del código se ha resumido en:

```
# Carga de Paquetes ----------------------------------------------------
install.packages("benford.analysis") # Análisis de Fraude
install.packages("fdth") # crear tablas de frecuencia
install.packages("ggplot2") # gramáticas de graficas
install.packages("dplyr") # manipulación de datos y estructuras de datos
# Activación de Paquetes -----------------------------------------------
library(benford.analysis)
library(fdth)
library(ggplot2)
library(readxl)
library(dplyr)
```

A continuación, se ejecuta la función Benford aplicándolo al df y vector numérico a analizar, df_libro_ventas_limpio_positivos$NETO, indicando que en este caso se analizará el primer digito significativo. Si bien se ha explorado hasta el 9° cifra significativa, con el consiguiente costo de procesamiento, benford recomienda ejecutar solo hasta 3 cifras significativas; es decir con montos por ejemplo $347.xxx.xxx o $134.xxx o $252. El argumento sign es para indicar si ese vector contiene valor negativo o positivo (pueden ser ambos) y la opción final de TRUE (verdadero) indica si es discreto o continuo. Su función es:

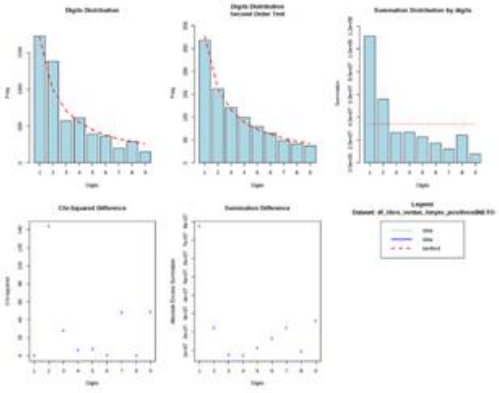
```
df_benford <- benford(df_libro_ventas$NETO,1,sign = "positive",TRUE)
plot(df_benford)
summary(df_benford $MAD)
```

Una vez aplicado benford, éste genera datos estadísticos[2] relevantes que, en este caso, son registrados en df_benford. A continuación, podemos generar las gráficas propias con la función plot() del proceso y subsiguientemente efectuar un rápido summary o resumen estadístico.

Tabla 2: Gráficas de salida de benford

Gráfico	Muestra	¿Para qué sirve en auditoría?
Digits Distribution	Frecuencia de digitos	Detectar patrones anómalos o manipulados en la frecuencia
Summation Distribution	Suma total por digito	Ver si el dinero se concentra en digitos sospechosos
Chi-squared Difference	Diferencia que contribuye al test	Identificar qué digitos alteran más la conformidad estadistica
Summation Difference	Diferencia entre suma observada y esperada	Encontrar montos "extraños" por digito

Figura 3: Gráficas de Análisis del Auditor



En este caso específico, la gráfica más significativa es de Digits Distribution donde se aprecian dígitos reveladores que comparados con la curva teórica calculadas por benford y la distribución real, muestran valores anómalos como las transacciones que comienzan con 2, 3, 5, 7 y 9. Asimismo, la gráfica de Chi-squared resalta aquellos dígitos anómalos en primer orden. A continuación, analizados el universo de 5781 transacciones del libro de ventas, con la función getSuspects() se detectan 1947 observaciones sospechosas. Si bien el paquete benford.analysis o benford, contiene una veintena de funciones (puede variar según la versión de este) las más importantes son las ejecutadas anteriormente. Detectadas las transacciones irregulares, corresponde ahora identificarlas e individualizarlas. Ello puede hacerse con cualquier paquete y funciones de depuración de datos distinto a benford. En este caso específico, dplyr. Las funciones aplicadas son filter, select, group by, ungroup, arrange, entre otras. Finalmente se identificaron, asociados a un vendedor determinado, transacciones sospechosas por un total de $4.590.000 con 170 facturas de $27.000 cada una. En este punto y con la información a la vista, el auditor debe llamar su atención para realizar las indagatorias que corresponda para esclarecer o efectuar cargos que correspondan.

Muestreo

Si bien con las herramientas ya comentadas es posible estudiar y analizar la totalidad del universo de datos, es evidente que, a la hora de contrastarlos con documentos físicos, éste se hace inviable para su tratamiento. Es por ello por lo que las técnicas de muestreo cobran especial relevancia constituyéndose en un instrumento irreemplazable para la auditoría, por ejemplo, de ventas, compras y/o pagos de remuneraciones, entre otras. El lenguaje R contiene varias

[2] La función benford genera además un conjunto de parámetros estadístico, entre otros, Kolmogorov-Smirnov test. En esta salida, el auditor debe prestar especial atención el indicador MAD (desviación media absoluta) y su valor. Según el experto internacional en fraude, Mark Nigrini, si éste es < 0.006 significa que los datos se ajustan excelentemente. Si fluctúa entre 0.006 y 0.012 su ajuste es aceptable, si está entre 0.012 y 0.015 ligera sospecha, si es > 0.015 existe una desviación altamente sospechosa.

funciones y argumentos sustentados en técnicas estadísticas que, por extensión del presente trabajo, éstas no serán estadísticamente explicadas. Sin embargo, se presentan algunas de las funciones aplicables más utilizadas en R. En efecto, sobre un dataset "df_auditoria" con 1000 valores aleatorios con montos que varían entre 1.000 y 10.000 m.u.m. con estructura: id., monto, sucursal, tipo_cuenta, control_aprobado, se muestrean las siguientes técnicas y funciones:

Tabla 3: Principales Técnicas de Muestro de R

Técnica	¿Por qué es clave en auditoría?
1. Muestreo Aleatorio Simple (MAS)	Base de muchos procedimientos. Asegura representatividad y es fácil de aplicar con R. **n** <- 50 muestra_mas <- df_auditoria %>% sample_n(n), # muestrea 50 transacciones aplicado en el df_auditoria dejando los 50 valores de interés en un df secundario (nrow→ número de filas)

2. Muestreo Sistemático	Eficiente para poblaciones ordenadas como libros contables o listados cronológicos.
	k <- floor(nrow(df_auditoria) / n) # calcula el número de intervalo k (1000/50)=20
	inicio <- sample(1:k, 1) # aleatoriamente selecciona 1 valor cada 20
	indices_sist <- seq(from = inicio, to = nrow(df_auditoria), by = k) # secuencia de índice inicio-término del df_auditoria
	muestra_sist <- df_auditoria[indices_sist,] #finalmente extrae valores numéricos del df_auditoria

3. Muestreo Estratificado	Usado para mejorar precisión, separando grandes montos o cuentas críticas (por riesgo o materialidad).
	# Se divide por 'tipo_cuenta' y se extrae muestra proporcional
	muestra_estrat <- df_auditoria %>% # df_auditoria
	group_by(tipo_cuenta) %>% #agrupado por tipo de cuenta
	sample_frac(size = 0.1) # se extrae el 10% por estrato

4. Monetary Unit Sampling (MUS)	Estándar en auditoría financiera. Favorece selección de partidas con mayor valor. Ideal para detectar errores materiales.
	# Probabilidad proporcional al monto (cuanto más alto el monto, mayor chance de ser elegido)
	probabilidades <- df_auditoria$monto / sum(df_auditoria$monto) # vectorialmente determina la proporcionalidad monto por monto y lo divide por la suma de este.
	muestra_mus <- df_auditoria[sample(1:nrow(df_auditoria), size = 50, prob = probabilidades),] # calcula vectorialmente la muestra desde 1 hasta el largo de filas de df_auditoria (1000), seleccionando una muestra de 50 valores con la proporcionalidad calculada.

5. Muestreo de Atributos	Fundamental para pruebas de cumplimiento de controles (control interno, compliance, etc.).
	# Ejemplo, estimar tasa de desviación de controles (proporción de registros NO aprobados)
	n_atributos <- 45 # ejemplo número de aprobaciones de pagos, en reemplazos de montos. (cumple o no cumple)
	muestra atributos <- df_auditoria %>% sample_n(n_atributos) # calcula la muestra de 45 elementos o pagos (no montos)
	tasa_error <- mean(!muestra_atributos$control_aprobado) # calcula la inversa de la tasa o proporción de pagos no autorizados. (! Simboliza lo contrario o la diferencia)

Fuente: elaboración propia en base a códigos ejecutados

Conclusiones

De acuerdo con lo mostrado, la utilización del lenguaje R con RStudio, junto a su homólogo Python, se consolida como instrumento de software referente, desafiante y obligatorio en el mundo de la auditoría. Sin desconocer las extraordinarias funcionalidades de las alternativas comerciales ya indicadas anteriormente, éstas, por su elevado costo anual y limitadas o inexistentes funciones de aprendizaje profundo, otorgan, entre otros atributos, la supremacía a las herramientas de código abierto. Desde la perspectiva de la auditoría, existen innumerables técnicas estadísticas —imposibles de abarcarlas todas en este documento— que pueden y deben ser estudiadas y aplicadas en el ejercicio de nuestra profesión, precisamente con los lenguajes de programación ya comentados. La auditoría, en particular la auditoría forense, con un enfoque de Auditoría Analítica Predictiva y apoyada por el lenguaje R o Python, se presenta como una alternativa irremplazable y atractiva para procesar y efectuar pruebas de auditoría, por ejemplo, sobre magnitudes estratosféricas de datos (big data), para hacer frente a las dinámicas y complejas caras del fraude.

El mundo académico y laboral deben revisar y redefinir no solo los perfiles de egreso de nuestra profesión, sino también las competencias deseables en las áreas de auditoría de las organizaciones.

Referencias

- Navarrete, O. (2019). Estadística para contadores y auditores con R. Universidad Politécnica Salesiana.
- Nigrini, M. J. (2020). Forensic Analytics: Methods and Techniques for Forensic Accounting Investigations. John Wiley & Sons.
- Pérez, J. (2020). Introducción a la ciencia de datos en R. UD Editorial.
- Redondo, C. (2024). El programa R: Herramienta clave en investigación. Universidad de Cantabria.
- Vargas, C. (2025). Estadística aplicada a la investigación con R. Ediciones de la U.
- Westland, C. (2020). Audit analytics: Data science for the accounting profession. Springer.