# REPORTE PROYECTO FINAL
# DETECCIÓN DE ANOMALÍAS EN TRANSACCIONES DE TARJETAS DE CRÉDITO


**GRUPO 25**
Andrés Felipe Gualdrón Gutiérrez
Jersson Hernán Morales Hernández
Carina Lizebeth Ordoñez Araque


Aprendizaje No Supervisado – MIAD 2024-14
Septiembre 2024

# 1 RESUMEN

Este proyecto propone un enfoque para la detección de anomalías en transacciones de tarjetas de crédito utilizando los algoritmos Isolation Forest y One-Class SVM, orientados a mitigar el fraude financiero. Se utilizan datos altamente desbalanceados de transacciones de tarjetas de crédito, con un total de 284,807 registros, de los cuales solo 492 son fraudulentos.

La metodología incluye la exploración, preprocesamiento de datos, y la implementación de los modelos. El análisis comparativo emplea métricas como la precisión, recall, F1-score, y AUC-ROC.

El modelo One-Class SVM mostró un rendimiento sólido con un AUC mínimo de 0.8842, pero generó falsos positivos que impactan financieramente ($577,396.79). El Isolation Forest alcanzó un rendimiento mucho mejor en la detección de fraudes con un AUC mínimo de 0.8919, pero igualmente, con una tasa significativa de falsos positivos ($2,190,967.50). Ambos modelos requieren ajustes en los umbrales para reducir los falsos positivos y mejorar la precisión.

El análisis financiero subraya el buen desempeño de los modelos seleccionados en la detección de anomalías, pero también la importancia de optimizarlos para minimizar costos asociados a transacciones erróneas clasificadas como fraudulentas.

# 2 INTRODUCCIÓN

Este trabajo final busca aplicar los conocimientos adquiridos en el curso de Aprendizaje No Supervisado a un proyecto práctico de detección de fraudes en tarjetas de crédito. Se enfoca en utilizar técnicas de detección de anomalías, un área importante del aprendizaje automático no supervisado, para identificar patrones inusuales en transacciones sin etiquetar. El proyecto pretende demostrar cómo los fundamentos teóricos y prácticos del curso pueden emplearse en una aplicación del mundo real dentro del amplio espectro de usos del aprendizaje no supervisado.

El fraude en tarjetas de crédito representa una amenaza significativa para las instituciones financieras y para sus clientes debido a su creciente sofisticación y al impacto financiero asociado. Por lo tanto, la elección adecuada de los algoritmos de detección es crucial para identificar transacciones fraudulentas con precisión y eficacia.

Este proyecto utiliza modelos avanzados de detección de anomalías, i.e., **ISOLATION FOREST** y **ONE-CLASS SVM.** Estos modelos son seleccionados debido a su capacidad para manejar datos desbalanceados y a su eficacia en la identificación de patrones anómalos.

# 3 MATERIALES Y MÉTODOS

## 3.1 Descripción de los Datos.

Para entrenar y probar los modelos de detección de fraudes, utilizamos un conjunto de datos de transacciones con tarjetas de crédito realizadas en septiembre de 2013 por titulares en Europa. Este conjunto de datos incluye el registro de las transacciones durante dos días, lo que corresponde a 284,807 transacciones de las cuales 492 fueron determinadas como casos de fraude lo que hace que el conjunto de datos sea extremadamente desbalanceado.

La fuente de los datos se encuentra disponible en https://www.openml.org/d/1597.

## 3.2 Exploración de los Datos.

El *dataset* está compuesto únicamente por variables numéricas, que fueron transformadas mediante Análisis de Componentes Principales (**PCA**) para proteger la confidencialidad de la información original mientras que las variables **Time** y **Amount** son las únicas variables no transformadas en la base de datos para un total de 30 variables independientes sin registros nulos o faltantes. La descripción de las variables se muestra a continuación:

- **V1, V2, ..., V27, y V28**: Son los componentes principales generados por el algoritmo PCA.

- **Time**: Indica los segundos transcurridos desde la primera transacción registrada en el dataset.
- **Amount**: Es el monto de la transacción registrada. Su unidad es desconocida pero no es necesaria para el análisis.
- **Class**: Es la variable de respuesta que indica si una transacción es clasificada como fraudulenta (1) o como legítima (0).

La visualización de la matriz de correlación para las observaciones clasificadas como legítimas muestra cierto nivel de correlación entre algunos de los componentes principales **PCA** y las variables **Time** y **Amount**; y no muestra ninguna o una mínima correlación entre sí. Contrariamente, la matriz de correlación para las transacciones clasificadas como fraudulentas muestra una fuerte correlación entre los primeros 18 componentes principales. (Refiérase al Anexo 2)

Por otro lado, el intervalo entre transacciones es bastante pequeño, inclusive muchos de las transacciones son simultáneos generando registros que se pueden interpretar como duplicados; sin embargo, estos no fueron removidos para evitar la reducir la cantidad de transacciones fraudulentas. Respecto al monto de las transacciones, estas están en un rango desde 0 hasta 25,000, con una media de 88 y una mediana de 22, lo que concluye una distribución asimétrica. Finalmente, en su mayoría, el rango de los componentes principales tiene mínimos en las decenas de valores negativos y máximos en las decenas de valores positivos con medias en cero (0), y medianas que dependen de la presencia de valores atípicos.

Al determinar la distribución de cada una de las variables discriminando la clase, es notorio que, para los componentes principales, las transacciones clasificadas como legítimas, Clase 0, siguen, prácticamente, una distribución normal con una dispersión, mayormente, baja; mientras que las clasificadas como fraudulentas, muestran una asimetría con una dispersión alta, condición que puede ser una ventaja al momento de la detección de anomalías. (Refiérase al Anexo 3)

La similitud en las distribuciones de algunas variables entre las dos clases sugiere que no aportarían a la detección de anomalía de manera efectiva, entonces se realizaron pruebas **Kolmogorov-Smirnov (K-S)** para determinar si estas eran estadísticamente diferentes. Para contrastar, se tomaron variables donde las distribuciones no son similares visualmente, obteniendo los siguientes resultados:

| Ítem | Variable | Distribuciones Similares [3] | Prueba K-S | Ítem | Variable | Distribuciones Similares [3] | Prueba K-S |
|---|---|---|---|---|---|---|---|
| 1 | V13 | Si | [1] | 6 | V25 | Si | [1] |
| 2 | V15 | Si | [2] | 7 | V26 | Si | [1] |
| 3 | V22 | Si | [2] | 8 | V27 | No | [1] |
| 4 | V23 | No | [1] | 9 | V28 | No | [1] |
| 5 | V24 | Si | [1] | 10 | Amount | No | [1] |

[1] Estadísticamente diferentes
[2] No concluyente
[3] Significancia: 1%

Con base en los resultados arriba tabulados, en esta etapa, no se descartaron ninguna de las variables presentes originalmente en el *dataframe*.

Respecto a la variable Time, esta no muestra diferencias significativas entre las clases en cuanto a su distribución. Sin embargo, en la densidad de probabilidad, se destacan picos a ciertas horas, lo que indicaría que las transacciones fraudulentas pueden ocurrir en momentos específicos del día (refiérase al Anexo 4). Con el propósito de capturar esta información, se creó la variable **Datetime** en el *dataframe* para ser usada en el entrenamiento del modelo.

### 3.3 Preprocesamiento de los Datos:

Para optimizar el entrenamiento del modelo y de disminuir su dimensionalidad, se determinó el nivel de importancia de las variables mediante algoritmos de ensamblaje usados en aprendizaje de máquina supervisado, i.e., **Random Forest (RF)** y **XGBoost (XGB).** Estos algoritmos manejan de

manera eficiente la alta dimensionalidad del *dataset*, capturan las interacciones complejas y no lineales entre las variables y, además, tienen integrado la determinación de la importancia de las variables basados en distintas métricas, e.g., el coeficiente de **gini** para el caso de RF.

Las variables que más aportan al propósito a la detección de anomalías (predictibilidad) determinadas por cada algoritmo son similares, pero no son las mismas. La lista final de las variables fue definida considerando hasta un 90% del nivel de importancia total, y de acuerdo con los resultados basados en la métrica AUC al correr los algoritmos de detección de anomalías con parámetros típicos. Con base en los resultados (Refiérase al Anexo 5), se optó por usar las variables determinadas por el algoritmo **RF**: 17 variables cuyo nivel de importancia suma 90.76% e incluyen **Amount**, **V2-V4**, **V7-V12**, **V14**, y **V16-V21**.

Como parte de la etapa del procesamiento de datos, se determinó la necesidad de balancear o no las clases mediante *Under_Sampling* y *Over_Sampling – SMOTE*. El balance de las clases se evaluó con los resultados de la métrica AUC al correr los algoritmos de detección de anomalías con parámetros típicos.

Para el caso de **IF**, existe una leve disminución de la métrica al hacer *Under_Sampling*, y un leve aumento al hacer *Over_Sampling*; de igual manera, los tiempos de ejecución del algoritmo. Por otro lado, para el algoritmo **OC-SVM**, el balance de las clases afecta negativamente el valor de la métrica, y el tiempo de ejecución acorde con la cantidad de observaciones. En conclusión, el balance sintético de las clases no es necesario para obtener un desempeño apropiado de los algoritmos de detección de anomalías usados en este proyecto.

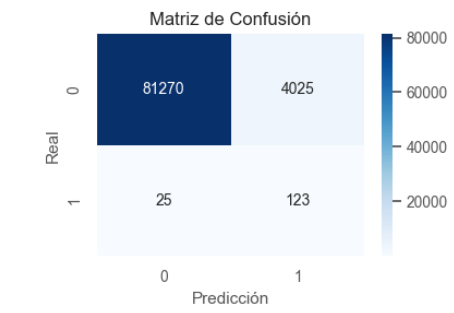
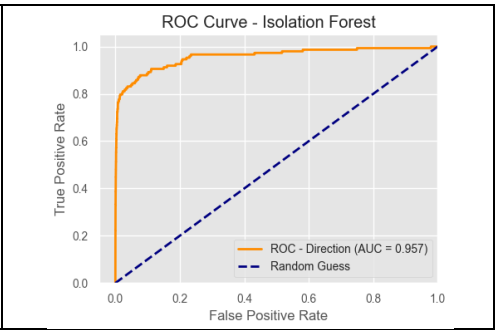
| Algoritmo | ROC_AUC - Dataset Desbalanceado | ROC_AUC - Dataset Under_Sampled | ROC_AUC - Dataset Over_Sampled |
|---|---|---|---|
| Isolation Forest | 0.9369 | 0.9051 | 0.9224 |
| One Class – SVM | 0.9111 | 0.7329 | 0.7321 |

# 4 RESULTADOS Y DISCUSIÓN

## 4.1 Isolation Forest (IF)

La función *IsoloationForest* de la librería *SciKit Learn* fue usada para correr este algoritmo. La función cuenta con hiper-parámetros que se ajustaron haciendo un *"Grid_Search"* manual con validación cruzada estratificada (*StratifiedKFold*) considerando que el entrenamiento del modelo se hace solo con la matriz correspondiente a las observaciones normales o en este caso, legítimas, y se evalúa con la matriz de prueba de los datos de las variables independientes y el correspondiente vector binario de la variable dependiente.

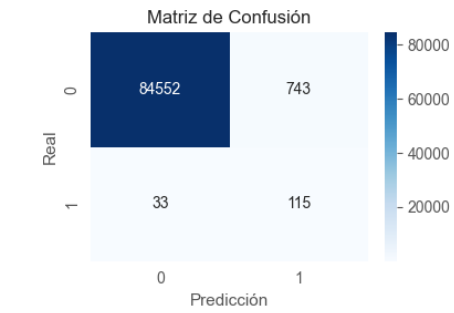
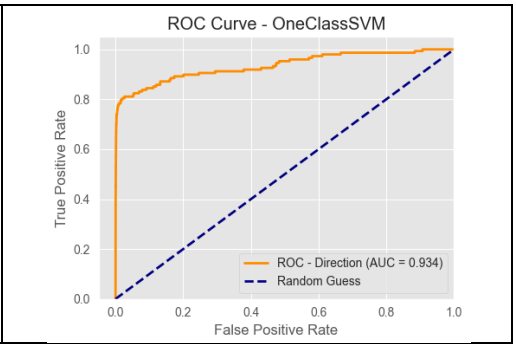
Tres (3) parámetros calibrados por medio del algoritmo *"Grid_Search",* se definieron anticipadamente debido a su fuerte impacto en los resultados: *contamination, max_sample, y bootstrap*. Entonces, los hiper-parámetros calibrados fueron *n_estimators* y max_features. Los resultados de las predicciones en la muestra de prueba (85.295 transacciones legitimas y 148 transacciones fraudulentas) para el modelo IF con los mejores hiper-parámetros se muestran en la siguiente tabla:

| | | | |
|---|---|---|---|
| **Isolation Forest** | *contamination = Auto*<br>*max_sample = Auto*<br>*bootstrap = True*<br>*n_estimators = 75*<br>*max_features = 0.315* |  |  |

### 4.2 One Class SVM (OC-SVM).

La función *OneClassSVM* de la librería *SciKit Learn* fue usada para correr este algoritmo. Los hiper-parámetros de la función dependen del kernel que se utilice; sin embargo, basado en las pruebas realizadas durante la etapa de calibración, el único kernel con resultados aceptables es el denominado *"rbf"*. El kernel *Radial Basis Function (rbf)* permite mapear los datos en altas dimensiones (infinitas) y medir su similitud. Entonces solamente el índice de contaminación: un, y el coeficiente del kernel: gamma, fueron los hiper-parámetros a calibrar.

Al igual que IF, la calibración se ejecutó por medio del algoritmo *"Grid_Search"* manual con validación cruzada estratificada (*StratifiedKFold*). Los resultados para el modelo OC-SVM con los mejores hiper-parámetros se muestran en la siguiente tabla:

| | | | |
|---|---|---|---|
| **One Class SVM** | *kernel = rbf*<br>*gamma = scale*<br>*un = 0.001767* |  |  |

### 4.3 Comparación del Desempeño de los Modelos:

Al comparar el rendimiento de los dos modelos, se observa que el **OC-SVM** mostró un rendimiento sólido en la detección de fraudes ($12,757.36), alcanzando un *ROC_AUC* entre 0.8919 y 0.934. Sin embargo, este modelo enfrentó un desafío significativo relacionado con los falsos positivos, ya que generó un número considerable de clasificaciones incorrectas de transacciones legítimas como fraudulentas. Esto se tradujo en un total de $577,396.79 en falsos positivos, lo que indica que, a pesar de su alta precisión en la identificación de transacciones legítimas, la capacidad de detectar efectivamente fraudes se vio comprometida. Esta situación puede generar inconvenientes financieros y de reputación para las instituciones, ya que las advertencias erróneas pueden llevar a la anulación innecesaria de transacciones legítimas y afectar la experiencia del cliente.

Por otro lado, el modelo **IF** logró una mejor métrica *ROC_AUC,* entre 0.8919 y 0.957, pero también tuvo dificultades en el conjunto de prueba, aunque con una tasa de detección de fraudes notablemente alta, resulta en una significativa cantidad de falso positivos, aún más que el modelo **OC-SVM**. En el análisis financiero, se destacó que, a pesar de identificar una cantidad considerable de fraude real ($14,828.48) el modelo también generó más de $2 millones en falsos positivos.

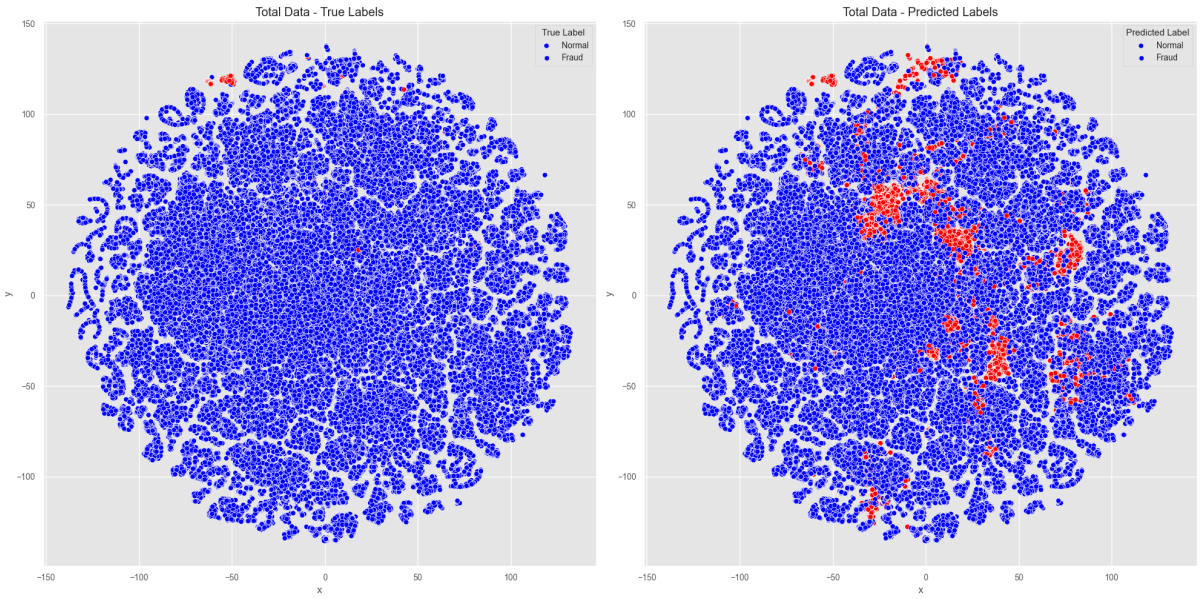
Ambos modelos presentan un costo considerable por falsos positivos, lo que subraya la necesidad de ajustes en los umbrales de decisión y parámetros para mejorar la precisión de las predicciones y reducir el impacto financiero de las clasificaciones incorrectas. **Esta evaluación destaca la importancia de considerar el análisis financiero al elegir y ajustar modelos de detección de fraudes.**

### 4.4 Visualización de las Predicciones

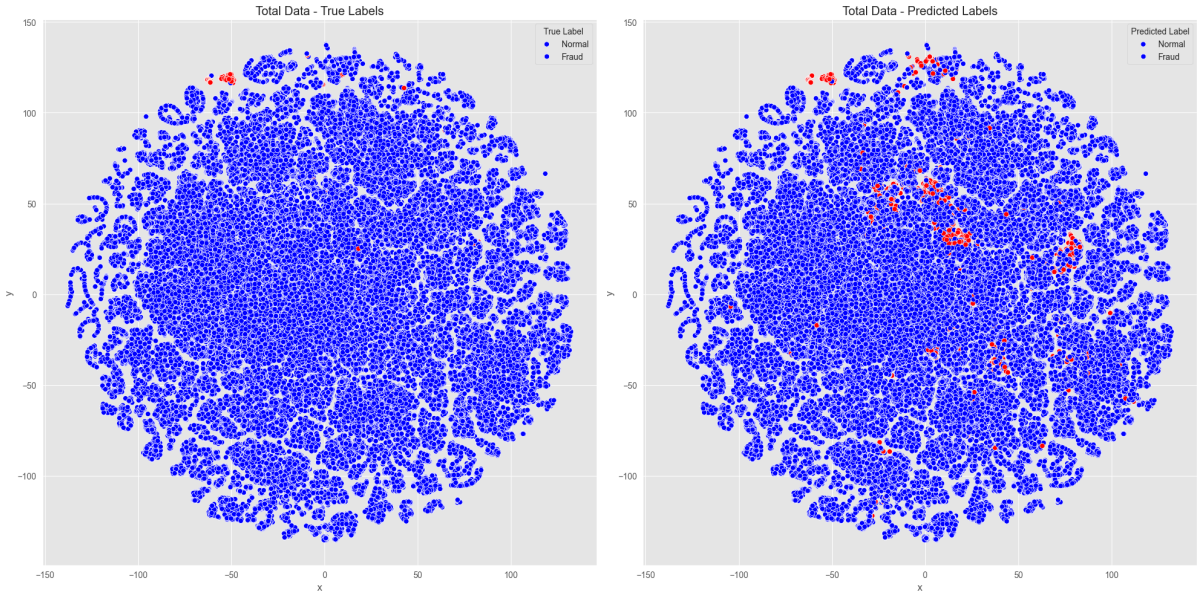
Mediante la técnica para disminución de la dimensionalidad t-SNE, del inglés *t-Distributed Stochastic Neighbor Embedding,* se descomponen las matrices para visualizar los resultados en 2-D aplicando cada uno de los modelos a la totalidad del *dataset*.

Se comparan el *dataset* original y las predicciones hechas por cada uno de los modelos calibrados. La visualización ratifica la efectividad en la detección de anomalías de los modelos y, además, la tasa elevada de Falsos Positivos, en especial, para el modelo de **Isolation Forest**.

## Isolation Forest




## One Class – SVM




# 5 CONCLUSION

La base de datos significativamente desbalanceada con transacciones de tarjetas de crédito fue usada para el entrenamiento de dos modelos basados en algoritmos de aprendizaje no supervisado: Isolation Forest y One Class-SVM. Los modelos tienen la ventaja de ser computacionalmente eficiente en bases de datos con una gran cantidad de dimensiones y con un significativo desbalance.

Aunque los modelos son altamente efectivos en la detección de anomalías; el reto está en optimizarlos para disminuir la significante tasa de falsos positivos, ya sea mediante el pre-procesamiento de los datos, e.g., haciendo Feature Engineering, mediante la calibración más ajustada de los umbrales de decisión y sus parámetros, o mediante su combinación.

Finalmente, la evaluación debe destacar la importancia del análisis financiero al momento de elegir y ajustar modelos de detección de fraudes.

## ANEXO 1. BIBLIOGRAFIA

La siguiente bibliografía fue consultada para la preparación del proyecto final:


- Alshameri, F., & Xia, R. (September 2024). An evaluation of variational autoencoder in credit card anomaly detection. Big Data Mining and Analytics, 7(99), 1-12.

- ObaidAli, Ola Imran, & Al-Sultan, Yakoob. (July 2024). Survey of fraud detection techniques for credit card transactions. Journal of University of Babylon for Pure and Applied Sciences.

- Smairi, Nadia & Abadlia, Houda. (March 2024). Enhanced particle swarm optimization-based hyperparameter optimized stacked autoencoder for credit card fraud detection. International Journal of Data Science and Analytics.

- Kowsalya, K., Vasumathi, M., & Selvakani, S. (March 2024). Credit card fraud detection using machine learning algorithms. EPRA International Journal of Multidisciplinary Research (IJMR).

- Bhakta, S. S., Ghosh, S., & Sadhukhan, B. (December 2023). Credit card fraud detection using machine learning: A comparative study of ensemble learning algorithms. In Proceedings of the 9th International Conference on Smart Computing and Communications (ICSCC), Kochi, Kerala, India.

- Lavanya, K. (April 24, 2024). Isolation Forest for Unsupervised Anomaly Detection. Techdevathe. https://medium.com/techdevathe/isolation-forest-for-unsupervised-anomaly-detection-4d4594e13451

- SPX. (March 25, 2024). Anomaly Detection with Isolation Forest. https://medium.com/@SPX701/anomaly-detection-with-isolation-forests-367e28e6a74e

- Castagno, P. (February 1, 2024). Isolation Forest Concept and Pseudocode. https://patriziacastagnod.medium.com/isolation-forest-if-70e4b6860fde

- Peters, M. (October 11, 2023). Understanding Support Vector Machine (SVM) and One-Class SVM. https://www.geeksforgeeks.org/understanding-one-class-support-vector-machines/

- Pierobon, G. (August 8, 2023). One-Class SVM (Support Vector Machine) for Anomaly Detection. https://medium.com/@gabrielpierobon/one-class-svm-support-vector-machine-for-anomaly-detection-a2e00c742ad7

- Kuo, C., & Dataman, Dr. (Ocotber 9, 2022). Handbook of Anomaly Detection (6) – One-Class SVM. https://medium.com/dataman-in-ai/handbook-of-anomaly-detection-with-python-outlier-detection-6-ocsvm-f746dae9f450

- Rahul, S. (May 4, 2023). PCA vs t-SNE (Dimensionality Reduction Techniques). https://ogre51.medium.com/pca-vs-t-sne-dimensionality-reduction-techniques-fdd7908973a4#:~:text=However%2C%20PCA%20assumes%20that%20the,for%20visualizing%20high%2Ddimensional%20data.

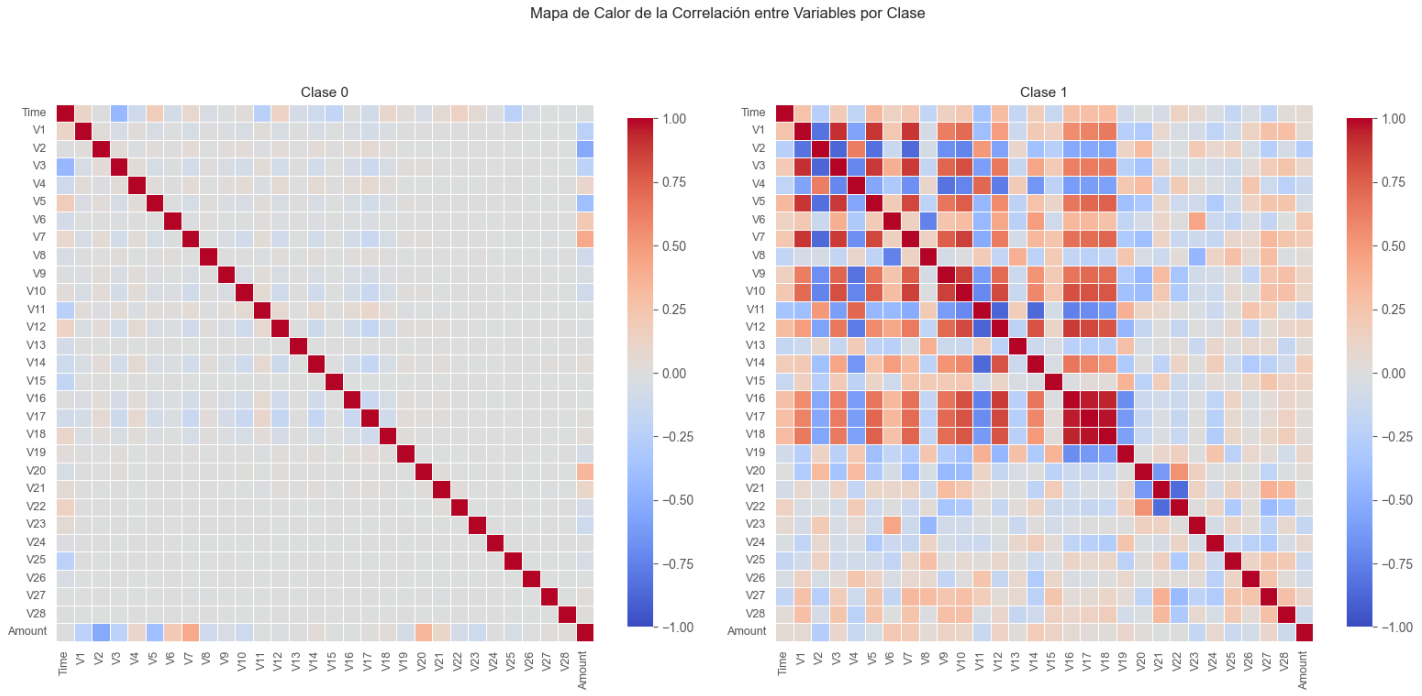
- Tregre, A. (January 12, 2012). Cyber Security: Machine Learning and Big Data Know It Wasn't You Who Just Swiped Your Credit Card. Tregre, A. (January 12, 2012). Cyber Security: Machine Learning and Big Data Know It Wasn't You Who Just Swiped Your Credit Card.

- Moser, R. (March 29, 2019). Fraud Detection with Cost-Sensitive Machine Learning. https://towardsdatascience.com/fraud-detection-with-cost-sensitive-machine-learning-24b8760d35d9

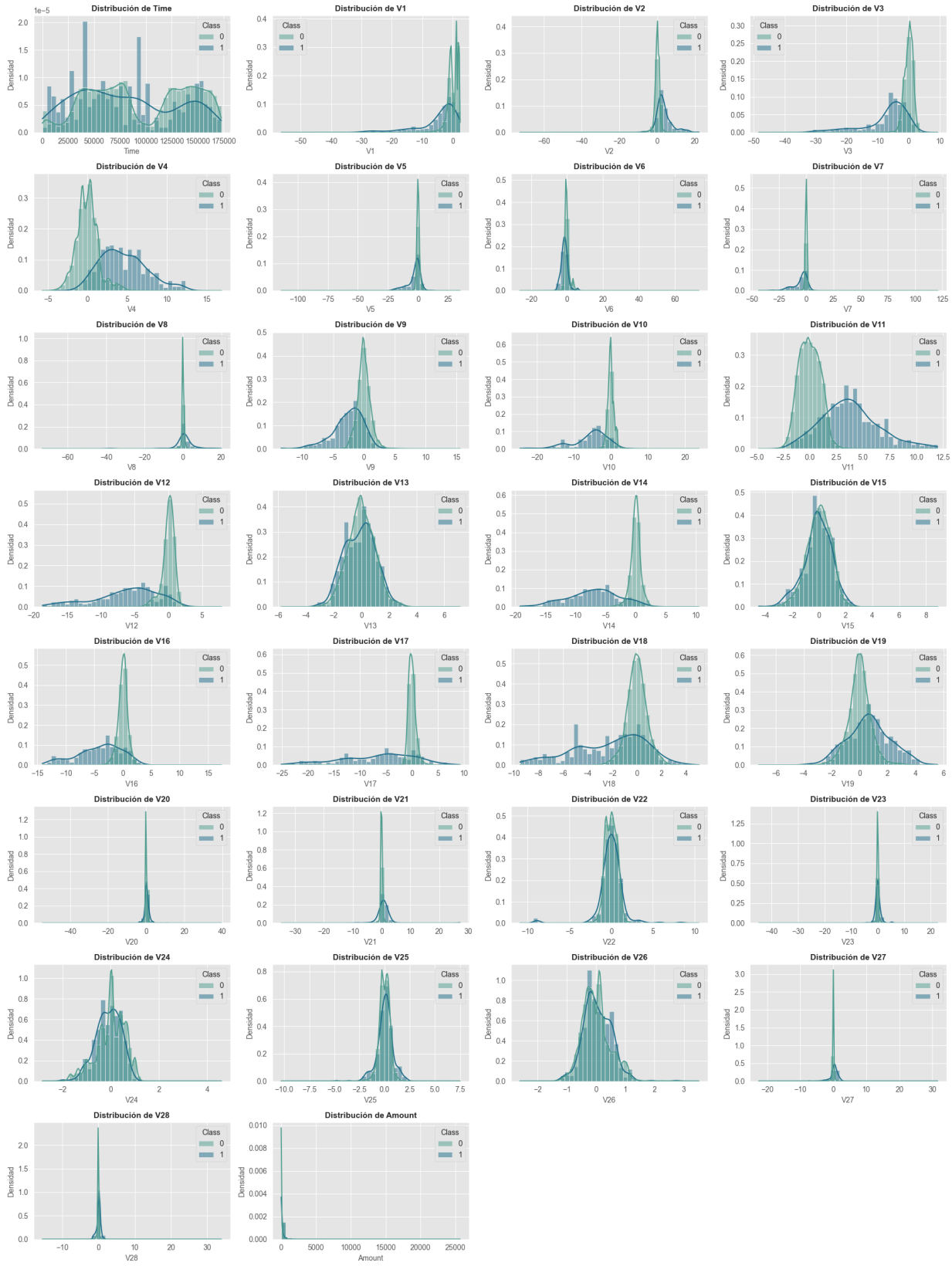
- Kuo, C., & Dataman, Dr. (August 10, 2018). Feature Engineering for Credit Card Fraud Detection. Dataman in AI. https://medium.com/dataman-in-ai/how-to-create-good-features-in-fraud-detection-de6562f249ef

- AltexSoft Co. (January 12, 2021). Credit Card Fraud Detection: How Machine Learning Can Protect Your Business from Scams. https://www.altexsoft.com/blog/credit-card-fraud-detection/

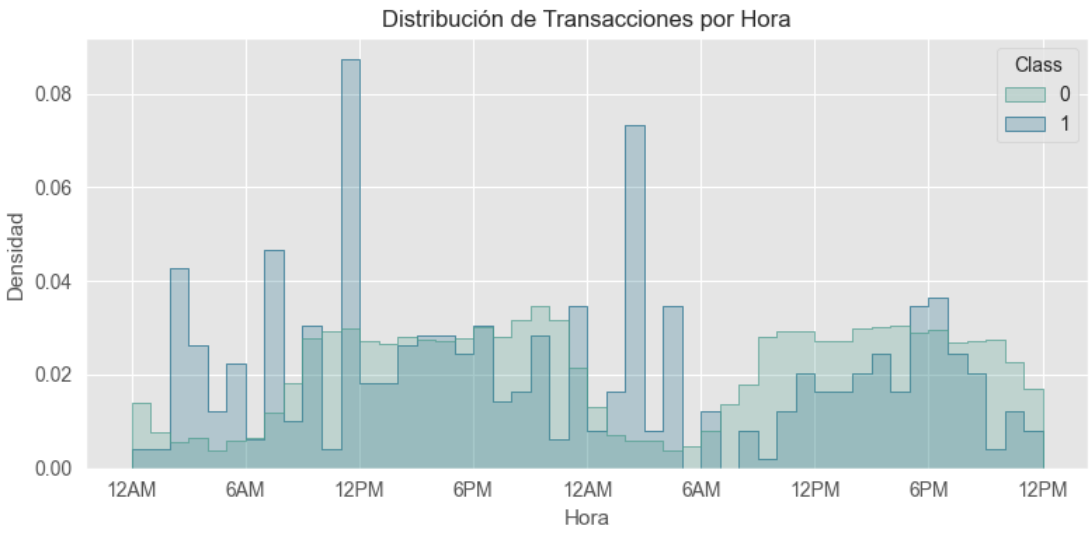
# ANEXO 2. CORRELACIÓN DE VARIABLES POR CLASE. 0: LEGÍTIMAS - 1: FRAUDULENTAS



# ANEXO 3. DENSIDAD DE LA DISTRIBUCIÓN DE LAS VARIABLES



## ANEXO 4. DISTRIBUCIÓNDE LAS TRANSACCIONES POR HORA POR CLASE



## ANEXO 5. IMPORTANCIA DE LA VARIABLES OBTENIDA POR ALGORITMOS DE ENSAMBLAJE

| Ítem | Var_rf | Random Forest | Var_xgb | XGBoost | Variable | Average Importance |
|---|---|---|---|---|---|---|
| 1 | V14 | 0.202918 | V14 | 0.575139 | V14 | 0.389029 |
| 2 | V10 | 0.138412 | V4 | 0.066500 | V4 | 0.077666 |
| 3 | V12 | 0.097837 | V12 | 0.036910 | V10 | 0.076825 |
| 4 | V4 | 0.088832 | V8 | 0.034249 | V12 | 0.067374 |
| 5 | V11 | 0.077571 | V20 | 0.024950 | V11 | 0.043243 |
| 6 | V17 | 0.063415 | V6 | 0.016907 | V17 | 0.037924 |
| 7 | V3 | 0.045047 | V27 | 0.015979 | V3 | 0.026306 |
| 8 | V16 | 0.036603 | V10 | 0.015238 | V8 | 0.023235 |
| 9 | V7 | 0.030287 | V7 | 0.014869 | V7 | 0.022578 |
| 10 | V9 | 0.022753 | V18 | 0.014184 | V16 | 0.020906 |
| 11 | V2 | 0.021949 | V23 | 0.013723 | V2 | 0.017805 |
| 12 | V21 | 0.018003 | V2 | 0.013662 | V20 | 0.017564 |
| 13 | V18 | 0.017352 | V26 | 0.013463 | V18 | 0.015768 |
| 14 | V19 | 0.012542 | V28 | 0.013371 | V9 | 0.015149 |
| 15 | V8 | 0.012221 | V17 | 0.012433 | V27 | 0.012402 |
| 16 | Amount | 0.011717 | V13 | 0.011212 | V6 | 0.012360 |
| 17 | V20 | 0.010178 | V19 | 0.010306 | V19 | 0.011424 |
| 18 | V27 | 0.008825 | Amount | 0.010166 | V21 | 0.010967 |
| 19 | V13 | 0.008489 | Time | 0.009741 | Amount | 0.010942 |
| 20 | V26 | 0.008227 | V5 | 0.008993 | V26 | 0.010845 |
| 21 | V6 | 0.007813 | V11 | 0.008915 | V13 | 0.009850 |
| 22 | V5 | 0.007595 | V3 | 0.007564 | V23 | 0.009813 |
| 23 | V1 | 0.007400 | V9 | 0.007544 | V28 | 0.009189 |
| 24 | V15 | 0.006524 | V1 | 0.007521 | V5 | 0.008294 |
| 25 | V22 | 0.006190 | V15 | 0.007463 | V1 | 0.007460 |
| 26 | V25 | 0.006079 | V22 | 0.006582 | Time | 0.007365 |
| 27 | V23 | 0.005903 | V16 | 0.005209 | V15 | 0.006993 |
| 28 | V28 | 0.005007 | Datetime | 0.005147 | V22 | 0.006386 |
| 29 | Time | 0.004989 | V24 | 0.004165 | V25 | 0.005021 |
| 30 | V24 | 0.004791 | V25 | 0.003962 | Datetime | 0.004839 |
| 31 | Datetime | 0.004531 | V21 | 0.003931 | V24 | 0.004478 |