Ensamblado *ad hoc* de clasificadores para la detección de cáncer de mama usando Scikit-learn

Andrés Maciel Cardozo, Gustavo Sosa-Cabrera, María E. García-Díaz

Facultad Politécnica
Universidad Nacional de Asunción
Asunción, Paraguay
amaciel@est.pol.una.py, {gdsosa, mgarcia}@pol.una.py
http://www.pol.una.py

Resumen En el presente, el cáncer de mama es uno de los cánceres más frecuentes y es la segunda causa de muerte en mujeres en todo el mundo. Asimismo, cada vez es más difícil ignorar el constante e intenso aumento de la importancia de los enfoques de minería de datos en los diagnósticos médicos. En este sentido, central a toda la disciplina de la minería de datos, encontramos la clasificación como la tarea preponderante en el proceso de toma de decisiones para los médicos. A día de hoy, una gran cantidad de clasificadores se han propuesto en la literatura. Sin embargo, teniendo en cuenta la cantidad de personas afectadas por el cáncer, merece la pena seguir desarrollando técnicas que puedan contribuir en mejores formas de diagnóstico. En lo que a este estudio concierne, se ha considerado introducir un *ensamblado de clasificadores*, propuesto en razón al balance entre sus factores individuales de *sesgo* y *varianza*. Demostrando la correctitud de la metodología adoptada para la conjunción *ad hoc* de los clasificadores, los resultados empíricos de este estudio proporcionan evidencia de una mejor clasificación de los tumores como maligno o benigno, en cuanto a precisión se refiere.

Palabras Claves: Cáncer de mama · Clasificación de tumores · Ensamblado de clasificadores · Sesgo · Varianza · Bagging · Boosting · Scikit-learn

1. Introducción

Durante el último siglo, el cáncer ha sido una de las principales causas de muerte donde el cáncer de mama es uno de los más frecuentes; y de hecho, es la segunda causante de muerte en mujeres en todo el mundo [13].

Por otra parte, entre las numerosas aplicaciones de la minería de datos y el aprendizaje automático, la tarea de clasificación es un área que cobra cada vez más importancia en diagnósticos médicos asistidos. En la tarea de clasificación cada instancia pertenece a una clase la cual se indica mediante el valor discreto de un atributo que llamamos la clase de la instancia.

Hasta la fecha, una considerable cantidad de propuestas de nuevos clasificadores, variantes de los ya existentes y *ensamblado* de los mismos ha sido publicada en la literatura científica.

En este sentido, se desarrolla un creciente interés por el ensamblado de clasificadores por ser considerado como el siguiente nivel de tendencia en la búsqueda de mejoras en la tarea de clasificación [9]. Básicamente, este enfoque consiste en la construcción de un clasificador más robusto y preciso, esto es, mediante la combinación de las predicciones de varios clasificadores atendiendo las fortalezas y debilidades que cada modelo algorítmico individualmente posee.

Entre tanto, el cáncer de mama es un tumor maligno que amenaza gravemente la salud de la mujer a nivel mundial y constituye una prioridad debido a su elevada incidencia[1]. Adicionalmente y como es bien conocido, es posible obtener mejores resultados en el contexto de los diagnósticos médicos apoyados por el análisis de los datos, ya que se equilibra el conocimiento de los expertos humanos al describir problemas y objetivos con las capacidades de clasificación de los computadores [12].

Surge, por tanto, la constante necesidad de nuevas investigaciones a los estudios previos [1,16] y que contribuyan para la toma de decisiones de los médicos en el diagnóstico de cáncer de mama mediante la aplicación de técnicas de minería de datos; siendo éste, en efecto, el propósito del presente estudio.

Los resultados de esta investigación demuestran que el proceso de *ensamblado de clasificadores* propuesto en razón al balance entre el *sesgo* y la *varianza* de cada clasificador, permite la construcción de un mecanismo mejorado de clasificación de tumores maligno o benigno, en cuanto a precisión se refiere.

De manera estructural, este trabajo primeramente sienta los fundamentos teóricos respecto a los clasificadores y los factores principales, sesgo y varianza, así como las técnicas de ensamblado. Posteriormente, en la sección de Materiales y Métodos, se presentan los conceptos de *Bagging* y *Boosting* seguido de los algoritmos de clasificación que se contemplan en este estudio. En la sección de experimentos se evalúan los planteamientos iniciales para finalmente converger las ideas en la conclusión.

2. Fundamentos Teóricos

Clasificador. Desde el punto de vista matemático, en [11], [4] se precisa que un problema de clasificación puede verse como una colección L de N ejemplos etiquetados que pueden definirse de la siguiente manera:

$$L = \{(x_n y_n), n = 1, 2, ..., N, y \in [y^1, y^2, ..., y^y]\} \quad (1)$$

donde cada ejemplo (x_n, y_n) viene determinado por un vector de atributos x_n y una etiqueta de clase y_n pertenecientes a algunas de las Y clases del problema $[y^1, y^2, ..., y^y]$. El algoritmo de clasificación resultante deberá de construir un clasificador que prediga, dado un nuevo vector de características x (no incluido

[1] https://seer.cancer.gov/statfacts/html/breast.html

en L) la clase Y a la que pertenece usando el conocimiento contenido en el conjunto inicial de datos L. Formalmente, un clasificador es un modelo o función M que predice la etiqueta de clase y para un ejemplo de entrada dado x, es decir, $\mathcal{Y} = M(x)$, donde $x = (x_1, x_2, ..., x_d)^{\mathcal{T}} \in \mathbb{R}^d$ es un punto en el espacio d-dimensional e $y =\in \{c_1, c_2, ..., c_k\}$ es su clase predicha.

Sesgo y Varianza. En [20] definen al sesgo y a la varianza como la pérdida esperada para la función de pérdida al cuadrado. Ofrece información importante sobre el problema de clasificación, ya que puede descomponerse en términos de sesgo y varianza. Ambas son herramientas poderosas de las estadísticas de la teoría de muestreo para analizar escenarios de aprendizaje supervisado que tienen funciones de pérdida cuadrática [2]. Dado un objetivo fijo y el tamaño del conjunto de entrenamiento, $sesgo^2$ mide hasta qué punto la estimación promedio del algoritmo de aprendizaje (sobre todos los conjuntos de entrenamiento posibles del tamaño del conjunto de entrenamiento dado) coincide con el objetivo. En tanto que la $varianza$, mide cuánto fluctúa la suposición del algoritmo de aprendizaje para los diferentes conjuntos de entrenamiento del tamaño dado. Matemáticamente Sesgo (Ses) y varianza (Var) se definen:

$$Ses_x^2 = \frac{1}{2}\sum_{y \in Y}[P(Y_f = y|x) - P(Y_H = y|x)]^2 \quad Var_x = \frac{1}{2}(1 - \sum_{y \in Y} P(Y_H = y|x)^2).$$

En general, la pérdida esperada puede atribuirse a un *alto sesgo* o a una *alta varianza*, que suele ser una compensación entre estos dos términos. Idealmente, se busca un equilibrio entre estas tendencias opuestas, es decir, se prefiere un clasificador con un *sesgo aceptable* (reflejando suposiciones específicas de dominio o conjunto de datos) y una variación tan baja como sea posible.

2.1. Técnicas de Ensamblado

En [20], se menciona que un clasificador se llama inestable si las pequeñas perturbaciones en el conjunto de entrenamiento resultan en grandes cambios en la predicción o el límite de decisión. Los clasificadores de alta varianza son inherentemente inestables, ya que tienden a sobre ajustar los datos. Por otro lado, los métodos de alto sesgo suelen adecuarse a los datos y, por lo general, tienen una varianza baja. Los métodos de conjunto crean un clasificador combinado utilizando la salida de múltiples clasificadores básicos que se entrenan en diferentes conjuntos de datos. Dependiendo de cómo se seleccionen los conjuntos de entrenamiento y de la estabilidad de los clasificadores básicos, los clasificadores de conjunto pueden ayudar a estabilizar la varianza y el sesgo, lo que lleva a un mejor rendimiento general del modelo.

3. Materiales y Métodos

Dadas las pruebas realizadas en el presente artículo, se describe a continuación tanto los algoritmos de clasificación seleccionados, como así también el

Bagging	Boosting
Genera múltiples conjuntos de entrenamiento y llama al algoritmo de aprendizaje modelo base con cada uno de ellos para producir una conjunción de modelos base [10]. $\mathbb{M}^{\mathbb{K}}(x) = sign(\sum_{i=1}^{K} M_i(x))$	En [2] se describe como una técnica que genera un conjunto de clasificadores y los vota. El algoritmo de aprendizaje genera los clasificadores en paralelo. $G(X) = sign(\sum_{m=1}^{M} a_m G_m(X))$
Intenta disminuir la varianza ya que cada modelo está construido de forma independiente.	Intenta disminuir el sesgo ya que los nuevos modelos están influenciados por el rendimiento de los modelos construidos previamente

Cuadro 1. Cuadro comparativo entre Bagging y Boosting.

entorno experimental utilizado, en el Cuadro 1 se puede observar la comparación entre *Bagging* y *Boosting*.

3.1. Algoritmos de Clasificación

Random Forest(RF). Cada árbol de decisión que se crea se hace a partir de una secuencia aleatoria de un subconjunto de datos del conjunto de entrenamiento [9].

Extra Trees Forest(ETF). Construye un conjunto de árboles de regresión o decisión sin afinar. Sus dos diferencias principales con otros métodos de conjunto basados en árboles son que divide nodos al elegir puntos de corte completamente al azar y que utiliza toda la muestra de aprendizaje [6] [17].

K-Neighbors(KNN). Encuentra un grupo de k objetos en el conjunto de entrenamiento que están más cerca del objeto de prueba, y basa la asignación de una etiqueta en el predominio de una clase en particular en este conjunto [19].

Linear Support Vector Machine(SVM). Técnica utilizada para el aprendizaje supervisado para resolver problemas de clasificación y regresión basados en hiperplanos [8].

Decision Tree(DT). Estructura en la cual los datos son divididos de acuerdo a un criterio. Cada nodo del árbol denota un test sobre un atributo. Cada rama representa un resultado del test, y las hojas del nodo representan clases [15] [3].

Redes Neuronales(RN). Las entradas $\mathbf{x_i}$ se recopilan de las neuronas anteriores (o el conjunto de datos) y se combinan mediante una función de combinación como $\sum$, que luego se ingresa en una función de activación (generalmente no lineal) para producir una respuesta de salida $\mathbf{y}$, que luego se canaliza otras neuronas. [4]:

Gaussian Naive Bayes(NB). Las Redes Bayesianas representan las dependencias que existen entre los atributos a través de una distribución de probabilidad condicional en un grafo dirigido acíclico [15].

AdaBoost(AB). Se encarga de analizar un número de algoritmos débiles, que serán árboles de decisión pequeños de profundidad 1, que tienen poca precisión

a través del conjunto de entrenamiento modificado en cada iteración. Secuencialmente formará uno nuevo más preciso con la combinación de todos ellos asignando un peso en la votación final para la elección de las mejores reglas de cada uno [2].

Logistic Regression(LR). Procedimiento cuantitativo donde la variable dependiente toma valores en un conjunto finito [9].

Gradient Boosting(GB). El modelo de conjunto construido es también una suma ponderada de clasificadores débiles. La conexión se realiza entre expansiones aditivas por etapas y minimización de descenso más pronunciado [5].

4. Experimentos y Análisis de Resultados

Para realizar los experimentos, se ha optado por el uso del *Scikit-learn*, ya que es un módulo de Python[2] que integra una amplia gama de algoritmos de aprendizaje automático de última generación para problemas supervisados y no supervisados de mediana escala [14], a su vez se opto por el uso del dataset de cáncer de mama de Wisconsin, ya integrado en el *Scikit-learn* para la realización de las pruebas [18]. Siguiendo con el proceso experimental, se seleccionan los siguientes algoritmos para entender las ventajas con respecto al *sesgo* y la *varianza* que se puede extraer. Al observar el Cuadro 2, las características con respecto al tipo de algoritmo, los modelos según su tipo, se hace el análisis sobre los mismos de manera de buscar el resultado adecuado. Se tiene en cuenta la precisión del modelo,debido a que el dataset utilizado se encuentra con una distribución de clases balanceada, lo que permite obtener un resultado aceptable para la interpretación final de los datos. La implementación de estas pruebas están disponibles en un repositorio público[3].

Alg.	Var.	Sesgo	Definición
RF	Alta	Bajo	$C_{rf}^{B}(x) = v\{C_b(x)_1^B\}$
ETF	Alta	Bajo	$f_t(x) = \sum_{x^i, y^i \in ls} a_i k_t(x^i, x)$
KNN	Alta	Bajo	$c_z = arg_{v \in L} max \sum_{y \in N} w_i x I(v = class(c_y))$
SVM	Alta	Bajo	$K(x, x_i) = sum(x * x_i)$
DT	Alta	Bajo	$I(P) = -(p_1 * log_2(p_1) + p_2 * log_2(p_2) + ... + p_n * log_2(p_n)$
RN	Alta	Bajo	$y_i = g_1(\sum_{j=1}^{L} w_{ij} s_j) = g_1(\sum_{j=1}^{L} w_{ij}(g_2(\sum_{r=1}^{N} t_{jr} x_r)))$
NB	Baja	Alto	$V_{nb} = argmax_v jcVP(V_j) \prod_i P(a_i \mid v_j)$
AB	Baja	Alto	$C * (x) = arg_{y \in Y} max \sum_{i:C_i(x)=y} log \frac{1}{\beta_i}$
LR	Baja	Alto	$logit(p(y = 1 \mid x)) = w0x0 + w1x1 + ... + wmxm = \sum_{i=0}^{m} wixi = w^t x$
GB	Baja	Alto	$F_m = (x) = F_{m-1}(x_i) + p_m h(x_i; \mathbf{a_m})$

Cuadro 2. Algoritmos ordenados por *varianza* y *sesgo*.

[2] https://www.python.org
[3] https://github.com/andresmacielc/Stacking

Lo primero es definir las variables a ser utilizadas y los objetivos a ser encontrados al realizar el análisis de aprendizaje supervisado, para que el paciente sepa si posee o no cáncer de mama. Una vez finalizado, se avanza al siguiente paso, el cual implicará normalizar los datos [7]; con esto se busca solución a diferentes ruidos, que se presentan en el *dataset*; luego se procede a dividir el *dataset* en conjunto de entrenamiento y testeo. Se utiliza el paquete *model selection()*, instalado por defecto en la biblioteca *scikit-learn*. Con la validación cruzada se hacen las pruebas a la precisión de la clasificación de algoritmos en las versiones *bagging* de cada uno, de modo de retratar las diferencias que presentan los resultados finales de dicha estimación.

```
clf_array = [rf, et, knn, svc, dt, nb, mlp]
for clf in clf_array:
    scores = cross_val_score(clf, X_train_scaled, y_train, cv=3,
    n_jobs=-1)
    bagging_clf = BaggingClassifier(clf,
       max_samples=0.8, max_features=1.0, random_state=seed)
    bagging_scores = cross_val_score(bagging_clf, X, y, cv=3,
       n_jobs=-1)
    print("Media de: {1:.3f}, std: (+/-) {2:.3f}
    [Bagging {0}]\n".format
    (clf.__class__.__name__,
    bagging_scores.mean(),bagging_scores.std()))
```

El código muestra los resultados en las ejecuciones estudiadas, con la aclaración que los algoritmos *RF*, *ETF* y *KNN* poseen un balance de 100 de estimaciones en los 2 primeros y 100 *neighbors* en el tercero. Los resultados en la figura 1.

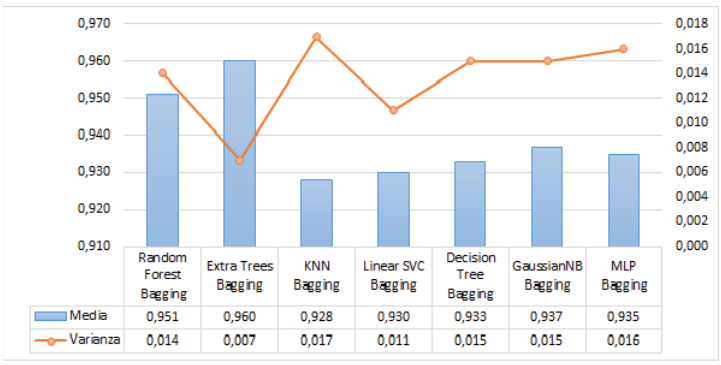

Figura 1. Comparación de precisión en la técnica del *bagging*.

Las versiones del *bagging* presentan mayor varianza, debido a que algunos de los algoritmos fueron ejecutados con baja cantidad de estimadores. En determinados casos el *bagging* es efectivo con datos limitados. El enfoque del *bagging*

consiste en usar un par de sub-muestras y agruparlas, si la precisión del conjunto es mayor que la de los modelos básicos, funciona. Nótese que el uso de submuestras más grandes no garantiza un mejor resultado. El *bagging* puede mejorar el conjunto en gran medida cuando tiene un modelo inestable; sin embargo, cuando sus modelos son más estables, entrenados en submuestras mayores, las mejoras de *bagging* se reducen.

Para el caso del *boosting*, se analizaa el funcionamiento de los algoritmos *AB* y *GB*, en los cuales no se tuvieron en cuenta los algoritmos utilizados en el *bagging*. Luego se realizan las comparaciones mediante la implementación del siguiente *script*:

```
boost_array = [ada_boost, grad_boost]
eclf = VotingClassifier(estimators=[('Ada Boost', ada_boost),
('GradientBoost', grad_boost)], voting='hard')
labels = ['Ada Boost', 'Grad Boost', 'Ensemble']
for clf, label in zip([ada_boost, grad_boost, eclf],
labels):
    scores = cross_val_score(clf, X_train_scaled, y_train, cv=10,
    scoring='accuracy')
    print("Media: {0:.3f}, std: (+/-) {1:.3f}
    [{2}]".format(scores.mean(), scores.std(), label))
```

En la Figura 2, se muestra los resultados comparativos obtenidos.

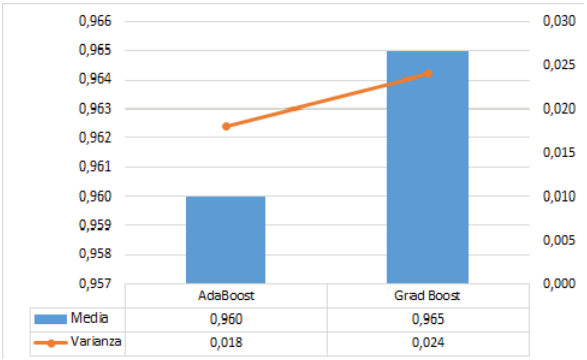

Figura 2. Comparación de precisión en la técnica del boosting.

En la Figura 2, las diferencias de resultados presentan una baja variación entre ellos, se observa a su vez que el *sesgo* es más preciso en el algoritmo *GB*. ya que al basarse en el descenso de gradiente, realiza búsquedas más precisas en el conjunto de datos entrenados erróneamente.

Realizadas las clasificaciones correspondientes, se procede a la combinación de los algoritmos que son implementados en el *ensamblado*, los cuales, basados

en las características que presentan los modelos descritos anteriormente, tanto en el *bagging* como en el *boosting*, permiten encontrar la mejor combinación de los métodos, aplicados a los clasificadores base y a los específicos, de manera que el resultante obtenga estabilidad en cuanto al *sesgo* y a la *varianza*. Para el ensamblado se utilizaron todos los algoritmos básicos implementados previamente, sumados a estos los algoritmos RF y ETF, los cuales ya implementan el método de *bagging* por defecto, y los algoritmos AB y GB que realizan lo propio desde el lado del *boosting*, de modo a que el ensamblado final realice el aprendizaje tanto de los clasificadores simples, del *bagging* y del *boosting*.

Para dicho propósito se utiliza la librería *StackingClassifier*, en donde para la implementación del *ensamblado*, es necesaria la selección de un clasificador final que pueda funcionar como base de los resultados propuestos por los anteriores, dado el hecho que la mayoría de los algoritmos utilizados previamente se caracterizan por poseer una *varianza* más alta, seleccionar un algoritmo de *sesgo* alto funciona como un nivelador de las variaciones que presentan los clasificadores, entonces se opta por el algoritmo de LR, resultando de la siguiente forma:

```
sclf = StackingClassifier(classifiers=
[rf, et, knn, svc, mlp, nb, dt, adab, gd], meta_classifier=lr)

classifier_array = [rf, et, knn, svc, mlp, nb, dt, adab, gd,
sclf]
labels = [clf.__class__.__name__ for clf in classifier_array]

for clf, label in zip(classifier_array, labels):
    cv_scores = cross_val_score(clf, X, y, cv=10,
    scoring='accuracy')
    print("Media: %0.4f (+/- %0.4f) [%s]" % (cv_scores.mean(),
    cv_scores.std(), label))
```

En la Figura 3 se muestran los resultados comparativos obtenidos.

Figura 3. Comparación de precisión de Algoritmos Individuales vs Ensamblado.

Puede observarse en la Figura 3 que la combinación de los algoritmos posee el mejor *sesgo* del conjunto, y una *varianza* estable respecto a los demás.

A diferencia de los ejemplos anteriores, esta vez se buscó una mayor estabilidad en cuanto a la precisión de los algoritmos se refiere, como se detalló en [20], el aumento de la *varianza* de un algoritmo implica indefectiblemente en su reducción en el *sesgo*, y viceversa, esta situación de por sí es algo inevitable, sin embargo, para casos particulares de algunos algoritmos, como serian el KNN y SVM o el RF, en los cuales aumentando la cantidad de *neighbors*, la cantidad de estimadores o de parámetros, puede obtener un aumento en el *sesgo*, y al mismo tiempo una reducción de la *varianza*.

Dicho de otra manera, en la figura 3 se puede observar la prueba de dicha afirmación al comparar las estimaciones de los algoritmos KNN y DT. En el caso del primero se definieron los *neighbors* lo que permitió el aumento del *sesgo*, por otra parte, en el segundo no se definieron mayores cantidad de parámetros, por lo que la cantidad de *varianza* se mantiene alta, a su vez, el *sesgo* permanece bajo. A través del balance obtenido con la combinación de los algoritmos se llega a un *multiclasificador* con estabilidad relativa en términos de *sesgo* y *varianza* y con una exactitud en la precisión mejorada tomando las características de su combinación.

5. Conclusión

En este estudio, se ha contribuido a evidenciar los resultados de la aplicación de un *ensamblado de clasificadores* propuesto en razón al balance entre el *sesgo* y la *varianza* de cada clasificador para la detección de cáncer de mama mediante el uso de herramientas de software libre. Tomados en conjunto, los resultados obtenidos proporcionan apoyo a la premisa que el ensamblado de los clasificadores en bruto, no garantiza una mejora automática en cuanto a la precisión de la clasificación, ya que al momento de realizar un apilado de algoritmos, deberían considerarse los factores identificados y referidos en esta investigación.

Finalmente, basándose en la presencia de un aumento en la precisión obtenida para la clasificación de tumores maligno o benigno, es posible subrayar la necesidad de seguir trabajando para incrementar el entendimiento sobre la conjunción de clasificadores y su generalización en otros dominios de aplicación.

Referencias

1. Agarap, A. F. M. On breast cancer detection: an application of machine learning algorithms on the wisconsin diagnostic dataset. In *Proceedings of the 2nd International Conference on Machine Learning and Soft Computing* (2018), ACM, pp. 5–9.
2. Bauer E., K. R. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning* (2018).
3. Blanco F., Ortiz A., R. G. M. R. C. Y. Clasificadores y multiclasificadores com cambio de concepto basados en arboles de decision. *Inteligencia Artificial. Revista Iberoamericana de Inteligencia Artificial* (2010).

4. D., L. *DISCOVERING KNOWLEDGE IN DATA An Introduction to Data Mining*. John Wiley and Sons, Inc., 2005.
5. G., F. Greedy function approximation:a gradient boosting machine. *The Annals of Statistics* (2001).
6. Geurts P., Ernst D., W. L. Extremely randomized trees. *Machine Learning* (2006).
7. Han J., K. M. *Data Mining: Concepts and Techniques, Second Edition*. Morgan Kaufmann Publishers, 2006.
8. Huincalef R., Urrutia G., I. G. M. D. Recognition of surface irregularities on roads: a machine learning approach on 3d models. *Congreso Argentino de Ciencias de la computacion* (2018).
9. J., Z. *Comparativa y Analisis de Algoritmos de Aprendizaje Automatico para la Prediccion del Tipo Predominante de Cubierta Arborea*. Benemérita Universidad Autónoma de Puebla, 2017.
10. Kotsiantis S. B., P. P. E. Combining bagging and boosting. *World Academy of Science, Engineering and Technology International Journal of Mathematical and Computational Sciences* (2007).
11. Martínez-Muñoz G., S. A. Comites de arboles igp.
12. Olarte, E., Panizzi, M. D., and Bertone, R. A. Segmentación de mercado usando técnicas de minería de datos en redes sociales. In *XXIV Congreso Argentino de Ciencias de la Computación (La Plata, 2018)*. (2018).
13. O'Sullivan, C. C., Loprinzi, C. L., and Haddad, T. C. Updates in the evaluation and management of breast cancer. In *Mayo Clinic Proceedings* (2018), vol. 93, Elsevier, pp. 794–807.
14. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. Scikit-learn: Machine learning in python. *Journal of machine learning research 12*, Oct (2011), 2825–2830.
15. Portugal R., C. M. Ensamble de algoritmos bayesianos con arboles de decision, una alternativa de clasificacion.
16. Salama, G. I., Abdelhalim, M., and Zeid, M. A.-e. Breast cancer diagnosis on three different datasets using multi-classifiers. *Breast Cancer (WDBC) 32*, 569 (2012), 2.
17. Wehenkel L., Ernst D., G. P. Ensembles of extremely randomized trees and some generic applications. *RTE-VT workshop* (2006).
18. Wolberg, W. H., Street, W. N., and Mangasarian, O. L. Breast cancer wisconsin (diagnostic) data set. *UCI Machine Learning Repository [http://archive.ics. uci. edu/ml/]* (1992).
19. Wu X., K. V. *The top ten algorithms in Data Mining*. Chapman and Hall, 2009.
20. Zaki M., M. W. *Data Mining and Analysis: Fundamental Concepts an Algorithms*. Cambridge University Press New York, NY, USA ©2014, 2013.