Universidad Jorge Tadeo Lozano
Facultad de Ciencias Naturales e Ingeniería

Trabajo de grado presentando como requisito para optar al título de Magister en Ingeniería y Analítica de Datos

Modelo de análisis de datos utilizando técnicas de aprendizaje supervisado y no supervisado, para identificar patrones en la información generada por los pacientes, sometidos a juegos diseñados como un instrumento de apoyo terapéutico

German Ruiz Manosalva

Profesor Asesor

PhD. Olmer García

Bogotá D.C – Colombia, junio 30 de 2019

Dedicatoria

A mi familia, esposa e hijos quienes son las personas que siempre me acompañaron en esta aventura y a mis hermanos y sobrinos

Resumen

En nuestros días gran cantidad de información es generada, esto es causado por los significativos avances de la computación que han permitido utilizar la tecnología en cada una de las actividades desarrolladas por el hombre, en este caso se puede notar en el campo de la medicina, precisamente en la parte sicológica. La combinación entre la tecnología y los tratamientos aplicados en los pacientes que sufren trastornos cognitivos leves, esto a través de juegos desarrollados especialmente para la estimulación de la atención, donde jugar creará gran cantidad de información que será almacenada.

Para este estudio se seleccionó una muestra de datos, a los cuales se le aplico la metodología de minería de datos CRISP-DM, donde se hizo uso de las teorías y técnicas de aprendizaje automatizado, como son los algoritmos de aprendizaje no supervisado y supervisado. Las etapas utilizadas fueron el procesamiento de los datos, la adecuación para el análisis, creación de unas variables y aplicación de los algoritmos de agrupamiento, clasificación y regresión para poder identificar en los datos los patrones, tendencia y características de los juego y jugadores. A partir de lo anterior se generaron gráficas que permiten entender de una forma más fácil la información y los resultados obtenidos.

Con la información que está contenida en los datos, se pudo establecer que, al realizar la aplicación de los modelos de aprendizaje automático, se encontraron patrones que permitieron demostrar que el aprendizaje se realizó de forma intuitiva y que cada vez que se jugaba, las posibilidades de hacer más aciertos aumentaban. Adicionalmente se pudo establecer que los juegos que fueron diseñados para cada uno de los tratamientos tienen características en común las cuales permitieron realizar agrupaciones con los datos existentes.

Palabras Clave: inteligencia artificial, aprendizaje automático, aprendizaje profundo, analítica de datos, algoritmos, identificación de patrones y tendencias, técnicas de análisis de datos, visualización de datos, bases de datos, metodologías, estadística, probabilidades, juegos serios, tratamientos pacientes.

Abstract

Now a day's we can talk about the high currency of information that is generate from the constant development of technology, it has allowed apply technology in every kind of activity that humans do from the above in this work we are going to considerate the implication of try technology in medicine through the match between computers and treatment that receive people who suffering cognitive disorders. In this case technology and the treatment are combine by games which are create for the stimulation of attention; while these games are played the information will be create and save.

For this study a sample of data was treated under data mining methodology CRISP-DM, where we use the theories and techniques of automated learning like unsupervised and supervised learning algorithms. the stages used was data processing, adequacy for analysis, creation of variables and application of grouping algorithms, classification and regression to be able to identify patterns, trend and characteristics about the games and the players. In addition, graphics were created for easy understanding of information and results.

Then with the information contained in the data, we can say that after to apply the models of automated learning, we found patterns that allows say that the learning was reached intuitively, also we found that the likelihood of success will increase while the game is play. So, these games which were designed for each treatment have things in common and it will be group into function of the data generated.

Keywords: Artificial Intelligence, machine learning, deep learning, data analytics, Algorithms, Identification of patterns and trends, techniques of data analysis, data visualization, databases, statistics, probabilities, serious games, patient treatments.

Tabla de Contenido

Lista de tablas

Lista de figuras

1. Presentación del trabajo

1.1 Introducción

Los procesos cognoscitivos en los niños se forman a medida que ellos crecen y se adaptan a la sociedad, donde aprenden habilidades para la supervivencia, el razonamiento, la cognición y la adaptación social [1]. Estas habilidades se agrupan en las Funciones Ejecutivas (FE) que según [2] las define como "las capacidades mentales necesarias para la formulación de objetivos y la planificación de estrategias idóneas para alcanzar dichos objetivos, optimizando el rendimiento", las funciones ejecutivas más importantes son: control atencional, memoria de trabajo, inhibición, flexibilidad cognitiva, fluidez verbal y de diseño, programación motora y planificación de la cotidianidad.

En la propuesta de trabajo se hace mayor énfasis en la planificación, ya que nos posibilita el estudio de las capacidades de los pacientes para entender un problema, elegir una meta, realizar la toma de decisiones para la formulación de un plan y realizar su auto aprendizaje. Para realizar la estimulación de las funciones ejecutivas se utilizan diferentes procedimientos, entre ellos, la estimulación sensorial que se realiza mediante el uso de programas de computación para el entrenamiento cognitivo.

Las universidades Católica de Colombia, Jorge Tadeo Lozano, El Bosque y El Hospital Militar se unieron para llevar a cabo un proyecto de investigación, teniendo como base la utilización de juegos de planificación y atención creados por computador, para pacientes con deterioro cognitivo leve y en niños con Trastorno por Déficit de Atención e Hiperactividad (TDAH), dichos juegos deben ser jugados por los pacientes como ayuda terapéutica utilizando un computador, el cual recopila, almacena y procesa toda la información que se produce de la relación paciente juego, y con la información recolectada realizar un análisis de datos aplicando técnicas de aprendizaje automático.

La primera fase del proyecto se realizó entre los años 2013 y 2014, con pruebas del módulo de juegos de planificación y fueron realizadas con la colaboración de 111 niños, y la asesoría de investigadores del programa de Doctorado en Psicología de la Universidad de la Laguna de España. Los programas computarizados demostraron ser efectivos en habilitar la función ejecutiva de la planificación en los niños participantes en proyecto. La segunda fase en los años 2015 y 2016 diseñó el módulo correspondiente al control atencional en el que se incluyeron ejercicios para atención sostenida, selectiva, dividida y alternante, aplicados a otro grupo de niños con déficit atencional.

En la actualidad se ejecuta la tercera fase dividida en tres partes, dirigida a pacientes con deterioro cognitivo leve que asisten a terapia neuropsicológica en el Hospital Militar Central de Bogotá. Uno de los objetivos de esta fase es extraer, organizar y transformar la información, y posteriormente construir un modelo de aprendizaje automático, de la información recolectada en la aplicación de estos juegos, que permita ser un instrumento de apoyo terapéutico para ayudar a la rehabilitación y realizar ajustes a los juegos aportando a un mejor rendimiento y usabilidad por los pacientes, con los resultados de los patrones arrojados en el modelo.

En el desarrollo de este trabajo se presenta como propuesta de analítica de datos la pregunta de investigación referente si: ¿es posible identificar patrones sobre los datos obtenidos de utilizar los videojuegos en tratamientos médicos utilizando técnicas de *Machine Learning*, mediante la implementación de la metodología de análisis de datos (*Cross Industry Standard Process for Data Mining* - CRISP–DM). A continuación, es presentado, primero un estado del arte en el uso y análisis de "videojuegos serios", seguido de un marco teórico que describe que es aprendizaje automático y como se pueden evaluar los resultados de los modelos. Posteriormente es presentado el desarrollo de la metodología donde se van presentando los resultados de las preguntas de investigación y los objetivos que se describen a continuación.

1.2. Modalidad

El presente trabajo se realiza bajo la modalidad de profundización.

1.3. Preguntas de investigación

- ¿Cómo podemos determinar si un juego es aprendido de forma intuitiva por los pacientes a partir de los datos?
- ¿Cómo poder estimar si los juegos generados para cada tratamiento de pacientes tienen características similares?
- ¿Existen patrones en el uso de los videojuegos que permitan identificar el tipo de función ejecutiva que se está tratando de estimular en el paciente?

1.4. Resultados obtenidos

En el presente trabajo se obtuvieron los siguientes resultados

- Se logró establecer por medio del análisis de los datos que los juegos presentan una forma fácil de aprender, es decir, son intuitivos.
- Se pudo comprobar que, si se utilizan los juegos de forma continua, aumentan las posibilidades de mejorar el número de aciertos en menos tiempo.

- Se obtuvo un modelo de agrupamiento que permitió establecer que los juegos diseñados para cada uno de los tratamientos tienen características en común.
- Se encontró que existe poca información para los juegos de atención dividida un solo juego y que esta información no es determinante.

1.5. Limitaciones y problemas vigentes

Para el desarrollo del presente trabajo no se cuenta con limitaciones que interrumpan el normal desarrollo del proyecto.

- Se cuenta con la asesoría de un docente de la maestría especialista en proyectos de analítica de datos.
- Acceso a una muestra de datos con calidad de información optima sin ninguna restricción.
- El software utilizado es Python como lenguaje de código abierto y de licenciamiento libre.

1.6. Objetivos

1.6.1. Objetivo general

Proponer un modelo de análisis de datos utilizando técnicas de aprendizaje supervisado y no supervisado, para identificar patrones en la información generada por los pacientes, sometidos a juegos diseñados como un instrumento de apoyo terapéutico.

1.6.2. Objetivos específicos

- Obtener un modelo utilizando las técnicas de aprendizaje supervisado que permita realizar la identificación de patrones en la información que se producen los juegos.
- Identificación de los datos en cuanto a volumen, variedad, velocidad, veracidad y el valor que aporta a la investigación, los cuales son generados como la respuesta de los juegos utilizados por los pacientes con trastornos.
- Obtener un modelo utilizando las técnicas de aprendizaje no supervisado que permita realizar la identificación de patrones en la información que se producen los juegos.

1.7. Aportes y contribuciones

El presente trabajo se enfocó en la aplicación de los algoritmos de aprendizaje automático, a la información que se tiene en archivos planos de las pruebas de realizados por niños a los juegos de atención. Con la aplicación de estos algoritmos se pudo establecer de forma cuantitativa la usabilidad de los juegos. Se identificaron las siguientes características en los juegos:

- En el nombre del archivo se identifica el jugador, nombre del juego, el intento y la fecha en la cual se jugó el juego.
- En cada uno de los archivos tienen una información resumen del juego, total Aciertos y tiempo total.
- Se identifico la relación que existe entre el número de intentos (número de registros del archivo) y el total de aciertos.
- Se encontró que el juego '*compañeros*', no se cumplió la relación existente entre número de intentos y el total de aciertos, ya que siempre el total de aciertos fue superior a la cantidad de intentos.
- Se encontró que los juegos se aprenden de forma intuitiva y se recuerdan después de un tiempo en el cual no se juegan y la calidad en los aciertos no disminuye ni se tiene que recapacitar a los usuarios.
- Finalmente, con el análisis de los datos también podemos identificar que tanto puede tardar un usuario en completar un cierto número de aciertos.

1.8. Mapa del documento

El presente documento se encuentra estructurado de la siguiente forma:

- **Presentación del trabajo:** contiene una descripción inicial del trabajo y el tema a trabajar en el presente documento.
- **Estado del arte:** realiza el estudio de la literatura científica, escrita hasta este momento sobre el tema de nuestra investigación.
- **Marco teórico:** describe de forma general los métodos de IA que serán utilizados para el desarrollo del trabajo.
- **Descripción de la investigación:** expone la información general del problema y la cantidad de información con la que se cuenta y sus características que permitan entender el problema y su posible solución.
- **Desarrollo de la metodología:** aplicar la metodología CRISP-DM diseñada para proyectos de minería de datos ya que se adapta a proyectos de esta naturaleza.

2. Marco teórico

3.1. Inteligencia artificial

La inteligencia artificial (*Artificial Intelligence, o AI*) es la simulación de procesos de inteligencia humana por parte de máquinas, especialmente en sistemas informáticos. Estos procesos incluyen el aprendizaje (la adquisición de información y reglas para el uso de la información), el razonamiento (usando las reglas para llegar a conclusiones aproximadas o definitivas) y la autocorrección. Hoy en día, es un término general que abarca todo, desde la automatización de procesos robóticos hasta la robótica actual.

Ha ganado prominencia recientemente debido, en parte, a los grandes volúmenes de datos, o al aumento de velocidad, tamaño y variedad de datos que las empresas están recopilando. AI puede realizar tareas tales como identificar patrones en los datos de manera más eficiente que los seres humanos, lo que permite a las empresas obtener más información sobre sus datos.

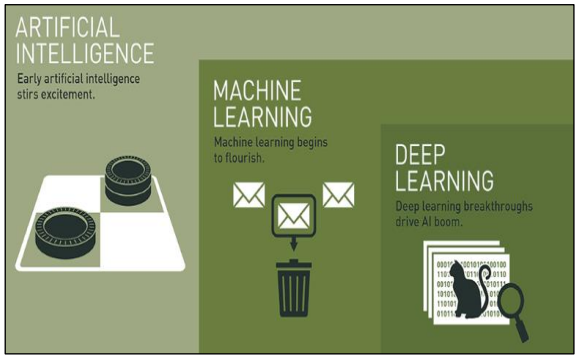

FIGURA 1. ARTIFICIAL INTELLIGENCE, MACHINE LEARNING Y DEEP LEARNING [12]

En la

Figura 1 se observa la relación existente entre *Artificial intelligence, Machine learning and Deep learning.*

3.2. Aprendizaje automático (Machine learning)

El aprendizaje automático (AA) o *machine learning* se identifica como una rama de la *AI* la cual se creó inicialmente para que pudieran responder preguntas como: [12] "¿puede un computador ir más allá de lo que le ordenamos como hacer y aprender por sí mismo como realizar una tarea específica? , ¿podría un computador automáticamente aprender reglas directamente de los datos que le pasamos?". Con estas preguntas se inicia la creación de una nueva forma de realizar la

programación, cambiando la forma clásica donde se le entrega al computador las reglas y los datos para obtener respuestas, con la AA se le entrega al computador datos y las respuestas que se obtiene para que el identifique patrones y genere sus propias reglas que serán aplicadas a los nuevos datos para obtener las respuestas esperadas. En la Figura 2,se puede observar las dos formas de programar un computador y las respuestas que se obtienen.

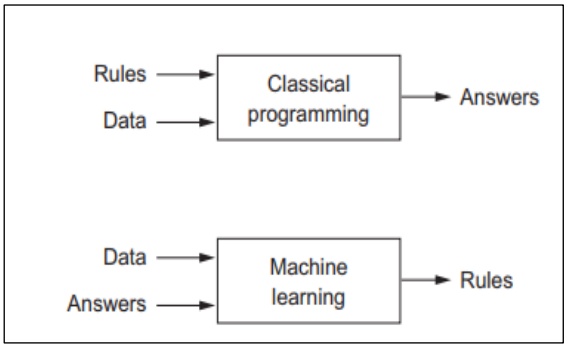

FIGURA 2. APRENDIZAJE AUTOMÁTICO: EL NUEVO PARADIGMA DE PROGRAMACIÓN [12]

Una de las características principales de un sistema de Aprendizaje Automático es que debe ser "entrenado" y no como los sistemas tradicionales que debe ser completamente "programado". La forma como se hace para entrenar un sistema es presentándole muchos datos que influyan en la realización de una tarea y que este logre la identificación de las estructuras estadísticas, matemáticas o los patrones buscando en la información entregada, esto le permitirá aprender las reglas para realizar la automatización de dicho proceso utilizando algoritmos. Existen varias formas de realizar el aprendizaje: supervisado, no supervisado y por refuerzo.

3.2.1. Aprendizaje supervisado

Los algoritmos de aprendizaje supervisado están basados en un modelo predictivo, el cual está compuesto por dos grupos de datos uno para realizar el entrenamiento, otro de prueba y un mecanismo que permita evaluar si el algoritmo está haciendo las cosas bien. El conjunto de datos previamente etiquetado y clasificado, del cual ya se sabe a qué grupo, valor o categoría pertenecen son los datos de entrenamiento y son utilizados para realiza el ajuste al modelo, los datos son utilizados por el algoritmo para ir "aprendiendo" a clasificar las muestras realizando una comparación del resultado obtenido por el modelo, y el valor inicial de la muestra, realizando compensaciones con respecto al modelo y a cada error en la estimación del resultado.

Un ejemplo, de este tipo de aprendizajes son los vehículos autónomos. Y las redes neuronales artificiales (*Artificial neural networks*) son uno de los algoritmos que se utilizan. El AA tiene algunas similitudes con la minería de datos en lo que se refiere a la búsqueda de información en grandes volúmenes de información, la minería de datos realiza la exploración de datos utilizando

la estadística para buscar y extraer patrones que sean analizados por el hombre y la AA utiliza los patrones y reglas identificadas en los datos para ser analizadas por sus algoritmos.

La descripción grafica de cómo funciona un algoritmo de aprendizaje supervisado lo podemos ver en la Figura 3.

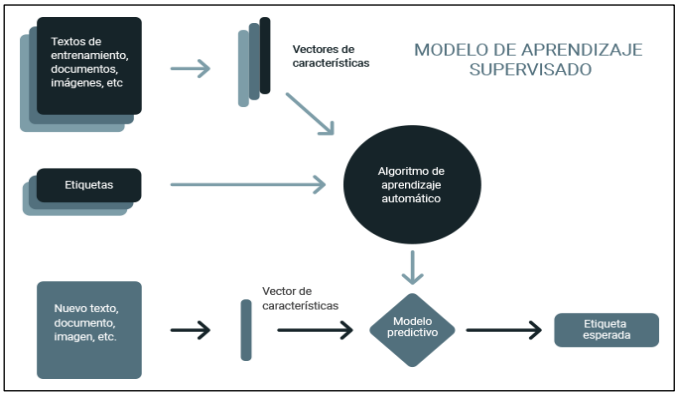

FIGURA 3. MODELO DE APRENDIZAJE SUPERVISADO [13]

3.2.2. Aprendizaje no supervisado

A diferencia del modelo anterior este solo tiene en cuenta para realizar sus ajustes al modelo predictivo, los datos de entrada, sin importar si están o no clasificados o etiquetados, estas características no son necesarias para realizar el entrenamiento del modelo un ejemplo de este modelo es la clasificación de imágenes y el algoritmo utilizado es K-medias (*K-means*). En el aprendizaje no supervisado se puede ver que las etiquitas de salida no son entregadas a los algoritmos, sino que tienen que ser identificadas como se muestra en la Figura 4.

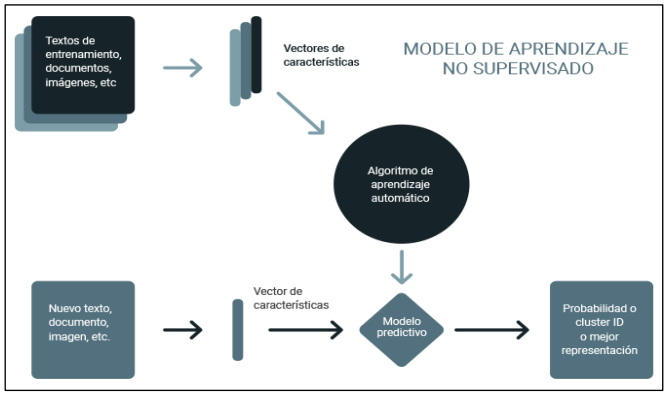

FIGURA 4. MODELOS DE APRENDIZAJE NO SUPERVISADO [13]

3.2.3. Aprendizaje por refuerzo (*Reinforcement learning*)

Esta clase de algoritmos de aprendizaje por refuerzo, están basado en maximizar una medida de "recompensas" la cual es el resultado de unas "acciones" teniendo en cuenta el ambiente y entorno

donde se desempeñan o desarrollan. Estos algoritmos están basados en una reacción de acción-recompensa, la cual busca que el algoritmo tenga como meta la mejor "recompensa" la cual es dada por el ambiente, teniendo siempre como su principal premisa que sus acciones están enfocadas a la recompensa. Este método es usado para que los robots aprendan a realizar tareas. Para el apréndase por refuerzo se puede identificar en la Figura 5, donde se entregan datos crudos y el sistema los clasifica y entrega las respuestas.

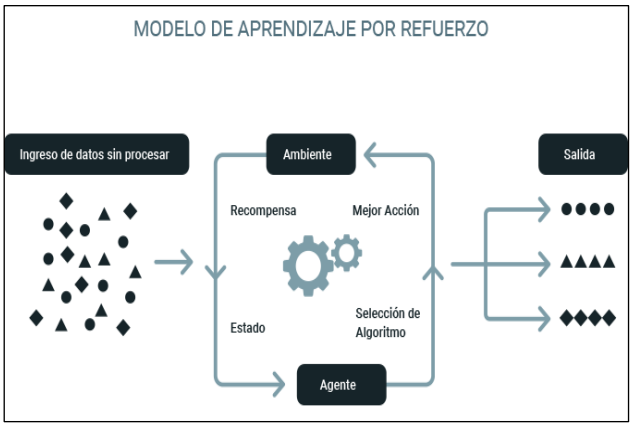

FIGURA 5 MODELO DE APRENDIZAJE POR REFUERZO [13]

3.3. Algoritmos de aprendizaje automático

Existe una gran cantidad de algoritmos desarrollados para realizar trabajos de aprendizaje automáticos entre los que se encuentra los siguientes:

- **Agrupamiento por K-medias** es un algoritmo de agrupamiento, segmentación o clasificación no supervisado, el cual realiza la clasificación de las observaciones en grupos teniendo como base sus características y las distancias entre cada una de las observaciones, los algoritmos que se utilizaran son de agrupamiento o de clúster. El agrupamiento consiste en realizar una minimización de la suma de distancias entre cada uno de los objetos y el centroide de su grupo (clúster).

  Se deben realizar los siguientes pasos: Inicialización: con el número de grupos, k, se establecen k centroides en el espacio de los datos. Asignación objetos a los centroides: se asigna el objeto a su centroide más cercano. Actualización centroides: la posición del centroide se ajusta tomando la posición del promedio de los objetos de cada grupo. Un ejemplo se muestra en la Figura 6 donde se pueden ver tres grupos que se diferencia por color y una estrella que identifica el centro de los datos.

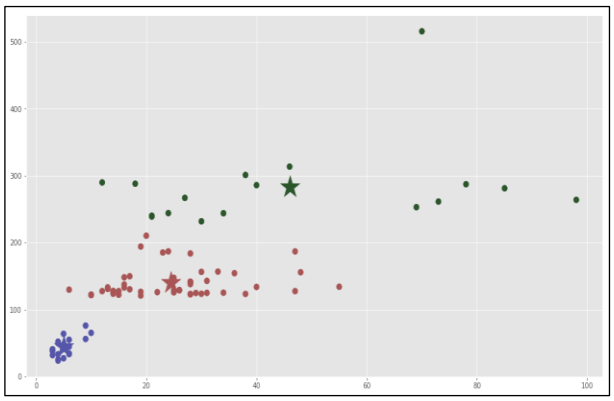

FIGURA 6. DESCRIPCIÓN DE MODELO DE AGRUPACIÓN GRÁFICA PROPIA

- **Regresión lineal**. Estos algoritmos esta basados en un modelo matemático, que consiste en explicar la relación que existe entre la variable dependiente (respuesta Y) y el conjunto de variables independientes que conforman el universo de datos de estudio (explicativas X1…Xn). Un ejemplo con una sola variable independiente se puede ver en la Figura 7 donde se muestran los puntos de las observaciones y la recta que se genera por la regresión lineal.

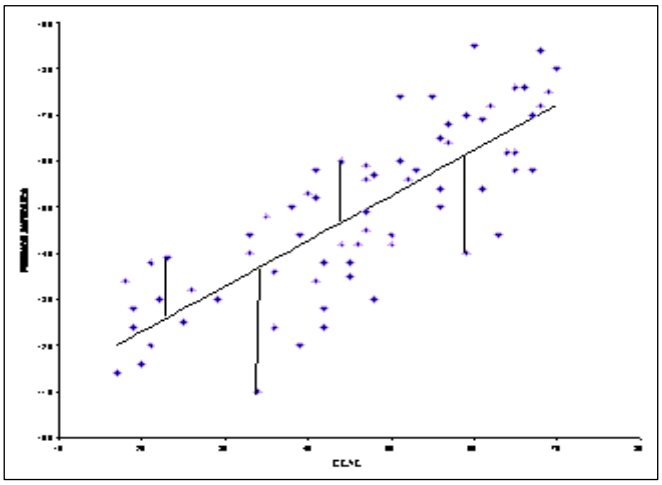

FIGURA 7. ILUSTRACIÓN DE UNA REGRESIÓN LINEAL [14]

- **Redes neuronales artificiales (RNA)**: estos algoritmos están dirigidos a encontrar patrones en los datos para realizar tareas como las de diagnosticar, clasificar, identificar o realizar la predicción que debe ser aprendida en tiempo de ejecución. Están inspiradas en el comportamiento de las neuronas y como se organizan la estructura del cerebro, mediante un modelo matemático utilizando las estadísticas y las probabilidades. A continuación, realizamos la descripción de una neurona llamada perceptrón múltiple. La cual consta de las

siguientes partes: capa de entrada, capa de salida y las capas ocultas. Las conexiones sinápticas (están representadas por las flechas que llegan y salen de las neuronas) están flechas indican el flujo de la señal a través de la red, y tienen asociadas un peso sináptico correspondiente. Un ejemplo donde se muestra la estructura física como funciona una red neuronal se muestra en la Figura 8 que utiliza dos capas.

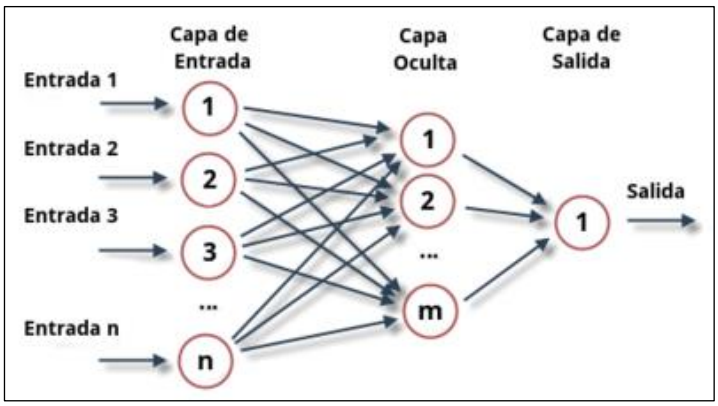

FIGURA 8. COMPONENTES DE UNA RED NEURONAL [15]

- **sklearn.naive_bayes.GaussianNB**: está formado por un algoritmo *Gaussian Naive Bayes* como clasificador esto hace que sea simple y rápido con buenos resultados con muy poca sintonía con los hiper parámetros del modelo y también puede trabajar con muy pocos datos.
- **Kneighborsclassifier** (vecino más cercano) este algoritmo de clasificación se basa en buscar en los datos los puntos más similares o cercanos, con el fin poder clasificar los nuevos puntos en una clase determinada, esto lo aprende en la etapa de entrenamiento. Este algoritmo también trabaja muy bien con pocos datos.
- **DecisionTreeClassifier** (Arboles de decisión) este algoritmo de clasificación está diseñado para trabajar con variables tipo binarias, por que maneja un conjunto de árboles de decisión. Es utiliza con muchas dimensiones y grandes volúmenes de información.
- **RandomForestClassifier** este algoritmo está basado en la utilización de un conjunto de árboles de poca profundidad que son entrenados de forma simultánea y que posteriormente se utiliza el aprendizaje de todos los árboles.
- **Support Vector Machines, SVMs** está conformada por un conjunto de algoritmos desarrollados por Vladimir Vapnik y fueron desarrollados para resolver problemas de clasificación y regresión con un conjunto de datos de entrenamiento para etiquetar las clases y construir un modelo para predecir las nuevas clases, mediante la representación de dos puntos con los datos de entrenamiento, separando ampliamente las dos clases con un hiperplano el cual se define como un vector entre los dos puntos.

3.4. Evaluación de algoritmos de AA

Una de las características más importantes en (*Machine Learning*) es poder realizar la evaluación del desempeño de los algoritmos de AA, para lo cual se utilizan diferentes técnicas, todo depende del campo de aplicación, la estructura y naturaleza de los datos a ser utilizados.

El siguiente autor Payam Refaeilzadeh [16], basa su estudio en la selección de atributos para el aprendizaje automático, su evaluación la realiza al comparar dos algoritmos y el efecto que tienen los atributos seleccionados sobre la exactitud de la clasificación, el desempeño de un algoritmo lo mide estadísticamente por su diferencia en la precisión.

Para otro autor como Rich Caruana [17]. Se realiza la comparación de diez algoritmos de aprendizaje supervisado, utilizando varios criterios de rendimiento. Además, utilizó varias métricas de rendimiento como: exactitud, F-score, Lift, ROC, precisión media, precisión de memoria del punto de equilibrio, error cuadrático y entropía cruzada.

Para nuestro caso en particular, realizaremos el estudio del método de validación cruzada [18] el cual se encarga de evaluar el desempeño del estimador, cuando se realiza un el entrenamiento y prueba de un modelos por lo general se utilizan datos los cuales son divididos en dos grupos uno de entrenamiento y otro de pruebas, pero al utilizar este método el algoritmo puede realizar un proceso de sobre entrenamiento, para resolver este problema utilizamos el método de validación cruzada, el enfoque básico es llamado k- fold CV, el cual consiste en seleccionar del total de los datos los datos de entrenamientos y estos a su vez se dividen en conjuntos más pequeños para el los cuales se sigue el siguiente procedimiento con los k "pliegues":

- Se realiza el entrenamiento de un modelo utilizando k-1 de los pliegues como datos de entrenamiento.
- Con el modelo resultante se realiza la validación con la parte restante de los datos (este resultado es utilizado como el conjunto de prueba, para el cálculo de una medida de rendimiento como es el caso de la precisión) esto se puede ver la Figura 9 .

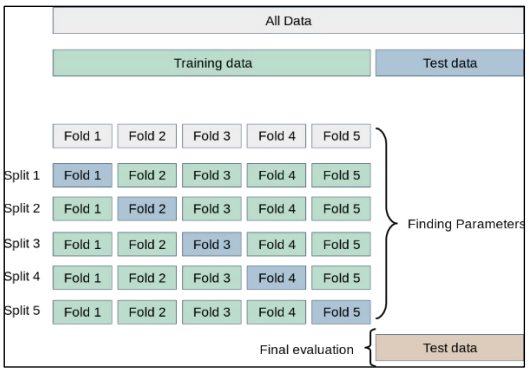

FIGURA 9. MODELO DE DIVISIÓN DE DATOS DE PRUEBAS EN PLIEGUES [18]

El cálculo de las métricas de validación cruzada se realiza dividiendo los datos de entrenamiento y calculando la puntuación obtenida según un número de repeticiones que se realizan los cálculos. La evaluación también se puede realizar para múltiples matrices el cual devuelve un conjunto de tiempos de ajustes, tiempos de puntuación y puntajes de pruebas. Se pueden obtener predictores por validación cruzada donde por cada elemento que es dado como entrada, devuelve la predicción que se obtiene cuando hacia parte de los datos de prueba, este método se puede utilizar para ejercicios de validación cruzada solo una vez.

Otra parte importante en la evaluación de los modelos [18] es poder realizar la evaluación de la calidad de las predicciones de los modelos que se desarrollaron.

- **Evaluación del rendimiento de la agrupación:** Para realizar la evaluación del rendimiento de un algoritmo de agrupamiento no se puede realizar contando el número de errores o la precisión y recuperación de un algoritmo supervisado. Las métricas que se utilizan para la evaluación no deben tener en cuenta los valores absolutos de las etiquetas con las cuales se están agrupando, sino que la agrupación se realiza de forma separada de los datos similares del conjunto de la clase que cumplan ciertos supuestos de tal forma que la agrupación se realice de los miembros más similares. A continuación, relacionamos algunas métricas como: Índice de Rand ajustado, Puntuaciones basadas en información mutua, Homogeneidad, integridad y medida V, Puntuaciones de Fowlkes-Mallows, Matriz de contingencia

- **Métricas de clasificación:** es otra forma que permite realizar la evaluación de los modelos, la cual utiliza la implementación de funciones de perdida, puntajes y utilidad para realizar la evaluación del rendimiento de la clasificación realizada. También se puede utilizar estimaciones de probabilidades de clases positivas, valores de confianza o de decisiones binarias. Estas implementaciones permiten que cada una de las muestras suministre una contribución ponderada a la puntuación general. Para nuestro trabajo en

particular se estudia el modelo de Matriz de Confusión esta función se encarga de evaluar la precisión de la clasificación al realizar el cálculo de la matriz de confusión con cada una de las filas correspondientes a la clase verdadera, en la Figura 10, se presenta una explicación grafica de la matriz de confusión donde se muestran las entradas i,j en una matriz representan el número de observaciones reales en el grupo i y que se predicen que se encuentran el grupo j.

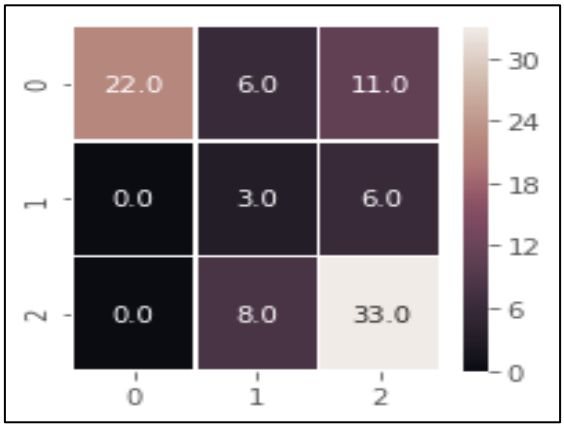

FIGURA 10. MATRIZ DE CONFUSIÓN [18] FIGURA PROPIA.

De la matriz de confusión se pueden identificar la cantidad de variables que fueron clasificadas como verdaderas y que son verdaderas, las que se clasifican como verdaderas pero que son falsas, las que se clasifican como falsa pero que son verdaderas y las que se clasifican como falsa y que son falsas. Con la evaluación de cada uno de los valores almacenados en cada casilla y los valores de la diagonal principal se puede determinar si el algoritmo está clasificando correctamente los datos en cada uno de los grupos.

- **Métricas de regresión (*Regression metrics)*:** las métricas que se utilizan en este trabajo se incluyen en la librería *sklearn.metrics* de Python, este módulo se basa en la implementación de algunas funciones de pérdida, puntaje y utilidad con la cuales realiza la medición del rendimiento y la evaluación del cálculo del error, que resulta de medir la distancia de la recta generada (o el hiperplano) a los puntos reales de la regresión, las funciones más utilizadas son:
    - Error Absoluto Medio: se representa como la media de la diferencia absoluta entre los puntos de los datos reales y los datos que son predichos.
    - Error cuadrático medio (MSE): esta métrica calcula la media de la diferencia entre los puntos reales de datos y los que fueron predichos, al cuadrado. Este método realiza una penalización de las mayores a las diferencias más altas.
    - Puntaje R^2 o el coeficiente de determinación: se identifica como la versión estandarizada del MSE, y es la que proporciona la mejor interpretación del

rendimiento del modelo. el R^2 muestra las varianzas de las respuestas que son registradas por el modelo.

3. Estado del arte

En la actualidad se identifican síndromes desde el punto de vista neuropsicológicos, teniendo como referencia la adaptación de los individuos a la vida cotidiana, en entornos sociales, familiares y profesionales de los pacientes, con deterioro cognitivo. Unos de los factores principales es la no utilización de fármacos en el tratamiento de estas enfermedades, con el objetivo de ayudar a la rehabilitación natural del paciente mediante el uso de juegos [3].

Los juegos que son utilizados como terapias, están incluyendo en su desarrollo los avance en la tecnología de la información y las Comunicación (TIC), para poder brindar a los pacientes una experiencia más real y la facilidad de poder almacenar y procesar la información. Se han creado juegos orientados a la rehabilitación con ayuda del computador, como son los siguientes: programas Smart Brain [4], [5] Rehacom, y plataformas como EuroNet (plataforma europea de tele-asistencia), Directosalud (plataforma de tele-estimulación cognitiva para dementes).

Los juegos anteriormente mencionados se desarrollaron para ser utilizados en tratamientos médicos en la Psicología, los juegos se utilizan como complemento a los tratamientos a pacientes con tratamiento psicoterapéutico, que ayuda a modificar el comportamiento cognitivo y en el desarrollo de tratamientos cognitivo-conductual para diversos trastornos. Sin embargo, el uso de juegos no se restringe solo en esta área, pues se están utilizando de diversas formas en la enseñanza y entrenamiento de las diferentes ramas de la medicina, ya que para la gran mayoría se deben utilizar cadáveres de personas y animales que no son tal fáciles de adquirir y conservar.

Beneficios a pacientes con demencia, favorece el costo económico en tratamientos médicos y facilita la rehabilitación neuropsicológica, apoyado con un seguimiento a la evolución del paciente, con el análisis a los datos recopilados en las rehabilitaciones de forma rápida, eficiente y precisa. Hoy en día la educación del profesional en salud se orienta por el uso pedagógico de cadáveres de humanos y animales para su examinación y diagnóstico de posibles enfermedades. Con la aplicación de las TIC [6], y en especial el uso de la inteligencia artificial, los sistemas de simulación y aprendizaje electrónico ofrecen nuevas formas de aprendizaje.

En este proceso, aparecen los llamados Juegos Serios, los cuales están diseñados para entretener a los jugadores mientras educan, entrenan o cambian el comportamiento [7], en esta rama del aprendizaje podemos identificar los juegos de simulación desarrollados para cada una de las especializaciones en la medicina como por ejemplo los procedimientos de alta precisión como son las cirugías, donde se pueden identificar juegos serios para procedimientos de ortopedia, ginecología y para los radiólogos, también se ofrecen juegos para el entrenamiento en las

imágenes de ultrasonido. En odontología, Los juegos se enfocan en el área de diagnóstico, para la toma de decisiones y el uso de protocolos para mejorar los resultados de los tratamientos y las terapias de los pacientes. A su vez en enfermería, los juegos desarrollados son complemento de la educación de las enfermeras y tienen que ver con las áreas que son difíciles de experimentar en la vida real como, manejo del dolor, laceraciones en la piel, transfusión de sangre y salud preventiva. En Cardiología existen varios juegos que permiten a los médicos mantenerse actualizados en sus capacidades, al igual que un juego que ayuda a los pacientes en la recuperación física después de un accidente cerebrovascular.

En Primeros auxilios, se utilizan juegos para la instrucción y enseñanza de los protocolos que conllevan al manejo de situaciones de emergencia mediante situaciones simuladas de eventos que ponen a prueba sus habilidades. En Dietista y Diabetes, los juegos desarrollados tienen que ver herramientas que permitan desarrollar habilidades como: identificar la terminología utilizada en el manejo de la enfermedad, la aplicación autónoma de inyecciones de insulina y la actividad física que puede realizar.

A nivel general, en la rama la salud la inclusión de juegos serios ayuda en el tratamiento de los pacientes en múltiples especialidades y en la capacitación de los profesionales de la salud teniendo gran importancia. Cada día se trabaja en la inclusión de nuevas herramientas de desarrollo y análisis de la información con la inteligencia artificial que contribuyan a la generación de nuevo conocimiento.

El autismo y los trastornos por falta de atención han tenido como centro de investigación el papel, el lápiz y las pruebas psicométricas computarizadas de funciones ejecutivas. Se plantean nuevas alternativas para que los niños interactúen en aulas virtuales con maestros humanos virtuales, ya que hoy en día la investigación en esta clase de ambientes está desarrollada y mide la evolución de los pacientes en habilidades sociales demostrando una evolución y un potencial en nuevas terapias [7]. La evaluación de los pacientes utilizando programas de computación, ofrece ventajas a las tradicionales de papel y lápiz, como la cantidad de tiempo utilizado en dar respuesta precisa, facilidad en la recopilación de datos y la administración de repeticiones de los ejercicios propuestos.

El uso de terapias alternativas como los juegos de computadora o videojuegos con es el caso de los juegos de Nintendo Wii, Xbox, Play Station, entre otros [8], aumentan la movilidad y control del cuerpo con respecto a la postura y su control mejorando la movilidad funcional en personas con trastornos neurológicos [9]. También identificaron otros beneficios con el uso de consolas de videojuegos, como el aumento en el cumplimiento de tareas en la fisioterapia de movimientos

para las manos, mientras desarrolla la resistencia, la fuerza y la coordinación, alcanzando una mayor socialización y aumento de su autoestima.

Existe evidencia que los *Serious Games*, puede ser utilizado exitosamente para entrenar habilidades físicas y cognitivas en personas mayores o con trastornos cognitivos, utilizados como complemento al tratamiento de trastornos de atención. [10]. Cuando se utilizan video juegos en tratamientos, los pacientes tienen la lógica para la resolución de problemas, con la función visoespacial que activa la secuencia motora y la memoria completando los objetivos de los juegos con estrategias de discusión, resolviendo conflictos para ejercicio del proceso de toma de decisiones y negociación sensorial.

Para desarrollar los juegos se tiene en cuenta el almacenamiento de la información que se aumenta a medida que se usa el juego, permitiendo realizar el seguimiento al aprendizaje, planificación, selección de estrategias, monitoreo del progreso, corrección de errores, análisis de la efectividad y cambio de conductas. Con la información recolectada se implementan alternativas de análisis cuantitativo con el uso de herramientas estadísticas como SPSS[1] y AMOS[2], implementando paquetes que realizan actividades como Análisis de correlaciones entre variables, Análisis de ruta entre las variables, Modelo de análisis de ruta, entre otras funcionalidades.

Los juegos utilizados como apoyo terapéuticos se tienen como procedimiento específico al inicio del juego, para realizar el seguimiento a la evolución del paciente, que este se debe identificar al iniciar el juego [11], para evaluar su comportamiento en el entorno del juego, esto nos lleva a la identificación de las variables que pueden estudiar en el desarrollo del juego como; la posición, la orientación, la puntuación, la salud, cuando disparan un arma, y demás. Las variables están representadas en componentes más prevalentes y cambiantes de los datos recopilados normalmente en el juego. Con toda esta información almacenada en un computador se nos presenta un nuevo desafío para realizar el análisis de información.

Nos surge la pregunta ¿Cómo extraer el conocimiento sobre el comportamiento de los jugadores a través de la minería de datos?, al igual que como podemos realizar la visualización de datos que nos permita generar una guía para mejorar el diseño de los juegos, mejorar la experiencia interactiva con el jugador, y poder aplicar este conocimiento a la identificación de una mayor cantidad de beneficiarios de los juegos.

[1] https://www.ibm.com/co-es/products/spss-statistics/pricing
[2] https://www.ibm.com/us-en/marketplace/structural-equation-modeling-sem

Los datos son las acciones y comportamientos realizados en el juego por los jugadores, que son almacenados en base de datos como variables numéricas [11]. El proceso de minería de datos se describe como el registro de datos, la limpieza y depuración de datos, el análisis de los datos y su visualización se está realizando hoy en día en tiempo real, a medida que se está jugando. Para realizar el análisis de datos basado en juegos requiere de planificación, dada la transmisión simultánea de datos de juego (gráficos), junto con los datos del juego por esta razón se deben identificar las siguientes especificaciones:

- Tiempo requerido para completar la sesión de juego
- Número de errores cometidos durante la sesión de juego
- Número de autocorrecciones hechas,
- Tiempo de acceso (inicio de sesión para cerrar sesión),
- Cantidad de contenidos de aprendizaje.

En la actualidad los profesionales que usan juegos para ayudas terapéuticas están realizando un seguimiento a sus pacientes, con la implementación de cuestionarios que determinen la evolución en un tiempo específico del tratamiento asignado [19], involucrando alternativas con mayor o menor relevancia que ayuden en definitiva al mejoramiento del paciente. Con las nuevas tendencias tecnológicas, se busca que, mediante procesos de captura de datos generados desde los juegos, se pueda avanzar aún más, en el seguimiento evolutivo a los trastornos psicomotrices de los pacientes de forma más rápida y eficiente, aplicando procesos y teorías de inteligencia artificial que ayuden a la toma de decisiones, sobre las acciones de tratamiento inmediato a seguir.

Con los datos capturados se realizan evaluaciones de rendimiento, que son comparados entre los pacientes jugadores, los cuales son clasificados como expertos o novatos según su productividad, aplicando las respectivas métricas de evolución según el caso, detallando el progreso del paciente con la usabilidad de juego, donde se involucran características de cantidad de tiempo, eventos, niveles alcanzados y si fue o no terminado. Al relacionar las métricas asignadas para cada juego, se establecen nuevas alternativas de medición con la intención de conocer la evolución de aprendizaje del jugador.

Para implementar analítica de datos, se inicia con la actividad de exploración de datos que ejecuta tareas de estadística de aprendizaje o segmentación [20], utilizada para dividir a los jugadores en dos o más grupos, basados en semejanzas o similitudes de acuerdo con la disponibilidad de los datos capturados, que incluyen acciones, actitudes, comportamientos y necesidades. Para hacer uso de la información con el fin de predecir comportamientos futuros de los datos generados por los jugadores, se busca implementar algoritmos de aprendizaje supervisado como por ejemplo

(redes neuronales, árboles de decisión, máquinas de soporte vectorial, regresiones lineales) y no supervisado como clusterización, formulando hipótesis a que su vez automaticen procesos y que confirmen la evolución del paciente en sus acciones y comportamientos.

Algunos métodos comunes de la analítica de juegos que aparecen en la actualidad, es el análisis de clúster (clasificación de datos), análisis arquetípico (conocimiento del jugador y la relación con los datos), factorización matricial no negativa (conjugación del álgebra lineal) y análisis de componentes principales (PCA), este último, que consiste en un método estadístico que facilita el análisis exploratorio que reduce la cantidad de variables a utilizar.

El análisis de los datos de los juegos involucra mediciones de carácter cuantitativo y herramientas de aprendizaje automático, con la finalidad de realizar posibles seguimientos futuros a los eventos generados por los jugadores, con relación a las múltiples variables utilizadas para el almacenamiento de datos permanentemente, que ayuden a un mejor entrenamiento de los algoritmos para decisiones futuras y de tratamiento del jugador.

Por otro lado los juegos comerciales son considerados como una industria de entretenimiento que utiliza técnicas *Game Analytics*(GA), [21] diseñada para que los jugadores aprendan en tiempo real utilizando el concepto de telemetría definida como "Un tecnología que permite medir en forma remota de magnitudes físicas y el posterior envío de la información hacia el operador del sistema" . el objetivo de esta tecnología es mantener al jugador el mayor tiempo posible en la interfaz de juego y establecer la ganancia que está generando con el desarrollo de destrezas que permitan el autoaprendizaje.

La telemetría detecta, mide y controla de forma automática los datos de un dispositivo remoto. Su principal ventaja es la transmisión de datos desde dispositivos a un punto de control central "*Telemetry*", que también incluye el envío de información de configuración y control a los dispositivos con un análisis incluido. La palabra telemetría procede de las palabras griegas τῆλε, que quiere decir a distancia, y la palabra μετρον, que quiere decir medida.

4. Desarrollo de la metodología

Con la implementación de la metodología CRIPS-DM para el proyecto a continuación se desarrollan cada una de las 6 fases enfocadas en los objetivos del trabajo.

4.1. Compresión del negocio

Para el desarrollo del presente trabajo se cuenta con la información que fue generada por la utilización de juegos de computadora por pacientes con trastorno cognoscitivos leves para un periodo de tiempo fijo (10 minutos por sesión) y solo para 9 juegos, la información esta almacenada en archivos tipo texto. Los juegos que son estudiados en este trabajo corresponden a juegos de atención como se describen a continuación en la Tabla 1.

TABLA 1. DESCRIPCIÓN DE LOS TIPOS DE JUEGOS

Tipos de Juegos	Descripción	Concepto	Nombre de los Juegos
Juego de Atención sostenida	Mantener una Respuesta conductual consistente durante una actividad continua y repetitiva. Para este tipo de juegos el temporizador es un mecanismo importante.	Respuesta conductual a una actividad continua	• Alimentar a los delfines • Donde están mis compañeros • Carrera en bici
Juego de Atención Dividida	Se debe dar prioridad algunos elementos sobre otros, se presenta un estímulo blanco dentro de otra serie de estímulos irrelevantes y se debe seleccionar el estímulo blanco en un límite de tiempo	Habilidad para responder simultáneamente a varias tareas	• Libéralos • Competencia de animales • Juguemos baloncesto
Juegos de Atención Selectiva	Dividir la atención entre varios procesos o estímulos potenciales; el participante debe responder de forma simultánea a varias tareas u obstáculos.	Inhibir estímulos irrelevantes en pro de los relevantes	• La conejera • De paseo por la ciudad • Encuentra a uno igual (Busca que lo encuentras)

En el caso particular del juego de atención sostenida ‘Alimentar a los delfines’, realizamos su descripción para realizar un entendimiento general de los juegos en cuanto a su utilización, interacción y resultado obtenidos.

Nombre del juego: Alimentando a los delfines

El área de estimulación de este juego es atención sostenida que involucra las siguientes conductas:

- La ejecución de la tarea varía en función de las características temporales

- Mantener una respuesta conductual consistente durante una actividad continua y repetitiva
- Niveles altos de vigilancia, pero pocas respuestas
- Eventos de baja frecuencia con una duración larga.

El objetivo principal del juego: Alimentar a los delfines

Mecanismo de juego:

- **Plataforma:** los juegos son desarrollados para ser trabajados en WEB
- **Introducción al juego:** En tus vacaciones de verano te llevaron de paseo al mar y en su recorrido se encontraron con el acuario, en donde puedes alimentar a los delfines mandándoles pescados cuando se acercan y asoman su cabeza en el agua
- **Descripción del contexto:** el juego se desarrolla en una piscina donde se encuentran varios delfines, la gran mayoría del tiempo los delfines permanecen sumergidos, el agua cristalina permite que se puedan ver y solo en algunas ocasiones salen a la superficie como se muestra en la Figura 11.

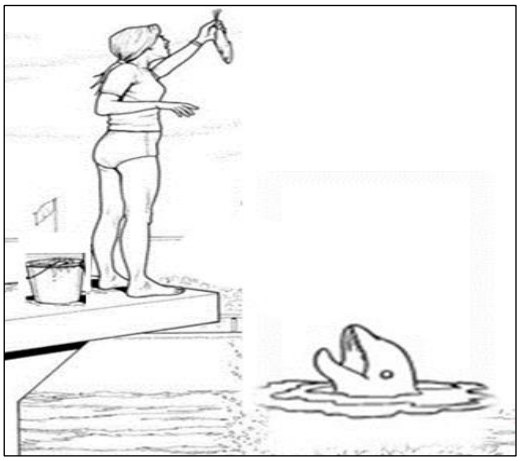

FIGURA 11. DIBUJO JUEGO ALIMENTANDO A LOS DES FINES

- **La jugabilidad:** Nuestros jugadores recibirán una cantidad de determina de peces en una cubeta, el adiestrador de los delfines del acuario es la persona encargada de impartir las instrucciones para alimentar a los delfines, que debe ser cada vez que el asomen la cabeza. La piscina es recorrida por los delfines y en determinadas ocasiones se dirigen a la superficie y asoman la cabeza para ser alimentados
- **Los personajes:** los actores de este juego son el niño (niña), los delfines y el adiestrador
- **Herramientas utilizadas**: son la cubeta y los pescados
- **Iteración con el ambiente:** el juego se desarrolla en el acuario, en la piscina de los delfines, donde son alimentados los delfines. Los niños se encuentran parado sobre una plataforma.

- **Cámara:** el juego se desarrolla en primera persona, donde se presenta una perspectiva lateral del tablero y los delfines están ubicados al frente del niño
- **Periféricos:** para la utilización del juego se utiliza el mousse
- Controles: Manipulación del mousse
- **Flujo del juego:** se muestra el
- Figura 12 del diagrama con sus flujos el desarrollo del juego.

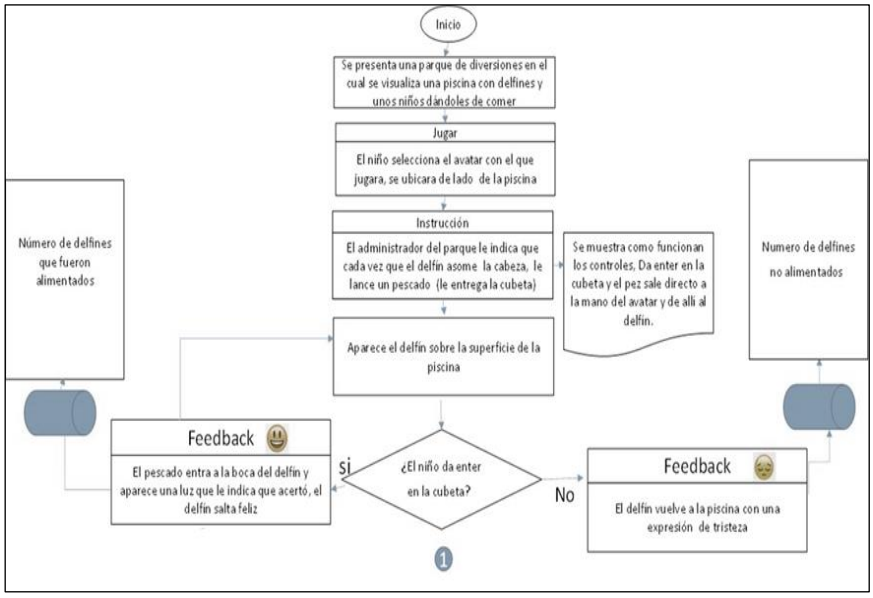

FIGURA 12. DIAGRAMA DE FLUJO DEL JUEGO

Los usuarios que realizaron las pruebas fueron niños sanos que no presentaban ningún trastorno cognoscitivo y con las debidas autorizaciones de sus padres en total se seleccionados 9 participantes.

La metodología utilizada en el desarrollo del programa siguió las siguientes pautas: se realizaron 9 sesiones con una periodicidad de tres por semana, con una duración de 1 a 10 minutos por juego y cada día se aplicaron los tres juegos siguiendo el siguiente orden:

- **Semana 1.** Juegos de atención sostenida (alimentando a los delfines, dónde están mis compañeros y carrera en bici).
- **Semana 2.** Juegos de atención selectiva. (la conejera, cuidando los animales pequeños, encuentra a uno igual).
- **Semana 3.** Juegos de atención dividida (Libéralos, competencia de animales y Juguemos baloncesto).

Una de las razones por las cuales se está realizando este estudio tiene que ver con poder entender el propósito u objeto por el cual se construyeron los juegos (que es como herramienta de apoyo psicológica para el tratamiento de traumas de carácter cognoscitivo leve) realizando un análisis de datos con la teoría de la AI.

4.2. Compresión de los datos

Se tienen 89 archivos planos de los diferentes juegos utilizados, ocupando 313 KB de espacio, el nombre del juego tiene la siguiente información: los primeros dos dígitos identifican al jugador, luego se identifica el nombre del juego, sigue el número del intento y por último la fecha en la que se realizó el juego(día/mes/año). De los jugadores no sé conoce ninguna información adicional y por esta razón no es una variable de interés para esta investigación.

Por cada sección de juego se genera un archivo tipo texto que tiene las siguientes características: los archivos tienen diferentes tipos y cantidad de variables, al igual que el número de registros, al terminar el juego se generan las variables de resumen los cual se encuentran al final del archivo y se almacena en las siguientes variables, Aciertos y tiempo totales.
Se realiza la descripción de las variables del juego: Alimentando a los delfines.

4.2.1. Descripción de variables:

- **Respuesta:** clic del niño sobre la cubeta, que genera que el pescado salte al delfín
- **Estímulo:** Delfín saliendo a la superficie (1)
- Delfín sumergido que se ve la sombra (2)
- Delfín en la superficie (3)
- **Ensayo:** total de veces en que se presenta el estímulo 1 en un intervalo de tiempo definido
- **Tiempo de presentación del estímulo:**

  Estímulo (1) el delfín que sale a la superficie (al comienzo son más saltos a la superficie y conforme avanza el juego disminuye)

  Estímulo (2) Delfín sumergido que se ve la sombra (al comienzo permanece menos tiempo, conforme avanza la dificultad aumenta el tiempo de permanencia bajo el agua)

  Estímulo (3) el delfín se asoma a la superficie 2 segundos
- **Número de errores:** número de veces en que el niño da clic en ausencia del estímulo 1 o el número de veces en que sigue dando clic, aunque ya se haya alimentado al delfín.

Datos resumen del juego como variables transpuestas

- **Número de aciertos:** aciertos de dar de comer al delfín cuando sale a la superficie es el estimulo1.
- **Tiempo total:** Tiempo que trascurre entre el inicio y el fin del juego.

Para comprender los conceptos de Big Data, se realizará la describen los datos que son utilizados en la investigación realizando las definiciones basados en los conceptos de las cinco V, como se describe a continuación:

- **Variedad:** tiene relación con los tipos de variables en los cuales se almacena la información según para muestro caso lo podemos ver en la Tabla 2. Descripción de Variables de los Archivos que conforma los archivos, también la posibilidad de almacenar diferentes tipos de información que puede venir en archivos planos, gráficos, Excel, Word y muchos otros formatos los cuales deben ser almacenado en la base de datos.

TABLA 2. DESCRIPCIÓN DE VARIABLES DE LOS ARCHIVOS

Nombre Variable	Total, Registros	Qué tipo de variable	Tipo
Aciertos	2122	Cuantitativas	Discretas
Clicks Ensayo	1606	Cuantitativas	Discretas
Clicks Post Ensayo	1954	Cuantitativas	Discretas
Clicks Pre-Ensayo	516	Cuantitativas	Discretas
Errores	55	Cuantitativas	Discretas
No. Compañeros	55	Cuantitativas	Discretas
Omisiones	55	Cuantitativas	Discretas
Teclas Oprimidas	481	Cuantitativas	Discretas
Tiempo Asoma Cabeza	348	Cuantitativas	Continua
Tiempo Duración Ensayo	1606	Cuantitativas	Continua
Tiempo Obstáculo En Rango Salto	113	Cuantitativas	Continua
Tiempo Primer Crick	2067	Cuantitativas	Continua
Tiempo Primer Crick Asoma Cabeza	348	Cuantitativas	Continua
Tiempo Primer Compañero	55	Cuantitativas	Continua
Tiempo Primer Tecla Oprimida	481	Cuantitativas	Continua
Aciertos totales	89	Cuantitativas	Discretas
Ensayo	2211	Cuantitativas	Discretas
Tiempo total	89	Cuantitativas	Discretas
Fecha	2211	Cuantitativas	Continua
Intento	2211	Cuantitativas	Discretas
Juego	2211	Cualitativa	Nominal
User	2211	Cualitativa	Nominal
tiempo por acierto promedio	89	Cuantitativas	Intervalo

- **Volumen:** está relacionado con la cantidad de información a procesar, donde esta almacenada y los recursos en tecnología: Se cuenta con 89 archivos planos que tienen en total 2211 registros y 22 columnas, ocupando 313 KB de espacio, la herramienta de tecnología utilizada para el procesamiento es Python en la cual se realizaran las tareas de cargue de la información, depuración, procesamiento y visualización. El proyecto está

diseñado para ser trabajado en internet y el almacenamiento se debe realizar en la nube para garantizar que el volumen de información que produzcan los juegos pueda ser almacenado sin inconveniente, en la Figura 13 se puede ver la estructura general del proyecto.

- **Velocidad:** con la estructura del proyecto se identifica que los juegos son utilizados por los usuarios desde internet y la información producida en la relación del jugador y el juego se almacenada de forma directa en una base de datos no estructurada, se debe garantizar que no existen problemas con el flujo de datos sin importar la cantidad y la velocidad con la cual son generados. Como se muestra en la Figura 13 en el esquema general del sistema de información está basado en HTML 5 para su utilización en internet como la plataforma del uso de los juegos, de esta forma se garantiza la disponibilidad, el almacenamiento y el acceso a los datos en tiempo real por el módulo que se creen para uso de la información como *machine learning*.

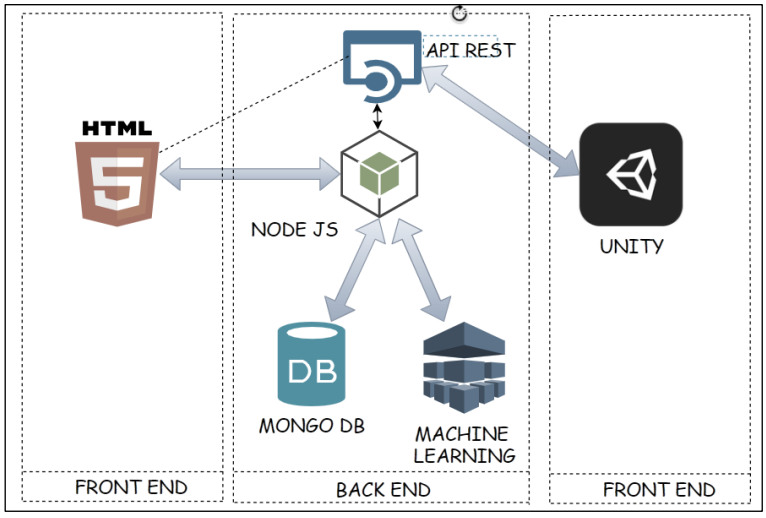

FIGURA 13. ARQUITECTURA DE LA APLICACIÓN DE JUEGOS

- **Veracidad:** los datos que se generan cuando un paciente utiliza un juego son cargados directamente a la base de datos en la cual son almacenados, con este proceso se garantiza que la información que se almacena en la base de datos no tiene errores generados por la manipulación de los archivos y las aplicaciones pueden trabajar con esta información.

- **Valor:** el valor de los datos esta dado por lo que podemos encontrar en ellos al realizar el análisis de cada una de las variables y sus relaciones, para poder establecer la evolución de los pacientes con base en la destreza que desarrolla en el manejo del juego. El análisis debe proveernos suficiente información para poder saber si el juego es efectivo como terapia complementaria en los pacientes que lo utilizan, con los resultados obtenidos en

el análisis de los datos se deben realizar los ajustes a los juegos para hacerlos más efectivo y eficaces como terapia complementaria a los pacientes con trastorno cognoscitivos leves.

4.2.2. Exploración inicial de los datos

Para el procesamiento de los datos se utiliza el lenguaje de programación Python, porque tiene las siguientes características: fue creado para ser leído con facilidad, fácil aprendizaje, filosofía de código abierto, permite trabajar con diferentes tipos de datos. Para nuestro trabajo se realiza una lectura de forma directa desde un directorio de donde están almacenados los archivos y los programas que realizan la carga y procesamiento.

A continuación, se muestra la descripción de los datos que se encuentran en los archivos planos para lo cual se utilizó Python como lenguaje de programación y Spyder como interfaz gráfica, donde se identifican las siguientes características de la información:

Los archivos están clasificados en tres grupos según el estímulo de atención para el que fue creado el juego, a continuación, relacionamos el inventario de los juegos por tipo de atención según la siguiente tabla:

TABLA 3. INVENTARIO DE JUEGOS POR TIPO DE ATENCIÓN

Tipo de Atención	Juegos creados por estimulo	Total, juegos
Sostenida	delfín, bici, compañeros	3
Selectiva	paseo, conejera, igual	3
Dividida	Libérelos	1

TABLA 4. ESTADÍSTICOS DE LAS VARIABLES

	Aciertos	Clicks Ensayo	Clicks Post Ensayo	Clicks Pre-Ensayo
Count	2122	1606	1954	516
Mean	0.848728	0.960149	0.212385	0.825581
Std	0.733274	1.437255	0.645144	2.180364
Min	0	0	0	0
25%	0	0	0	0
50%	1	1	0	0
75%	1	1	0	1
Max	5	17	6	35
	Errores	**No. Compañeros**	**Omisiones**	**Teclas Oprimidas**
Count	55	55	55	481
Mean	0.127273	3.527273	0.436364	0.997921

Std	0.33635	0.978558	0.995613	0.549617
Min	0	1	0	0
25%	0	3	0	1
50%	0	4	0	1
75%	0	4	0	1
Max	1	5	4	5
	Tiempo Asoma Cabeza	**Tiempo Duracin Ensayo**	**Tiempo Primer Tecla Oprimida**	**Aciertos totales**
Count	348	1606	481	89
Mean	5.194542	5.046875	2.111441	15.617978
Std	2.720585	5.29635	1.779208	15.106821
Min	1.29658	1.032308	0	0
25%	2.833052	2.833155	1.216334	5
50%	4.880447	3.831716	1.783232	10
75%	8.043617	5.56326	2.681676	23
Max	10.12837	114.566	22.89838	77
	Ensayo	**Tiempo total**	**fecha**	**intento**
Count	2211	89	2.21E+03	2211
Mean	20.504749	143.371105	1.53E+09	2.847128
Std	18.377818	85.487528	4.08E+06	1.911424
Min	0	23.78282	1.52E+09	1
25%	7	120.9538	1.53E+09	2
50%	15	129.1009	1.53E+09	3
75%	28	183.8481	1.53E+09	3
Max	98	515.7602	1.54E+09	12
	Juego	**User**	**tiempo acierto promedio**	**tiempo intento promedio**
Count	2211	2211	89	89
Mean	5.602442	8.971958	0.124627	0.160148
Std	2.165917	2.700114	0.093563	0.077857
Min	1	4	0	0.041387
25%	4	7	0.04096	0.101158
50%	6	8	0.093853	0.135722
75%	8	11	0.193647	0.201266
Max	8	13	0.360614	0.410162

El primer paso que realizamos es la estadística descriptiva de las variables que conforman la base de datos. En la Tabla 4, podemos identificar las principales características estadísticas de las variables de los archivos trabajados, dicho resultado se obtuvo utilizando comando de Python.

Se realiza una descripción grafica de las variables como se puede ver en los siguientes gráficos:

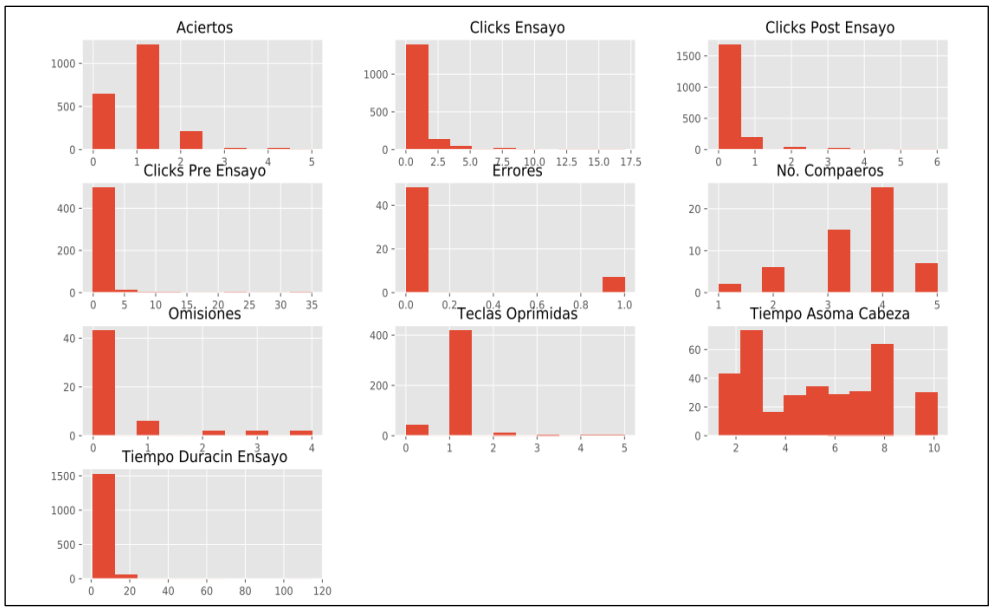

FIGURA 14. HISTOGRAMA DE DESCRIPCIÓN GRAFICA DE VARIABLES 1

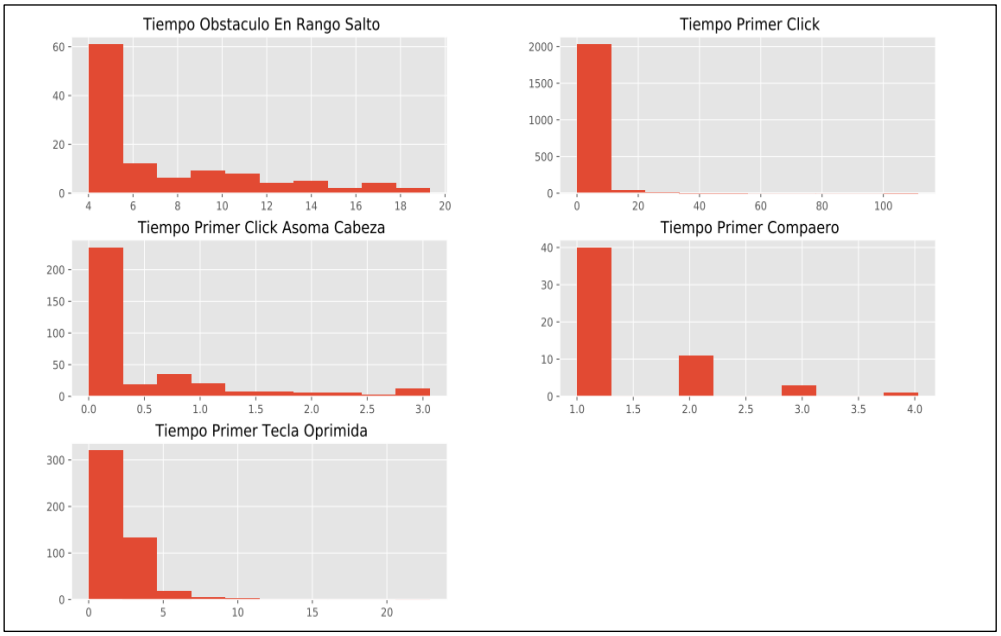

FIGURA 15. HISTOGRAMA DE DESCRIPCIÓN GRAFICA DE VARIABLES 2

Las figuras utilizadas para la descripción de variables son histogramas las cuales nos permiten identificar la información por cada una de las variables con Python como se puede ver en la Figura 14 y la Figura 15, a cada una de las variables seles genero su histograma el cual permitió realizar la identificaron sus características.

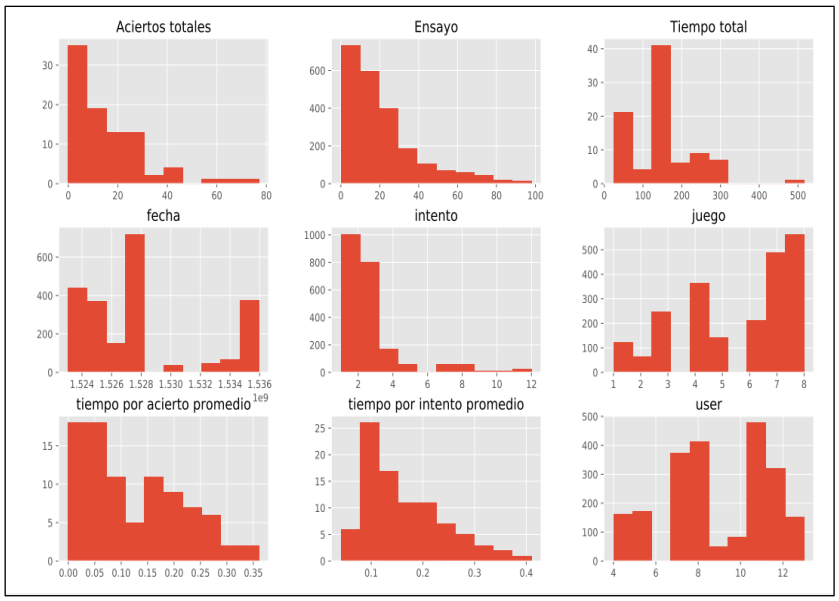

FIGURA 16. HISTOGRAMA DE DESCRIPCIÓN GRAFICA DE LAS VARIABLES RESUMEN

Como se pudo observar en los archivos se encuentran unas variables resumen las cuales se complementaron con otras derivadas (tiempo por acierto promedio, tiempo por intento promedio) que permitieron tener más información de los juegos. La visualización de las frecuencias simples se puede ver en la Figura 16, para la variable Aciertos totales se identifica que existen usuarios que realizaron más de 30 aciertos.

4.2.3. Analítica descriptiva

La analítica descriptiva está determinada en el grado de satisfacción y comprensión de los objetivos del juego por los usuarios lo que se conoce como usabilidad, que responde la primera pregunta de investigación del trabajo "¿Como podemos determinar si un juego es aprendido de forma intuitiva por los pacientes a partir de los datos?", según numeral **1.3 Preguntas de Investigación**, mediante la elaboración de una guía que permite evaluar la usabilidad de los juegos, tomando como base los trabajos de Ferrari Alve, Santiago Iván Mariño, Sonia I [19] . Los parámetros seleccionados para la medición de la usabilidad de los juegos se relacionan a continuación:

- **Facilidad de aprendizaje:** con este parámetro se pretende medir la facilidad de uso del juego, evaluando su comprensión de las metas a cumplir y la facilidad de manejo del juego. La medición de la facilidad de aprendizaje se propone medir con la relación de intentos, aciertos totales por juego.
- **Recuerdo en el tiempo:** Para cada uno de los usuarios, en cada uno de los intentos debe ser capaz de recordar la mecánica del juego y mejorar la cantidad de aciertos.

- **Entendibilidad:** Es evaluar si a los nuevos usuarios se les facilita poder entender para que fueron diseñados los juegos. Los atributos del parámetro son:
  - Funciones de la interfaz entendibles. Poder entender las funciones y los elementos de los juegos, para saber si están diseñados para una fácil comprensión por los jugadores.
  - Explicación clara de requisitos de entrada y salida.
  - Determinar la medida en la que el jugador entiende los datos iniciales para el inicio del juego y los que son retornados como salida.
  - Lenguaje sencillo y breve. Determinar si el lenguaje que es utilizado en el trascurso del juego como ayuda y la documentación están escritos de forma sencilla y son acordes con las características de los jugadores.
- **Atractivo:** Cuando estamos evaluando un juego de computadoras se debe tener especial atención a la apariencia estética y su atracción visual que logra atraer el interés del jugador mediante el uso de los colores, gráficos y textos que permiten una interacción con el jugador y se pueda entender de forma fácil el objetivo del juego.

Se realiza una Evaluación heurística con el propósito de identificar problemas relacionados con la usabilidad que permitan identificar entre otras cosas las características de funcionamiento, los errores y las posibles mejoras de los juegos. Para lo cual se sigue la guía [19] la cual cumple con las siguientes tareas:

- En nuestro caso en particular para realizar el primer juego se dejó al niño que lo explorara de forma intuitiva sin entrenamiento y para el segundo juego se realizó una capacitación sobre el juego y su objetivo.
- Los parámetros que se definieron para evaluar la usabilidad son los siguientes: Aciertos totales, Ensayo, Tiempo total, intento, juego, tiempo por acierto promedio, tiempo por intento promedio, Aciertos por Intentos.
- Con los parámetros definidos, se realizaron las preguntas que permiten determinar si se cumplen o no las tareas.

Crear la guía para evaluar cada una de las preguntas para obtener las frecuencias en cada uno de los juegos, al igual que el impacto que causan. La estimación del impacto se expone en la Tabla 5.

TABLA 5. CLASIFICACIÓN DE LAS DEFINICIONES DE IMPACTO

Impacto	Explicación
Bajo	Se recomienda que se cumplan las afirmaciones, si se incumplen esto no debe implicar confusión ni error en el jugador. No generan un problema de usabilidad importantes.

Medio	Si se presenta un incumplimiento y esto provoca algún problema no muy grave de usabilidad, estos deben ser resueltos de inmediato para que se facilite el funcionamiento del juego.
Alto	Si produce un problema grave de comprensión y además de funcionalidad en el juego, es urgente que el problema sea resuelto. Porque esto puede provocar problemas graves de usabilidad.

Se realiza un análisis a cada una de las categorías y los juegos que la conforman:

Juego de Atención sostenida: para esta categoría de juegos se encontró las siguientes características en los archivos. En los juegos del delfín y bici la relación de las variables total de aciertos y ensayos se cumple que: el número de ensayos es superior a los aciertos totales lo cual no sucede con el juego compañeros donde el número de aciertos totales es superior al número de ensayos lo cual se evidencia en la Figura 17. Donde para 3 ensayos (Total tests), se obtuvieron 6,7,11,12 Aciertos (Total hits).

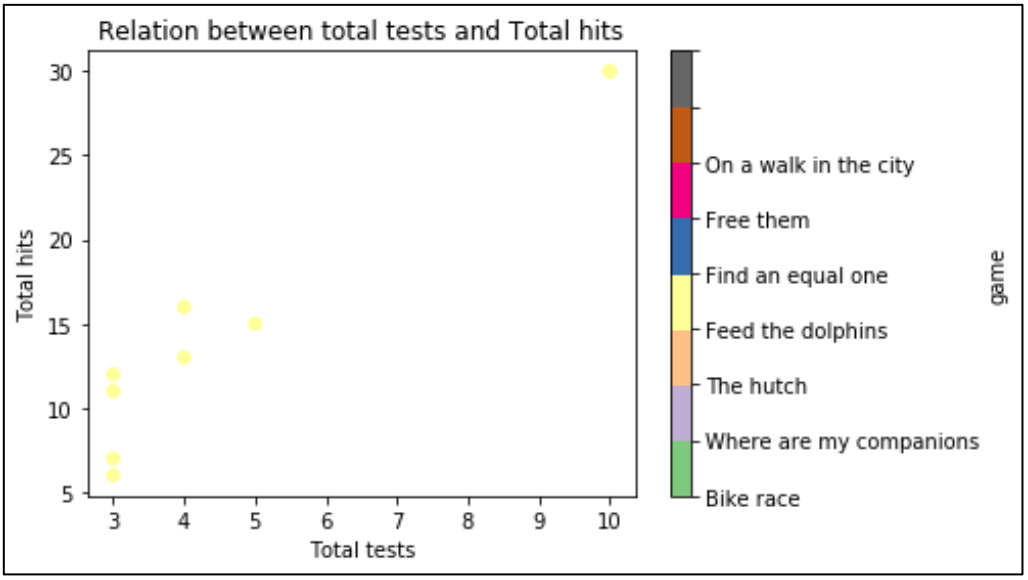

FIGURA 17. RELACIÓN ENTRE ENSAYOS Y ACIERTOS

Juegos de Atención selectiva: (paseo, conejera, igual) de esta categoría los juegos tienen la misma estructura en los archivos generados y se cumple que el número de ensayos es superior a los aciertos totales como se puede ver en la Figura 18.

Juegos de atención dividida: (libérelos) de esta categoría de juegos solo se cuenta con los archivos para un solo juego y se cumple que el número de ensayos es superior a los aciertos totales como se puede ver en la Figura 18.

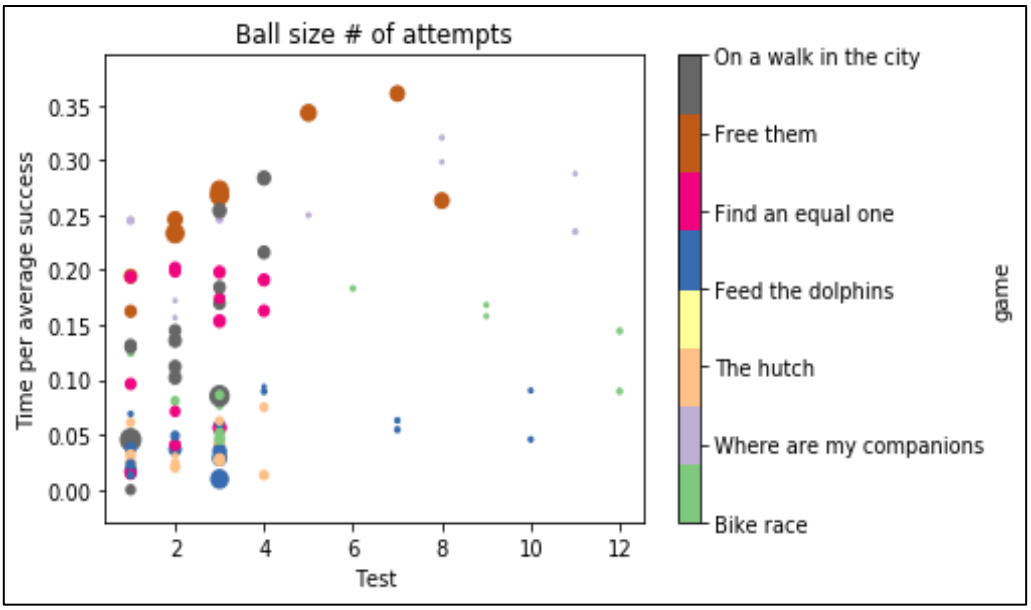

FIGURA 18. RELACIÓN ENTRE ENSAYOS Y ACIERTOS TOTALES

Después de la identificación de los juegos por categorías, se realiza una exploración inicial para buscar relaciones existentes en los datos utilizando una matriz de correlación, como lo muestra en la Figura 19.

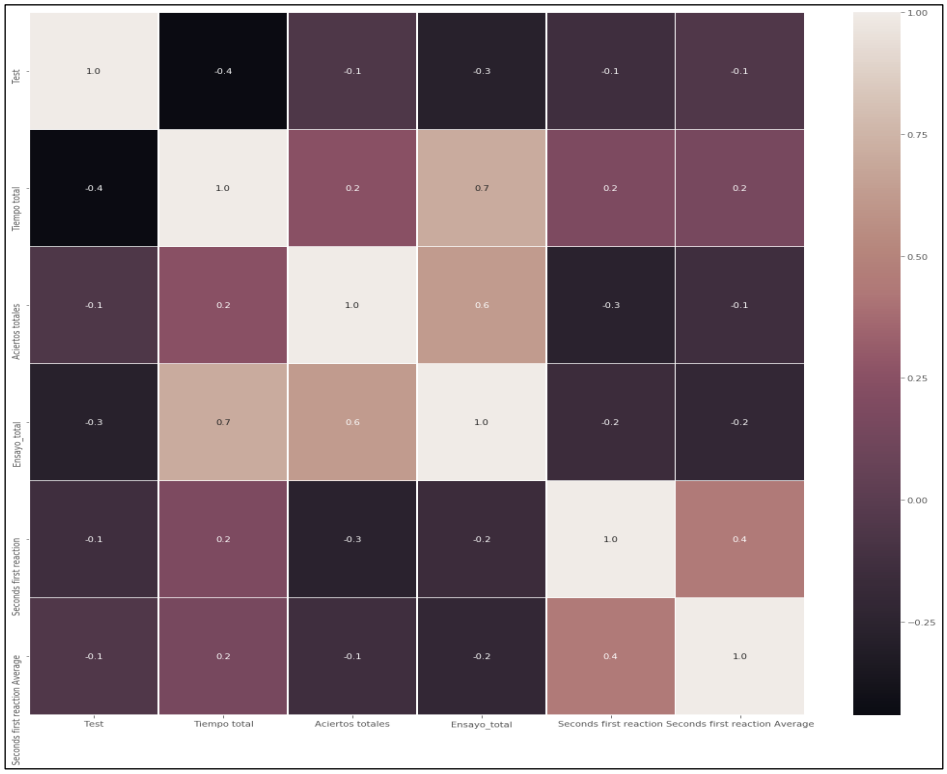

FIGURA 19. MATRIZ DE CORRELACIÓN

Con la información de la gráfica anterior se identifica la relación existente entre las variables aciertos totales y el ensayos totales, para realizar una mejor comprensión utilizamos gráficas de

histogramas y cajas de bigotes, lo cual permitió comprobar la existencia de una relación significativa, según la Figura 20 y la Figura 21. Las variables se conservan para la evaluación de la usabilidad ya que estas variables se encuentran en todos los archivos.

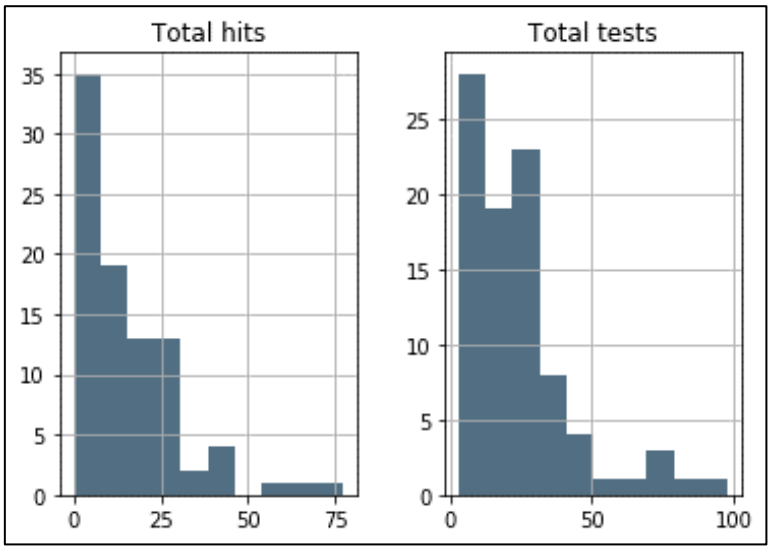

FIGURA 20. HISTOGRAMA DE ACIERTOS TOTALES Y TOTAL ENSAYOS

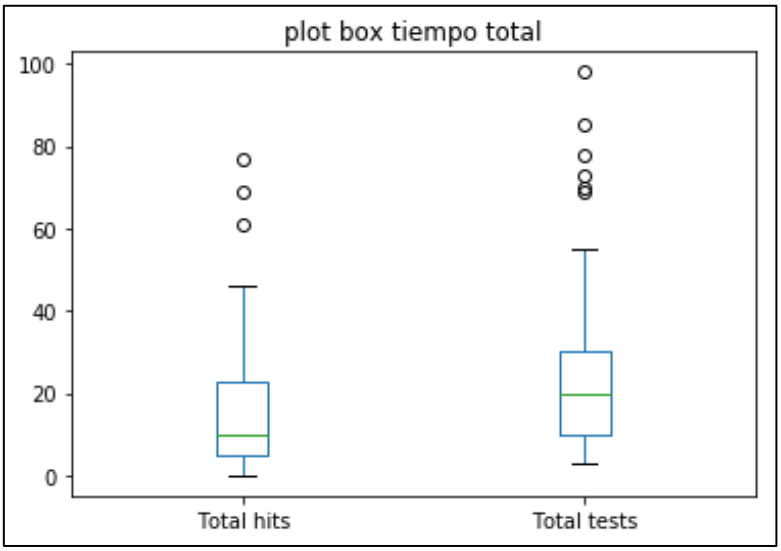

FIGURA 21. CAJAS DE BIGOTES DE ACIERTOS TOTALES Y TOTAL DE ENSAYOS

Evaluación del impacto de la facilidad de aprendizaje de los juegos, a continuación se realiza la Figura 22, donde se muestra la comparación entre el primer y segundo intento en los juegos, por un usuario, para el primer intento se deja al jugador que realice una exploración intuitiva y juegue, para el segundo intento se capacito al jugador y luego se deja que juegue, se puede observar que existe un dato que se aleja de la media por 100 ensayos realiza 11 aciertos, para los demás datos se observa que para el segundo intento la cantidad de aciertos se aumentaron gracias

a la capacitación. Para la facilidad de aprendizaje al evaluar los datos muestran que el aprendizaje de los juegos es progresivo y no presentan problemas grabes de usabilidad.

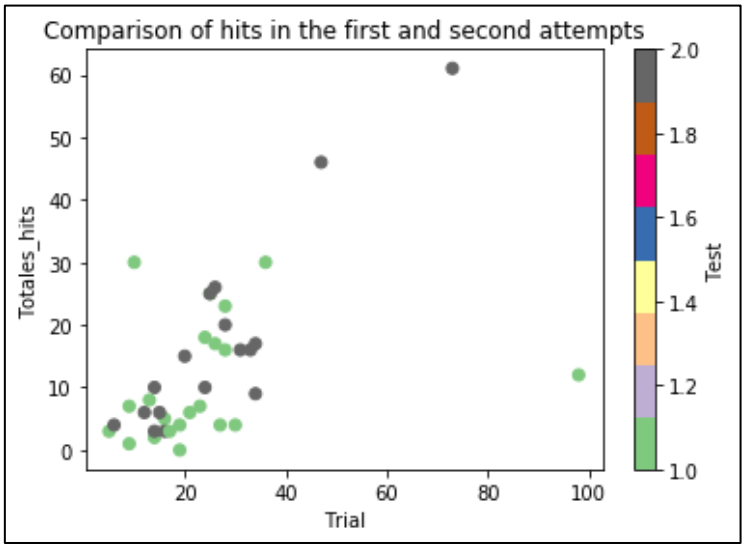

FIGURA 22. COMPARACIÓN PRIMER Y SEGUNDO ENSAYO

Evaluación del recuerdo en el tiempo: para realizar esta evaluación se crea una nueva variable que resulta de la relación entre el número de aciertos totales y el tiempo total de juego, a la variable se le asigna el nombre de "tiempo por acierto promedio". En la Figura 23, se presenta la relación entre tiempo por acierto promedio y el número del intento por usuario, se puedo identificar que la mayoría de los jugadores realizaron los primeros cuatro intentos, solo tres jugadores realizaron más intentos y su tiempo promedio al realizar un acierto disminuyo a medida que realizaban más intentos. Se presentaron una excepción con el *user* 8, donde el tiempo de respuesta para el intento 3 el tiempo por aciertos promedios es muy grande 100 segundos.

El recuerdo en el tiempo entre los datos evaluados se puede identificar, que entre el primer y último intento los tiempos promedios no aumentan, esto demuestra que los usuarios no olvidaron el objetivo y la forma de jugar el juego, podemos afirmar que los juegos no presentan problemas de usabilidad.

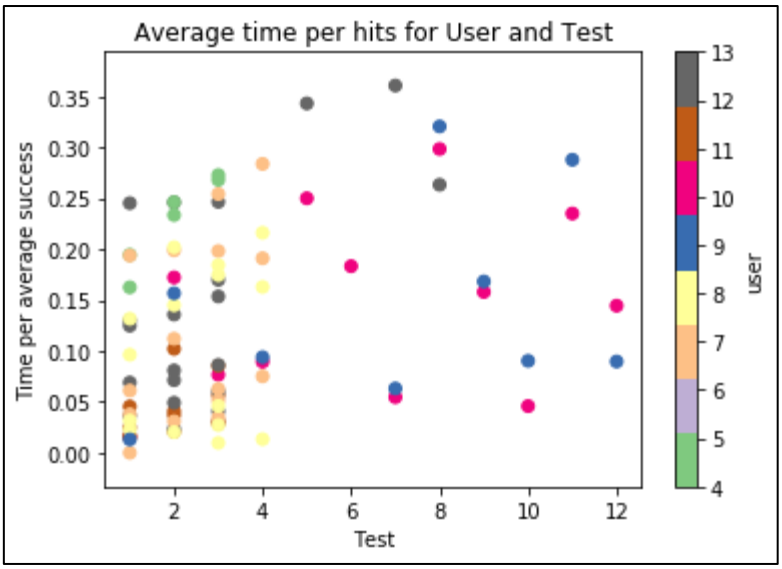

FIGURA 23. TIEMPO POR ACIERTOS PROMEDIO PARA INTENTOS POR USUARIO

La entendibilidad se evalúa en la Figura 24, donde se muestran las relación entre las variables de tiempo por acierto promedio, número del intento realizado para cada uno de los juego y se adiciona la cantidad de ensayos que se realizan para cada uno de los intentos, el cual es representado por el grosor de las esferas, para un caso en particular se puede identificar que existió un tiempo muy grande para realizar un acierto y que se realizaron muchos ensayos esto ocurrió en el intento tres para el juego delfín, pero a medida que aumentaron los intentos se disminuyó drásticamente el tiempo por aciertos y los ensayos.

Existe un caso donde se realizan muchos ensayos en cada uno de los intentos, pero el tiempo promedio por acierto es muy pequeño en el juego libérenlos (para este juego en particular evalúa la forma de jugarlo que es el teclado lo cual multiplica el número de intentos en un menor tiempo). Se establece que la comprensión y entendimiento de los juegos por los usuarios es bueno y no tiene mayores inconvenientes para comprender los objetivos a ser alcanzados.

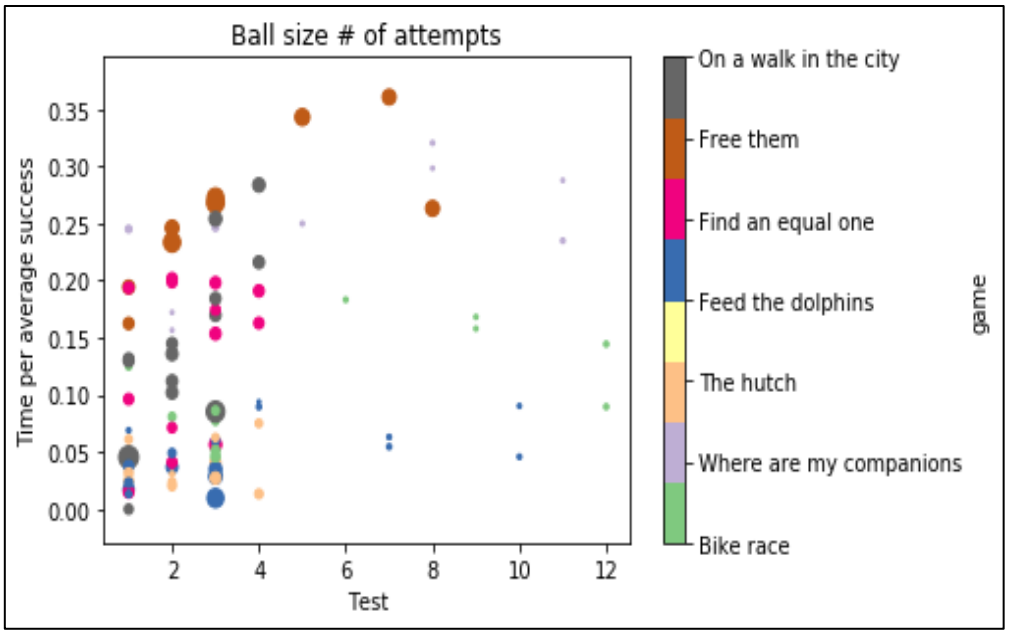

FIGURA 24. TIEMPO POR ACIERTO PROMEDIO PARA LOS INTENTOS DE CADA JUEGO Y LA CANTIDAD DE ENSAYOS

Evaluación de la Atracción de los juegos, En la Figura 25, en esta grafica podemos identificar el tiempo que tarda un jugador en iniciar su primera jugada después de haber ingresado al juego, en el intento 3 se observa que los tiempos son mayores y en especial para el juego de la bici, y existe un caso especial del juego igual que utiliza más de 20 segundos, para los demás intentos por juego se observa que los tiempos de respuestas son buenos y no presentan inconvenientes para los usuarios. Se deduce que no existen problema de rechazo y poca atención por parte de los usuarios con los juegos.

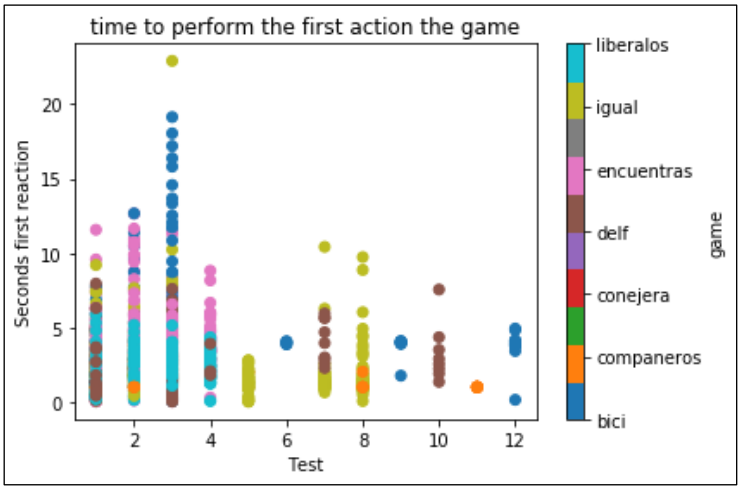

FIGURA 25. TIEMPO QUE TARDA EL USUARIO EN REALIZAR LA PRIMERA ACCIÓN EN EL JUEGO

4.3. Preparación de los datos

Con la información de los archivos planos, se procede a realizar las siguientes tareas que permitirán desarrollar del presente trabajo.

4.3.1. Selección de los datos

Para realizar la selección de las variables a utilizar se realizó una evaluación a cada una de las 26 variables que forman el archivo de las cuales se identificaron seis Tabla 6, que lo gran caracterizar todos los archivos y no se tienen en cuenta para este estudio las otras 20, porque solo tienen información que describe un juego en específico u intento.

TABLA 6 . VARIABLES SELECCIONADAS DE LOS JUEGOS

Variables	Descripción
Game	Número del juego
User	Número del Jugador
Test	Numero de Intentos
Total_time	Tiempo total jugado
Total_hist	Total, aciertos
Total_test	Total, intentos

4.3.2. Limpieza de los datos

Para realizar este proceso, se excluyeron las columnas que tenían muchos datos nulos, no se realizó el complemento de datos faltantes por cada uno de los juegos generan diferente número de datos y se clasifican en diferentes variables, sin embargo, se crearon nuevas variables que agrupaban información de varios juegos que tenían igual significado.

4.3.3. Nuevos datos derivados

En el proceso de exploración se pudo identificar que la información de los diferentes juegos se guardaba en sus propias variables pero que existían juegos que tenían la misma información en otra variable, con esta información se procedió a crear las nuevas variables Tabla 7. Esto permitió poder contar con la mayor cantidad de información se crearon las siguientes:

TABLA 7. DESCRIPCIÓN NUEVAS VARIABLES

Variable derivada	Descripción	Formula
Time per average success	Esta variable se obtiene de dividir los aciertos totales en tiempo total	df['Time per average success'] = df['Aciertos totales']/df['Tiempo total']
Seconds first reaction	Esta variable se obtiene al unifican los datos de 'Tiempo Primer Tecla Oprimida', 'Tiempo Primer Click', 'Tiempo Primer Compaero' y se toma la primera reacción del jugador	df['Time first reaction']=df[' Tiempo Primer Tecla Oprimida'].fillna(0)+df[' Tiempo Primer Click'].fillna(0)+ df[' Tiempo Primer Compaero'].fillna(0)

Reaction Average	se calcula el promedio de tiempo de las mismas variables de 'Second first Reaction' para el total de los intentos que realizo durante el juego.	

4.3.4. Formato de los datos

Con las nuevas variables creadas Tabla 8 y las seleccionadas de los archivos se realizaron la aplicación de los algoritmos de *Machine Learning*.

TABLA 8. DESCRIPCIÓN DE VARIABLES A EVALUAR POR LOS ALGORITMOS

Nombre Variable
Game
User
Test
Total Time
Total hist
Total t**est**
tiempo por acierto promedio
Time per average success
Seconds first reaction
Reaction Average

El total de registros utilizados es de 89 y las variables son de tipo Cuantitativas y de tipo discretas.

4.4. Modelamiento

Para esta fase de la metodología se implementarán 8 modelos. Uno con aprendizaje no supervisado para responder la segunda pregunta de investigación del trabajo y los demás con algoritmos de aprendizaje supervisado para responder la tercera pregunta de investigación. véase numeral **1.3. Preguntas de investigación**. Es importante aclarar que para la primera pregunta de investigación esta se respondió dentro ítem de **4.2.3. Analítica descriptiva**.

4.4.1. Técnicas de modelamiento

Con el conocimiento adquirido del negocio y el análisis realizado a cada una de las variables y la identificación del grado de correlación que existente, esto permitió contar con la suficiente información para la selección de las técnicas de modelado, que permiten responder las preguntas que están planteadas en el presente trabajo y las técnicas seleccionadas. Las técnicas seleccionadas se dividen en dos grupos las que se utilizan para aprendizaje no supervisado y supervisado.

Aprendizaje No supervisado

Clustering por K-medias: es uno de los algoritmos más utilizados para realizar clasificaciones que no requieren supervisión, los cuales realizan una agrupación de objetos en un determinado número de k grupos, los cuales se basan en sus principales características. Como uno de los propósitos del presente trabajo es poder identificar las características que permiten relacionar los diferentes juegos que fueron desarrollados para evaluar los diferentes trastornos de atención como son: atención sostenida, atención dividida y atención selectiva.

El algoritmo de *K-means* [20] requiere como insumos los datos y el número de grupos en los cuales se deben segmentar la población. Con esta información se crean los centroides de forma aleatoria e iniciar la asignación de los puntos con las menores distancias y el punto se va desplaza a la media de las muestras más cercanas. Con esto se está creando una nueva asignación de muestras, produciéndose de forma iterativa hasta que se ajustan los valores. El resultado final de las iteraciones realiza es el ajuste Maximino entre las distancias de los grupos y la minimización en cada uno de los inter-grupos.

Análisis de componentes principales o PCA

Para la valoración de los datos [21] se utilizamos la técnica estadística que permite realizar la reducción de dimensiones con lo cual se logra: que se disminuya la cantidad de almacenamiento, se disminuye el costo de computación y el tiempo de respuesta en el procesamiento, el proceso realizando una selección de las más importantes características que represente mejor los datos. La forma en la cual se selecciona las variables esta determina por la varianza que producen en la salida se tiene en cuenta la característica que produce mayor variación es la que se toma como componente principal, para la segunda varianza se selecciona la que produce la segunda varianza más alta y se continua con este método, se debe garantizar que los componentes principales no tienen correlación entre sí.

Para la implementación de PCA utilizaremos la biblioteca *scikit-learn* de Python, pero primero debemos realizar la normalización de las características, es el primer paso para la implantación de PCA seguimos los siguientes pasos:

- Paso 1 se realiza la división de los datos en dos conjuntos de entrenamiento y pruebas.
- Paso 2 se realiza una normalización escalar estándar de los datos seleccionados.
- Paso 3 Aplicamos PCA a los datos seleccionados, primo se crea un objeto PCA y luego se utilizan los métodos *fit y transform*, tiene otro componente que es *explained_variance_ratio* "relación de variación explicada".

Aprendizaje Supervisado

- **Regresión lineal:** Este modelo nos permite evaluar la dependencia entre variables y la identificación de patrones que están presente en la información pero que no son fáciles de identificar y este algoritmo permite el entrenamiento y las posteriores pruebas para evaluar si está o no cumpliendo con la predicción de la información.

- **Redes neuronal artificial (RNA):** este algoritmo usa un modelo matemático basado en la estadísticas y probabilidades, para simular el funcionamiento del de las neuronas cerebrales. Tiene una capa de entrada que es la encargada de recibir la información, luego la procesa en las capas oculta y por último se tiene la capa de salida, este modelo permite utilizar datos de entrenamiento y luego se evalúa con los datos de prueba para poder determinar si cumple con predecir la salida con respecto a los datos entregados.

- **Regresión lineal de Ridge:** Es otro modelo de regresión lineal La regresión lineal busca la optimización *w* y *b*, de manera que minimiza la función de costo.

- **Naïve-Bayes (NBC):** este algoritmo de clasificación está basado en análisis probabilístico simple y una suposición de independencia de los atributos. Este clasificador proporciona una buena precisión en conjuntos de pocos datos de entrenamiento en tiempo real.

- **KNeighbors Classifier:** este algoritmo de aprendizaje automático es simple, fácil de entender, aplicar y es uno de los que más avanzados y utilizado en diferentes tipos de aplicaciones como las de finanzas, salud, ciencias políticas y reconocimiento de imágenes.

- **DecisionTreeClassifier:** Es uno de los algoritmos más fáciles de interpretar, se inicia el proceso en el punto más alto del árbol, dónde se clasifican las observaciones y de las cuales se generan dos ramas, que están relacionadas por la misma región donde se el proceso de la división del predictor de manera sucesiva busca la mejor división en cada uno de los pasos.

- **RandomForestClassifier:** es un algoritmo aleatorio que de aprendizaje supervisado y se utiliza en modelos de clasificación, regresión y es fácil de utilizar, está compuesto de árboles que le dan su robustez en la medida que existan más árboles. Se obtienen

predicciones por cada uno de los árboles y se realiza la selección por votación de la mejor solución.

- **Support Vector Machines**: Este algoritmo tiene una precisión muy buena cuando se realiza la comparación con otros clasificadores. Maneja la posibilidad de utilizar el kernel para el manejar de los datos no lineales, se utiliza en aplicaciones como detección de rostros, detección de intrusiones, clasificación de correos electrónicos, artículos de noticias y páginas web, clasificación de genes y reconocimiento de escritura a mano.

- **MLPClassifier:** Este algoritmo se maneja mejor con datos normalizados por que se entrena de forma iterativa y para cada uno de los pasos de tiempo calculas las derivadas parciales de la función de perdida teniendo como referencia los parámetros del modelo para actualizar los parámetros.

5. Resultados

5.1. ¿Como poder estimar si los juegos generados para cada tratamiento de pacientes tienen características similares?

5.1.1. Estrategia con aprendizaje no supervisado

Aplicando el algoritmo de K.-medias se pretende poder identificar si los datos se agrupan siguiendo el patrón de la Categoría de los juegos (atención sostenida, selectiva, dividida) para lo cual se determina el número de clúster de 3. Primero utilizamos para cálculo del número de grupos la construcción del *Elbow curve*, que se pude ver en la Figura 26, la cual nos permite identificar que el número óptimo de grupos es 3, Con esta prueba confirmamos nuestra hipótesis inicial de los números de grupos a utilizar como 3.

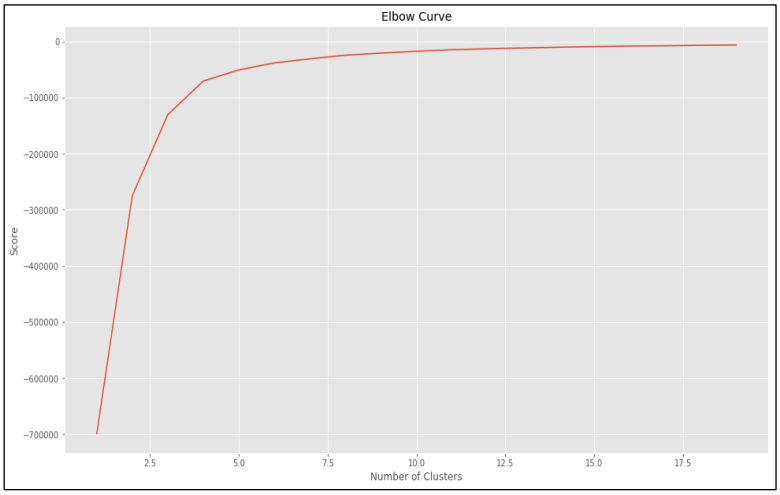

FIGURA 26. FIGURA DE ELBOW CURVE

Se obtienen los 3 clúster para las 6 variables, los datos obtenidos para las etiquetas y los centroides se muestran en la Tabla 9.

TABLA 9. CENTROS PARA LAS SEIS VARIABLES EN LOS TRES GRUPOS

	Ubicación variable1	Ubicación variable2	Ubicación variable3	Ubicación variable4	Ubicación variable5	Ubicación varible6
Grupo 1	4.611.764.706	2.827.774	229.411.765	2.017.647.059	342.197.172	389.165.029
Grupo 2	509.090.909	4.385.050.091	6.5	613.636.364	23.714.102	276.785.102
Grupo 3	24.52	13.976.203	2.5	18.24	300.229.861	301.643.572

En el Figura 27, con las variables Total_time y Test, se identifican de forma clara los tres agrupamientos de las variables y un círculo para poder identificar el centroide de cada uno de los clústeres.

- **El primer clúster** está formado por círculos que corresponden a la función cognitiva de atención sostenida
- **El segundo clúster** está formado por círculos que corresponden a juegos de atención sostenida, estrellas que corresponden a juegos de atención selectiva y triángulos que corresponden a juegos de atención dividida.
- **El tercer clúster** está formado por círculos que corresponden a juegos de atención sostenida, estrellas que corresponden a juegos de atención selectiva y triángulos que corresponden a juegos de atención dividida.

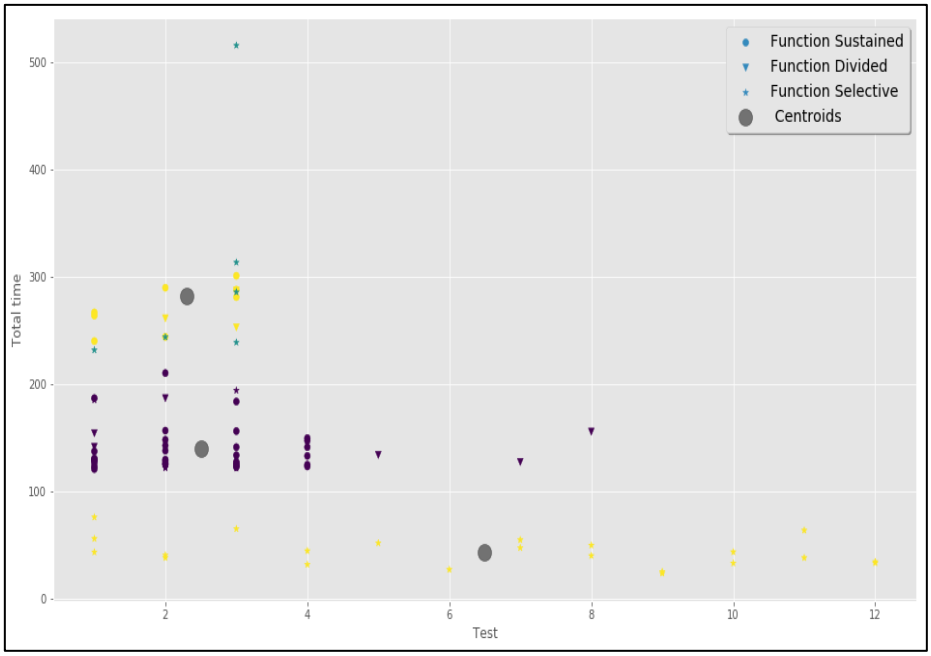

FIGURA 27. REPRESENTACIÓN GRÁFICA DE LOS CLÚSTER

Se muestra el Figura 28, que corresponde a la matriz de confusión que dio como resulta al ser aplicada a los datos que dieron como resultado del algoritmo de clasificación *K-means* para las 6 variables seleccionadas, con la cual podemos evaluar el modelo de clasificación ejecutado, donde se puede observar que la diagonal principal tiene un número de 58 registros, de un total de 86 lo cual permite clasificar de forma exitosa el 67% de los datos.

- Para el grupo 0 se clasificaron correctamente 22 registros.
- Para el grupo 1 se clasificaron correctamente 3 registros.
- Para el grupo 2 se clasificaron correctamente 33 registros.

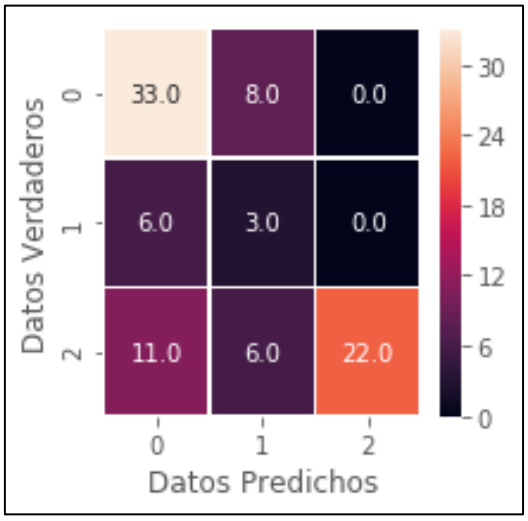

FIGURA 28. MATRIZ DE CONFUSIÓN PARA LAS 6 VARIABLES

Sin embargo, se aplicará el método de reducción de variables PCA, con el propósito de conformar los datos obtenidos con la totalidad de la información. Este método se basa en el análisis de los componentes principales de todas las variables el cual sirve para reducir el ruido y poder realizar una mejor visualización de los datos, ya que reduce el número de variables a dos y los gráficos son más sencillos de graficar.

Se calcula la varianza por cada uno de los componentes con *explained_variance_ratio_* y luego se calcula la sumas acumuladas de dichas varianza para dibujar la Tabla 10 donde podemos identificar que el primer componente representa el 38% de la varianza, el segundo el 28%, el tercero el 14%, lo cual nos indica que el 80% de la información clasifica está contenida en los tres primeros componentes. Este 80% que se muestra en la Figura 29 permite establecer que la variabilidad está contenida en las primeras tres variables. Resultado de la matriz de relación de varianza Tabla 10 de cada uno de los componentes principales

TABLA 10. MATRIZ DE RELACIÓN DE VARIANZA CON PCA

Matriz de relación de varianza					
Componente 1	**Componente 2**	**Componente 3**	**Componente 4**	**Componente 5**	**Componente 6**
38%	28%	14%	9%	8%	2%

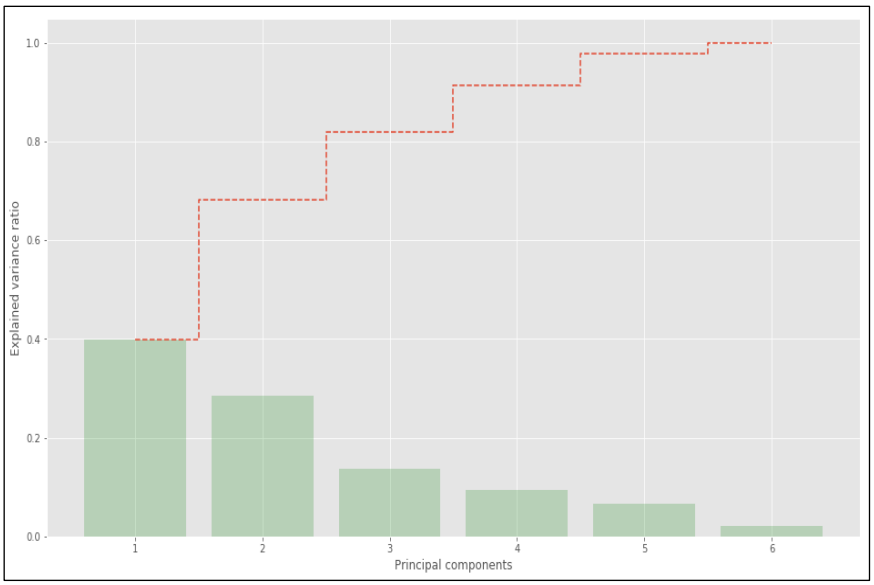

FIGURA 29. RELACIÓN ENTRE COMPONTES PRINCIPALES Y VARIABLES EXPLICADAS

En la Figura 30 muestra los tres clústeres, donde se pueden identificar que existe una mejor clasificación para cada uno de los grupos y son similares a los que se formaron cuando se utilizaron todas las variables.

- **El primer clúster** está formado por círculos de color morado que corresponden a la función cognitiva de atención sostenida
- **El segundo clúster** está formado por estrellas que corresponden a juegos de atención selectiva y algunos círculos que corresponden a juegos de atención sostenida.
- **El tercer clúster** está formado por triángulos invertidos los cuales representan los datos de los juegos de atención sostenida, solo se cuenta con información de un solo juego el cual no es significativa.

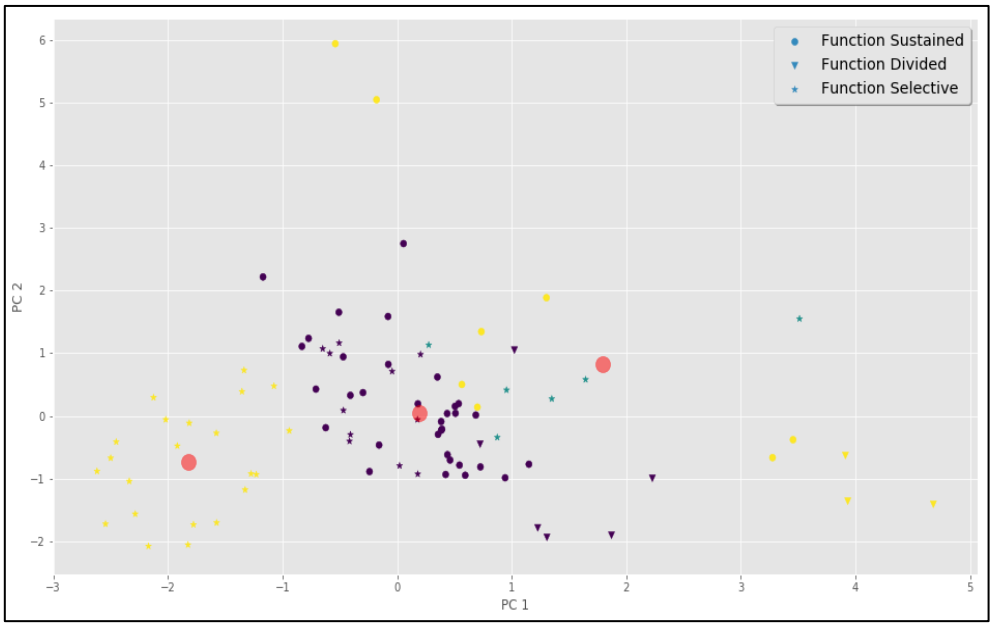

FIGURA 30. AGRUPAMIENTO RESULTANTE VISTOS EN LOS DOS MOMENTOS PRINCIPALES QUE REPRESENTAN EL 66% DE LA VARIABILIDAD DE LOS DATOS.

5.1.2. Estrategia con aprendizaje supervisado

Se realiza un segundo ejercicio de clasificación, pero esta vez se utilizan algoritmos de clasificación supervisados y se realiza una división de los datos, del 70% por ciento para entrenamiento y el 30% por ciento para pruebas, los algoritmos seleccionados para este trabajo fueron: *naive bayes, k-neighbors, decision tree, random forest, logistic, svm radial, best MLP*.

Se creo una función (Clasificadores) para que realizara la evaluación de los diferentes algoritmos a los cual se le pasaron los datos de entrenamiento y tes según el caso para cada uno de los

Para sklearn.naive_bayes.GaussianNB(), se utilizó el algoritmo gaussianos.

Para KNeighborsClassifier(n_neighbors=3), se utilizó el algoritmo de Classifier

Para DecisionTreeClassifier(max_depth=3), se utilizó el algoritmo de Classifier

Para RandomForestClassifier(max_depth=3), se utilizó el algoritmo de Classifier

Para LogisticRegression(), , se utilizó el algoritmo deRegression

Para la VSM.SVC(C= 10, gamma = 0.0001, kernel='rbf'), se tuvieron en cuenta los siguientes valores para los hiperparametros, los cuales dieron el mejor valor en la predicción:

C: penalización del término de error. Controla la compensación entre el límite de decisión suave y la clasificación correcta de los puntos de entrenamiento.

Gamma: ajustarse exactamente al conjunto de datos de entrenamiento

Kernel: El uso de 'lineal' usará un hiperplano lineal (una línea en el caso de datos 2D). 'rbf' y 'poly' usan un hiperplano no lineal.

MLPClassifier (activation = 'relu', hidden_layer_sizes = (100,), learning_rate = 'adaptive', learning_rate_init = 0.0001, max_iter = 10000, solver = 'lbfgs'),].

Se entrenaron los algoritmos de clasificación anteriormente mencionados, con el conjunto de datos de entrenamiento que corresponden al 70%. Luego se realizó la evaluación del resultado de los modelos, creando un vector para almacenar los resultados obtenidos de cada modelo al realizar validación cruzada, el resumen de este proceso se puede ver en la Tabla 11, que corresponde a la exactitud alcanzada, por cada uno de los algoritmos, se utilizó la estrategia de validación cruzada con un *k-folds* de 3.

TABLA 11. DESCRIPCIÓN DEL RESULTADO DE LA EXACTITUD EN LOS MODELOS DE PREDICCIÓN SUPERVISADOS

Modelos	***Accuracy:***
Naive bayes	0.68 (+/- 0.10)
k-neighbors	0.71 (+/- 0.06)
Decision tree	0.68 (+/- 0.07)
Random Forest	0.68 (+/- 0.11)
Logistic	0.68 (+/- 0.04)
Svm radial	0.76 (+/- 0.06)
Best MLP	0.74 (+/- 0.22)

El mejor algoritmo de clasificación, que fue clasificado con el set de datos de train, utilizando el método de evaluación de validación cruzada, fue el algoritmo de SVM radial, con una exactitud del 76%. Luego se realiza la evaluación del algoritmo de SVM, con el set de datos de *tests* y se obtuvo una exactitud del 70%.

Realizamos una comparación de las matrices de confusión, que se generan con los datos de prueba y los datos totales el cual muestra los siguientes resultados en las Gráficos 31 y 32.

- **Para el grupo 0** con los datos de test realizo la clasificación del 30% bien y con el total de los datos el 50% bien.
- **Para el grupo 1** con los datos de test realizo la clasificación el l% bien y con el total de los datos el 1 bien%.
- **Para el grupo 2** con los datos de test realizo la clasificación del 30% bien y con el total de los datos el 40% bien.

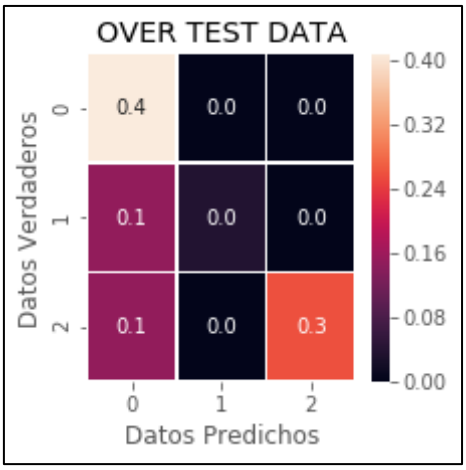

FIGURA 31. MATRIZ DE CONFUSIÓN CON DATOS DE TEST

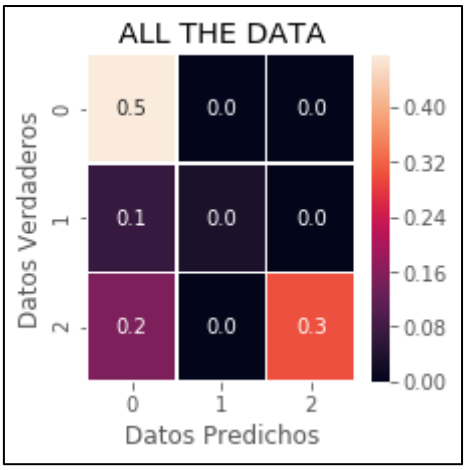

FIGURA 32. MATRIZ DE CONFUSIÓN CON EL TOTAL DE LOS DATOS

También se tuvieron en cuenta las métricas de precisión para el algoritmo de SVM, como las puntuaciones de precisión, recuperación, F1 y soporte como se puede ver en la Tabla 12

TABLA 12. MÉTRICAS DE PRECISIÓN DE LOS DATOS DE PRUEBA ALGORITMO DE SVM

Grupo	precision	recall	f1-score	support
0	0.58	1	0.73	11
1	1	0.2	0.33	5
2	1	0.64	0.78	11

Se concluye que si existe patrones que permiten relacionar los juegos y el trastorno para el cual fueron desarrollados, sin embragó se pudo identificar que para el caso de los juegos desarrollados opción 1 trastorno de atención dividida, solo se tiene información de un solo juego lo cual no es significativo.

5.2. Evaluación de los resultados

Con los datos obtenidos por los modelos anteriormente utilizados se procede a realizar la evaluación de los modelos y poder contestar las preguntas de investigación:

La primera pregunta del trabajo se contestó al realizar la analítica descriptiva de los datos donde se pudo evaluar el desempeño de los jugadores de un juego a otro al igual que la facilidad de aprendizaje.

La segunda pregunta del trabajo se desarrollaron dos métodos: uno con un modelo de aprendizaje no supervisado que utilizó el método de segmentación con el algoritmo de *K-meas*, el cual permitió evaluar si los juegos desarrollados como soporte terapéutico para cada uno de los

tipos de los tres tipos de atención se agrupaban según sus características y patrones que se encontrar en los datos.

Para completar este proceso se realizó la aplicación de la técnica de reducción de variables o PCA, que permitió trabajar un menor número variables para nuestro caso dos, se realizó una comparación de resultados obtenidos, donde se obtuvieron las tres agrupaciones para el caso de PCA se encontró un mejor agrupamiento con la clasificación del 66% de las variables.
También se realizó la aplicación de algoritmos de clasificación supervisados, para los cuales se realizado la implementación de 7 algoritmos, se encontró que el mejor clasificador fue el SVM radial (*support vector Machine*), el cual tiene una exactitud: *Accuracy*: 0.73 (+/- 0.05).

Se encontró que el mejor método para realizar la clasificación es el modelo de clasificación supervisado SVM radial (*support vector Machine*) con un 73% de acierto.

6. Conclusiones y recomendaciones

En este estudio se aplicaron las fases de la construcción de un proyecto de aprendizaje automático, lo que permitió terminar a satisfacción el presente. El cual tenía como objetivo proponer un modelo de análisis de datos utilizando técnicas de aprendizaje supervisado y no supervisado, para identificar patrones en la información generada por los pacientes, quienes son sometidos a juegos diseñados como un instrumento de apoyo terapéutico.

A partir de lo anterior se encontró que con algunas variables se identifican los juegos, como también se identificó la necesidad de crear nuevas variables para la unificación de datos, ya que los juegos no tienen las mismas variables. Además, se identificaron características y patrones en los datos que permitieron comprobar la facilidad de usar y recordar los objetivos de los juegos.

Con lo anterior y en conjunto de la depuración de datos, se generó un archivo para la aplicación de las técnicas de aprendizaje automático de agrupamiento, clasificación y regresión. Como también con la aplicación de las técnicas de agrupamiento se encontró que los juegos si tienen características en común que hacen que se agrupen según el tipo de atención para el que fueron creados. Por medio de los métodos de clasificación se identificaron el mismo comportamiento en los datos con el fin de ser clasificados en cada uno de las categorías de los juegos del tipo de atención. Con los métodos de regresión no se pudo identificar resultados satisfactorios porque no se contó con la cantidad suficiente de información.

A manera de recomendación se sugiere que se utilicen los datos suficientes para el correcto desarrollo de los juegos, ya que esto permite una correcta aplicación de los modelos. Además, se recomienda que, para incluir el módulo de aprendizaje automático al sistema de juegos se realice una verificación de la información que se desea obtener.

7. Trabajos futuros

En el presente trabajo no se contempló la realización de la implementación e incorporación al sistema de juegos, sino que esta debe ser incluidas en las nuevas fases del proyecto.

Con la información existente se plantea la posibilidad de implementación de nuevos modelos que permitan poder predecir la cantidad de espacio a ser ocupado por la información que se produce de la interacción de los pacientes con los juegos. Se realizaron pruebas con algoritmos de regresión, para estimar la cantidad de espacio a ser utilizado, con la información del número de intentos realizados por un paciente para conseguir una cantidad determinada de aciertos, en un tiempo específico. Se crea un anexo con los resultados obtenidos hasta el momento, para que se continúe en próximos trabajos donde se pueda contar con más información de los juegos.

Bibliografía

[1] S. B. H. G. K. &. M. C. Rios, «https://dialnet.unirioja.es/servlet/articulo?codigo=4897626,» *Revista Vanguardia Psicológica Clínica Teórica y Práctica,* vol. 5, nº 1, pp. 22-31, 2014.

[2] M. D. H. D. B. &. L. D. W. (. Lezak, Neuropsychological assessment., New York: Oxford Univer. Press. E., & JAckuns, 1995.

[3] E. M. D.-M. J. P. &. C. S. J. Arroyo-Anlló, «Técnicas de rehabilitación neuropsicológica en demencias:,» *Pensamiento psicológico,* vol. 10, nº 1, pp. 107-127, 2012.

[4] M. D. T. M. L. &. F.-B. R. Zamarrón Cassinello, «Plasticidad cognitiva en personas con la enfermedad de Alzheimer que reciben programas de estimulación cognitiva,» *Psicothema,* vol. 20, nº 3, 2008.

[5] S. &. F. M. Giaquinto, «computerized cognitive remediation: First results.,» *Acta neurologica,* 1992.

[6] F. &. P. L. T. D. Ricciardi, «A comprehensive review of serious games in health professions. International Journal of Computer Games Technology, 2014,,» *International Journal of Computer Games Technology,* vol. 2014, 2014.

[7] B. G. Stokes, «Videogames have changed: time to considerserious games'? ,» *Development Education Journal,* vol. 11, nº 3, p. 12, 2005.

[8] T. D. (. Parsons, «Virtual teacher and classroom for assessment of neurodevelopmental disorders. In Technologies of inclusive well-being ,» [En línea]. Available: https://link.springer.com/chapter/10.1007/978-3-642-45432-5_7.

[9] J. E. B. M. F. J. H. K. &. G.-B. P. Deutsch, «Use of a low-cost, commercially available gaming console (Wii) for rehabilitation of an adolescent with cerebral palsy,» *Physical therapy,* vol. 88, nº 10, pp. 1196-1207, 2008.

[10] K. F. J. A. M. &. L. Zettergren, «The effects of Nintendo Wii Fit training on gait speed, balance, functional mobility and depression in one person with Parkinsons disease,» *Applied Innovations and Technologies,* vol. 5, nº 2, pp. 38-44, 2011.

[11] B. P. H. &. B. Y. Kim, «Not just fun, but serious strategies: Using meta-cognitive strategies in game-based learning,» *Computers & Education,* vol. 52, nº 4, pp. 800-810, 2009.

[12] M. Cajamarca, «Inteligencia Artificial, Aprendizaje Automático y Aprendizaje Profundo,» 2018. [En línea]. Available: https://planetachatbot.com/inteligencia-artificial-aprendizaje-autom%C3%A1tico-y-aprendizaje-profundo-862ca9790bb9.

[13] J. L. Gonzalez, 2 2018. [En línea]. Available: https://medium.com/soldai/tipos-de-aprendizaje-autom%C3%A1tico-6413e3c615e2.

[14] Pértega Díaz, S., Pita Fernández, S.,, «Fisterra,» 20 08 2001. [En línea]. Available: https://www.fisterra.com/formacion/metodologia-investigacion/tecnicas-regresion-regresion-lineal-simple/.

[15] V. Y. Piqueras, «victoryepes.blogs.upv.es,» 07 01 2017. [En línea]. Available: https://victoryepes.blogs.upv.es/2017/01/07/que-es-y-para-que-sirve-una-red-neuronal-artificial/.

[16] L. T. H. L. Payam Refaeilzadeh, «On comparison of feature selection algorithms,» *Proceedings of AAAI workshop on evaluation methods for machine learning II,* vol. 3, nº 4, p. 5, 07 2007.

[17] R. Caruana, «Rich Caruana's Home Page,» [En línea]. Available: https://www.cs.cornell.edu/~caruana/.

[18] scikit-learn@python.org. [En línea]. Available: https://scikit-learn.org/stable/modules/cross_validation.html.

[19] S. I. M. S. I. Ferrari Alve, «Guía de evaluación de la usabilidad para herramientas de minería de datos,» *No Solo Usabilidad,* vol. 13, 14 04 2014.

[20] Duk2, «ESTRATEGIAS DE TRADING,» [En línea]. Available: https://estrategiastrading.com/k-means/.

[21] U. Malik, «Implementing PCA in Python with Scikit-Learn,» 10 5 2018. [En línea]. Available: https://stackabuse.com/implementing-pca-in-python-with-scikit-learn/.

[22] D. L. L. R. K. M. G. M. &. P. N. Martínez, «Perfiles de rendimiento académico: un modelo basado en minería de datos.,» *Campus Virtuales,* vol. 4, nº 1, pp. 12 - 30, 2015.

[23] J. I. Bagnato, «APRENDE MACHINE LEARNING,» 03 12 2018. [En línea]. Available: http://www.aprendemachinelearning.com/k-means-en-python-paso-a-paso/.

Anexos

1. Descripción de términos

1. **Algoritmo:** Se entiende como una definición para matemáticas, estadística, probabilidades y ciencias de la computación, como una secuencia estructurada de sentencias que permiten realizar una tarrea siguiendo una serie de pasos que garantizan que se cumple con el objetivo.
2. **Inteligencia Artificial:** El termino se aplica a la simulación del proceso de la inteligencia humana que es realizado por las maquinas que utilizan sistemas informáticos. Dichos sistemas utilizan algoritmos para procesos de aprendizaje, razonamiento y autolesiones.
3. **Aprendizaje automático:** Es la parte de la IA, que se encarga de conseguir que una computadora realizase sus tareas sin programación. Exciten tres tipos de algoritmos de aprendizaje que son: supervisado, no supervisado y de refuerzo.
4. **Analítica de datos:** es una técnica que se utiliza para realizar un examen a los datos históricos y actuales que se encuentran almacenados en las empresas, con el fin de poder sacar conclusiones de la información procesada y poder identificar patrones y tendencias. El propósito general de las AD es poder permitir que las compañías y las organizaciones realicen la tomen de decisiones basada en la información extraída de los datos, la ciencia también utiliza la AD para la verificación de modelos o teorías.
5. **Técnicas de análisis de datos:** Es tan basadas en herramientas que son utilizadas para la organización, descripción y análisis de los datos que son recogidos con los instrumentos de investigación. El análisis de datos encierra dos procedimientos: La organización de los datos, la descripción y análisis de los datos
6. **Visualización de datos:** La visualización de datos es la encargada de la generación de informes sobre los resultados que se obtienen en el análisis de datos, que permitan identificar de forma gráfica las tendencias y propiedades de la información analizada.
7. **Bases de datos:** Es la forma en la cual se almacena la información y archivos, la cual debe ser organizada de tal forma que sea fácil de consultar por programas de computadoras. En la actualidad existen varios tipos de base de datos como: estructuradas, semi estructuradas y no estructuradas al igual que muchos motores para realizar su gestión.
8. **Metodologías:** Hace relación a la serie de reglas y pasos que deben seguirse para llevar a cabo un proyecto de minería de datos, existen varias metodologías para el análisis de datos, pero hay teres que son las más reconocida: KDD, CRISP-DM, SEMMA.
9. **Estadística:** Es una rama de las matemáticas que se encarga de estudiar la variabilidad y los procesos aleatorios que se generan siguiendo las leyes de la probabilidad, utilizando

las siguientes actividades como reunir, clasificar y recontar todos los hechos que tienen una característica común, que permitan llegar a conclusiones.

10. **Probabilidades:** Es la medida de incertidumbre que está asociada a un evento futuro que se expresa como un entero entre 0 y 1.
11. **Minería de datos:** (*Datamining*) Se refiere al proceso de extracción de patrones y tendencias en los datos que son generados por los programas de computador y dispositivos electrónicos, utilizando modelos matemáticos y estadísticas.

2. Descripción de la Metodología CRISP-DM

Para realizar el análisis de la información recolectada en la utilización de los video juegos, se debe utilizar una metodología que garantice el cumplimiento de todas las etapas del proyecto, para garantizar que se cumpla estos postulados se utilizara la metodología CRISP-DM (Chapman, 2015) [22], cuyas siglas significan *Cross-Industry Standar Process for Data Mining*. La cual proporciona la descripción del ciclo de vida del proyecto, que contiene las fases del proyecto, sus tareas, y las relaciones entre estas tareas.

Los niveles van desde lo general hasta lo particular, organizando el desarrollo del proyecto en una serie de seis fases según Figura 33 que son:

- Business Understanding (Comprensión del negocio)
- Data Understanding (Comprensión de los Datos)
- Data Preparation (Preparación de los Datos)
- Modeling (Modelado)
- Evaluation (Evaluación)
- Deployment (Implementación)

Cada fase se estructura en varias tareas generales de segundo nivel. Estas tareas se planean para labores específicas, describiendo las acciones que deben ser desarrolladas para cada uno de los casos, pero no son de carácter impositivo, son una guía práctica que el desarrollador del proyecto debe tener encuentra.

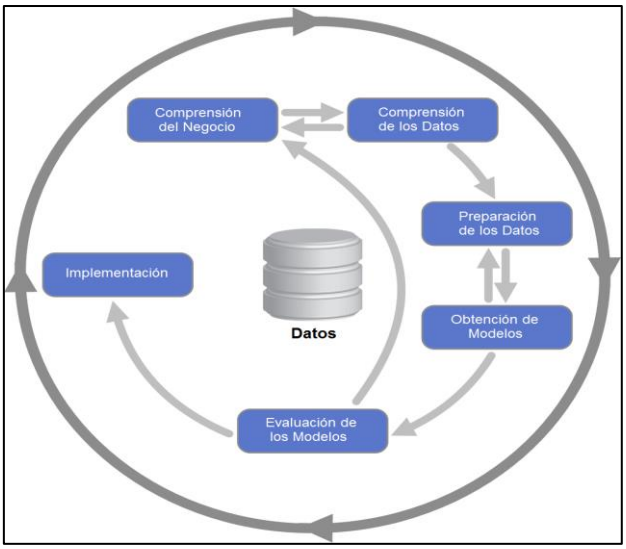

FIGURA 33. ESQUEMA DE LAS FASES DE CRISP-DM

Data understanding (Comprensión del Negocio)

Es la primera fase del proyecto donde se definirá los objetivos, se evaluará el estado actual, como se aplica la minería de datos a los objetivos y la elaboración del plan del proyecto como se ve en la Figura 34:

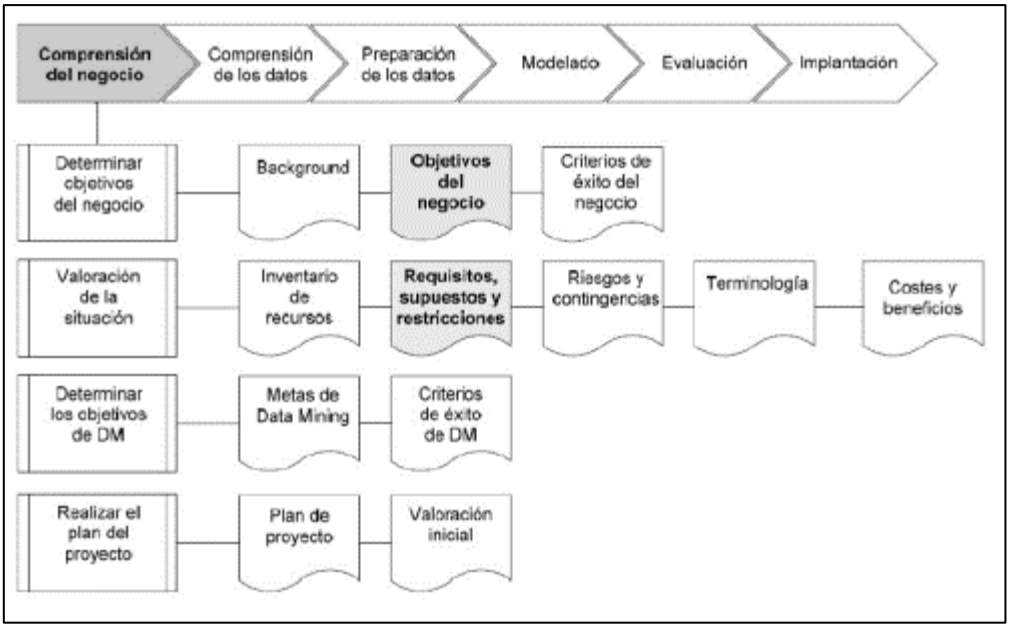

FIGURA 34. PRIMERA FASE DEL MODELO CRISP-DM

Determinar los objetivos del negocio: se determina cuál es el problema que queremos resolver, porqué utilizamos aprendizaje automático para cumplir nuestro propósito y definir los criterios de éxito.

Evaluar la situación actual: En esta tarea debe evaluar antecedentes y requisitos del problema, tanto en términos del negocio como en términos del aprendizaje automático. Se deben tener en cuenta la cantidad de datos a trabajar, ventajas de aplicar aprendizaje automático al problema.
Determinar los objetivos del aprendizaje automático: se deben representar los objetivos del negocio en términos de las metas del proyecto.

Producir un plan de proyecto: Se generar un plan de proyecto, que considere los pasos a seguir y los métodos a utilizar para cada paso.

Data understanding (Comprensión de los datos)

En esta fase inicia con la recolección de datos, describiéndolos, explorándolos y verificando su calidad, las tareas se muestran en la Figura 35:

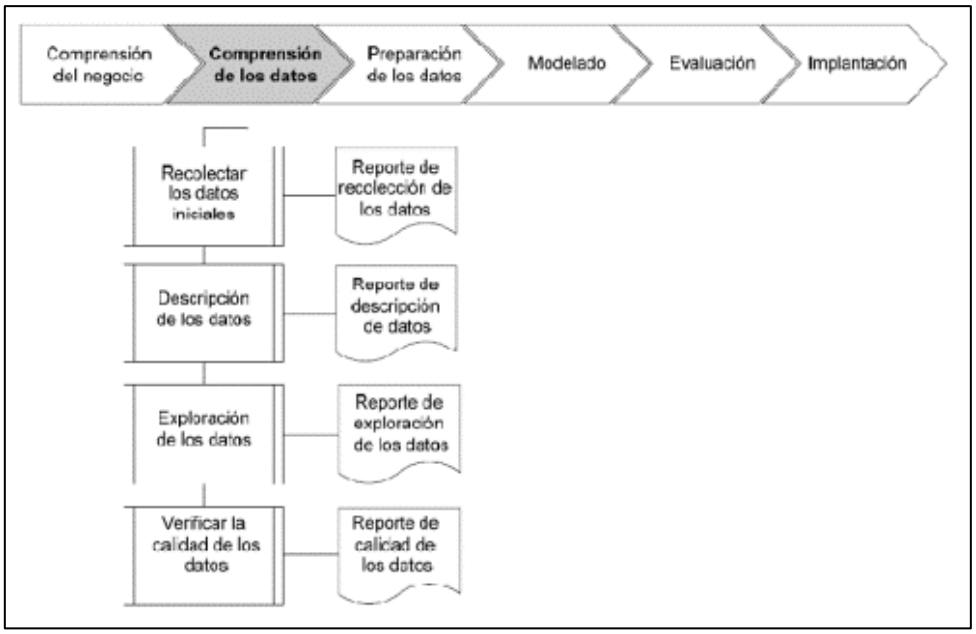

FIGURA 35. SEGUNDA FASE DEL MODELO CRISP-DM

Recolectar datos iniciales: Su objetivo es la recolección de datos y la adecuación para su procesamiento. Se debe elaborar informes con una lista de los datos adquiridos, su localización, las técnicas utilizadas en su recolección y los problemas y soluciones inherentes a este proceso.
Describir los datos: se realiza la una descripción de los datos obtenidos, como número de registros y campos por registro, su identificación, el significado de cada campo y la descripción del formato inicial.

Explorar los datos: Se identifica la estructura general para los datos. Se deben aplicar pruebas estadísticas básicas, que identifican las propiedades de los datos, creando tablas de frecuencia y gráficos de distribución. Se debe generar un informe de exploración de datos.

Verificar la calidad de los datos: debemos determinar en los datos, la consistencia de los valores de los campos, la cantidad y distribución de los valores nulos, encontrar valores fuera de rango que pueden ser ruido para el proceso. cuyo objetivo es asegurar la completitud y corrección de los datos.

Data preparation (Preparación de los datos)

El objetivo de esta fase es la selección de los datos, limpieza, estructuración, integración y el formato de datos según se muestra en la Figura 36 :

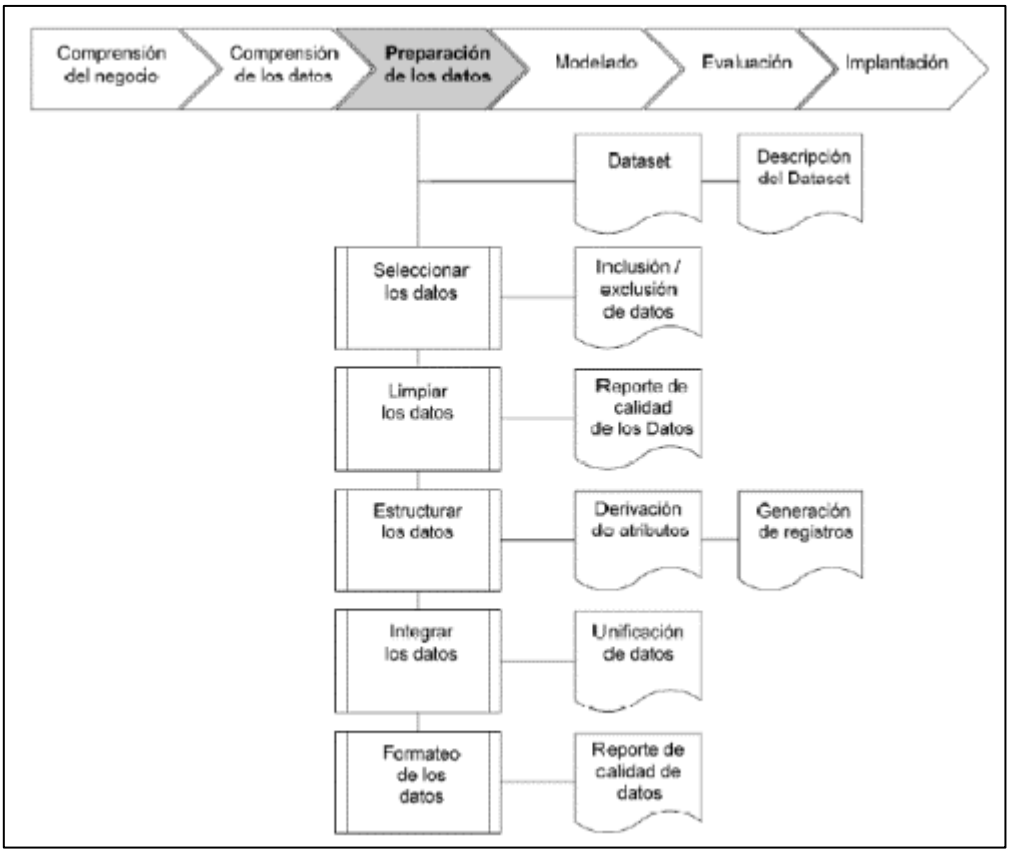

FIGURA 36. TERCERA FASE DEL MODELOS CRISP-DM

Seleccionar los datos: seleccionamos un subconjunto de datos que garanticen su calidad, las limitaciones en volumen y en los tipos de datos están determinadas por las técnicas de aprendizaje automático que se seleccionen.

Limpiar los datos: para realizar esta tarea se pueden utilizar diversas técnicas con el propósito de garantizar la calidad de los datos para la fase de modelación. se pueden utilizar las técnicas de: normalización de los datos, discretización de campos numéricos, tratamiento con valores vacíos, reducción del volumen de datos, etc.

Estructurar los datos: Las tareas a realizar son: la generación de nuevos atributos a partir de atributos ya existentes, integración de nuevos registros o transformación de valores para atributos existentes.

Integrar los datos: las tareas son: creación de nuevas estructuras, por ejemplo, crear nuevos campos, nuevos registros, fusión de tablas o crear nuevas tablas.

Formatear los datos: Consiste en la transformación sintáctica de los datos, sin modificar su significado, para permitir o facilitar el empleo de alguna técnica de aprendizaje automático en particular. Por ejemplo, eliminar comas, tabuladores, caracteres especiales, espacios, máximos y mínimos para las cadenas de caracteres, etc.

Modeling (Modelado)

El objetivo de esta fase es la selección técnica de modelado, generar plan de pruebas, construcción del modelo y su evaluación como se identifica en la Figura 37:

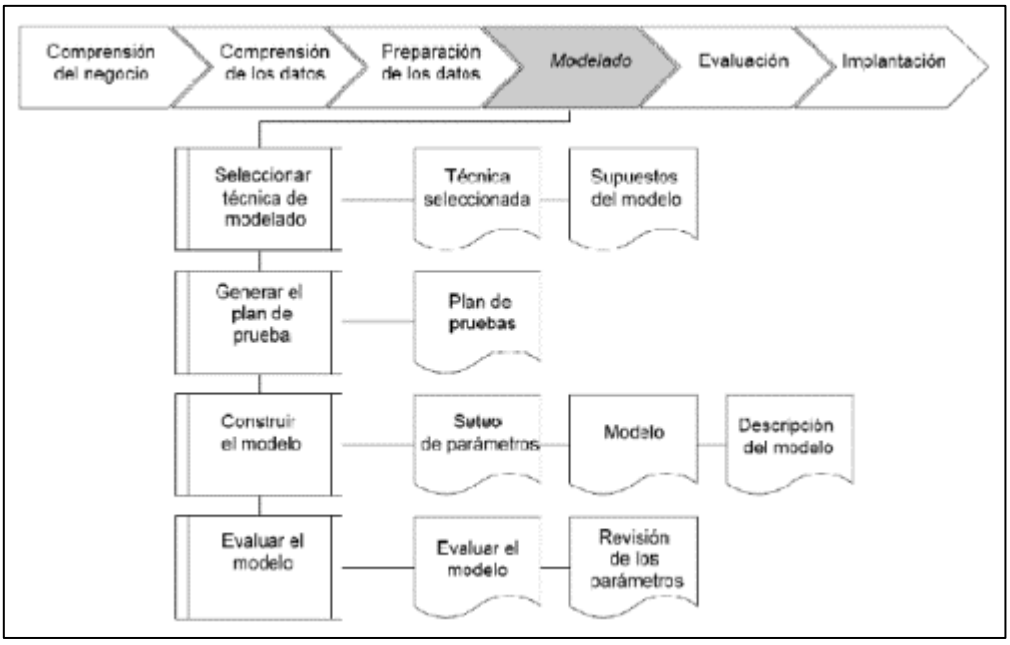

FIGURA 37. CUARTO FASE DEL MODELO CRISP-DM

Seleccionar la técnica del modelado: Debemos elegir la técnica de aprendizaje automático más apropiado para el proyecto. Para elegir las técnicas de aprendizaje automático se pueden tener encuentra los siguientes criterios:

- Ser apropiada al problema
- Disponer de los datos adecuados
- Cumplir requisitos del problema

- Tiempo adecuado para obtener un modelo
- Conocimiento de la técnica

Ejemplo: para la utilización del aprendizaje automático se tiene las opciones de dos modelos como son: Aprendizaje automático supervisado, que pueden ser, regresión lineal, *Artificial neural networks*. Y aprendizaje automático no supervisado que puede ser: K-medias (K-means).
Generar plan de prueba: El plan de pruebas debe garantizar la calidad y validez del modelo construido. Por ejemplo, para tareas de clasificación es posible usar la razón de error para medir la calidad. Se deben separan los datos en dos conjuntos, uno de entrenamiento y otro de prueba.
Construir el modelo: se debe aplicar la técnica seleccionada, a los datos preparados para la generación de uno o más modelos. Todas las técnicas del modelado tienen un conjunto de parámetros que determinan características del modelo a generar. La tarea de selección de los mejores parámetros es iterativa basado en los resultados generados. Estos deben ser interpretados y su rendimiento justificado.

Evaluar el modelo: La interpretación de los modelos se debe realizar de acuerdo con el conocimiento del dominio y los criterios de éxitos preestablecidos.

Evaluation (Evaluación)

El objetivo de esta fase es evaluar los resultados, revisión del proceso y determinar los próximos pasos como se muestra en la Figura 38:

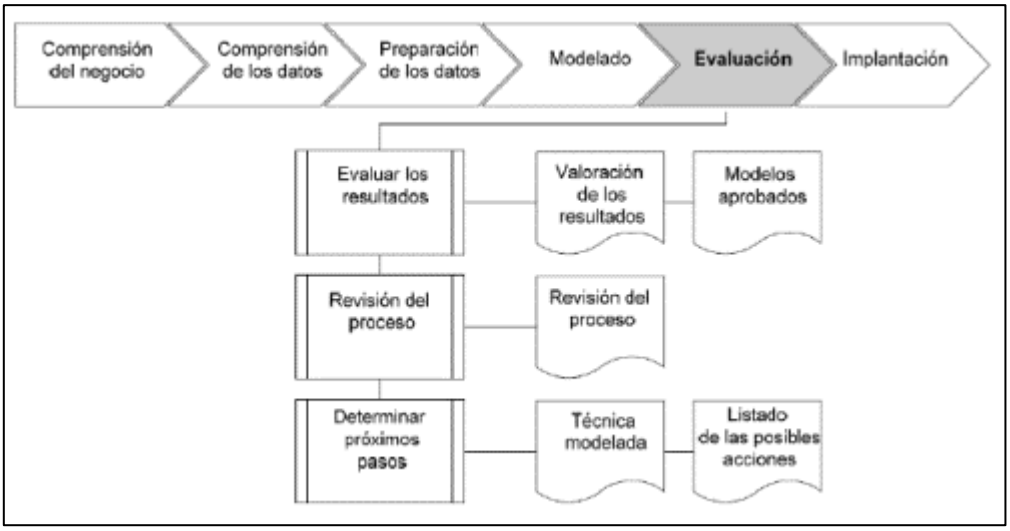

FIGURA 38. QUINTA FASE DEL MODELOS CRISP-DM

Evaluar los resultados: En esta tarea se evalúan los modelos con relación a los objetivos del negocio y se busca determinar si los modelos cumplen y ayudan a responder alguna las preguntas del problema planteado, para lo cual el modelo fue desarrollado.

Revisar el proceso: Esta tarea evaluar el proceso completo de aprendizaje automático, para identificar los elementos que pueden ser mejorados.

Determinar próximos pasos: En esta tarea se debe identificar los casos que no han generado resultados satisfactorios, para realizar nuevas iteraciones desde la fase de preparación de datos o para realizar nuevos ajustes a los modelos.

Deployment (Implementación)

El objetivo de esta fase es realizar la planeación, para la Implementación del proyecto, realizar el plan de Monitoreo y Mantenimiento y realizar el informe final como se muestra en l Figura 39:

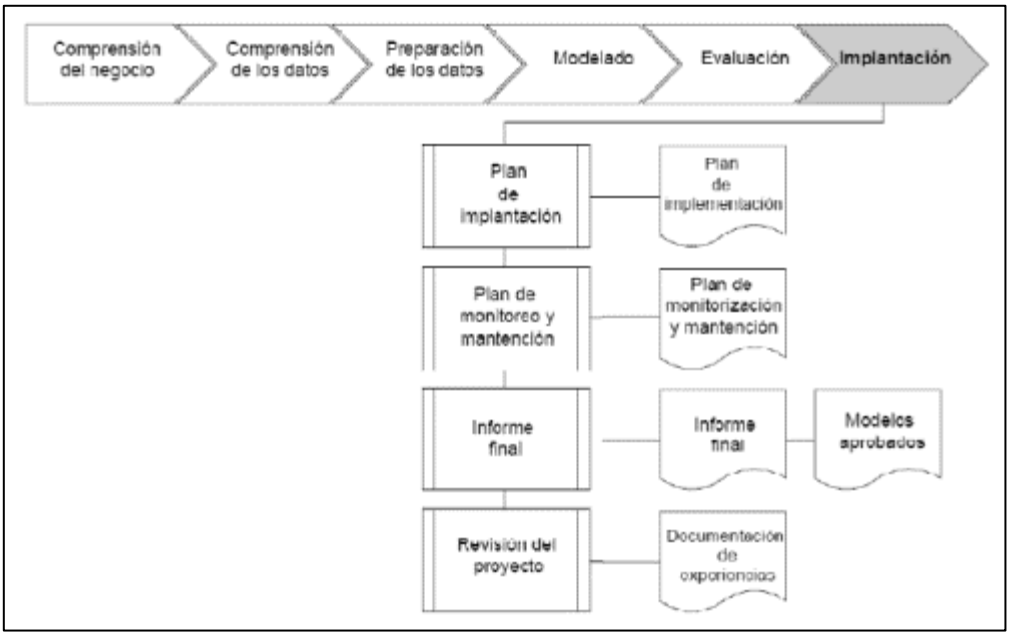

FIGURA 39. SEXTA FASE DEL MODELO CRISP-DM

Planear la implementación: En esta etapa se evalúan los resultados obtenidos y se crea una estrategia para su implementación. Todo procedimiento que se ha utilizado en la creación del modelo debe ser documentados para su implementación.

Monitorizar y mantener: Se deben crear estrategias de monitorización y mantenimiento para ser aplicadas a los modelos Implementados.

Informe final: Con las dos etapas anteriores cumplidas se debe realizar un documento resumen del proyecto y las experiencias logradas o también puede ser una presentación final explique los resultados y logrados obtenidos en proyecto.

3. Aplicación de modelos de regresión para determinar espacio para utilizar por juego

Se realizaron pruebas para realizar la implementación de algoritmos de regresión para ser aplicados a los datos de entrenamiento que corresponden al 70% y de prueba del 30%, para lo cual se utilizó la librería de aprendizaje automática *Scikit-Learn* de Python. Y se realizan las siguientes tareas:

- Se realiza la selección de las variables a ser utilizadas del total que tiene el archivo: la variable dependiente es 'Total tests' y las independientes son 'Total hits', 'Total time'. A los datos seleccionados se les realiza una verificación, para ser utilizados por los algoritmos de aprendizaje automático supervisado y se realiza la descripción sus valores estadísticos en la Tabla 13:

TABLA 13. DESCRIPCIÓN ESTADÍSTICA DE LAS VARIABLES A EVALUAR EN LOS MODELOS

	User	Test	Tiempo Total	Aciertos Totales	Game	Ensayo_total
Count	89.000000	89.000000	89.000000	89.000000	89.000000	89.000000
mean	9.303371	3.449438	143.371105	15.617978	4.067416	23.842697
std	2.263367	2.808312	85.487528	15.106821	1.929424	19.574972
min	4.000000	1.000000	23.782820	0.000000	1.000000	3.000000
25%	8.000000	2.000000	120.953800	5.000000	3.000000	10.000000
50%	9.000000	3.000000	129.100900	10.000000	4.000000	20.000000
75%	11.000000	4.000000	183.848100	23.000000	6.000000	30.000000
max	13.000000	12.000000	515.760200	77.000000	7.000000	98.000000

- Como siguiente paso se crean los data set de entrenamiento y pruebas, utilizando la función *train_test_split*() de Python, con el 70% para entrenamiento y el 30% de prueba.
- Se crea el objeto tipo regresión que se quiera utilizar.
- Se entrenan los modelos con el conjunto de datos de entrenamiento.
- Después de tener el algoritmo entrenado procedemos a calcular el coeficiente de regresión y las validaciones a los modelos.

Se realiza el entrenamiento de los algoritmos de regresión que fueron seleccionados (Regresión lineal, ridge 0.7, MLP, SVM) con el 70% de los datos, luego se realiza la aplicación del modelo de evaluación que para este caso se utilizó validación cruzada y se obtuvieron los siguientes resultados que se muestran en la Tabla 14. Resultado de validación cruzada para algoritmos de regresión con k-fold donde el algoritmo que presenta el mejor valor de exactitud es el algoritmo de ridge 0.7 con un valor del 62%, std de 0.19 y mean 0.62.

TABLA 14. RESULTADO DE VALIDACIÓN CRUZADA PARA ALGORITMOS DE REGRESIÓN CON K-FOLD

Modelos	mean	std	Accuracy
model lineal	0.62	0.19	0.62 (+/- 0.39)
model ridge 0.7	0.62	0.19	0.62 (+/- 0.39)
model MLP	0.21	0.37	0.21 (+/- 0.74)
model SVM	0.46	0.17	0.46 (+/- 0.34)

Y se completó la evaluación del algoritmo al cuantificar y evaluar la calidad de las métricas que miden la distancia entre el modelo y los datos según Tabla 15.

TABLA 15. CUANTIFICACIÓN DE LAS MÉTRICAS DEL MODELO

Regression ridge 0.7:		
regressor score	0.42	
Mean squared error	252.73	mejor valor 0.0 +, no es bueno
mean_squared_error	252.73	mejor valor 0.0 +, no es bueno
mean_squared_log_error	0.27	mejor valor 0.0 +, es bueno
mean_absolute_error	8.13	el mejor valor es 0.0 +, no es muy bueno
median_absolute_error	4.17	el mejor valor es 0.0 +, no es muy bueno
explained_variance_score	0.51	mejor es 1 más bajos peores, es un valor aceptable
r ^ 2 score	0.42	mejor es 1 más bajos peores, es un valor malo

```
recalcularon los valores para el:
Intercepto 0.6609977543044454
Regresor   0.0
```

Con esta modelo podemos realizar una buena predicción de la cantidad de ensayos que se realizan en un determinado tiempo para cumplir con un numero de aciertos dados.

En la Figura 40 se muestra la distribución de los datos que son generados, con los datos de entrenamiento y pruebas, también se dibuja el regresor del algoritmo seleccionado en el punto anterior que es **ridge 0.7,** y se verifico la relación existente entre los datos de test, train y la forma de la recta del regresor.

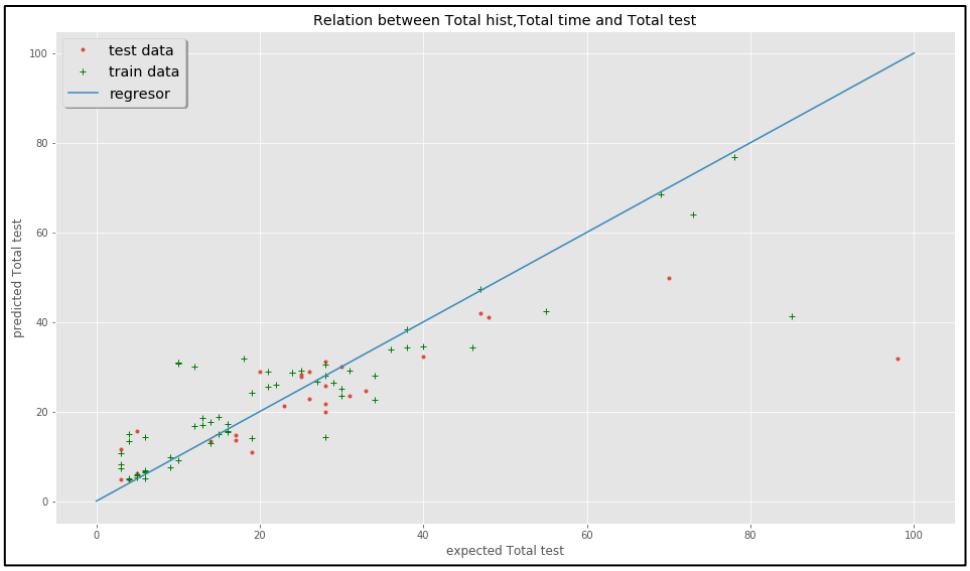

FIGURA 40. RELACIÓN DE DATOS DE ENTRENAMIENTO Y PRUEBAS MOSTRADOS EL IDEAL, EL CUAL ES CALCULADO POR MÉTODO DE RIDGE 0.7

En la Figura 41. se muestra la relación existe entre los resultados que se obtienen con los datos de prueba del algoritmo de ridge 0.7, y los datos que son predichos por el modelo. En la gráfica podemos establecer que todo se encuentran relacionados entres si las variables son evaluadas en este algoritmo son Total test y Total Hits.

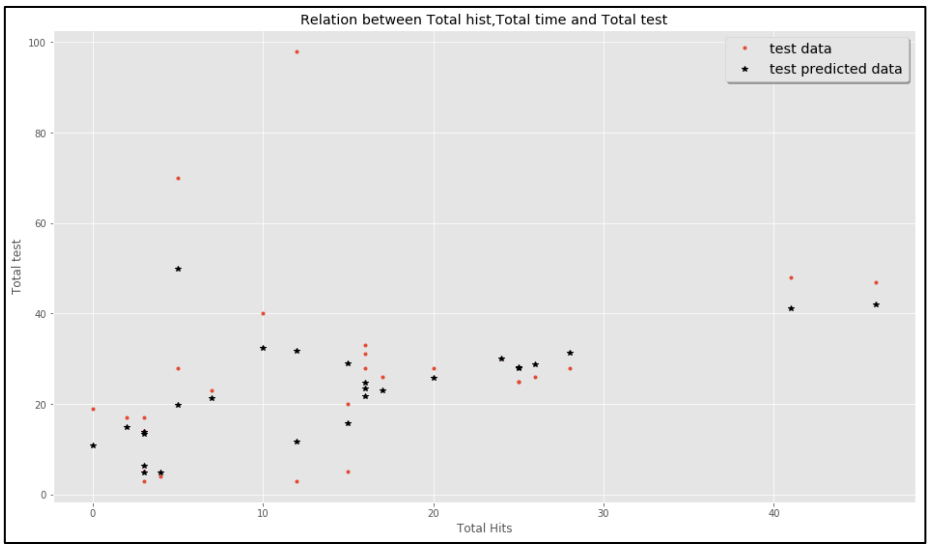

FIGURA 41. RELACIÓN ENTRE DATOS DE PRUEBA Y LOS QUE SON PREDICHOS.

En la Figura 42 se muestra la relación existe entre los resultados que se obtienen con la aplicación del algoritmo de **ridge 0.7,** que dio la mejor puntuación y los que predice el método. Se puede observar que todo están muy cercano y la evaluación se realiza en las variables Total time y Total Hits.

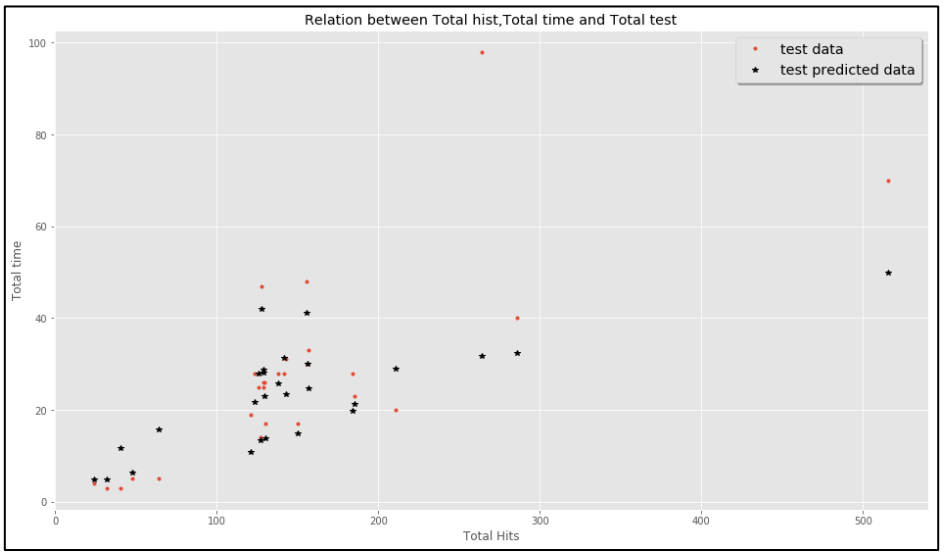

FIGURA 42. RELACIÓN DE DATOS DE ENTRENAMIENTO Y PRUEBAS

Se realiza una conclusión inicial con respecto al proceso de aplicación de los modelos de regresión. Después de aplicar los conceptos de *machine learning* de aprendizaje supervisado en algoritmos de regresión, no se llegó a una conclusión definitiva ya que los modelos no mostraron una buena calificación en su validación, esto debido a la poca información con la que se dispone, que para el caso de los juegos de atención dividida solo se contó con información de un juego.

4. Código Python utilizado en el desarrollo del trabajo

```
#Importación de librerías a ser utilizadas
# -*- coding: utf-8 -*-

import pandas as pd
import matplotlib.pyplot as plt
import re
from os import listdir
from os.path import isfile, join
import time
import datetime
import numpy as np
from pandas.plotting import scatter_matrix
import pickle #for save the model


# función plot_scatter utilizada para crear gráficas de la descripción
# de la información cargada
def plot_scatter(df, x,y,c,title='',s=None,n_color=None,cmap='Accent'):
    plt.figure()
    if s:
        f=plt.scatter(df[x],df[y],c=df[c],cmap=cmap,s=df[s])
    else:
        f=plt.scatter(df[x],df[y],c=df[c],cmap=cmap)
    plt.title(title)
    plt.xlabel(x)
    plt.ylabel(y)
    cbar=plt.colorbar(f)
    if n_color:
        cbar.ax.set_yticklabels(n_color)
    cbar.set_label(c)
```

```
    plt.savefig(title+'.pdf',format='pdf')

# directorio de donde se encuentran los archivos tipo texto
folder=folder='C:/Users/carlos/Downloads/proyecto_ tesis_dic/datos/'

# conversión de variables categóricas a cuantitativas
juegos=dict()
juegos['bici']=1
juegos['companeros']=2
juegos['conejera']=3
juegos['delfin']=4
juegos['igual']=5
juegos['liberalos']=6
juegos['paseo']=7

print([juegos])

#actualización de nombres a Ingles
juegos_nombre=dict()
juegos_nombre['bici']='Bike race'
juegos_nombre['companeros']='Where are my companions'
juegos_nombre['conejera']='The hutch'
juegos_nombre['delfin']='Feed the dolphins'
juegos_nombre['igual']='Find an equal one'
juegos_nombre['liberalos']='Free them'
juegos_nombre['paseo’] ='On a walk in the city'

print([juegos_nombre])

# lectura del encabezado de cada uno de los archivos
files = [f for f in listdir(folder) if (isfile(join(folder, f)) and f.endswith('txt'))]
#print(files)

lista_final= []
for f in files:
    est=re.match('[0-9]{1,2}',f) #número participante
    nombre=re.search('([a-z]+)',f)#nombre del juego
    intento=re.search('[0-9$]{8,10}',f)#intento juego
    if(est):
        est=est.group(0)
    else:
        est=-1
    if(nombre):
        nombre=nombre.group(0)
        if(nombre=='encuentras'):
            nombre='igual'
        nombre=juegos[nombre]
    else:
        nombre=-1

    if(intento):
        fecha=f[-12:-4]
        if len(intento.group(0))==8:
            intento=1
        elif len(intento.group(0))==9:
            intento=intento.group(0)[0:1]
        else:
            intento=intento.group(0)[0:2]
#        convertir a timestamp dato tipo datetime
        try:
            fecha=time.mktime(datetime.datetime.strptime(fecha, "%d%m%Y").timetuple())
        except:
            print(f,fecha)
            fecha=-1
```

```
    else:
        intento=-1
        print(f)
        fecha=-1

            # leer contenidos del archivo txt
            file=open(folder+'/'+f, "r") # leer archivo
        #renombrar variables
        dd={'user':est ,'game':nombre, 'Test':intento,'fecha':fecha}
        #creamos la variable Ensayo_total, con el total de Ensayo
        n_ensayo=0
        for line in file:
            data=line.split(";")

            #print(dd)
            i=0
            for d in data:
                tag=d.split(":")
                #if tag[0]!='Aciertos totales' and tag[0]!='Tiempo total'
:
                dd[str(tag[0].encode('ascii', 'ignore' ).decode( 'utf-
8'))] =float(re.sub('\.?\n','',tag[1]))
                if(tag[0]=='Ensayo'):
                    n_ensayo=int(tag[1])
                if tag[0]!='Tiempo total':
                    dd['Ensayo']=n_ensayo
                if tag[0]=='Tiempo total':
                    dd['Ensayo_total']=n_ensayo
                i=i+1
            if len(dd.keys())>6:
               lista_final.append(dd)
               dd={'user':est ,'game':nombre, 'Test':intento,'fecha':fech
a}
        file.close()

    #creamos el DataFrame  df
    df = pd.DataFrame(lista_final)
    print(df)

    df=df.apply(pd.to_numeric, errors='ignore')
    #Mostramos los encabezados
    df.keys()
    #creamos nueva varaible 'Time per average success'
    df['Time per average success']=df['Aciertos totales']/df['Tiempo tota
l']

    #primera grafica de descripción de datos
    plot_scatter(df,'Test','Time per average success','game',title='Ball
size # of attempts',s='Ensayo',n_color=juegos)

    plt.figure()
    #Actualizamos nombres de variables
    df['Total hits']=df['Aciertos totales']
    df['Total tests']=df['Ensayo_total']

    #grafias de Histograma de las variables 'Total hits','Total tests' pa
ra verififcar corelacion de varaibles
    df[['Total hits','Total tests']].hist()

    #grafías de cajas de bigotes de las variables 'Total hits','Total tes
ts' para verificar correlación y datos atípicos
    plt.figure()
    df[['Total hits','Total tests']].plot.box()
    plt.title('plot box tiempo total')

    # creación de nuevas variables
```

```
154.        df['Time first reaction']=df[' Tiempo Primer Tecla Oprimida'].fillna(
    0)+df[' Tiempo Primer Click'].fillna(0)+ df[' Tiempo Primer Compaero'].filln
    a(0)
155.
156.        df['Time reaction Average']=df[' Tiempo Primer Tecla Oprimida'].filln
    a(0)+df[' Tiempo Primer Click'].fillna(0)+ df[' Tiempo Primer Compaero'].fil
    lna(0)
157.
158.        #actualizamos la variable 'Time reaction Average'
159.        first=0
160.        n_media=0
161.        mean=0
162.        i=0
163.        for fr,tt,j,k in zip(df['Time first reaction'],df['Tiempo total'],ran
    ge(0,len(df['Time first reaction'])),range(0,len(df['Time reaction Average']
    ))):
164.            if np.isnan(tt):
165.                if(i==0):
166.                    first=fr
167.                mean+=fr
168.                i+=1
169.            else:
170.                df['Time first reaction'][j]=first
171.                n_media=mean/i
172.                df['Time reaction Average'][k]=n_media
173.                #print('valor para media',n_media)
174.                n_media=0
175.                i=0
176.                mean=0
177.
178.        print(df)
179.
180.        #verificamos los textos del encabezado del DataFrame
181.        df.keys()
182.
183.        #selección de datos para crear gráficas de descripción de los datos
184.        df9=df[df['Total hits']>0]
185.        df9=df9[df['game']==2]
186.        plot_scatter(df9,'Total tests','Total hits','game',title='Relation be
    tween total tests and Total hits',n_color=juegos)
187.
188.        # creamos nueva grafica
189.        plt.figure()
190.        df['Total hits']=df['Aciertos totales']
191.        df['Total tests']=df['Ensayo_total']
192.        df['Total time']=df['Tiempo total']
193.        plot_scatter(df,'Total tests','Total hits','game',title='Relation bet
    ween total test and total hits',n_color=juegos)
194.
195.        #número intentos y total aciertos para el primer intento
196.        df6=df[(df['Test'] == 1)|(df['Test']==2) & (df['game']!=2) ]
197.        df6['Trial']=df6['Ensayo']
198.        df6['Totales_hits']=df6['Aciertos totales']
199.        plot_scatter(df6,'Trial','Totales_hits','Test',title='Comparison of h
    its in the first and second attempts')#,n_color=intento)
200.        #generación y guardado del archivo para aplicación de modelos
201.        df.to_csv('datos1.csv',sep=';', encoding='utf-8')
```

```
#Código Utilizado en el Desarrollo del Presente Trabajo de Machine #Learning

#Importación de librerías a ser utilizadas
# -*- coding: utf-8 -*-

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sb
import sklearn
from sklearn.cluster import KMeans
from sklearn.metrics import pairwise_distances_argmin_min
from mpl_toolkits.mplot3d import Axes3D
from sklearn.decomposition import PCA
plt.rcParams['figure.figsize'] = (16, 9)
plt.style.use('ggplot')
import sklearn.cluster as cluster
#nuevas
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from matplotlib.colors import ListedColormap
import seaborn as sns
from sklearn.metrics import classification_report

# lectura del archivo
df5 = pd.read_csv('datos1.csv',sep=';')

# Código utilizado para la aplicación de los modelos de closterización
# se realiza la selección de variables a tener en cuenta para la closterizac
ión.
df=df5[['game',"user","Test",'Total time','Total hits','Total tests','Time f
irst reaction','Time reaction Average']]
df=df.dropna()

df=df5[['game',"user","Test","Total_Time","Total_Hits",'Total_Test','Seconds
 first reaction','Seconds first reaction Average']]

df.head()
df.isnull().any()

#creación de la matriz de Dispersión de los datos seleccionados
df1=df.loc[:,'Test':'Time reaction Average']
f,ax = plt.subplots(figsize=(18, 18))
sns.heatmap(data=df1.corr(),annot=True,linewidths=.5, fmt= '.1f',ax=ax)
plt.show()

# Concretamos la estructura de datos que utilizaremos para los algoritmo.
# Como se ve, sólo cargamos las columnas "Ensayo_total","Tiempo total",'Test
','Aciertos totales" en nuestra variable X
X1 = np.array(df[["Test",'Total time','Total hits','Total tests','Time first
 reaction','Time reaction Average']])

# creamos una nueva variable 'congition function' que clasifica los #juegos
con el tipo de atención para la cual fue Desarrollo
lst=[]
for i in range(0,len(df['game'])):
    g=df['game'].iloc[i]
    if(g==1 or g==2 or g==4):
        lst.append(2)
    elif(g==3 or g==5 or g==7):
        lst.append(0)
    else:
```

```
        lst.append(1)
df['congition function']=lst
X1.shape

y1 = np.array(df['game'])

## aplicación de los modelos de Agrupamientos
## utilizando algoritmos de K-meas Clustering

# hallar el “punto de codo”
fig = plt.figure()
Nc = range(1, 20)
kmeans = [KMeans(n_clusters=i) for i in Nc]
score = [kmeans[i].fit(X1).score(X1) for i in range(len(kmeans))]

plt.plot(Nc,score)
plt.xlabel('Number of Clusters')
plt.ylabel('Score')
plt.title('Elbow Curve')
plt.show()

# la gráfica del punto de codo nos muestra la mejor opción de 3 grupos
# Ejecutamos el algoritmo para 3 clusters y obtenemos las etiquetas y los centroides.
# creamos un objeto k-means
kmeans = KMeans(n_clusters=3, random_state=100).fit(X1)
centroids = kmeans.cluster_centers_
print(centroids)

x = np.linspace(0, 100,50)
# veremos si se diferencian: (las estrellas marcan el centro de cada cluster)
# Predicting the clusters
labels = kmeans.predict(X1)

# Getting the cluster centers
C = kmeans.cluster_centers_

# se realiza el agrupamiento por el tipo de función cognitiva a tratar
# creamos la gráfica de las tres agrupaciones
groups = ("Function Sustained", "Function Divided", "Function Selective")
#colocar las lebel de los vértices
      #plt.scatter(X1[:, 0], X1[:, 1], c=labels, s=50, cmap='viridis', marker=df['congition function'])
      markers = ["o","v",'*']
      for i, c in enumerate(np.unique(df['congition function'])):

          plt.scatter(X1[df['congition function']==c, 0], X1[df['congition function']==c, 1],c=labels[df['congition function']==c], marker=markers[i],label=groups[i])

      plt.scatter(C[:, 0], C[:, 1], c='black', s=200, alpha=0.5,label=' Centroids');
      plt.xlabel('Test')
      plt.ylabel('Total time')
      legend = plt.legend(shadow=True, fontsize='x-large')

      # generamos la Matriz de confusion 'Confusion Matrix about cognition functions'
      #función para dibujar la matriz
      def print_confusion(m):
          #print is never easy :)
          plt.figure()
          for i in range(0,3):
              for j in range(0,3):
                  L=m[i,j]
```

```
                    plt.scatter(i,j,s=100,marker='$'+str(L)+'$')
                    plt.scatter(i,j,s=20*L,marker='o',alpha=0.5)
            plt.show()

        #creación de la gráfica
        M=confusion_matrix(df['congition function'], labels)
        print(M)
        f,ax = plt.subplots(figsize=(3, 3))
        sns.heatmap(data=M/89,annot=True, linewidths=.5, fmt= '.1f',ax=ax)
        plt.show()

        #utilización del método de reducción de variables 'Visualization thro
    ugh PCA'
        C1 = kmeans.cluster_centers_
        #estandarizamos los datos de entrenamiento
        #para utilizar PCA
        sc = StandardScaler()
        X_train_std = sc.fit_transform(X1)

        C1_train_std = sc.transform(C1)

        cov_mat = np.cov(X_train_std.T)
        print(cov_mat)

        ##Utilizamos reducción de la dimensionalidad con el algoritmo PCA()
        pca = PCA()
        X_train_pca = pca.fit_transform(X_train_std)

        #print(X_train_pca.shape)
        X_center = pca.transform(C1_train_std)

        #print(X_center)
        pca.explained_variance_ratio_

        plt.figure()
        varianza = pca.explained_variance_ratio_
        print(varianza)
        kk=np.cumsum(varianza)
        print(kk)

        plt.bar (range(1,7), varianza, alpha=0.2, align='center',label='Varia
    nza individual explicada', color='g')
        plt.step(range(1,7), kk, where='mid', linestyle='--
    ', label='Varianza explicada acumulada')
        plt.ylabel('Explained variance ratio')
        plt.xlabel('Principal components')
        plt.show()

        x = np.linspace(0, 100, 50)

        groups = ("Function Sustained", "Function Divided", "Function Selecti
    ve")
        plt.figure()

        #plt.scatter(X_train_pca[:, 0], X_train_pca[:, 1], c=labels, s=50, cm
    ap='viridis')

        for i, c in enumerate(np.unique(df['congition function'])):

            plt.scatter(X_train_pca[df['congition function']==c, 0], X_train_
    pca[df['congition function']==c, 1],c=labels[df['congition function']==c], m
    arker=markers[i],label=groups[i])

        plt.scatter(X_center[:, 0], X_center[:, 1], c='red', s=200, alpha=0.5
    );
        plt.xlabel('PC 1')
```

```
        plt.ylabel('PC 2')
        legend = plt.legend(shadow=True, fontsize='x-large')

        ## la gráfica 3d con 3 momentos principales para una mejor visualizac
    ión
        colores=['blue','red','green','blue','cyan','yellow','orange','black'
    ,'pink','brown','purple']
        asignar=[]
        for row in labels:
            asignar.append(colores[row])

        fig = plt.figure()
        ax = Axes3D(fig)
        ax.scatter(X1[:, 0], X1[:, 1], X1[:, 2], c=asignar,s=60)
```

```
## métodos de CLASSIFICATION
# Importación de librerías a ser utilizadas
from sklearn.model_selection import train_test_split
import sklearn.naive_bayes
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.neural_network import MLPClassifier
#print only 3 decimals
np.set_printoptions(precision=3)

## Aplicando clasificación

names=['naive bayes','k-
neighbors','decision tree','random Forest' ,'logistic' ,'svm radial','best M
LP']
clasificadores=[sklearn.naive_bayes.GaussianNB(),
                KNeighborsClassifier(n_neighbors=3),
                DecisionTreeClassifier(max_depth=3),
                RandomForestClassifier(max_depth=3),
                LogisticRegression() ,
               SVC(C= 10, gamma = 0.0001, kernel='rbf'),
               MLPClassifier(activation= 'relu', hidden_layer_sizes= (100,),
 learning_rate= 'adaptive', learning_rate_init= 0.0001, max_iter= 10000, sol
ver= 'lbfgs'),]

from sklearn.model_selection import train_test_split,cross_val_score

y2=df['congition function']
x_train,x_test, y_train, y_test  = train_test_split(X1,y2, test_size = 0.3,
random_state=0 ) # remove random state o change it
lst=[]
for clf,name in zip(clasificadores,names):
    print(name)
    scores = cross_val_score(clf, x_train, y_train, cv=3)
    print(scores)
    lst.append(scores.mean())
    # 95% de confidencia en el error (dos desviaciones estandar)
    print("Accuracy: %0.2f(+/-%0.2f)"%(scores.mean(),scores.std()* 2))
    print()

#identificamos el mejor algoritmo de clasificación
best=sorted(range(len(lst)), key=lambda k: lst[k], reverse=True)
print("The best model is")
print(names[best[0]])
```

```
43.
44. #creamos la matriz de confusión con los datos de test para el mejor #algorit
    mo
45. # with the best
46. print('OVER TEST DATA')
47. clasificadores[best[0]].fit(x_train,y_train)
48. y_pred=clasificadores[best[0]].predict(x_test)
49. M=confusion_matrix(y_test, y_pred)
50. f,ax = plt.subplots(figsize=(3, 3))
51. sns.heatmap(data=M/27,annot=True, linewidths=.5, fmt= '.1f',ax=ax)
52. plt.title('OVER TEST DATA')
53. plt.show()
54. print(classification_report(y_test, y_pred))
55.
56. #creamos la matriz de confusión con el total de los datos el mejor algoritmo

57. print('ALL THE DATA')
58. y_pred=clasificadores[best[0]].predict(X1)
59. M=confusion_matrix(y2, y_pred)
60. f,ax = plt.subplots(figsize=(3, 3))
61. sns.heatmap(data=M/86,annot=True, linewidths=.5, fmt= '.1f',ax=ax)
62. plt.title('ALL THE DATA')
63. plt.show()
64. print(classification_report(y2, y_pred))
65.
66. ##búsqueda de los mejores parámetros para support vector machines
67. import warnings
68. warnings.filterwarnings('ignore')
69. from sklearn.model_selection import GridSearchCV
70. from sklearn import svm
71.
72. # Set the parameters by cross-validation
73. tuned_parameters = [{'kernel': ['rbf'], 'gamma': [0.1,1e-3, 1e-4],
74.                      'C': [1, 10, 100, 1000,10000]},
75.                     {'kernel': ['linear'], 'C': [1, 10, 100, 1000]},
76.                     {'kernel': ['poly'],'degree':[3,5,7], 'C': [1, 10, 100,
    1000],'gamma': [1e-3]}
77.                    ]
78.
79. scores = ['precision' , 'recall','f1' ]
80.
81. for score in scores:
82.     print("# Tuning hyper-parameters for %s" % score)
83.     print()
84.
85.     clf = GridSearchCV(svm.SVC(), tuned_parameters, cv=5,
86.                        scoring='%s_macro' % score)
87.     clf.fit(x_train, y_train)
88.
89.     print("Best parameters set found on development set:")
90.     print()
91.     print(clf.best_params_)
92.     print()
93.     print("Grid scores on development set:")
94.     print()
95.     means = clf.cv_results_['mean_test_score']
96.     stds = clf.cv_results_['std_test_score']
97.     for mean, std, params in zip(means, stds, clf.cv_results_['params']):
98.         print("%0.3f (+/-%0.03f) for %r"
99.               % (mean, std * 2, params))
100.          print()
101.
102.          print("Detailed classification report:")
103.          print()
104.          print("The model is trained on the full development set.")
105.          print("The scores are computed on the full evaluation set.")
```

```
            print()
            y_true, y_pred = y_test, clf.predict(x_test)
            print(classification_report(y_test, y_pred))


        ##búsqueda de los mejores parámetros para una red neuronal
        from sklearn.neural_network import MLPClassifier
        # Set the parameters by cross-validation
        tuned_parameters = [{'hidden_layer_sizes':[(100,),(50,20,),(30,20,5,)
    ], 'activation' :['logistic', 'tanh','relu'],'learning_rate_init': [1e-
    3, 1e-4],  'solver' : ['lbfgs', 'sgd', 'adam'],'max_iter':[2000]  }]
        scores = ['precision' ]#, 'recall','f1' ]

        for score in scores:
            print("# Tuning hyper-parameters for %s" % score)
            clf = GridSearchCV(MLPClassifier(), tuned_parameters, cv=5,  scor
    ing='%s_macro' % score)
            clf.fit(x_train, y_train)
            print("Best parameters set found on development set:")
            print(clf.best_params_)
            print("Grid scores on development set:")
            means = clf.cv_results_['mean_test_score']
            stds = clf.cv_results_['std_test_score']
            for mean, std, params in zip(means, stds, clf.cv_results_['params
    ']):
                print("%0.3f (+/-%0.03f) for %r"
                      % (mean, std * 2, params))
            print()
            print("Detailed classification report:")
            print()
            print("The model is trained on the full development set.")
            print("The scores are computed on the full evaluation set.")
            print()
            y_true, y_pred = y_test, clf.predict(x_test)
            print(classification_report(y_test, y_pred))

        # se hace cross-validation para los resultados optenidos
        from sklearn.neural_network import MLPClassifier
        # Set the parameters by cross-validation
        tuned_parameters=[{'hidden_layer_sizes':[(100,),(30,5,),(30,),(30,20,
    5)],'activation' :['relu'], 'learning_rate_init': [1e-5, 1e-
    4], 'solver' : ['lbfgs'],'max_iter':[2000,10000],'learning_rate' :['adaptive
    ' }]
        scores = ['precision']# , 'recall','f1' ]
        for score in scores:
            print("# Tuning hyper-parameters for %s" % score)
            print()
            clf = GridSearchCV(MLPClassifier(), tuned_parameters, cv=3,
                               scoring='%s_macro' % score)
            clf.fit(x_train, y_train)
            print("Best parameters set found on development set:")
            print()
            print(clf.best_params_)
            print()
            print("Grid scores on development set:")
            print()
            means = clf.cv_results_['mean_test_score']
            stds = clf.cv_results_['std_test_score']
            for mean, std, params in zip(means, stds, clf.cv_results_['params
    ']):
                print("%0.3f (+/-%0.03f) for %r"
                      % (mean, std * 2, params))
            print()

            print("Detailed classification report:")
            print()
```

```
            print("The model is trained on the full development set.")
            print("The scores are computed on the full evaluation set.")
            print()
            y_true, y_pred = y_test, clf.predict(x_test)
            print(classification_report(y_test, y_pred))

        ##se realiza la aplicación de los modelos de Regresión
        # Importación de librerías a ser utilizadas
        import pandas as pd
        import numpy as np
        from matplotlib import pyplot as plt
        from sklearn import linear_model
        from sklearn.model_selection import train_test_split
        from sklearn.metrics import accuracy_score,median_absolute_error
        from sklearn.metrics import mean_squared_error, mean_absolute_error
        from sklearn.metrics import r2_score,mean_squared_log_error
        from sklearn.metrics import explained_variance_score
        from sklearn.model_selection import cross_val_score, cross_validate
        from sklearn.neural_network import MLPRegressor
        from sklearn import svm

        #read the file
        df=pd.read_csv('datos1.csv',sep=';')

        #selection the variable
        #df2=df[["user","Test","Tiempo total","Aciertos totales",'game','Ensayo_total']]
        df2=df[['game',"user","Test",'Total time','Total hits','Total tests']]#,'Seconds first reaction']]
        df2=df2.dropna()

        print(df2.describe())

        #creation in the matrixs
        X=df2.as_matrix(['Total hits','Total time'])
        y=df2.as_matrix(['Total tests'])

        #selection the set date for train and test
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=4200)

        #model the regression
        #si tenemos varios modelos debemos hacer cross validation
        regressors = [linear_model.LinearRegression(fit_intercept = False),
                      linear_model.Ridge(alpha=0.7,fit_intercept = False),
                       MLPRegressor(hidden_layer_sizes=(100,),activation='relu',
                                    max_iter=100000),
                      svm.SVR(C= 10, gamma = 0.0001, kernel='rbf')]

        names = ['lineal','ridge 0.7','MLP','SVM']
        #from https://scikit-learn.org/stable/modules/model_evaluation.html #scoring-parameter
        scoring =['explained_variance',
                  'neg_mean_absolute_error',
                  'neg_mean_squared_error',
                  'neg_mean_squared_log_error',
                  'neg_median_absolute_error',
                  'r2']
        lst=[]
        for reg,name in zip(regressors,names):
            scores = cross_val_score(reg, X_train, y_train.ravel(), scoring=scoring[5],
                                     cv=5)
            print('---------------------------------------')
            print('model {0:20} | score {1:20}'.format(name,scoring[5]))
```

```
            print('mean {0:22.2f}| std   {1:<22.2f}'.format(scores.mean(),scores.std()))
            print("Accuracy(95.7%): {0:23.2f} (+/- {1:.2f})" .format(scores.mean(), scores.std() * 2))
            lst.append(scores.mean())

        best=sorted(range(len(lst)), key=lambda k: lst[k], reverse=True)
        print("The best model is")
        print(names[best[0]])

        #make cross-validation for validate the model
        import warnings
        warnings.filterwarnings('ignore')
        from sklearn.model_selection import GridSearchCV
        from sklearn import svm


        # Set the parameters by cross-validation
        tuned_parameters = [{'kernel': ['rbf'], 'gamma': [0.1,1e-3, 1e-4],
                             'C': [1, 10, 100, 1000,10000]},
                            {'kernel': ['linear'], 'C': [1, 10, 100, 1000]},

                            {'kernel': ['poly'],'degree':[3,5,7], 'C': [1, 10, 100, 1000],'gamma': [1e-3]}
                           ]

        scoring =['explained_variance',  'r2']

        for score in scoring:
            print("# Tuning hyper-parameters for %s" % score)
            print()

            clf = GridSearchCV(svm.SVR(), tuned_parameters, cv=5,
                               scoring='%s' % score)
            clf.fit(X_train, y_train)

            print("Best parameters set found on development set:")
            print()
            print(clf.best_params_)
            print()
            print("Grid scores on development set:")
            print()
            means = clf.cv_results_['mean_test_score']
            stds = clf.cv_results_['std_test_score']
            for mean, std, params in zip(means, stds, clf.cv_results_['params']):
                print("%0.3f (+/-%0.03f) for %r"
                      % (mean, std * 2, params))
            print()

            print("Detailed classification report:")
            print()
            print("The model is trained on the full development set.")
            print("The scores are computed on the full evaluation set.")
            print()
            y_true, y_pred = y_test, clf.predict(X_test)
            print(classification_report(y_test, y_pred))

        #the classification is done for the model LinearRegression
        # for check the quality of the models
        regr = linear_model.LinearRegression()
        regressors[best[0]].fit(X_train,y_train)
        #prediction
        y_pred=regressors[best[0]].predict(X_test)


```

```
285.        # https://scikit-
    learn.org/stable/modules/model_evaluation.html#regression-metrics
286.        print('{0:30} | {1:9.2f}'.format('regressor score', regr.score(X_test
    ,y_test)))
287.        print('{0:30} | {1:9.2f}'.format('Mean squared error', np.mean((y_pre
    d - y_test) ** 2)))
288.        print('{0:30} | {1:9.2f}'.format('mean_squared_error', mean_squared_e
    rror(y_test,y_pred)))
289.        print('{0:30} | {1:9.2f}'.format('mean_squared_log_error', mean_squar
    ed_log_error(y_test,y_pred)))
290.        print('{0:30} | {1:9.2f}'.format('mean_absolute_error', mean_absolute
    _error(y_test,y_pred)))
291.        print('{0:30} | {1:9.2f}'.format('median_absolute_error', median_abso
    lute_error(y_test,y_pred)))
292.        print('{0:30} | {1:9.2f}'.format('explained_variance_score', explaine
    d_variance_score(y_test,y_pred)))
293.        print('{0:30} | {1:9.2f}'.format('r ^ 2 score', r2_score(y_test,y_pre
    d)))
294.
295.        ##gráficas
296.        # plot prediction and actual data
297.        plt.plot(y_test, y_pred, '.',label='test data')
298.
299.        plt.plot(y_train,regressors[best[0]].predict(X_train), '+',color='gre
    en',label='train data')
300.
301.        # plot a line, a perfit predict would all fall on this line
302.        x = np.linspace(0, 100, 50)
303.        y = x
304.        plt.plot(x, y,label='regresor')
305.        plt.xlabel('expected Total test')
306.        plt.ylabel('predicted Total test')
307.        plt.title('Relation between Total hist,Total time and Total test')
308.
309.        legend = plt.legend( shadow=True, fontsize='x-large')
310.        plt.show()
311.
312.        ##gráficas
313.        # plot prediction and actual data
314.        cual=0
315.        x = X_test[:,cual]
316.
317.        plt.plot(x, y_test, '.',label='test data')
318.
319.        #plt.plot(X_train[:,cual],y_train, '+',color='green',label='train dat
    a')
320.        # plot a line, a perfit predict would all fall on this line
321.        y = regressors[best[0]].predict(X_test)
322.        plt.plot(X_test[:,cual],y, '*',color='black',label='test predicted da
    ta')
323.        x = np.linspace(0, 100, 50)
324.
325.        #plt.plot(x,regressors[best[0]].coef_[0][cual]*x+regressors[best[0]].
    intercept_ ,label='regresor')
326.        plt.xlabel('Total Hits')
327.        plt.ylabel('Total test')
328.        plt.title('Relation between Total hist,Total time and Total test')
329.        legend = plt.legend( shadow=True, fontsize='x-large')
330.        plt.show()
331.
332.        ##gráficas
333.        # plot prediction and actual data
334.        cual=1
335.        x = X_test[:,cual]
336.
337.        plt.plot(x, y_test, '.',label='test data')
```

```
338.
339.        #plt.plot(X_train[:,cual],y_train, '+',color='green',label='train dat
    a')
340.
341.        # plot a line, a perfit predict would all fall on this line
342.
343.        y = regressors[best[0]].predict(X_test)
344.        plt.plot(X_test[:,cual],y, '*',color='black',label='test predicted da
    ta')
345.        x = np.linspace(0, 100, 50)
346.
347.        #plt.plot(x,regressors[best[0]].coef_[0][cual]*x+regressors[best[0]].
    intercept_ ,label='regresor')
348.        plt.xlabel('Total Hits')
349.        plt.ylabel('Total time')
350.        plt.title('Relation between Total hist,Total time and Total test')
351.
352.        legend = plt.legend( shadow=True, fontsize='x-large')
353.        plt.show()
354.
355.        #get the results for the coefficient and the intercept_print(regresso
    rs[best[0]].coef_[0][0])
356.        print(regressors[best[0]].intercept_)
357.        sklearn.metrics.SCORERS.keys()
```