Método de la ingeniería

Fase 1: Identificación del problema

Estados Unidos es el tercer país más poblado del mundo con un total de 330 millones de habitantes. En él, aproximadamente se encuentran 106 millones de viviendas donde las personas de los diferentes estados habitan. Además, se tiene en cuenta que en un futuro este mercado avance de forma directa con el aumento de habitantes en cada una de las regiones nativas. Es por esta razón que en algunas ocasiones las compañías de seguro, impuestos de registro y la administración gubernamental, se ven paralizadas por el número de peticiones o encargos que cada día manejan.

Tomando en cuenta esta situación, se necesita realizar un software de aprendizaje autónomo con la idea de agilizar los procesos de compra y venta de cada una de las viviendas solicitadas, dominando los siguientes requerimientos funcionales:

1. Leer y cargar el dataset “House Sales in King County, USA” proveniente de Kaggle, tomando en cuenta variables como: fecha, precio, número de pisos, pies cuadrados construidos y cantidad de habitaciones.

2. Visualizar los datos de manera correcta y ordenada desde el momento en que el usuario cargue el dataset, hasta cuando deje de utilizar el software. Esto, por medio de un data view y de un splitcontainer.

3. Clasificar un nuevo datapoint designando un rango de precio con base a las opciones establecidas con anterioridad, analizando los datos brindados en el dataset seleccionado por medio de un aprendizaje autónomo.

4. Diseñar una interfaz gráfica de usuario donde se logre realizar cada una de las funcionalidades deseadas para el proyecto, que básicamente se desarrolla en dos aspectos importantes: carga, visualización y toma de datos para su análisis, y la generación de diferentes componentes como gráficas, contenedores, entre otros.

5. Diagramar la base del software por medio de clases (UML), y objetos (con base al primero respectivamente). En este caso, se tomará en cuenta la herramienta Visual Paradigm Online para visualizar, modificar y agregar todas las clases, requerimientos y paquetes encontrados en el sistema.

6. Generar gráficos estadísticos analizando los datos brindados por el dataset seleccionado con anterioridad. Serán un total de 5 designados de la siguiente manera: gráfico de líneas, circular, relieve, barras (2 con diferentes datos analizados).

7. Implementar un algoritmo que permita generar un árbol de decisiones con una buena efectividad para predecir el precio del inmueble a partir de los demás factores presentados en el dataset, estos son: número de pisos, habitaciones y baños, pies cuadrados construidos, entre otros.

8. Utilización de una librería externa de C# que nos ayude a resolver la clasificación y análisis de datos planteado en el enunciado, tomando en cuenta el dataset elegido con anterioridad. Para hacerlo, se toma en cuenta este video como una base para iniciar.

9. Diseñar, desarrollar e implementar el experimento con el cual se va a comprobar los resultados obtenidos del dataset, analizando algunos factores variables de salida, número de repeticiones, niveles o tratamientos.

10. Ejecutar el experimento anteriormente implementado para lograr obtener algunos resultados frente a los módulos programados (comparación entre código propio y externo [librería]).

11. Realizar un análisis ANOVA para conocer la naturaleza, características, estado y factores que intervienen en todo el experimento planteado. Para hacerlo, se evidencia una base sólida con gráficas, dependencias e independencias.

12. Evaluar de manera significativa el experimento, valorando los factores estudiados, pero principalmente, concluir una comparación exhaustiva entre la implementación propia y la librería externa.

Fase 2: Recolección de la información necesaria

Para solucionar este problema y realizar un correcto software, es indispensable conocer algunas herramientas importantes que permitan cumplir con cada uno de los requerimientos funcionales anteriormente descritos. Algunas de estas son:

Machine Learning: Es una habilidad indispensable para hacer sistemas capaces de identificar patrones entre los datos para hacer predicciones. Esta tecnología está presente en un sinfín de aplicaciones como las recomendaciones de Netflix y Spotify o las respuestas inteligentes de Gmail y el habla de Siri y Alexa. Además, es capaz de convertir una muestra de datos en un programa informático capaz de extraer inferencias de nuevos conjuntos de datos para los que no ha sido entrenado previamente

Los algoritmos de Machine Learning se dividen en tres categorías, siendo las dos primeras las más comunes:

- Aprendizaje supervisado: estos algoritmos cuentan con un aprendizaje previo basado en un sistema de etiquetas asociadas a unos datos que les permiten tomar decisiones o hacer predicciones. Un ejemplo es un detector de *spam* que etiqueta un *e-mail* como *spam* o no dependiendo de los patrones que ha aprendido del histórico de correos (remitente, relación texto/imágenes, palabras clave en el asunto, etc.).

- Aprendizaje no supervisado: estos algoritmos no cuentan con un conocimiento previo. Se enfrentan al caos de datos con el objetivo de encontrar patrones que permitan organizarlos de alguna manera. Por ejemplo, en el campo del *marketing* se utilizan para extraer patrones de datos masivos provenientes de las redes sociales y crear campañas de publicidad altamente segmentadas.

- Aprendizaje por refuerzo: su objetivo es que un algoritmo aprenda a partir de la propia experiencia. Esto es, que sea capaz de tomar la mejor decisión ante diferentes situaciones de acuerdo a un proceso de prueba y error en el que se recompensan las decisiones correctas. En la actualidad se está utilizando para posibilitar el reconocimiento facial, hacer diagnósticos médicos o clasificar secuencias de ADN.

Algunas aplicaciones de esta rama tecnológica pueden ser:

- Motores de búsqueda
- Diagnóstico médico
- Detección de fraudes con el uso de tarjetas de crédito
- Análisis del mercado de valores
- Clasificación de secuencias de ADN
- Reconocimiento del habla
- Robótica
- Minería de datos
- Big Data

Árbol de decisiones: Esta herramienta nos ayuda a la hora de tomar decisiones correctas cuando se tienen grandes cantidades de datos, y se presenta como posibles ramas en las que se plantean diferentes soluciones a un escenario continuo de sucesos. Asimismo, tomando en cuenta cada complejidad situada en diferentes planteamientos, la tecnología hablada presenta diferentes uso como:

- Plantear problemas desde diferentes puntos de vista.
- Analizar todas las posibles soluciones a una misma decisión.
- Estimar factores cuantitativos cuando el problema lo permita.
- Tomar decisiones en base a información real y confiable.
- Analizar diferentes alternativas y probabilidades que cada rama puede generar.

Los árboles de decisión están formados por nodos, vectores de números, flechas y etiquetas.

- Cada nodo se puede definir como el momento en el que se ha de tomar una decisión de entre varias posibles, lo que va haciendo que a medida que aumenta el número de nodos aumenta el número de posibles finales a los que puede llegar el individuo. Esto hace que un árbol con muchos nodos sea complicado de dibujar a mano y de analizar debido a la existencia de numerosos caminos que se pueden seguir.

- Los vectores de números serían la solución final a la que se llega en función de las diversas posibilidades que se tienen, dan las utilidades en esa solución.

- Las flechas son las uniones entre un nodo y otro y representan cada acción distinta.

- Las etiquetas se encuentran en cada nodo y cada flecha y dan nombre a cada acción.

Además, los árboles de decisión debe cumplir una serie de reglas:

1. Al comienzo del juego se da un nodo inicial que no es apuntado por ninguna flecha, es el único del juego con esta característica.

2. El resto de los nodos del juego son apuntados por una única flecha.

3. De esto se deduce que hay un único camino para llegar del nodo inicial a cada uno de los nodos del juego. No hay varias formas de llegar a la misma solución final, las decisiones son excluyentes.

Por último, estas decisiones que se eligen son lineales, ya que a medida que vas seleccionando entre varias opciones se van cerrando otras, lo que implica normalmente que no hay marcha atrás. En general se podría decir que las normas siguen una forma condicional: Opción 1 -> opción 2 -> opción 3 -> Resultado Final X. Estas reglas suelen ir implícitas en el conjunto de datos a raíz del cual se construye el árbol de decisión.

Gráficos: Una parte indispensable del programa es generar gráficos estadísticos que permitan entender el problema de manera significativa. Por ende, es de gran importancia reconocer cuales pueden ser algunos tipos de gráficos que podemos utilizar, y cuales son de gran necesidad para el problema planteado.

Lector de archivos: Para utilizar el sistema de forma eficaz, es necesario leer rápidamente un dataset con toda la información existente hasta el momento. De esta manera no solo se evita perder datos significativos, sino que también permite filtrar casillas por cualquier tipo de limitación que el mismo usuario interponga.

Además de esas funcionalidades, se complementa la información con los requerimientos no funcionales encontrados en el sistema:

1. Depositar el código fuente, y cualquier otro cambio en el sistema en la plataforma GitHub.

2. Utilizar el readme del repositorio fuente para explicar el funcionamiento del software, y enlazar todos los documentos utilizados.

3. Grabar el uso Beta del sistema para mostrar su funcionamiento a las personas interesadas en utilizar o adquirir el software.

4. Programar e implementar el programa en el lenguaje C# (Visual Studio).

5. Codificar e implementar un árbol de decisión como la estructura primaria del programa.

6. Diagramar 3 secuencias importantes del programa para tener un mayor entendimiento en algunos aspectos del software.

7. Manejar buenas prácticas de programación al momento de utilizar la librería externa, comparado con el código propio implementado.

8. Al momento de ejecutar el experimento, evidencia una buena toma de datos para realizar un futuro análisis exhaustivo, donde por medio de ANOVA logremos encontrar conclusiones importantes del software.

9. Manejar una complejidad temporal baja, ya que si estamos manejando grandes cantidades de datos, es mejor evidenciar una buena conducción de tiempo y resultados.

10. Al momento de diseñar y desarrollar el experimento en la iteración 4, definir apropiadamente cada uno de los factores que se deben de tomar en cuenta, incluyendo el número de repeticiones y la cantidad de niveles apropiados.

Fase 3: Búsqueda de soluciones creativas

Para llegar a una solución que resulte efectiva para todos los autores del proyecto, cada uno de estos debe analizar los requerimientos anteriormente indicados para así proceder a dialogar sobre las posibles soluciones a las que se pueden llegar.

Mediante el diálogo y la resolución de unas cuantas preguntas, se llegará a la solución que todos consideren adecuada. Las preguntas son las siguientes:

1. ¿Cuál es el problema a resolver?

   R//: Dificultad durante el proceso de compra y venta de vivienda.

2. ¿Qué información se necesita para resolverlo?

   R//: Precios, ubicación e información general de cada casa.

3. ¿Cómo se analizará la información?

   R//: Se sacarán porcentajes de características en las casas para graficarlos.

4. ¿Qué entradas recibirá el programa?

   R//: Filtros de búsqueda que acoten la cantidad de casas resultado.

5. ¿Qué información se debe mostrar para facilitar el correcto uso del programa?

   R//: Campos de filtro, gráficas, tablas.