UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

Generación de cuestionarios sobre reglas de asociación (AssoQuiz-Generator)

Presentado por Sergio Revilla Merino
en Universidad de Burgos — 13 de febrero de 2023
Tutor: Juan José Rodríguez Díez

UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

D. Juan José Rodríguez Díez, profesor del departamento de Ingeniería Informática, área de Lenguajes y Sistemas Informáticos.

Expone:

Que el alumno D. Sergio Revilla Merino, con DNI 71314289Y, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado Generacion de cuestionarios sobre reglas de asociación (AssoQuiz-Generator).

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 13 de febrero de 2023

Vº. Bº. del Tutor:	Vº. Bº. del co-tutor:
Firmado por RODRIGUEZ DIEZ JUAN JOSE - ***9371** el día 13/02/2023 con un certificado emitido por	
D. Juan José Rodríguez Díez	D. nombre co-tutor

Resumen

Desarrollo de una aplicación para la Universidad de Burgos que consta de una interfaz gráfica en la que el usuario debe de seleccionar entre unos tipos de preguntas a generar, configurando unos parámetros diferentes en cada una. La aplicación genera un archivo en formato .XML para ser importado en la plataforma Moodle con la finalidad de generar de forma automática cuestionarios sobre reglas de asociación para la asignatura de minería de datos.

Descriptores

Generación de cuestionarios, minería de datos, reglas de asociación, algoritmo apriori.

Abstract

Development of an application for the university of Burgos that consists of a graphical interface in which the user must select between some types of questions to generate, configuring different parameters in each one. The application generates a file in .XML format to be imported into the Moodle platform in order to automatically generate questionnaires on association rules for the subject of data mining.

Keywords

Quiz generation, data mining, association rules, apriori algorithm.

Índice general

Índice de figuras

Índice de tablas

Introducción

La enseñanza en línea ha experimentado un gran aumento en la actualidad, especialmente en el ámbito universitario. Esto se ve reflejado en la creciente demanda de educación a distancia. Un ejemplo evidente es UBUVirtual, la plataforma virtual de la Universidad de Burgos, basada en Moodle, que es ampliamente utilizada. Por lo tanto, es esencial proporcionar las herramientas necesarias para mejorar la experiencia de estudiantes y profesores. Esto incluye la capacidad de crear y publicar exámenes en la plataforma.

El proyecto presentado consiste en una aplicación de escritorio que permite generar cuestionarios en formato .XML sobre reglas de asociación para poseriormente ser importados en Moodle.

Las preguntas que se pueden generar mediante esta aplicación son las siguientes:

1. ***Generación Item Sets***: dado un conjunto de datos con una cantidad determinada de atributos y transacciones obtener sus item sets con soporte mayor o igual que un valor determinado.

   En la siguiente imagen se muestra un ejemplo de pregunta de los apuntes de la asignatura de Minería de Datos que sirvió como inspiración para esta pregunta.

Sea el siguiente conjunto de datos:

A1	A2	A3	A4	Clase
T	51	T	F	c0
F	77	T	T	c1
T	71	F	F	c0
T	64	T	T	c0
T	36	F	T	c1
T	86	F	F	c0
F	48	T	F	c0
T	75	F	F	c0
F	25	F	F	c0
F	14	T	F	c0
F	69	T	T	c1

Genere los *item sets* con soporte mayor o igual que 5.

Figura 1.1: Inspiración pregunta *Generación Item Sets*

2. ***Generación Reglas Asociación***: dado un conjunto de datos con una cantidad determinada de atributos e iteraciones, obtener las reglas de asociación con soporte y confianza mayores o iguales que unos valores específicos.

   En la siguiente imagen se muestra un ejemplo de pregunta de los apuntes de la asignatura de Minería de Datos que sirvió como inspiración para esta pregunta.

Sea el siguiente conjunto de datos:

a1	a2	a3	a4	clase
F	31	T	T	N
T	86	T	F	P
T	91	T	F	P
F	22	F	F	N
T	98	T	T	P
T	08	F	F	P
T	82	T	F	P
F	07	T	F	N
F	84	T	F	P
F	63	F	T	N
T	75	F	T	N

Obtenga aquellas reglas de asociación con soporte y confianza mayores o iguales que, respectivamente, 5 y 80 %. Discretice el atributo a2 usando el umbral 50.

Figura 1.2: Inspiración pregunta *Generación Reglas Asociación*

3. ***Ampliación Item Sets***: dado un conjunto que contiene un numero determinado de n-item sets, calcular los posibles n+1-item sets.

   En la siguiente imagen se muestra un ejemplo de pregunta de los apuntes de la asignatura de Minería de Datos que sirvió como inspiración para esta pregunta.

Se dispone de los siguientes 3-item sets: (A,B,D), (A,B,F), (A,D,F), (B,C,E), (B,C,G), (B,D,F), (B,E,G), (C,E,G), (D,F,G)

¿Cuáles son los posibles 4-item sets? Justifique la respuesta.

Figura 1.3: Inspiración pregunta *Ampliación Item Sets*

Objetivos del proyecto

2.1. Objetivos técnicos

Los principales objetivos técnicos perseguidos mediante el desarrollo de la aplicación han sido:

1. Desarrollar una Interfaz Gráfica de Usuario para facilitar el uso de la aplicación.
2. Establecer los parámetros necesarios para generar los enunciados de las preguntas.
3. Generar los conjuntos de datos aleatorios para las preguntas en base a los parámetros establecidos.
    - Para la pregunta de *Ampliación Item Sets* generar un conjunto de n-item sets aleatorios.
    - Para las preguntas de *Generación Reglas Asociación* y *Generación Item Sets* generar un conjunto de datos aleatorios que pueden tomar los valores T o F y en ocasiones, valores numéricos.
4. Generar los enunciados de las preguntas en base a los conjuntos de datos generados.
5. Generar opciones verdaderas y falsas para las preguntas en base a los conjuntos de datos generados.
    - Para las preguntas de *Ampliación Item Sets* las opciones consistirán en n+1-item sets generados a partir de un conjunto de n-item sets.

- Para las preguntas de *Generación Reglas Asociación* las opciones consistirán en reglas de asociación generadas a partir de un conjunto de datos.
- Para las preguntas de *Generación Item Sets* las opciones consistirán en item sets generados a partir de un conjunto de datos.

6. Establecer una puntuación determinada para cada opción que se calcule automáticamente dependiendo de la cantidad de opciones que sean verdaderas y falsas.
7. Generar preguntas a partir de los enunciados y opciones que se han generado.
8. Generar un banco de preguntas de un tipo concreto a partir de las preguntas que se han obtenido.
9. Traducir ese banco de preguntas a un formato .XML importable en Moodle.
10. Exportar el fichero .XML en una ruta especificada y con un nombre especificado.

2.2. Objetivos académicos

Algunos de los objetivos académicos que se han conseguido son los siguientes:

1. Aplicar los conocimientos adquiridos en la carrera para desarrollar una aplicación práctica.
2. Mejorar las habilidades de programación.
3. Aprender a generar preguntas y respuestas automáticamente en el ámbito de minería de datos.
4. Entender y aplicar los conceptos de item sets y reglas de asociación en la minería de datos.
5. Comprender el proceso de discretización de atributos numéricos en los conjuntos de datos.
6. Adquirir experiencia en la creación de ficheros en formato MoodleXML y su importación en el sistema de gestión de aprendizaje Moodle.

7. Adquirir nuevos conocimientos y habilidades en herramientas y tecnologías relacionadas con la ingeniería informática.

8. Mejorar las habilidades en presentación y comunicación, así como en la defensa y justificación del trabajo realizado.

2.3. Objetivos personales

Los objetivos personales que se han llevado a cabo han sido los siguientes:

1. Adquirir experiencia práctica en el desarrollo de un proyecto.

2. Mejorar habilidades de resolución de problemas y pensamiento lógico.

3. Desarrollar habilidades para la generación automática de preguntas y respuestas.

4. Aumentar la confianza en las propias habilidades técnicas y académicas.

5. Aprender a trabajar de manera independiente y a cumplir con un plazo establecido.

Conceptos teóricos

3.1. Introducción

Las reglas de asociación en minería de datos son una técnica utilizada para encontrar patrones y relaciones ocultas en un conjunto de datos. Estas reglas se basan en el principio de que si dos o más elementos aparecen juntos con frecuencia en un conjunto de datos, es probable que estén relacionados de alguna manera. [20]

Para entender esto del todo hay que mencionar el concepto de item set. Un item set es simplemente un conjunto de elementos o ítems que aparecen juntos con frecuencia en un conjunto de datos. Por ejemplo, en un supermercado, un item set podría ser un conjunto de productos que un cliente compra juntos, como leche y pan. [3]

Las reglas de asociación son conclusiones que se pueden sacar de los item sets. Por ejemplo, si se encuentra que muchos clientes compran leche y pan juntos, se puede crear una regla de asociación que indique que los clientes que compran leche también suelen comprar pan.

Estas técnicas se utilizan en la vida real para tomar decisiones basadas en patrones y relaciones en grandes conjuntos de datos. Por ejemplo, en un supermercado, se pueden utilizar para decidir cómo colocar los productos en las estanterías o para personalizar las promociones y ofertas para los clientes. En la publicidad en línea, se pueden utilizar para personalizar los anuncios que se muestran a los usuarios basados en sus intereses y comportamientos.

3.2. Generación de item sets

Para generar un conjunto de n-item sets en minería de datos, se siguen los siguientes pasos: [2]

1. Seleccionar un conjunto de datos: El primer paso es seleccionar un conjunto de datos que contenga información sobre los elementos o ítems que se quieren analizar.
2. Identificar los elementos o ítems individuales: El siguiente paso es identificar los elementos o ítems individuales en el conjunto de datos. Por ejemplo, en un supermercado, los elementos individuales podrían ser productos específicos como leche, pan, huevos, etc.
3. Calcular la frecuencia de cada item set: A continuación, se cuenta la frecuencia con la que cada item set aparece en el conjunto de datos. Un item set es un conjunto de n elementos o ítems que aparecen juntos. Por ejemplo, en un supermercado, un item set de dos elementos sería leche y pan.
4. Filtrar los item sets con una frecuencia mínima: Una vez que se han calculado las frecuencias de cada item set, se puede filtrar los item sets que no cumplen con una frecuencia mínima establecida previamente. Esto se hace para evitar item sets que no sean significativos o que aparezcan con poca frecuencia.
5. Generar el conjunto de n-item sets: Finalmente, se pueden generar el conjunto de n-item sets a partir de los item sets que cumplen con la frecuencia mínima.

Para obtener los n+1-item sets a partir de los n-item sets, se siguen los siguientes pasos:

1. Tomar dos item sets de n elementos: El primer paso es tomar dos item sets de n elementos y compararlos para ver si tienen n-1 elementos en común.
2. Unir los item sets: Si los dos item sets tienen n-1 elementos en común, se pueden unir para formar un item set de n+1 elementos.
3. Calcular la frecuencia del item set unido: A continuación, se cuenta la frecuencia con la que aparece el item set unido en el conjunto de datos.

4. Filtrar los item sets unidos con una frecuencia mínima: Finalmente, se pueden filtrar los item sets unidos que no cumplen con una frecuencia mínima establecida previamente.

Este proceso se podría repetir para obtener item sets de n+2 elementos, n+3 elementos, etc. hasta que se alcance el número de elementos deseado.

3.3. Generación de reglas de asociación

Para obtener reglas de asociación de un conjunto de datos, se siguen los siguientes pasos: [2]

1. Discretizar atributos numéricos, si los hay: en caso de que haya atributos numéricos, es necesario discretizarlos. Esto significa convertirlo en un atributo categórico, es decir, dividir un rango continuo de valores en un conjunto finito de categorías o intervalos.

2. Generar el conjunto de item sets: El primer paso es generar el conjunto de item sets a partir del conjunto de datos, siguiendo los pasos que se describieron en la respuesta anterior.

3. Establecer el soporte mínimo: El siguiente paso es establecer un soporte mínimo, que es un porcentaje o un número que representa la frecuencia mínima con la que un item set debe aparecer en el conjunto de datos para ser considerado significativo.

$$\text{soporte}(A \implies C) = \text{soporte}(A \cup C)$$

4. Generar las reglas de asociación: A continuación, se pueden generar las reglas de asociación a partir de los item sets que cumplen con el soporte mínimo. Una regla de asociación es una relación entre dos item sets que indica que si un item set aparece en el conjunto de datos, es probable que también aparezca el otro item set. Por ejemplo, una regla de asociación podría ser "si un cliente compra leche, es probable que también compre pan".

5. Establecer la confianza mínima: El siguiente paso es establecer una confianza mínima, que es un porcentaje que representa el grado de confianza en la regla de asociación. La confianza se puede calcular

como la proporción de veces que aparece el item set de destino en el conjunto de datos, dada la presencia del item set de origen.

$$\text{confianza}(A \implies C) = \frac{\text{soporte}(A \cup C)}{\text{soporte}(A)}$$

6. Filtrar las reglas de asociación con una confianza mínima: Finalmente, se pueden filtrar las reglas de asociación que no cumplen con la confianza mínima establecida previamente.

3.4. Algoritmo Apriori

Apriori es un algoritmo de minería de datos utilizado para generar reglas de asociación en un conjunto de datos. El algoritmo se basa en la idea de que si un item set es frecuente, entonces todos los subconjuntos de ese item set también deben ser frecuentes. [8]

El algoritmo se divide en dos fases similares a lo explicado anteriormente:

1. Generación de item sets: En esta fase, se generan todos los item sets posibles del conjunto de datos, y se eliminan aquellos que no cumplen con el soporte mínimo establecido previamente.

2. Generación de reglas de asociación: En esta fase, se generan reglas de asociación a partir de los item sets frecuentes obtenidos en la primera fase. Se utiliza un proceso de *candidate generation* para generar reglas de asociación, en el que se combinan item sets para generar nuevas reglas, y se eliminan aquellas que no cumplen con la confianza mínima establecida previamente.

El algoritmo Apriori se caracteriza por su eficiencia en términos de tiempo de ejecución y memoria, ya que utiliza un proceso de eliminación de item sets infrecuentes para reducir el espacio de búsqueda. Sin embargo, también tiene algunas limitaciones, como la necesidad de establecer un soporte mínimo y una confianza mínima adecuados, y la incapacidad de manejar datos no estructurados.

```
Apriori(T, ε)
    L_1 ← {large 1 − itemsets}
    k ← 2
    while L_{k−1} ≠ ∅
        C_k ← {a ∪ {b} | a ∈ L_{k−1} ∧ b ∉ a} − {c | {s | s ⊆ c ∧ |s| = k − 1} ⊈ L_{k−1}}
        for transactions t ∈ T
            C_t ← {c | c ∈ C_k ∧ c ⊆ t}
            for candidates c ∈ C_t
                count[c] ← count[c] + 1
        L_k ← {c | c ∈ C_k ∧ count[c] ≥ ε}
        k ← k + 1
    return ⋃_k L_k
```

Figura 3.1: Algoritmo Apriori

[8]

3.5. Ejemplo práctico

Se va a hacer un ejemplo de cómo calcular las reglas de asociación que superan un soporte mínimo y una confianza mínima en el contexto de una tienda.

Supongamos que tenemos el siguiente conjunto de datos de transacciones de compra en una tienda:

- Transacción 1: pan, leche, huevos
- Transacción 2: pan, leche
- Transacción 3: leche, huevos
- Transacción 4: pan, huevos
- Transacción 5: pan, leche

Lo primero es calcular el soporte de cada item set en el conjunto de datos.

Luego, supongamos que se establece un soporte mínimo de 6 y una confianza mínima del 70 %.:

Para el item set $\{pan, leche\}$, el soporte es $3/5 = 0.6$, que cumple con la condición de ser igual o mayor que el soporte mínimo de 6 (0.6), por lo que se puede generar una regla a partir de este item set.

- Regla: "si un cliente compra pan, es muy probable que también compre leche"(soporte: 3/5, confianza: 3/4)

La confianza de la regla se puede calcular dividiendo el soporte del item set $\{pan, leche\}$ por el soporte del item set $\{pan\}$. En este caso, la confianza es $3/4 = 0.75$.

La confianza es mayor que la confianza mínima del 70 % (0.7), por lo que se puede considerar esta regla.

De este modo, se puede afirmar que la regla "si un cliente compra pan, es muy probable que también compre leche"(soporte: 3/5, confianza: 3/4) supera tanto el soporte mínimo como la confianza mínima y, por lo tanto, es una regla de asociación válida.

Este proceso se puede repetir para todos los item sets y sus respectivas reglas para encontrar todas las reglas de asociación que superan el soporte mínimo y la confianza mínima.

Item set	**Soporte**
$\{pan\}$	4/5
$\{leche\}$	4/5
$\{huevos\}$	2/5
$\{pan, leche\}$	3/5
$\{pan, huevos\}$	2/5
$\{leche, huevos\}$	2/5
$\{pan, leche, huevos\}$	1/5

Tabla 3.1: Ejemplo práctico item sets

Técnicas y herramientas

4.1. Técnicas

SCRUM

SCRUM es una metodología ágil de gestión de proyectos que se enfoca en la entrega continua de valor al cliente mediante un proceso iterativo e incremental. Se divide en ciclos cortos de trabajo, llamados sprints, en los cuales un equipo trabaja de forma colaborativa en tareas específicas para completar una parte del proyecto. El objetivo es entregar un producto o servicio funcional al final de cada sprint. [16]

En este proyecto se ha seguido este tipo de metodología al organizarse en sprints de una duración de dos semanas con diferentes tareas a completar en cada uno de ellos.

UML

UML (Unified Modeling Language) es un lenguaje de modelado de sistemas que permite representar de manera gráfica y estandarizada los distintos aspectos de un sistema, como su estructura, comportamiento, procesos y relaciones. UML se utiliza comúnmente en el diseño y planificación de proyectos de software, así como en la documentación y comunicación del mismo. [15]

En este proyecto, se ha aplicado UML de la siguiente manera:

1. Se han identificado los requisitos del sistema y definido su alcance.

2. Se ha diseñado la arquitectura y estructura del sistema mediante diagramas de clases.

3. Se ha modelado el comportamiento del sistema mediante diagramas de estado y actividad.

4. Se ha utilizado el modelo como base para el desarrollo del sistema y como documentación para su mantenimiento y evolución.

4.2. Herramientas

Java

Java es un lenguaje de programación de alto nivel, orientado a objetos y de propósito general. Fue desarrollado por Sun Microsystems (ahora propiedad de Oracle) a finales de los años 90. Se caracteriza por ser multiplataforma, lo que significa que los programas escritos en Java pueden ejecutarse en diferentes sistemas operativos sin necesidad de ser recompilados, y su arquitectura "write once, run anywhere"(escribir una vez, ejecutar en cualquier lugar).

Java es un lenguaje orientado a objetos, lo que significa que utiliza el paradigma de programación orientada a objetos (POO) para construir programas. La POO se enfoca en la encapsulación, la herencia y el polimorfismo, tres conceptos fundamentales que permiten crear un código más organizado y reutilizable. [17]

Java cuenta con una gran cantidad de librerías y paquetes pre-construidos para facilitar el desarrollo de aplicaciones.

Para la realización de este proyecto se ha utilizado Java JDK 19.

Página web de la herramienta: https://www.java.com/es/

Eclipse

Eclipse es un entorno de desarrollo integrado (IDE) de código abierto ampliamente utilizado para la programación en varios lenguajes de programación, incluyendo Java, C++, y PHP. Ofrece características como depuración, control de versiones, y herramientas de construcción integradas. También tiene un sistema de plugins para extender las funcionalidades del IDE. Es muy popular entre los desarrolladores debido a su capacidad de adaptarse a diferentes necesidades de desarrollo y su gran comunidad de usuarios. [11]

En este proyecto se ha utilizado Eclipse IDE 2022-09.

Página web de la herramienta: https://eclipse.org/org/

Maven

Maven es una herramienta software que se utiliza en la construcción y administración de proyectos Java.

Esta herramienta utiliza un POM (Project Object Model) mediante la creación de un fichero en formato .XML en el que se describe el proyecto software a construir y se especifican las dependencias externas del proyecto, así como otras tareas.[18]

En este proyecto se ha utilizado Apache Maven 3.8.7.

Página web de la herramienta: https://maven.apache.org/

Weka

Weka es una herramienta de minería de datos open source que proporciona una amplia variedad de algoritmos de aprendizaje automático para la clasificación, regresión, agrupamiento y visualización de datos.[9]

Weka proporciona varios algoritmos de aprendizaje automático para generar reglas de asociación a partir de datos de transacciones, entre ellos, el algoritmo *Apriori*.

En este proyecto se ha utilizado weka 3.8.6.

Página web de la herramienta: https://www.cs.waikato.ac.nz/ml/weka/

Git

Git es un sistema de control de versiones de código. Es una herramienta que permite a los desarrolladores llevar un registro detallado de los cambios realizados en un proyecto de software a lo largo del tiempo. Con Git, se pueden almacenar y gestionar de forma eficiente las diferentes versiones de un proyecto, lo que facilita la colaboración y la resolución de conflictos entre los desarrolladores.

GitHub

GitHub es una plataforma en línea que proporciona almacenamiento y control de versiones para proyectos de software. Los desarrolladores pueden crear repositorios para su proyecto, trabajar en ellos y colaborar con otros desarrolladores. También proporciona herramientas para ayudar a los desarrolladores a colaborar y manejar sus proyectos de manera eficiente, como un sistema de seguimiento de errores, gestión de tareas, comentarios y discusiones. Es un lugar donde se almacena y se comparte el código fuente y se sigue las últimas tendencias en tecnología.[13]

Enlace al repositorio de este proyecto: `https://github.com/srevilla/TFG-22.11`

ZenHub

ZenHub es una herramienta de gestión de proyectos en línea que se integra con GitHub. Es una plataforma de colaboración que permite a los equipos de desarrollo de software planificar, organizar y seguir el progreso de sus proyectos de manera eficiente.[6]

Con ZenHub, los equipos pueden crear tableros de proyectos personalizados, donde pueden asignar tareas a miembros del equipo, establecer prioridades y seguir el progreso en tiempo real.

Página web de la herramienta: `https://www.zenhub.com/`

Moodle

Moodle es un sistema de gestión del aprendizaje de código abierto que ayuda a los profesores a crear y administrar cursos en línea, proporciona herramientas de colaboración y una gran cantidad de características y plugins que pueden ser utilizadas para personalizar el funcionamiento y la apariencia de un curso.[4]

En este proyecto se ha utilizado para importar los cuestionarios generados en formato .XML, ya que era el principal objetivo de la aplicación.

Página web de la herramienta: `https://docs.moodle.org/`

XML

XML es un lenguaje de etiquetas utilizado para describir y transmitir información estructurada. Es un lenguaje de etiquetado similar al HTML, pero más flexible y menos estricto en su sintaxis.[12]

En este proyecto se ha utilizado XML para crear cuestionarios. Los cuestionarios se pueden escribir en un archivo XML y luego importar al sistema de Moodle. Al utilizar XML, los cuestionarios son fáciles de crear y editar, y también son fácilmente transferibles entre diferentes sistemas.

LaTeX

LaTeX es un sistema de composición de texto utilizado principalmente para la creación de documentos científicos y académicos, es especialmente útil en la creación de ecuaciones matemáticas, tablas, gráficos y figuras y proporciona herramientas para crear índices, bibliografías y tablas de contenido automáticamente.[14]

Overleaf

Overleaf es una plataforma de colaboración de documentos basada en *LaTeX* en línea, que permite a los usuarios crear y editar documentos académicos y científicos de forma colaborativa y fácil de usar, con características de colaboración en tiempo real y compilación y visualización en línea.[19]

Página web de la herramienta: https://es.overleaf.com/

Diagrams.net

Diagrams.net (anteriormente llamado Draw.io) es una herramienta en línea gratuita para la creación y edición de diagramas y gráficos. Se ha usado para crear los diagramas UML de los anexos, entre los que se incluyen diagramas de clases, diagramas de secuencia, etc... Es una opción popular para quienes buscan una herramienta fácil de usar para crear diagramas profesionales sin la necesidad de software de pago costoso. Además, permite la colaboración en tiempo real y la integración con diferentes servicios en línea, como Google Drive y Dropbox. [10]

Página web de la herramienta: https://app.diagrams.net/

Aspectos relevantes del desarrollo del proyecto

Este apartado recoge los aspectos más interesantes del desarrollo del proyecto. Se abordan los aspectos relacionados con la justificación de las herramientas y los caminos elegidos para desarrollar el software. Se explica la importancia de estas decisiones y cómo afectan a la eficacia, la eficiencia y la calidad del proyecto. También se describen las ventajas y desventajas de las herramientas y los procesos elegidos y cómo se hizo la elección final.

En este apartado también se explica cómo se llevó a cabo el desarrollo del proyecto desde el principio hasta el final.

5.1. Inicio del proyecto

La idea del proyecto se tomó de la lista de TFGs ofertados por la UBU. Hacer una aplicación para la universidad de Burgos de forma que ayude a los profesores a generar cuestionarios, ya sea para exámenes online o para cuestionarios de autoevaluación se vio como una propuesta interesante, por lo que se contactó al tutor correspondiente a este proyecto. Una vez se contactó al tutor, se hizo una reunión para planificar el proyecto. En esta reunión se establecieron las herramientas a utilizar, las metodologías, los objetivos y los plazos del proyecto.

Se decidió utilizar el lenguaje de programación Java para el desarrollo de esta apliación y el entorno de desarrollo Eclipse. El motivo de esto es porque tanto Eclipse como Java son herramientas que ya se habían utilizado en proyectos anteriores y se habían trabajado con ellas en la universidad, de modo que se conocían más a fondo que otro tipo de herramientas. Además de

eso, Eclipse es una herramienta muy poderosa que facilita la programación, especialmente en la refactorización y depuración del código, que cuenta con muchos plugins disponibles, como el control de tareas, control de versiones, pruebas, etc.

Para la gestión de dependencias externas se decidió utilizar Maven. El motivo de esto es que Maven es una herramienta de gestión de proyectos y dependencias en Java muy potente ya que permite automatizar la descarga e integración de bibliotecas externas en un proyecto, de forma que se simplifica y agiliza el proceso de desarrollo al resolverse dependencias de forma rápida y confiable. Además, Maven proporciona un formato estándar para la estructura del proyecto y una serie de reglas para la construcción y el despliegue de aplicaciones, lo que aumenta la consistencia y la calidad del código.

Para la realización de este proyecto se decidió utilizar una metodología ágil. El motivo de esto es que ofrecen varias ventajas en comparación con las metodologías tradicionales no ágiles. La principal ventaja de las metodologías ágiles es su flexibilidad y capacidad de adaptación a los cambios. En lugar de seguir un plan rigido y establecido, las metodologías ágiles permiten un enfoque iterativo e incremental que permite una mejor gestión de los cambios.

Se decidió utilizar SCRUM como metodología ágil a implementar en este proyecto. Las razones de esto fue que se trata de una de las metodologías ágiles más populares y utilizadas, que promueve la transparencia y la inspectabilidad en todas las fases del proyecto, lo que permite una mejor toma de decisiones y una mayor eficacia en la resolución de problemas. Además, SCRUM utiliza sprints regulares para mantener el enfoque en los objetivos y mejorar la productividad. También ofrece una estructura sólida para la gestión de tareas y el seguimiento de los progresos, lo que permite una mejor gestión del tiempo y una mejor gestión del riesgo.

Al ser un proyecto educativo, no se ha seguido esta metodología al 100 % ya que no se ha trabajado en equipo si no que ha sido un proyecto individual, por lo tanto, no se han hecho reuniones diarias y otras tareas típicas de la metodología SCRUM. No obstante, se ha intentado adaptar esta metodología lo mejor posible a un proyecto educativo e individual de la siguiente forma:

- Se planificaron *sprints* como estrategia para completar una cantidad de trabajo establecida.

- Se realizaron reuniones con el tuto al final de cada sprint, en las que se comentaban los objetivos que se habian conseguido, los que quedaban por hacer y se establecían los objetivos del próximo *sprint*.
- La duración de cada *sprint* se estableció de dos semanas.
- Se utilizaron *story points* como medida para estimar el esfuerzo que se realizó en cada tarea.
- Se utilizaron gráficos *burndown* para monitorizar el progreso.

En cuanto al sistema de cotrol de versiones a utilizar, se optó por *Git* ya que permite gestionar proyectos de una forma eficiente y cuenta con muchas ventajas como la facilidad de resolución de conflictos, permite revertir cambios, ofrece una amplia documentación y soporte comunidad, y permite una fácil integración con otros servicios y herramientas.

Para alojar el proyecto en la nube se acordó, siguiendo los consejos del tutor, usar *GitHub* porque ofrece varias ventajas como su amplia comunidad de desarrolladores, herramientas de colaboración en tiempo real, integración con diferentes servicios y herramientas, control de versiones robusto y seguimiento de errores, así como una amplia documentación y soporte en línea. Además, ofrece opciones gratuitas y de pago para proyectos tanto personales como empresariales. Además de esto, se estaba familiarizado con esta herramienta ya que su uso había sido habitual en alguna asignatura impartida en la universidad de Burgos por lo que se vio como la opción más completa y viable.

Conocimientos necesarios

El proyecto requería recordar conocimientos sobre asignaturas ya cursadas en la carrera, entre las que se encuentran las siguientes:

- **Metodología de la programación**: se han aplicado los conocimientos aprendidos en esta asignatura para desarrollar una aplicación en java orientada a objetos utilizando Eclipse. Para ello, se han utilizado habilidades como la abstracción, encapsulamiento, herencia o polimorfismo.
- **Minería de datos** se ha repasado el tema de reglas de asociación correspondiente a esta asignatura para poder implementar el algoritmo. En este tema, caben destacar conceptos como el de item sets, reglas de asociación o el algoritmo apriori.

- **Ingeniería del software**: los conocimientos aprendidos en esta asignatura se han utilizado para diseñar y desarrollar el proyecto asegurando su fiabilidad, seguridad y calidad, así como la capacidad para diseñar y evaluar interfaces persona computador que garanticen la accesibilidad y usabilidad a la aplicación.
- **Gestión de proyectos**: esta asignatura ha contribuido a la organización, planificación y gestión del proyecto, utilizando técnicas como la estimación o planificación de tareas, así como a la utilización de la metodología *SCRUM*.

Además de repasar estos conocimientos, se tuvieron que aprender algunos nuevos, siendo el más destacable el de cómo crear interfaces gráficas en Java. Para ello se buscó información en diferentes sitios web y se vieron algunos videos explicativos del proceso.

5.2. Desarrollo

1. Se comenzó escribiendo el código para la pregunta *Ampliación Item Sets*. Para ello, se construyeron las clases y los métodos necesarios para desarrollar esta pregunta. Estas clases consistía en una versión *primitiva* de lo que actualmente se tiene en los siguientes paquetes:
   - `/src/es/ubu/inf/tfg/generador/datos/conjuntoitemsets`
   - `/src/es/ubu/inf/tfg/generador/preguntas/ampliacionitemsets`
   - `/src/es/ubu/inf/tfg/traductor`

   Es decir, se hizo el generador de conjuntos de n-item sets y se calcularon todos sus posibles n+1-item sets. Una vez hecho esto, se hizo un clasificador para que clasificara esas posibles soluciones en verdaderas y falsas, comprobando previamente que existían soluciones para ese conjunto de datos con los parámetros especificados. Posteriormente, se hizo un traductor simple para traducir esta pregunta a un formato .XML importable en *Moodle*. Para ello se creó una plantilla que es utilizada por la clase Traductor para traducir la pregunta.
2. Para la pregunta *Reglas Asociación* se siguió una lógica similar a la anterior. Es decir, se creó una clase en la que se generaba el conjunto de datos a utilizar y se clasificaban las reglas de asociación a partir de ese conjunto en verdaderas y falsas, en caso de que tuviese solución. La

lógica del traductor era la misma, por lo que se empezaron a utilizar más conceptos de POO como la utilización de interfaces y clases abstractas, de forma que no era necesario crear otro traductor específico para esta pregunta sino que se podían utilizar estas técnicas para que el traductor funcionase independientemente del tipo de pregunta que sea. Para ello, se creó una interfaz común para todos los tipos de preguntas que se encarga de generar el banco de preguntas. Por lo tanto, se reorganizó el código en diferentes paquetes a como estaban originalmente, de forma que la estructura fuera algo así:

- Se tiene un paquete generador que contiene diferentes subpaquetes.
- Cada subpaquete se corresponde con una pregunta.
- En cada subpaquete hay una clase para la configuración, otra clase para generar el banco de preguntas y otra gente para generar las preguntas necesarias para generar el banco de preguntas.

3. Hasta este punto, todo funcionaba por comandos introducidos desde consola. Para facilitar la utliziación de la aplicación se decidió crear una interfaz gráfica. Con la interfaz gráfica es posible decidir, en la configuración de cada pregunta, qué campos se quieren generar aleatoriamente y cuales no, introduciendo el valor a mano en estos últimos. Por lo tanto, se cambiaron las clases anteriores de forma que esto fuese posible, y además, se creó usando *JFrame* las ventanas de la interfaz, siendo necesario crear un *Main* desde cero totalmente nuevo.

4. En un principio, estas preguntas eran las que originalmente se iban a poder generar en la aplicación. Posteriormente, se decidió añadir una nueva, bajo el título de *Generación Item Sets*. Para esta pregunta se vio que se necesitaba generar un conjunto de datos de la misma forma que para la pregunta anterior. Por lo que se decidio organizar de nuevo la estructura de paquetes de forma que dentro del paquete del generador haya dos subpaquetes, uno para generar los datos y otro para generar las preguntas. En el de preguntas se colocaron los tres subpaquetes que había para cada pregunta, y en el de datos se crearon dos subpaquetes nuevos, uno para generar el conjunto de item sets y otro para generar el conjunto de datos para reglas de asociación. De esta manera, ambas preguntas pueden utilizar el mismo generador de datos sin tener que escribir el mismo código.

5. Una vez se terminó el programa, se incluyó un tratamiento de excepciones creando para ello una nueva clase en el dominio llamada

UnexpectedException. Esta excepción se lanza cuando el programa se encuentra con una excepción que no extiende de *RunTimeException* que no sabe como tratar. De esta forma, se traduce a *UnexpectedException* que sí que extiende de *RunTimeException*.

5.3. Proceso de generación de pregunta de reglas de asociación

En este apartado se va a explicar cómo se genera una pregunta de tipo *Generación Reglas de Asociación*, ya que es la más importante del programa. La estructura es similar para el resto de preguntas.

Para generar una pregunta se necesita un generador, tanto datos como de preguntas.

El generador de datos se compone de dos clases:

- **GeneradorConjuntoDatos**: esta clase recibe la configuración para crear un conjunto de datos. Tiene un método public llamado *crearConjuntoDatos()*. Este método llama a otros métodos private para generar crear los atributos del conjunto de datos, crear las transacciones con sus respectivos datos (pueden ser T o F) y, además, tratar atributos numéricos en caso de que haya. Esta clase utiliza la librería *Weka* para generar el conjunto de datos necesario.
- **ConjuntoDatos**: esta clase se compone de:
  - *datosEnunciado*: conjunto de datos de tipo *Instances* en el que los valores de los atributos numéricos se muestran en formato decimal. Es el que se muestra en el enunciado.
  - *datosCalculos*: conjunto de datos de tipo *Instances* en el que los valores de los atributos numéricos se muestran en formato nominal. Es el que se usa en el código para las operaciones necesarias.
  - *intervalos*: cadena de texto que contiene los intervalos en los que se discretizarán los atributos numéricos. Se usa en el enunciado.

El generador de preguntas se compone de tres clases:

- **ConfigReglasAsociacion**: contiene todos los valores de los parámetros establecidos en la ventana de configuración de la pregunta.

- **GeneradorPreguntaReglasAsociacion**: tiene un metodo public llamado *generarPregunta()*. Este metodo es muy importante, ya que en él se instancia el conjunto de datos y se genera toda la pregunta como tal. Este método llama a otras funciones private, de forma que, en total, realiza los siguientes pasos:

  1. Inicializa un objeto *random* de la clase *Random*, que es utilizado para generar números aleatorios.
  2. Calcula el número de respuestas verdaderas y el número de respuestas falsas en base al número de opciones. El número de respuestas verdaderas es igual a un número aleatorio entre 1 y (*numOpciones*-1) y el número de respuestas falsas es igual a *numOpciones* menos el número de respuestas verdaderas.
  3. Inicializa una variable *tieneSolucion* en false.
  4. Se entra en un bucle *while* que se repetirá hasta que *tieneSolucion* sea igual a true.
  5. Se crea un objeto *apriori* de la clase *Apriori*. Para ello se utiliza la librería *Weka*.
  6. Se crean dos listas, *reglasVerdaderas* y *reglasFalsas*, que almacenarán las reglas que cumplen ciertos criterios.
  7. Se establecen los valores para el soporte mínimo, la confianza mínima y el número de reglas en *apriori*. Esto se hace para que este objeto cree reglas en base a estos valores fijos, y son siempre menores que los valores de soporte y confianza que se establecen en la configuración.
  8. Se crea un objeto *conjuntoDatos* utilizando el método *crearConjuntoDatos* en el objeto *gcd*.
  9. Se obtienen los datos de cálculo utilizando el método *getDatosCalculos* en *conjuntoDatos*.
  10. Se ejecuta el método *buildAssociations* en *apriori* utilizando los datos de cálculo, de forma que construye reglas de asociación a partir de un conjunto de datos y de los valores mínimos fijos establecidos previamente.
  11. Se obtienen las reglas asociadas utilizando el método *getAssociationRules* en *apriori*.
  12. Se recorren las reglas y se evalúan su confianza y soporte. Si la confianza y el soporte cumplen con los criterios establecidos, la regla se añade a *reglasVerdaderas*, de lo contrario se añade a *reglasFalsas*.

13. Se ejecuta el método *tieneSolucion* para verificar si hay suficientes reglas para generar una pregunta con opciones de respuesta.
14. Si *tieneSolucion* es igual a true, se generan las opciones de respuesta utilizando el método *generarOpciones*.
15. Se establece la puntuación de las opciones verdaderas y falsas.
16. Se crean las opciones verdaderas y falsas a partir de las reglas verdaderas y falsas y se le asigna un peso a cada una de ellas.
17. El peso de las opciones verdaderas se establece con la función establecerPuntuacion y es igual a la puntuación que se le asigna a una opción verdadera, que se obtiene multiplicando el número de opciones verdaderas por una constante. El peso de las opciones falsas se establece restando la puntuación que se le asigna a una opción falsa (obtenida de la misma forma que la verdadera) de 0.
18. Por último, se crea un objeto *Pregunta* con las opciones generadas, un enunciado obtenido de *obtenerEnunciado* y un título obtenido de *obtenerTítulo*, y se retorna.

- **GeneradorBancoPreguntasReglasAsociacion**: implementa el método *generarBancoPreguntas(T)* de la interfaz. En este caso, *T* toma el valor de *ConfigReglasAsociacion*. Este método comienza declarando una variable *contador* que se inicializa en 0 y una lista de preguntas *listaPreguntas* que se inicializa como una lista vacía. Luego, dentro de un bucle *while*, se crea un objeto *GeneradorPreguntaReglasAsociacion* y se usa para generar una pregunta, llamando al método explicado anteriormente, *generarPregunta()*. Si esta pregunta no está en la lista de preguntas, se agrega a la lista y se incrementa el contador. El bucle se repite hasta que el contador sea igual al número de preguntas especificado en la configuración. Finalmente, se devuelve un nuevo objeto de tipo *BancoPreguntas* que contiene la lista de preguntas generadas.

5.4. Pruebas

Se generaron varios bancos de preguntas con diferentes valores para sus parámetros de todos los tipos de preguntas, y se resolvieron a mano cada una de ellas, comprobando de esta forma el correcto funcionamiento de la aplicación. Se importaron estos bancos de preguntas en *Moodle* para ir comprobando que las opciones que la apliación marcaba como verdaderas, realmente lo eran, de la misma forma que para las opciones falsas.

Trabajos relacionados

El tutor proporcionó los enlaces de los repositorios de *GitHub* de algunos proyectos similares a este en los que poder inspirarse. Dichos proyectos son también generadores de cuestionarios para *Moodle*, pero de diferentes asignaturas y temas.

6.1. PLQUIZ

PLQUIZ[1] es una herramienta de escritorio que permite generar preguntas de test aleatorias (tipo quiz, cloze, de texto libre...) sobre problemas de algoritmos de análisis léxico. El formato utilizado para generar las preguntas es .XML para importarse a entornos virtuales de aprendizaje (Moodle), y .TEX para su impresión en papel a través de la obtención de código LaTeX.[1]

Se han seguido sus pasos en el formato del traductor, al decidir crear una clase *Plantilla* y tener las plantillas en formato .XML en el paquete *resources*

6.2. PLGRAM

PLGRAM[2] es una aplicación que a partir de una gramática obtiene el análisis sintáctico descendente $LL(1)$ y los análisis sintácticos ascendente $SLR(1)$, $LR(1$)y $LALR(1)$. Se obtienen los conjuntos *FIRST* y *FOLLOW* así como la tabla de análisis sintáctico predictivo (*TASP*) y las tablas de *ACCIÓN y de IR A*. Asimismo se puede obtener para los distintos análisis la traza de una cadena.

[1]Roberto Izquierdo Amo, Julio 2014, Universidad de Burgos
[2]Víctor Renuncio Tobar, Julio 2016, Universidad de Burgos

Con la aplicación se puede generar un archivo .XML para importar a la plataforma *Moodle* con la finalidad de generar de forma automática cuestionarios de análisis sintáctico. También se puede generar un fichero en formato LaTeX que se puede utilizar para realizar pruebas escritas. Se puede obtener un documento en el que el cuestionario esté completo, con las respuestas correctas y otro documento en el que el cuestionario está vacío, pensado para ser respondido por un tercero.[7]

6.3. ProgDinQuiz

ProgDinQuiz[3] es una herramienta de escritorio que ayuda a generar preguntas de test aleatorias (tipo quiz, cloze...) sobre tipos de problema de programación dinámica.[5]

[3]Asier Alonso Morante, Septiembre 2016, Universidad de Burgos

Conclusiones y Líneas de trabajo futuras

7.1. Conclusiones

En el desarrollo de este proyecto, no solo se ha logrado el objetivo de crear una herramienta útil para los profesores de la Universidad de Burgos, sino que también se han perfeccionado habilidades relacionadas con la programación y el desarrollo de software, al ser la primera vez que se hace un programa totalmente desde cero. Durante el desarrollo de la aplicación, se han puesto en práctica conocimientos adquiridos en materias relacionadas con la programación y se han adquirido nuevos conocimientos sobre la creación de interfases gráficas y la importación de archivos en plataformas externas, así como ampliar los conocimientos relacionados con la gestión de proyectos.

Además, la colaboración con el tutor ha permitido mejora habilidades relacionadas con la comunicación y la resolución de problemas.

En conclusión, el desarrollo de esta aplicación ha sido un proceso enriquecedor tanto en términos de conocimientos técnicos como personales. Se ha logrado el objetivo de crear una herramienta útil y eficiente para los profesores de la universidad, y se han perfeccionado habilidades relacionadas con la programación y la resolución de problemas.

7.2. Líneas de trabajo futuras

En este apartado se presentan algunas sugerencias y posibilidades para el desarrollo futuro del proyecto.

- **Desarrollo de tests unitarios**: Una de las áreas en las que se podría trabajar es en la implementación de tests unitarios para garantizar la calidad del software y detectar posibles errores de manera temprana.
- **Adición de preguntas nuevas**: Se podría ampliar los tipos de preguntas que la aplicación puede generar para mejorar la experiencia del usuario.
- **Traducción a más lenguajes**: Se podrían exportar los bancos de preguntas generados en diferentes formatos además del .XML, ya que *Moodle* acepta diferentes tipos de formatos.
- **Mejora de la interfaz gráfica de usuario**: Se podrían realizar mejoras en la interfaz gráfica del programa, de manera que sea más visual e intuitiva para el usuario.

Estas son solo algunas ideas, pero existen muchas otras áreas en las que se podría trabajar para mejorar y ampliar el proyecto.

Bibliografía

[1] Roberto Izquierdo Amo. *PLQUIZ*. Universidad de Burgos, 2014.

[2] Juan José Rodríguez Díez. *4. Reglas de Asociación*. Universidad de Burgos, 2019.

[3] Cristina Gil. Reglas de asociación, 2020. [Online; visitado 11-Febrero-2023].

[4] Moodle. Acerca de moodle, 2022. [Online; visitado 02-Febrero-2023].

[5] Asier Alonso Morante. *ProgDinQuiz*. Universidad de Burgos, 2016.

[6] Juan Diego Polo. Github, 2015. [Online; visitado 02-Febrero-2023].

[7] Víctor Renuncio Tobar. *PLGRAM*. Universidad de Burgos, 2016.

[8] Wikipedia. Algoritmo apriori — wikipedia, la enciclopedia libre, 2020. [Online; visitado 02-Febrero-2023].

[9] Wikipedia. Weka (aprendizaje automático) — wikipedia, la enciclopedia libre, 2020. [Online; visitado 02-Febrero-2023].

[10] Wikipedia. Diagrams.net — wikipedia, la enciclopedia libre, 2022. [Online; visitado 02-Febrero-2023].

[11] Wikipedia. Eclipse (software) — wikipedia, la enciclopedia libre, 2022. [Online; visitado 02-Febrero-2023].

[12] Wikipedia. Extensible markup language — wikipedia, la enciclopedia libre, 2022. [Online; visitado 02-Febrero-2023].

[13] Wikipedia. Github — wikipedia, la enciclopedia libre, 2022. [Online; visitado 02-Febrero-2023].

[14] Wikipedia. Latex — wikipedia, la enciclopedia libre, 2022. [Online; visitado 02-Febrero-2023].

[15] Wikipedia. Lenguaje unificado de modelado — wikipedia, la enciclopedia libre, 2022. [Online; visitado 02-Febrero-2023].

[16] Wikipedia. Scrum (desarrollo de software) — wikipedia, la enciclopedia libre, 2022. [Online; visitado 02-Febrero-2023].

[17] Wikipedia. Java (lenguaje de programación) — wikipedia, la enciclopedia libre, 2023. [Online; visitado 02-Febrero-2023].

[18] Wikipedia. Maven — wikipedia, la enciclopedia libre, 2023. [Online; visitado 02-Febrero-2023].

[19] Wikipedia. Overleaf — wikipedia, la enciclopedia libre, 2023. [Online; visitado 02-Febrero-2023].

[20] Wikipedia. Reglas de asociación — wikipedia, la enciclopedia libre, 2023. [Online; visitado 01-Febrero-2023].