CIMAT

Centro de Investigación en Matemáticas, A.C.

Visualización de datos a partir de patrones de despliegue en lenguaje de programación R

T E S I S

Que para obtener el grado de

Maestro en Ingeniería de Software

P r e s e n t a

Luis Angel Sustaita Guerrero

Director de Tesis:

Dr. José Arturo Mora Soto

Zacatecas, Zacatecas. 9 de Noviembre de 2017

Resumen

Actualmente para los científicos de datos representa un reto el muestreo gráfico de los resultados del análisis de datos, regularmente los científicos de datos son expertos en programación y en estadística, pero no en técnicas de diseño de gráficos, por esta razón se propone el uso de patrones de visualización, estos patrones están publicados en el trabajo de tesis de Christian Behrens en 2008.

El alcance del siguiente trabajo fue hasta los patrones de despliegue, puesto que Behrens propone la separación de los patrones en tres categorías, patrones de despliegue es la primera categoría. Consistió en la programación de cada patrón en el lenguaje de programación R con uno o más paquetes, dependiendo del patrón y los tres paquetes elegidos como prioritarios para el experimento de este trabajo. Una vez programados los patrones con los paquetes mencionados se aplicaron las métricas de calidad de software elegidas mediante AHP (Proceso Analítico Jerárquico), que consiste en la elección de las métricas a partir de la encuesta a un grupo de expertos.

Con los resultados obtenidos de la aplicación de las métricas de calidad de software se hizo una comparación con los resultados obtenidos de la programación de los mismos patrones y sus métricas, pero con los resultados programados en Python, y a partir de esto llegar a las conclusiones de cual lenguaje es más recomendable para hacer ciencia de datos y visualización de patrones con software de calidad.

Palabras clave: Patrón, librería, paquete, métrica de calidad, visualización de datos.

Agradecimientos

Agradezco a Dios por la oportunidad de vivir esta experiencia, ha sido una de las mejores etapas de mi vida.

Agradezco mi familia por todo el apoyo a lo largo de mi vida personal y académica, gracias a Mi madre, Esthela Guerrero y su pareja Carlos Aguilar, a mis hermanas Leyra y Candy y también a mi hija Betzabé. Ustedes siempre serán el motor de mi vida.

A mi asesor de tesis, al Dr. Arturo Mora Soto y al coordinador académico de la maestría Mtro. José Hernández Reveles, Gracias por confiar en mí aun cuando todo se veía en contra y ser parte de este que fue uno de mis sueños, son grandes profesionales en su área y con una calidad moral que se encuentra en pocas personas.

A mis compañeros y amigos de maestría no encontraría iguales ni hay mejores.

Agradezco a CONACYT y a su programa de becas por el apoyo para la realización de esta tesis, estaré eternamente agradecido.

Al Centro de investigación en Matemáticas (CIMAT) y a todo su personal que me han permitido el crecimiento profesional y personal, al cursar un posgrado en este prestigiado centro de investigación.

Finalmente, pero no menos importante a los colaboradores con el desarrollo de esta tesis, sin ustedes el trabajo no se hubiera logrado concluir.

Índice General

Índice de Figuras

Índice de Tablas

Capítulo 1

Motivación de la investigación y descripción del problema

En este capítulo se presenta una breve descripción del contenido en este trabajo de tesis. El objetivo es dar a conocer la motivación que llevó a la implementación de patrones de visualización de datos y sus respectivos casos prácticos, desarrollados en un lenguaje de programación, a los cuales se les aplicaron métricas de calidad de software para poder compararlos contra otro lenguaje de programación.

1.1. Motivación y descripción del problema

En la actualidad uno de los principales problemas que abordan los científicos de datos es encontrar la mejor manera para representar gráficamente los resultados que dan respuesta a una pregunta que ha requerido análisis de información, y que dicha respuesta, sea fácil de entender e interpretar por el público o la audiencia a la que se quiere presentar ya sean del campo de la ciencia de datos o no.

En el intento de dar respuesta al problema de la representación gráfica de datos, estadistas, artistas y diseñadores gráficos han incursionado en el mundo de la ciencia de datos en una nueva disciplina denominada “Visualización de Información” o también conocida como “Visualización de Datos”; sin embargo, la convergencia entre personas con un perfil de artes con personas con un perfil en ciencias, no ha sido fácil. Como una propuesta para introducir a los profesionales del diseño en el mundo de la visualización de información, en el año 2008 se presentó un trabajo de tesis en la Universidad de Ciencias Aplicadas de Potsdam (Alemania) en el que se define un conjunto de patrones para el diseño de soluciones gráficas de visualización de información (Behrens, 2008).

La visualización de datos es la disciplina que se encarga de hacer la comunicación visual de un cierto análisis de datos (Friendly, 2009). Esto implica la creación y el estudio de representaciones visuales de los datos y tiene como objetivo principal comunicar de forma clara la interpretación de los mismos a partir de gráficos.

La importancia de la aplicación de patrones de visualización de datos radica en mostrar la interpretación de los datos a partir de gráficos, dado que cualquier cosa de la vida real puede

ser interpretada como un dato, es necesario el uso de patrones de visualización de datos para exponer diferentes tipos de datos. Por ejemplo, para una empresa es más sencillo hacer una comparación de sus ganancias por mes en su año actual frente al año pasado con un gráfico de multilíneas que lo muestre todo de forma compacta, con alzas y bajas a usar un gráfico de barras simples o barras múltiples que muestra aumentos puntuales en sus barras.

1.2. Descripción general de la solución

Tomando en cuenta la problemática generada en la visualización grafica de los datos, basados en el trabajo realizado por Behrens en 2008, se tomaron los patrones de despliegue propuestos por el mismo Behrens y se desarrollaron en el lenguaje de programación R. A dichos patrones se les emplearon métricas de calidad de software seleccionadas a partir de la técnica AHP (Analytic Hierarchy Process), y con eso dar respuesta a la siguiente pregunta: ¿Es R mejor lenguaje estadístico que Python para realizar visualización de datos? Para dar respuesta a esta pregunta también se toma en cuenta el trabajo de tesis presentado por Mota (2017), donde aplica las mismas técnicas seleccionadas en AHP a los patrones de despliegue de Behrens, pero en el lenguaje de programación Python.

Capítulo 2

Estado de la práctica

En este capítulo se presentan los trabajos previamente revisados, los cuales podrían presentar una respuesta a la pregunta: ¿Es R mejor lenguaje estadístico que Python para realizar visualización de datos?, también se mostrará una breve descripción de la literatura referente a la visualización de datos, su aplicación en el área software y la calidad del mismo, los cuales son los elementos más relevantes en que se basa este trabajo de tesis.

2.1. Marco Teórico

2.1.1 Visualización de datos y patrones de visualización

Hoy en día estamos viviendo una época donde las formas de enseñar, aprender, de trabajar e incluso de interactuar son distintas. La revolución tecnológica que ha traído el uso de computadora, internet, teléfonos inteligentes, aplicaciones y distintos tipos de multimedios ha llevado tanto la industria, la academia, al estado y a cualquier persona civil a utilizar programas de software. Tan solo en México según la investigación de Guerrero (2015) en 2011 se estimaba que había 98.9 millones de dispositivos móviles, que se traduce a 98.5 dispositivos por cada 100 habitantes, quienes hacen descargas de aplicaciones de las cuales 35% son redes sociales. Esto quiere decir que tan solo en redes sociales se manejan miles de millones de datos personales que en algún lado están guardados y que también se debería de tener acceso a todos ellos. Aquí es donde reside la importancia de la visualización de datos, en hacer muestras gráficas de los datos que se tiene guardados.

La visualización de datos permite a los mismos ser expresados de forma rápida, clara y sencilla a través de gráficos y símbolos, gracias a ello podemos describir la visualización de datos como un lenguaje universal. El objetivo principal de la visualización de datos es comunicar la información de forma clara y eficaz a través de medios gráficos. No significa que la visualización de datos tiene que parecer aburrida para ser funcional o muy sofisticado para verse hermosa. Para transmitir ideas eficazmente, tanto la forma estética y funcionalidad necesitan ir de la mano Friendly (2008).

Lo cierto es que Aunque para algunas investigaciones científicas, el muestreo de los datos puede no ser estrictamente visual, es una extensión adicional incorporar las visualizaciones, ilustraciones y gráficas visuales de ideas abstractas con expresión simbólica previa (Cat, 2017). Cualquier objeto del mundo real puede ser traducido a datos estadísticos, y estos mismos se pueden representar de forma gráfica, El proceso de representación y del cual se

encarga el científico de datos que según Josh Willls “es una persona que sabe más de estadística que cualquier programados y que a la vez sabe más de programación que cualquier estadista". Suele ser una tarea donde se emplea recolección de datos, estadística, ingeniería y diseño. Puesto que no todos los datos son iguales tampoco se deberían representar igual gráficamente, pero tampoco puede haber una gráfica por cada tipo de objetos, es aquí donde se recomienda el uso de patrones, estos contribuirán a tomar la decisión de qué forma gráfica se podrían representar mejor los datos.

Del mismo modo que en la arquitectura, los patrones son métodos que describen el proceso de planeación y construcción de un producto. De hecho, el término diseño de patrones fue implementado por primera vez por Christopher Alexander en arquitectura urbana. Behrens (2008) propone el uso de patrones para la visualización de datos dividiéndolos en categorías, las cuales a su vez tienen meta patrones que contienen los patrones de diseño. Si bien los patrones propuestos por Behrens son un gran, avance aún le falta la automatización de patrones a partir del software (Figura 2.1).

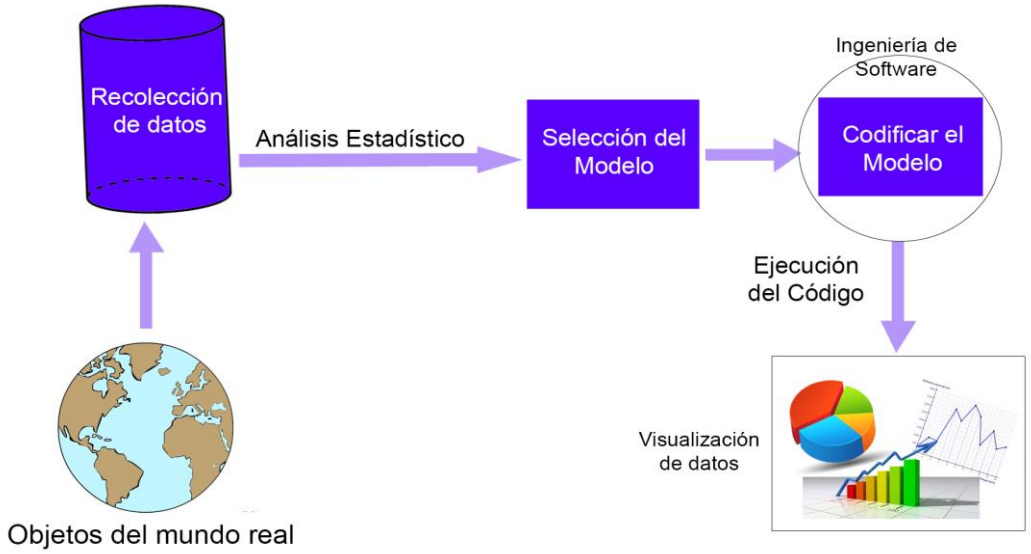

Figura 2.1 Análisis y visualización de datos

2.1.2 Lenguajes de programación y la visualización de datos

Los lenguajes de programación juegan un papel muy importante a la hora de hacer el análisis y la graficación de los datos, es imprescindible seleccionar el lenguaje conveniente para hacer el desarrollo. Existen muchos lenguajes de programación e incluso programas de software con entornos de desarrollo integrado, que permiten hacer uso de manipulación de datos. Según el sitio KDnuggets (2017), los lenguajes más utilizados por los científicos de datos, del año 2012, 2013, 2014 son R, SAS, Python y SQL, también figuran algunos programas con entorno de desarrollo como Matlab y GNU Octave.

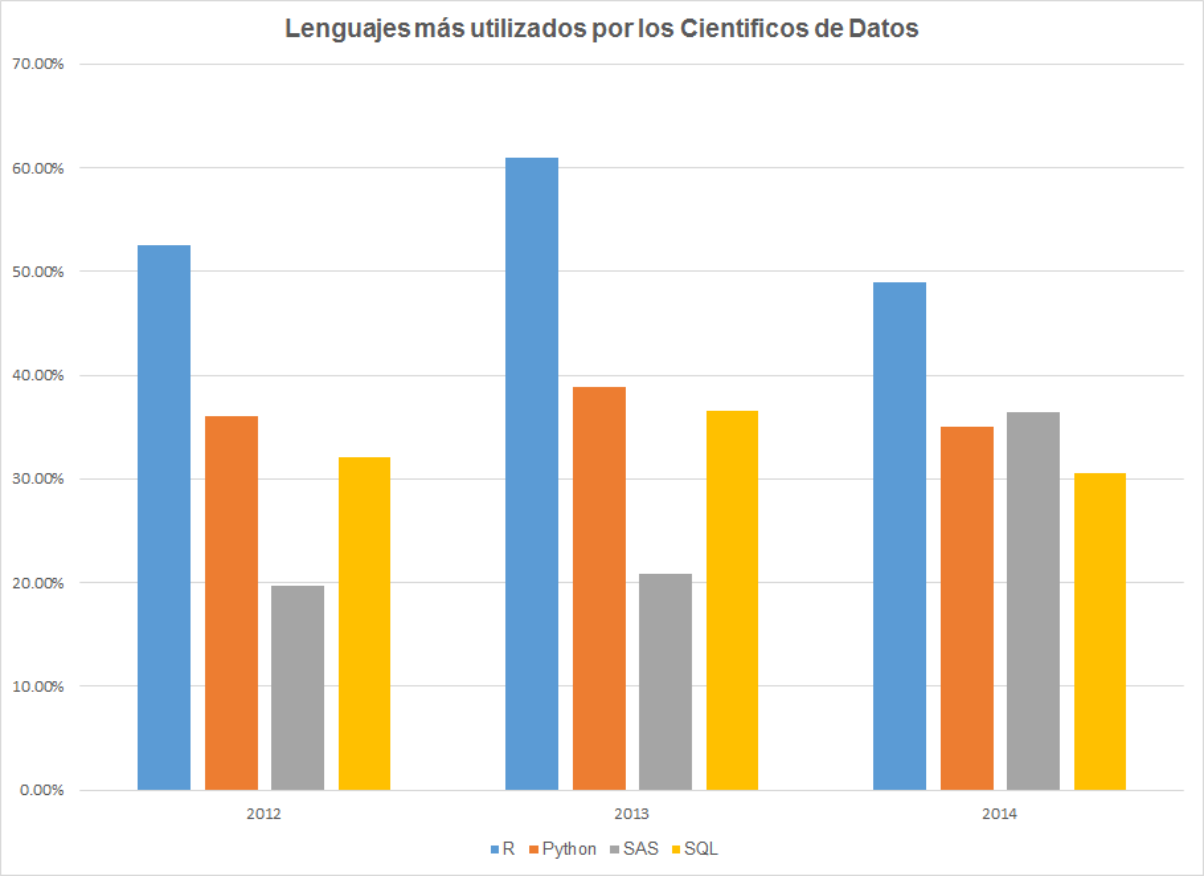

Figura 2.1.2 Lenguajes más utilizados por los científicos de datos según KDnuggets

.

Para este trabajo de tesis se utilizará el lenguaje R para la implementación de patrones puesto que cuenta con las siguientes ventajas frente a los anteriores 3 más utilizados.

- Es uno de los lenguajes más utilizados por los científicos de datos

- Tiene una de las comunidades más grandes de desarrollo y de soporte
- Es un lenguaje de código libre
- Fue creado especialmente para uso estadístico incluso para personas que no están relacionadas en el campo de software
- Es sencillo de aprender
- Existe un gran número de librerías de código abierto para el uso de gráficos

2.1.3 La calidad de Calidad de Software y sus métricas

Cuando se habla de calidad, generalmente se refiere a un objeto que por sus cualidades y características puede tener más o menos valor frente a otro objeto de su misma clase, cualquier producto que no se somete a estándares rigurosos de calidad, es propenso a ser defectuoso y en su defecto menos vendido. El software es un producto que día con día toma más fuerza, la tecnología está complementando y rebasando las tareas humanas, por esta razón es de gran importancia someter los productos de software a procesos de calidad, puesto que de él dependen muchas veces la economía de una empresa, la comunicación entre usuarios e incluso vidas humanas. Uno de los fallos más famosos fue en 2012 cuando la empresa financiera Knight Capital por un error en la actualización de su software provocó que la empresa perdiera 460 millones de dólares en cuestión de minutos. Como este caso existen muchos otros casos sonados por fallas en la calidad de software donde inclusive se llegaron a perder vidas.

Otra parte importante de la calidad es la competitividad, el mercado de software está creciendo cada vez más, entre programas de software libre y software propietario se ha desatado una guerra por hacer mejores desarrollos, pues, aunque parece que no, también en el software libre hay remuneraciones económicas. Tan solo en México según la secretaría de economía (2014) entre el año 2013 y 2014 hubo un aumento de consumos de servicio de software y tecnologías de información de 12.25% lo que quiere decir que en 2014 se generó un flujo económico ascendente a los 20 mil millones de dólares.

En software, en el sentido más estrecho de producto la calidad es comúnmente reconocida por la ausencia de fallos en el producto (Kan, 1999). Arquitecturas, métricas, modelos, entre otros hacen un software de mayor calidad, Las métricas son empleadas para conocer qué tan cerca o tan lejos está el software para ser de calidad. Algunas de las métricas más comunes son, Número de líneas de código, complejidad ciclomática, número de librerías externas, promedio de líneas por función, entre otras, para este trabajo la definición de métricas se hizo según la técnica AHP (analytic hierarchy process), donde un grupo de expertos a partir de su

criterio de la práctica recomendaron utilizar las métricas más usadas por ellos para el análisis y visualización de datos.

Características	***Buenas prácticas de código a primer nivel***	***Buenas prácticas de arquitectura***
Confiabilidad	Estado de protección en entornos multiproceso. Uso seguro de la herencia y el polimorfismo. Gestión de límites de recursos, Código complejo, Gestión de recursos asignados, Tiempo de espera, Direcciones remotas integradas.	Cumplimiento del diseño multicapa El software administra la integridad de los datos Manejo de excepciones a través de transacciones Arquitectura de clases
Eficiencia de Rendimiento	Cumplimiento de las mejores prácticas orientadas a objetos. Cumplimiento de las mejores prácticas de SQL. Cálculos costosos en bucles. Conexiones estáticas versus conexiones. Cumplimiento con las mejores prácticas de recolección de basura.	Interacciones apropiadas con recursos costosos o remotos Rendimiento de acceso a datos y gestión de datos Gestión de memoria, red y espacio en disco Manejo centralizado de solicitudes de clientes Uso de componentes de nivel medio versus procedimientos y base de datos Funciones
Seguridad	Uso de credenciales codificadas Desbordamientos del búfer	Validación de entrada Inyección SQL Scripting entre sitios

	Fracasos o arriesgados algoritmos criptográficos Falta la inicialización Validación incorrecta del índice de matriz Bloqueo incorrecto Referencias a los recursos publicados Cadena de formato no controlada	Fracaso en el uso de bibliotecas o marcos aprobados Cumplimiento del diseño de arquitectura segura
Mantenimiento	Código no estructurado y duplicado Alta complejidad ciclomática Nivel controlado de codificación dinámica Sobre parametrización de métodos Codificación de literales Tamaño excesivo del componente Cumplimiento con las mejores prácticas de SO	Cumplimiento del diseño inicial de la arquitectura Estructura estricta de la llamada entre capas arquitectónicas Capas horizontales excesivas

Tabla 2.1.3 Requerimientos de calidad para productos de software según CISQ (Consortium for IT Software Quality, 2013)

2.2. Soluciones existentes

Actualmente existen programas de software, lenguajes de programación, paquetes para lenguajes y aplicaciones para la visualización de los datos, estas herramientas proporcionan distintos gráficos para proporcionar de forma amigable la visualización de los datos. Pero, ¿Realmente existe una aplicación, programa, paquete o lenguaje en el cual se puedan implementar todos los patrones propuestos por Behrens? La respuesta en definitiva es no, dado que el uso específico de patrones no es del todo común en la visualización de datos programadores y diseñadores se han enfocado a los gráficos más usuales en sus aplicaciones. Algunos programas como Matlab y Octave son frecuentemente utilizados para

el desarrollo e implementación de gráficos. Sin embargo, no todo está perdido lenguajes de programación como R y su amplia comunidad han desarrollado un gran número de librerías para la visualización de datos, dado que R es un lenguaje de sintaxis simple, es sencillo de aprender y se pueden generar casi cualquier tipo de gráficos o patrones.

2.2.1 Soluciones existentes en R

Tomando en cuenta que R es un lenguaje de programación orientado totalmente a la estadística, tiene una gran cantidad de paquetes para generar gráficos, sin embargo no existe ninguna aplicación programada en el lenguaje que sea capaz de mostrar cada uno de los patrones propuestos por Behrens, incluso algunos de los patrones de despliegue propuestos no ha sido posible desarrollarlos con algunos paquetes de los más utilizados en el lenguaje, como se puede apreciar en la tabla 2.2.1, donde se muestran tres de los paquetes de visualización más comunes en R, con los cuales se observa si es posible o no codificar los patrones de despliegue propuestos.

En la tabla se puede apreciar del lado izquierdo los metapatrones y patrones de despliegue, seguidos de ellos, los tres paquetes más utilizados para la visualización de datos en R. En el siguiente enlace se encuentra una lista detallada de más paquetes con los cuales se podrían codificar los patrones que no se pudieran codificar con los tres paquetes más comunes.

https://drive.google.com/open?id=1hIWSd0iX9zXsoEtqx7xI7MQqW06_hYYx8cYY83x5-CM

		Paquetes		
Meta Patrones/Patrones		Ggplot2	Graphics	Lattice
Correlaciones	Gráfico de dispersión	Si	No	Si
	Gráfico de burbujas	Si	No	Si

Cantidades Continuas	Gráficos de Línea Simple	Si	Si	Si
	Gráficos Multilínea	Si	Si	Si
	Gráfica de Área Apilada	Si	Si	No
	Gráficos de destello	No	No	No
Cantidades Discretas	Gráficos de Barra simple	Si	Si (barplot)	Si (barchart)
	Gráficos de Multibarra	Si	Si	Si
	Matriz de Puntos	Si	Si	Si
	Gráfico de Barra de destellos	Si	Si	Si
	Gráficas de Barras Isométricas	No	No	Si
	Gráficos de Lapsos	Si	Si	Si
Proporciones	Gráfico de Pastel Simple	Si	Si	Si

	Gráficos de anillo	Si	Si	Si
Flujos	Diagrama de Sankey	No	Si	No
	Diagramas de Hilos de Arco	No	No	No
Jerarquías	Diagramas de Árbol	No	No	No
	Mapa de Árbol	Si	Si	Si
Redes	Diagrama de Árbol	Si	Si	Si
	Círculos de Relación	Si	Si	Si
	Collar de Perlas	Si	Si	Si
Espacio	Mapas Topográficos	Si	Si	Si
	Mapas temáticos	Si	Si	Si

Tabla 2.1.1. Meta patrones y patrones, solución con paquetes en R

En la tabla anterior se puede ver que si existen paquetes y gráficos con los que se pueden desarrollar los patrones, pero son necesarios más de un paquete para poder implementar todos y cada uno de los patrones, si bien el patrón es uno, cada paquete contiene diferentes estilos de gráficos convenientes para el patrón y eso da mayor gama de estilos visuales para los patrones.

2.2.2 Soluciones con Python

Si bien Python no es un lenguaje enfocado totalmente a la estadística, es un lenguaje que también es muy popular entre los científicos de datos. Por el mismo motivo que Python no es un lenguaje totalmente enfocado para la manipulación de datos, los códigos elaborados en el lenguaje para algunos patrones suelen ser más extensos que los códigos desarrollados en R. El lenguaje cuenta con una gran cantidad de librerías para hacer visualización de datos, pero igual que con R no existe una aplicación o librería que contenga todos los elementos para la visualización de los patrones propuestos por Behrens. En el siguiente enlace, en la pestaña Python se encuentran las librerías que se podrían utilizar para la implementación de patrones de despliegue.

https://drive.google.com/open?id=1hIWSd0iX9zXsoEtqx7xI7MQqW06_hYYx8cYY83x5-CM

2.3 Conclusiones acerca del estado de la práctica

2.3.1 Soluciones en el mercado

Tomando en cuenta lo importante que se ha vuelto la manipulación de los datos, y la visualización gráfica de los mismos, no existen herramientas que den solución total al problema de la visualización. En el mercado existen algunos programas que haciendo uso de ellos se pueden generar algunos gráficos predeterminados en el mismo programa, pero ninguno sigue la especificación de patrones ni mucho menos cuenta con todos, puesto que la mayoría de esos programas tiene el uso de gráficos más comunes y no aquellos que se utilizarían en casos específicos.

Softwares comerciales y con entornos de desarrollo como Matlab y Octave suelen ser muy utilizados para el análisis de datos a un bajo nivel y no tanto en la ciencia de datos, no es posible la generación de gráficos avanzados ni la personalización de todas las gráficas.

2.3.2 Soluciones desarrollo de software

Lenguajes de programación orientados a la manipulación de datos como R y Python tienen un gran número de paquetes y librerías que se utilizan en el campo del análisis y visualización de datos, pero para poder programar todos y cada uno de los patrones se tendría que hacer una búsqueda detallada de las librerías y paquetes para conocer cuales podrían programar ciertos patrones.

Una de las desventajas es que no todas las librerías son de código libre y se tiene que hacer un pago para el uso de ellas, la ventaja de que existan muchas librerías y paquetes es que buscando detenidamente se pueden encontrar algunas que son de código libre y pueden ser utilizadas con facilidad.

El lenguaje R cuenta con una sintaxis sencilla de comprender y aprender y eso aminora la curva de aprendizaje a la hora de hacer manipulación de datos utilizando los paquetes para R, no tiene un paquete donde se puedan graficar cada uno de los patrones, pero, facilita el trabajo el poder conocimiento del lenguaje y los ejemplos de propietarios de paquetes.

Capítulo 3

Solución Propuesta

En este capítulo se describe el desarrollo de la solución propuesta al problema de la visualización de datos a partir de patrones, se describirán las tecnologías y herramienta utilizadas para la visualización de datos implementados en el lenguaje de programación “R” y se justificará el uso de las mismas, además se describe el uso de las métricas de software utilizadas en las mismas tecnologías y herramientas.

3.1. Descripción general de la solución

Para el desarrollo de la solución aquí propuesta se tomó como punto de partida el trabajo de tesis “The Form of Facts and Figures” desarrollado por Christian Behrens en el año 2008, este trabajo propone una solución al problema de visualización de datos a partir de patrones, los cuales forman parte de meta patrones y así mismo los meta patrones forman parte de tres categorías propuesta por el mismo Behrens como se muestra en la figura 3.1.

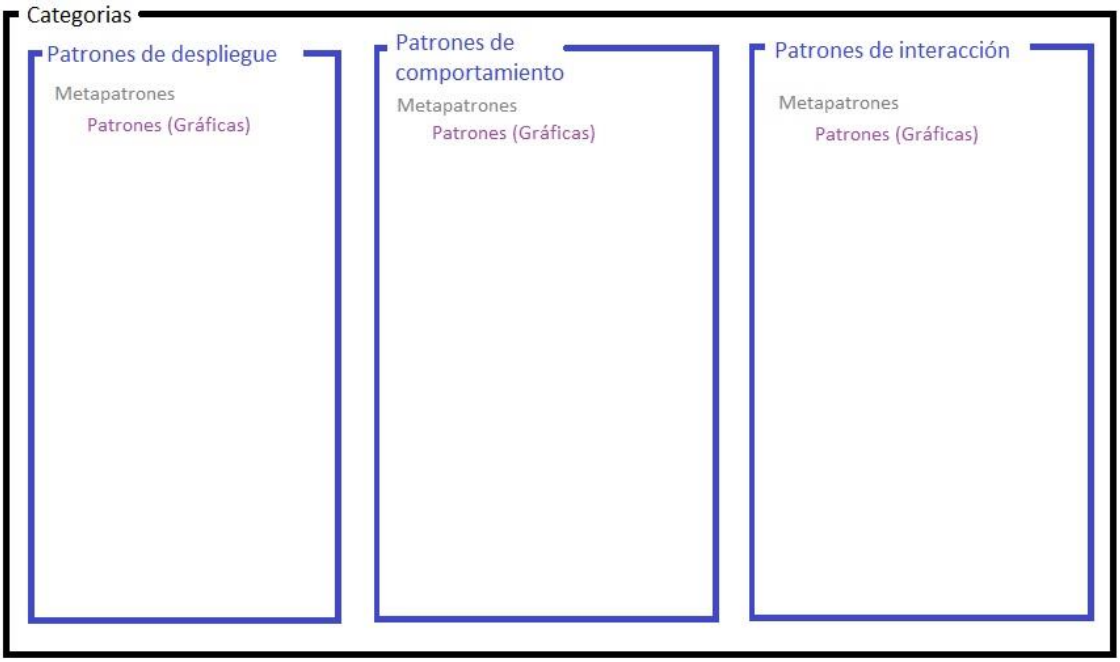

Figura 3.1: Estructura de Categorías, Meta patrones y Patrones.

Para este trabajo únicamente se utilizó la primera categoría: “Patrones de Despliegue”. Esta categoría está conformada por ocho meta patrones, cada meta patrón contiene un número diferente de patrones, pero la suma de todos los patrones de despliegue contenidos en los

meta patrones es un total de veintitrés. Para cada patrón se hizo una búsqueda de herramientas y tecnologías para su implementación en el lenguaje de programación “R”, a dichas implementaciones se les aplicaron métricas de calidad de software para validar el uso del lenguaje de programación R en la implementación de los patrones.

Una vez implementados los patrones en el lenguaje de programación R se realizó una wiki (que se encuentra disponible en: https://github.com/cimat/data-visualization-patterns/wiki), en donde se pueden consultar los patrones que fueron utilizados, descripción del patrón, código de la implementación y una gráfica del patrón implementado en el lenguaje R.

3.2 Descripción de las métricas de software utilizadas

Para la definición de las métricas de ingeniería de software a utilizar, se empleó una técnica llamada AHP (analytic hierarchy process), donde un grupo de expertos a partir de criterios dados, ponderan los mismos según la relevancia haciendo comparaciones a partir de criterios y experiencias personales. La esencia principal del uso de AHP es darle valor al juicio humano para la realización de evaluaciones (Saaty, 2008). Para este caso, los objetivos son las métricas, y a partir de las respuestas de los expertos determinar cuáles métricas son las apropiadas para utilizar en este trabajo.

3.2.1 Selección de expertos

Para lograr que los resultados del uso de métricas sean confiables, también es importante que las personas que darán su punto de vista, conozcan el tema que se les presenta, por lo tanto, es de suma importancia definir criterios de aceptación para la elección de expertos.

Los criterios de aceptación que deberían cumplir los expertos son los siguientes:

- Experiencia programando en R mínima de dos años.
- Experiencia en visualización de datos y reportes.
- Experiencia en librerías de visualización de datos en R para escritorio o en la web.
- Experiencia en las métricas de ingeniería de software para código fuente.

3.2.1 Ponderación de las métricas

Después de hacer la selección del grupo de expertos, se les envió una encuesta en línea para que pudieran proporcionar un grado de importancia a las métricas más comunes de ingeniería de software. La encuesta realizada se puede encontrar en la siguiente liga: http://fluidsurveys.com/surveys/cimatzac/comparativa-de-metricas-de-software/.

Dicha encuesta presenta comparativas de métricas entre sí, donde cada experto puede dar su opinión sobre el grado de importancia que le da a las métricas dadas para su comparación, todo con la finalidad de valorar su implementación en el código de los distintos patrones de visualización de datos.

Las métricas elegidas, a partir del análisis crítico del estado del arte, las más utilizadas y confiables para generar software de calidad fueron las siguientes:

- Líneas totales de código (SLOC)
- Número de funciones (FunNum)
- Número promedio de líneas de código por función (AvSLOCPF)
- Complejidad ciclomática (CC)
- Número de librerías externas (ExLibNum)

La ponderación del peso de cada actividad se realizó en base a la siguiente tabla de escala fundamental de pesos absolutos, donde se muestra la relevancia de las ponderaciones de una métrica frente a otra:

Intensidad de Importancia	*Definición*	*Explicación*
1	Igual importancia	Dos actividades contribuyen igualmente al objetivo
2	Débil o Leve	
3	Importancia Moderada	La experiencia y el juicio favorecen ligeramente una actividad sobre otra
4	Más que moderado	
5	Fuerte Importancia	La experiencia y el juicio favorecen fuertemente una actividad sobre otra
6	Más que fuerte	

7	Importancia muy fuerte o demostrada	Una actividad es favorecida muy fuertemente sobre otra; Su dominio demostrado en la práctica
8	Muy, muy fuerte	
9	Extrema importancia	La evidencia que favorece una actividad sobre otra es del orden más alto posible de la afirmación
1.1-1.9	Cuando las actividades están muy cerca de un decimal se agrega a 1 para mostrar su diferencia como inapropiado	Una mejor manera alternativa de asignar los decimales pequeños es comparar dos actividades cercanas con otras muy contrastantes, favoreciendo la más grande un poco más pequeña que la menor cuando se usan los valores 1-9.
Recíproco de enzima	Si la actividad i tiene uno de los números no nulos anteriores asignados a él cuando se compara con la actividad j, entonces j tiene el valor recíproco cuando se compara con i	Una suposición lógica
Mediciones de escalas de proporción		Cuando se desea utilizar tales números en aplicaciones físicas. Alternativamente, a menudo uno estima las razones de tales magnitudes usando el juicio

Tabla 3.1. Escala Fundamental de Números Absolutos (Saaty, 2008)

Los resultados obtenidos de la encuesta arrojaron que las métricas con más peso para software de visualización de datos son las siguientes en el siguiente orden:

1. Número de Líneas de Código.
2. Complejidad Ciclomática.
3. Número de Funciones.
4. Número de librerías Externas.

Durante el desarrollo de la codificación de patrones se tomó la decisión de no tomar en cuenta las métricas de número de funciones (FunNum) y complejidad ciclomática (CC), puesto que para la codificación de los patrones en el lenguaje R para este trabajo se pueden programar dentro de una sola función, y para todos los patrones la complejidad ciclomática daría como resultado 1, lo cual quiere decir que son códigos de complejidad simple.

Las métricas a medir para este trabajo fueron las siguientes:

- **Número de líneas de código**: El conteo de las líneas de código representa el tamaño y la complejidad de un programa, esto no representa una sorpresa dado que, entre más líneas de código en un programa, se esperan más defectos en el mismo (Kan, 1999).
- **Número de librerías externas:** Se refiere al número de librerías, módulos o paquetes externos al núcleo del lenguaje, que se requieren para poder crear un gráfico (Mora-Soto et al, 2016).

3.3 Automatización de las métricas de software

Con la finalidad de facilitar el conteo de las líneas de código y el número de librerías externas, se decidió hacer uso de herramientas para la automatización del conteo de las métricas de software utilizadas. Las herramientas utilizadas para la automatización deberán de ser herramientas que no sean propietarias, con el afán de que los usuarios no necesiten ningún tipo de licencia al momento de usarlas y querer replicar los resultados presentados en este trabajo de tesis.

Para hacer el conteo de las líneas de código se realizó una búsqueda de herramientas de código libre que fuera capaz de soportar el conteo de líneas de código del lenguaje R, que consiga exportar los resultados a un tipo de archivo de texto y que fuera capaz de correr en el sistema operativo Ubuntu, puesto que la programación de los patrones se realizó bajo la misma plataforma, ya que facilita el uso de paquetes utilizados.

Se decidió usar la herramienta de código libre **CLOC**, la cual, es una herramienta que cuenta las líneas de código físicas, así como las líneas en blanco de diferentes lenguajes de programación y permite exportar los resultados a diferentes tipos de archivos que son clasificados por lenguaje y proyecto (Danial, 2006-2015), esto facilita la exportación de resultados para el desarrollo de la wiki de este trabajo.

Como parte de las aportaciones de este trabajo de tesis se desarrolló un script en el lenguaje de programación Python con licencia Creative Commons para poder llevar cabo el conteo de número de librerías externas. Por medio del mismo script es posible unir los resultados exportados por el programa CLOC, además, este script es capaz de hacer un mapeo de su

respectivo patrón con el número de librerías externas utilizadas, y por último genera un archivo markdown, el cual se utiliza en para publicar los resultados en la wiki. El código del script desarrollado se puede encontrar en el siguiente enlace: https://github.com/cimat/data-visualizationatterns/blob/master/templates/LibraryScript.py

3.4 Justificación del lenguaje de programación utilizado (R)

Uno de los lenguajes más utilizados entre los científicos de datos para el análisis de datos es R, además R es un lenguaje y entorno de computación y gráficos estadísticos (The R Foundation, 2016), Este lenguaje de programación está basado en un lenguaje de análisis estadístico llamado S. Puesto que la mayoría de los científicos de datos no son expertos en programación este lenguaje permite a los usuarios aprender de forma rápida y sencilla su sintaxis y su uso para la generación de gráficos, además hay un sin fin de librerías de código abierto en la web para la mejora de gráficos en R.

Las ventajas principales de usar R es la gran cantidad de usuarios que utiliza este lenguaje, el soporte que el proyecto R continuamente brinda y que es fácil de usar en cualquier plataforma y sistema operativo. Además, La Red Completa de Archivos R (CRAN por sus siglas en inglés) tiene servidores en todas partes del mundo para dar comodidad a los usuarios según sus ubicaciones geográficas. La simplicidad del lenguaje permite que con pocas líneas de código se pueda analizar grandes cantidades de datos y se logren graficar los mismos, R es el lenguaje apropiado tanto para principiantes como para expertos en estadística, el campo de análisis de datos y la programación. R es un lenguaje ampliamente utilizado en clases de ciencias políticas para gráficos de datos, y es ahora el paquete de elección más común en las clases avanzadas de métodos cuantitativos (Boef et al, 2002).

3.5 Descripción de las librerías utilizadas

Como se mencionó anteriormente, R tiene un gran número de usuarios y desarrolladores, lo cual permite encontrar un sinfín de librerías para la creación de gráficos en el lenguaje. Las librerías, también llamadas paquetes en el lenguaje R, son las unidades fundamentales de reproducción del código en este lenguaje de programación, estas incluyen funciones reusables, la documentación que describe el uso de las mismas y datos de muestra (Wickham, 2015).

Las librerías utilizadas para este trabajo fueron librerías seleccionadas de acuerdo a las descargas (que librerías tienen más descargas) en el CRAN de R, el reporte de bugs, que sean versiones en constante actualización, que sean multiplataforma, que sean capaces de graficar más de cinco patrones, y que sean librerías de código abierto para que los usuarios no tengan ningún problema de licencias al querer hacer uso de las mismas. En el Anexo 3.1 se presenta una descripción detallada del análisis de librerías realizado

Los paquetes utilizados fueron los siguientes:

- **Ggplot2:** Es un sistema de trazado para R, basado en la gramática de gráficos. Trata de tomar las mejores partes de Lattice y ninguna de las malas partes. Ggplot2 se ocupa de hacer trazos detallados, así como proporcionar un potente modelo de gráficos y esto hace que sea fácil de producir gráficos multicapa (Wickham, 2013).
- **Graphics:** Es el paquete predeterminado de gráficos en R, contiene las funciones básicas de los gráficos tradicionales en S, a diferencia de los gráficos reticulados (Murrell, 2005).
- **Lattice:** Es un sistema potente y elegante de visualización de datos inspirado en Trellis, con un enfoque especial en datos multivariados, Lattice es capaz de implementar tanto gráficos típicos, como lo suficientemente flexible para manejar requisitos no estándar (Graphics, Sarkar, & Sarkar, 2016).

Para efectos del desarrollo de la wiki, también se utilizaron algunos paquetes que no cumplían con todos los criterios de aceptación, pero que podían crear gráficos de algunos patrones que las tres librerías anteriores no originaron. Los paquetes utilizados fueron los siguientes:

- **GoogleVis:** El paquete de GoogleVis provee una interface entre R y Google Chart Tools, las funciones de este paquete les permiten a los usuarios visualizar los datos almacenados en R (Gesmann, 2016).
- **Arcdiagram:** Es un paquete de R minimalista diseñado para el único propósito de ayudar a trazar bastantes diagramas de arco (Sánchez, 2010).
- **Diagram:** Es un paquete que permite visualizar gráficos sencillos de redes en base a una matriz de transición, útil para graficar diagramas de flujo, visualización de webs y redes electrónicas (Soetaert, Cash, & Springer, 2015).
- **Treemap:** Es un paquete de visualización para llenar el espacio de las estructuras jerárquicas. Este paquete ofrece una mayor flexibilidad para dibujar diagramas de árbol (Tennekes, 2016).

3.6 Descripción del proceso de generación de la Wiki de Patrones

Para publicar los resultados del experimento y la solución de este trabajo de tesis se generó una Wiki donde se publican los patrones de despliegue implementados en el lenguaje de programación R junto con un conjunto de datos real y por lo menos una gráfica de la mayoría patrones propuestos por Behrens. Cabe mencionar que no todas las librerías utilizadas pudieron graficar el total de los patrones y por lo tanto no todos los patrones fueron graficados, sin embargo se pueden consultar los patrones en la wiki sin ejemplo de gráfica. La Wiki generada puede ser consultada en el siguiente enlace: https://github.com/cimat/data-visualization-patterns/wiki

En la figura 3.2. Se puede apreciar el proceso de creación de la Wiki.

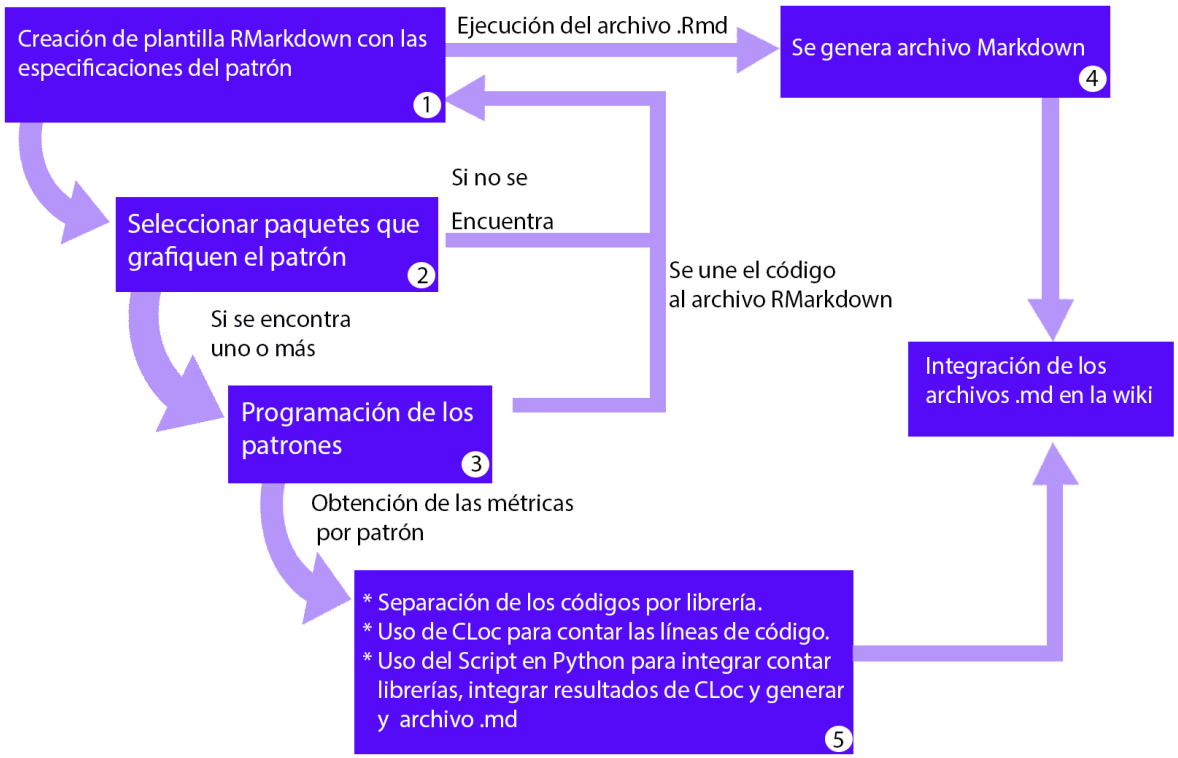

Figura 3.2. Proceso de creación de la wiki

Como se observa en la figura 3.2, para el proceso de creación de la Wiki primero se hace una plantilla que contenga las especificaciones del patrón y el conjunto de datos que se utilizara, después se seleccionan las librerías a utilizar, por lo menos una que pueda graficar el patrón, cuando se encuentra la librería se codifica y se genera una gráfica, el código se agrega a la

plantilla de RMarkdown (Rmd) para después ejecutarlo y generar un archivo Markdown (md) y unirlo a la wiki. De no encontrarse por lo menos una librería para graficar el patrón, pasa a generarse el archivo Markdown y unirlo a la wiki.

Ya que se tienen los códigos de los patrones se toman cada uno y se escanean con la herramienta de CLOC, la cual genera un archivo con los resultados de líneas de código, después se ejecuta el script desarrollado en Python para leer el número de librerías utilizadas, juntar los resultados arrojados por el CLOC y generar un archivo Markdown el cual también se integra en la wiki.

Capítulo 4

Validación de la solución propuesta

4.1 Alcance de la validación

A continuación, se presenta la ratificación de la solución a partir de comparativas de los patrones y sus codificaciones tanto en R como en Python a partir de las técnicas de ingeniería de software utilizadas para realizar software de calidad, se busca hacer una reflexión de las pruebas desarrolladas y también se pretende validar la hipótesis donde se plantea que, R es mejor lenguaje de programación para ciencia de datos que Python. Se presentan descripciones del experimento realizado, líneas de código de programación de patrones y sus respectivos gráficos, tanto en R como en Python.

4.2 Objetivo de la validación

La finalidad principal de la validación es mostrar los resultados del experimento realizado y dar respuesta al planteamiento hipotético donde se expresa que R es mejor lenguaje para codificación de patrones de visualización de datos que Python a partir de técnicas de calidad de software.

4.3 Hipótesis de la investigación

A partir de la problemática generada para hacer visualización de datos, siguiendo las recomendaciones de patrones, se puede llegar a la conclusión que no existen herramientas totalmente desarrolladas que den solución a la visualización de datos a partir de patrones, los programas más utilizados no dan una solución clara y completa, y con lenguajes de programación ninguna de sus librerías o paquetes presentan la solución completa a todas las vistas de patrones propuestas por Behrens (2008).

Tomando en cuenta lo anterior se decidió hacer la codificación propia de los patrones de despliegue propuestos por Behrens (2008) en el lenguaje de programación R dado que es un lenguaje de uso específico para la ciencia de datos y compararlo con los mismos patrones desarrollados por Mota (2016) en el lenguaje de programación Python con las mismas métricas de calidad de software utilizados. Partiendo de lo previo, se plantea la siguiente hipótesis:

“Programando patrones en el lenguaje R se proporcionan códigos de mejor calidad para el desarrollo de patrones de visualización de datos, R es mejor lenguaje para hacer análisis y visualización de datos que Python”.

4.3.1 Método de Comprobación de la Hipótesis

A partir del experimento realizado se hizo una comparación con el mismo experimento, pero realizado con Python por Mota (2016) y se tomaron en cuenta los siguientes puntos para la validación de la hipótesis.

- Se seleccionaron 16 de los patrones de despliegue propuestos por Behrens programados tanto en Python como en R.
- Se compararon las métricas de calidad utilizadas seleccionadas mediante AHP (líneas de código y librerías utilizadas) entre los patrones desarrollados con Python y con R.
- Se compara hasta cuantas librerías de las seleccionadas pueden utilizarse para programar el patrón.

4.4 Descripción del Experimento Realizado

A partir de los patrones de despliegue los cuales son la primera de tres categorías que Behrens propone, esta contiene a su vez ocho subcategorías (meta patrones) las cuales cada una proponen distintos gráficos (patrones), de estos se hizo la programación de quince de los primeros patrones propuestos para el experimento. Los patrones programados se muestran en la figura 4.4.

Dentro de los patrones de cantidades continuas, se propone el patrón de gráficos de destellos, este tipo de gráfico no se programó ni fue agregado, puesto que ninguna de los paquetes utilizados para este trabajo contenían módulos para programarlo.

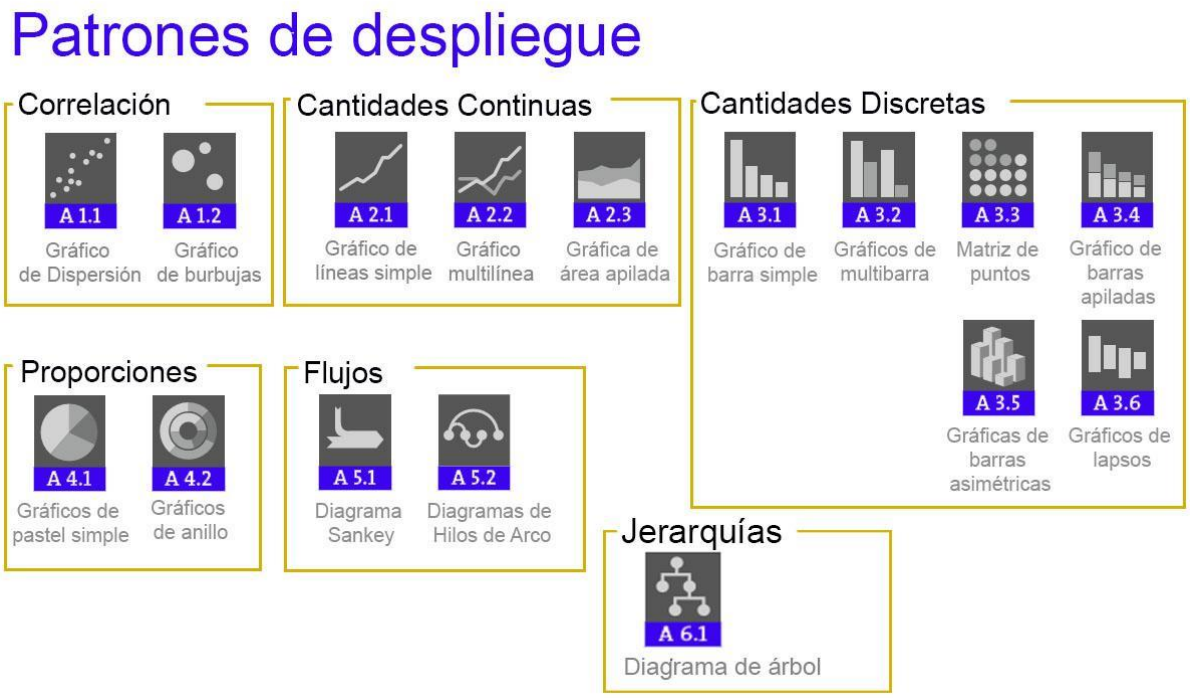

Figura 4.4 Metapatrones y patrones programados para validación

Después de programar los patrones se les aplicaron las métricas de calidad de software a cada uno de los códigos de los patrones programados, para la automatización de las métricas se desarrolló un script para el conteo automático de los paquetes externos utilizados para la programación del patrón, también se utilizó una herramienta de código libre llamada CLOC para contar las líneas de código de los patrones desarrollados.

Una vez con los patrones programados en R se procede a hacer la comparativa entre los códigos y las métricas de calidad en patrones programados en Python. Todos los códigos de la programación de patrones se publicaron en una Wiki en GitHub para que otros desarrolladores tengan acceso a la programación de patrones.

4.5 Análisis de resultados

A continuación, se muestran los ejemplos de programación de patrones, se presentan el ejemplo del desarrollo en R y su respectivo gráfico generado, también se señalan con cuantas librerías en R se puede generar el gráfico de dicho patrón y se compara con la información de los patrones programados en Python.

4.5.1 Patrones de correlación

Los patrones de correlación son capaces de mostrar una relación predictiva entre datos estadísticos, que regularmente con otros gráficos como en tablas no se podrían mostrar.

4.5.1.1 Gráfico de dispersión

Un diagrama de dispersión muestra correlaciones entre dos variables métricas. Se describe visualmente una tabla de dos columnas con pares de variantes que no proporciona mucha información significativa en la forma tabular, especialmente cuando los conjuntos de datos subyacentes se vuelven grandes. En un diagrama de diagrama de dispersión, cada par de variantes está representado por un punto en un sistema de coordenadas cartesiano bidimensional (Behrens, 2008).

Los gráficos de dispersión son más sencillos de programar en R puesto que para realizarlos se requiere de menos paquetes (librerías) externos y las líneas de código generadas son mucho menos en comparación con las de Python como se puede apreciar en la tabla 4.5.1.1. Donde se muestran las librerías con las que se puede programar el patrón tanto en R como en Python, las líneas de código y las librerías externas, además se agrega la liga al código en GitHub.

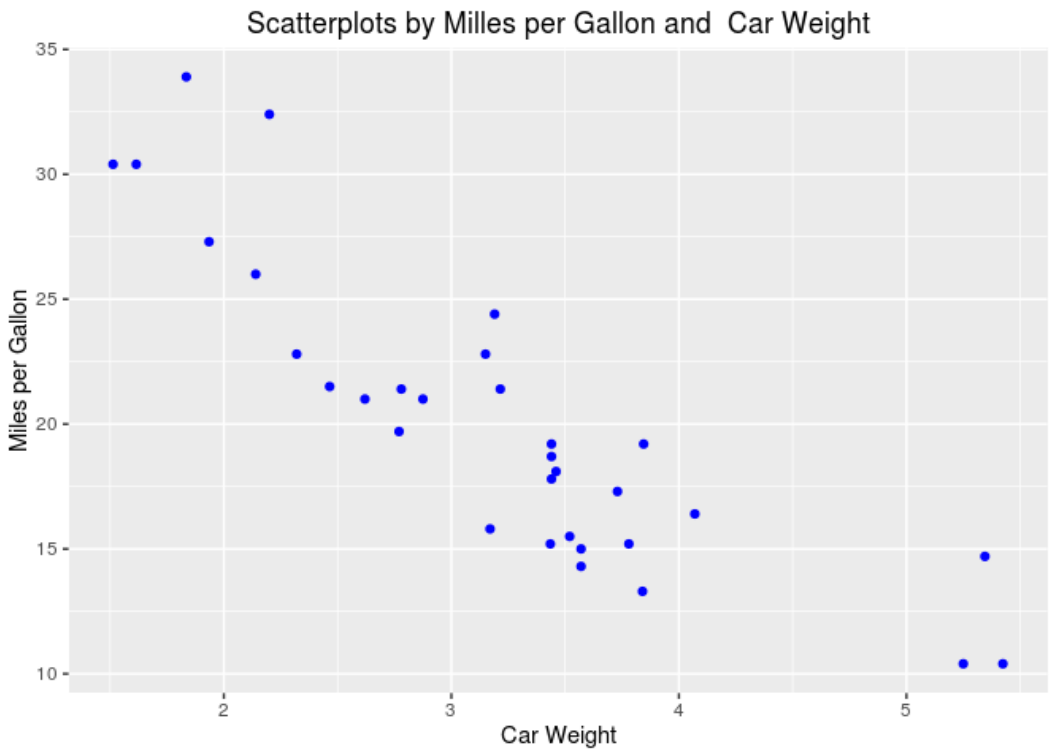

Figura 4.5.1.1 Gráfico de dispersión en R con ggplot2 (millas/galón y peso del vehículo)

Lenguaje	Librería utilizada	líneas de código	librería externas	Liga a repositorio
R	Graphics	3	0	https://github.com/cimat/data-visualization-patterns/blob/master/display-patterns/correlations/A11ScatterplotR.rmd#graphics
R	Lattice	4	1	https://github.com/cimat/data-visualization-patterns/blob/master/display-patterns/correlations/A11ScatterplotR.rmd#lattice
R	ggplot2	3	1	https://github.com/cimat/data-visualization-patterns/blob/master/display-patterns/correlations/A11ScatterplotR.rmd#ggplot2
Python	Matplotlib	9	2	https://github.com/cimat/data-visualization-patterns/blob/master/display-

				patterns/correlations/A11ScatterplotPy.md#code-example
Python	Seaborn	10	2	https://github.com/cimat/data-visualization-patterns/blob/master/display-patterns/correlations/A11ScatterplotPy.md#seaborn
Python	Pyqtgraph	16	2	https://github.com/cimat/data-visualization-patterns/blob/master/display-patterns/correlations/A11ScatterplotPy.md#pyqtgraph

Tabla 4.5.1.1. Tabla de librerías de gráficos de dispersión en R y Python

4.5.1.2 Gráfico de burbujas

Las gráficas de burbujas comparten ciertas similitudes con los diagramas de dispersión: se dibujan en un sistema de coordenadas cartesianas y proporcionan información sobre la correlación entre los atributos cuantitativos representados por los dos ejes de coordenadas. Pero en oposición a un diagrama de dispersión, los datos sin procesar de un gráfico de burbujas no consisten en una matriz de pares anónimos de variantes que sólo se vuelven significativos en el contexto de un grupo más grande de elementos. En cambio, cada conjunto de datos tiene asignada una etiqueta única, normalmente un nombre de texto sin formato para identificar el objeto correspondiente en la cuadrícula de coordenadas (Behrens, 2008).

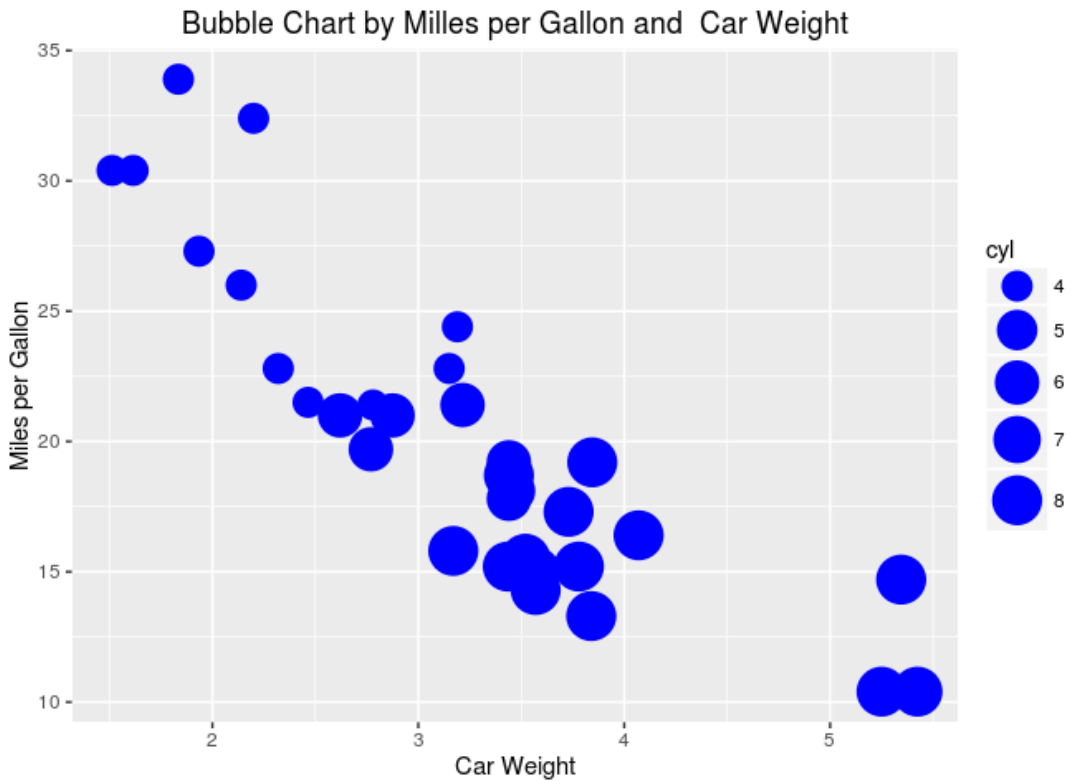

Figura 4.5.1.2 Gráfico de burbujas en R con ggplot2 (millas/galón y peso del vehículo según el número de cilindros)

Como se puede apreciar, la gráfica de burbujas es muy similar a la gráfica de dispersión, pero con la diferencia que la gráfica de burbuja agrega el elemento del volumen de las burbujas donde también representan otro tipo de dato.

Aunque la programación del patrón (gráfico de burbujas) en R con lattice tiene la cantidad de líneas de código similar que, en las librerías en Python, las otras dos librerías están por debajo del promedio de líneas de código en Python para este patrón, por lo tanto parece más simple la codificación de gráficos de burbujas en R por las líneas de código y las librerías externas utilizadas para la codificación.

Lenguaje	Librería utilizada	líneas de código	librería externas	Liga a repositorio
R	Graphics	3	0	https://github.com/cimat/data-visualization-patterns/wiki/A12-Bubble-Chart#graphics
R	Lattice	10	1	https://github.com/cimat/data-visualization-patterns/wiki/A12-Bubble-Chart#lattice
R	ggplot2	3	1	https://github.com/cimat/data-visualization-patterns/wiki/A12-Bubble-Chart#ggplot2
Python	Matplotlib	10	2	https://github.com/cimat/data-visualization-patterns/wiki/A12-Bubble-Chart#matplotlib
Python	Seaborn	12	3	https://github.com/cimat/data-visualization-patterns/wiki/A12-Bubble-Chart#seaborn
Python	Pyqtgraph	16	2	https://github.com/cimat/data-visualization-patterns/wiki/A12-Bubble-Chart#pyqtgraph

Tabla 4.5.1.2. Tabla de librerías de gráficos de burbujas en R y Python

4.5.2 Patrones de Cantidades Continuas

Los diagramas de cantidades continuas, están diseñados para plasmar datos estadísticos con puntos en un plano cartesiano, donde las cantidades son continuas un tras de otra en el eje de las “x” y llegan a plasmar su valor en el eje de las “y”, es decir la cantidad continua no puede quedar empalmada en un mismo punto del plano, a diferencia de las gráficas de correlación.

4.5.2.1 Gráfico de Línea Simple

El diagrama de líneas es un tipo de diagrama ampliamente utilizado, y debido a su estructura familiar fácil de entender. Además de los valores individuales, la información más significativa que se puede derivar de ella es el gradiente de la curva, que proporciona información sobre la intensidad del cambio del atributo en el tiempo. Además, los valores mínimos y máximos pueden identificarse fácilmente a partir de tal representación (Behrens, 2008).

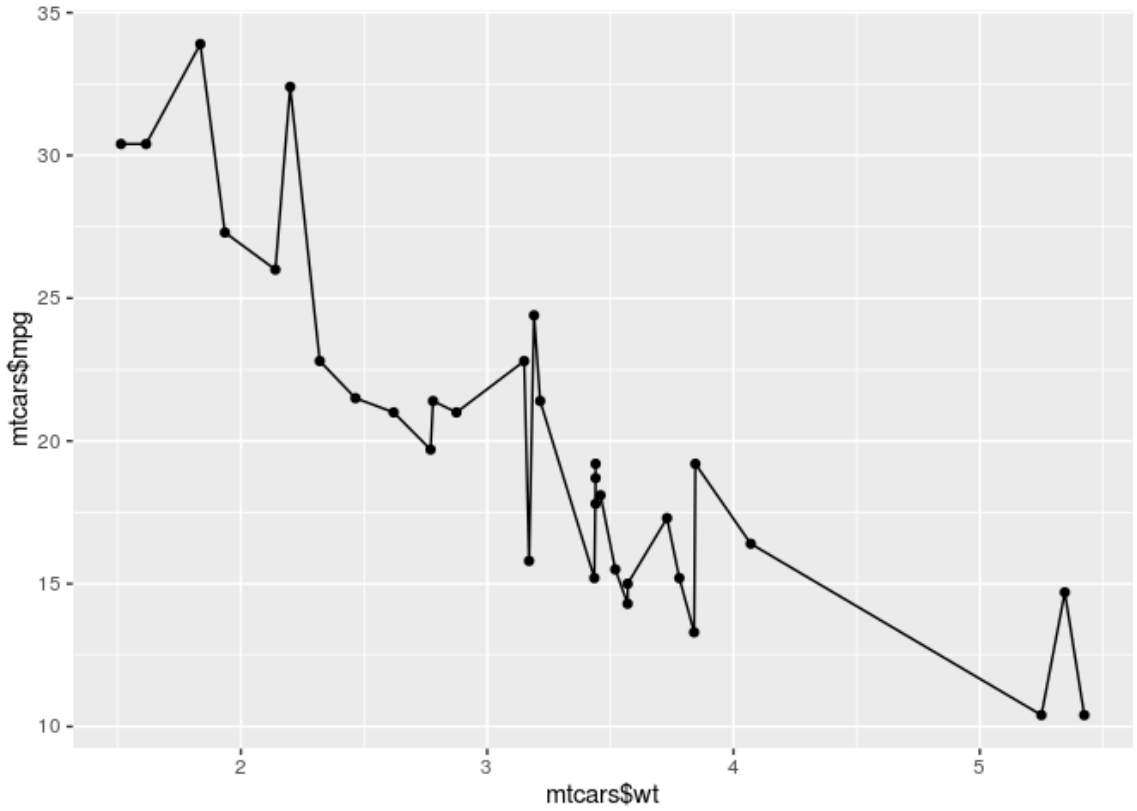

Figura 4.5.2.1 Gráfico de líneas en R con ggplot2 (millas/galón y peso del vehículo cada 1000 libras)

Dado que el gráfico de líneas es uno de los gráficos que se utilizan más comúnmente, la programación en R y en algunos otros lenguajes es muy sencilla, con no más de diez líneas de código por librería, generalmente para vistas más complejas es donde se vuelve más detallada la información de la programación de los gráficos, pero en general suele ser fácil de comprender y de ejecutar. En la tabla 4.5.2.1 se pueden apreciar con cuantas librerías se programó este tipo de gráfico en Python como en R y se puede ver que en los dos lenguajes fue sencilla su programación y con un número menor de líneas de código.

Lenguaje	Librería utilizada	líneas de código	librería externas	Liga a repositorio
R	Graphics	3	0	https://github.com/cimat/data-visualization-patterns/wiki/A21-Simple-Line-Chart#code-example-with-graphics
R	Lattice	8	1	https://github.com/cimat/data-visualization-patterns/wiki/A21-Simple-Line-Chart#code-example-with-lattice
R	ggplot2	4	1	https://github.com/cimat/data-visualization-patterns/wiki/A21-Simple-Line-Chart#code-example-with-ggplot
Python	Matplotlib	12	2	https://github.com/cimat/data-visualization-patterns/wiki/A21-Simple-Line-Chart#code-example-with-matplotlib
Python	Seaborn	17	4	https://github.com/cimat/data-visualization-patterns/wiki/A21-Simple-Line-Chart#code-example-with-seaborn
Python	Pyqtgraph	17	1	https://github.com/cimat/data-visualization-patterns/wiki/A12-Bubble-Chart#pyqtgraph

Tabla 4.5.2.1. Tabla de librerías de gráficos de línea simple en R y Python

En la tabla anterior se puede apreciar claramente que el patrón programado en R con las tres diferentes librerías es más sencillo de programar que en Python, puesto que no se utilizan tantas librerías externas ni tantas líneas de código como cuando es programado en Python.

4.5.2.2 Grafico Multilínea

En la mayoría de los casos, se utiliza un gráfico de líneas para mostrar el comportamiento de un solo valor en un intervalo. Sin embargo, hay situaciones en las que es importante permitir al usuario comparar directamente varias variables y su desarrollo en el mismo intervalo. Los gráficos multilínea son gráficos que permiten hacer las varias visualizaciones del patrón de línea simple en un solo gráfico, este gráfico permite hacer una comparación de los datos mostrados al mismo tiempo.

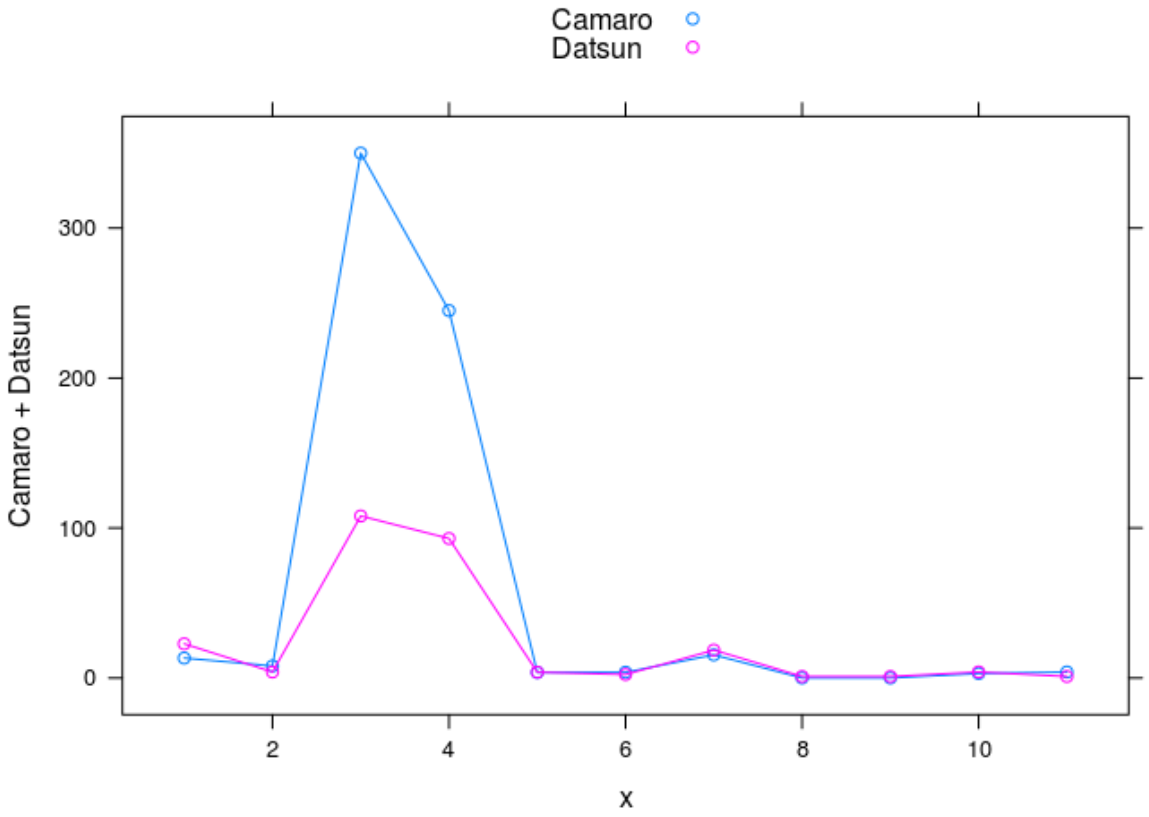

Figura 4.5.2.2 Comparación de desplazamiento entre Camaro Z28 y Datsun 710 Programada en R con el paquete de Latice.

Lenguaje	Librería utilizada	líneas de código	librería externas	Liga a repositorio
R	Graphics	5	0	https://github.com/cimat/data-visualization-patterns/wiki/A22-Multiset-Line-Chart#code-example-with-graphics
R	Lattice	6	1	https://github.com/cimat/data-visualization-patterns/wiki/A22-Multiset-Line-Chart#code-example-with-ggplot
R	ggplot2	5	1	https://github.com/cimat/data-visualization-patterns/wiki/A22-Multiset-Line-Chart#code-

				example-with-lattice
Python	Matplotlib	9	2	https://github.com/cimat/data-visualization-patterns/wiki/A22-Multiset-Line-Chart#code-example-with-matplotlibalization-patterns/wiki/A21-Simple-Line-Chart#code-example-with-matplotlib
Python	Seaborn	10	3	https://github.com/cimat/data-visualization-patterns/wiki/A22-Multiset-Line-Chart#code-example-with-seaborn
Python	Pyqtgraph	15	4	https://github.com/cimat/data-visualization-patterns/wiki/A22-Multiset-Line-Chart#code-example-with-pyqtgraph

Tabla 4.5.2.2. Tabla de librerías de gráficos de Multilínea en R y Python

El promedio de líneas de código utilizadas en R para este patrón es de 5 líneas de código por paquete, en cambio en la codificación en Python el promedio de líneas de código por librería utilizada es de 11 líneas de código, también el promedio de paquetes externos a R por paquete es menor a 1, puesto que R tiene paquetes predeterminados para la visualización del modelo, el promedio de librerías utilizadas en Python para el desarrollo del modelo es de 3 librerías externas para generación del gráfico. El número de líneas de líneas de código también puede variar en qué tan simple se muestra el gráfico, pero para estos gráficos se manejaron gráficos simples, es decir no tan detallados tanto colores como en texto.

4.5.2.3 Gráfica de área apilada

Un gráfico de área muestra el desarrollo de valores cuantitativos a lo largo de un intervalo. Se parece a un gráfico de líneas, ya que utiliza líneas que conectan puntos de datos entre sí. La característica específica de la gráfica de área es que el área debajo de la línea está llena de cierto color o textura. Cuando se agregan más líneas al gráfico, se "apilan" encima de la línea anterior. Esto hace que el gráfico de área sea un tipo de diagrama preferido para mostrar varios conjuntos de datos dentro del mismo gráfico con la tensión en una magnitud total que se añaden varias variables.

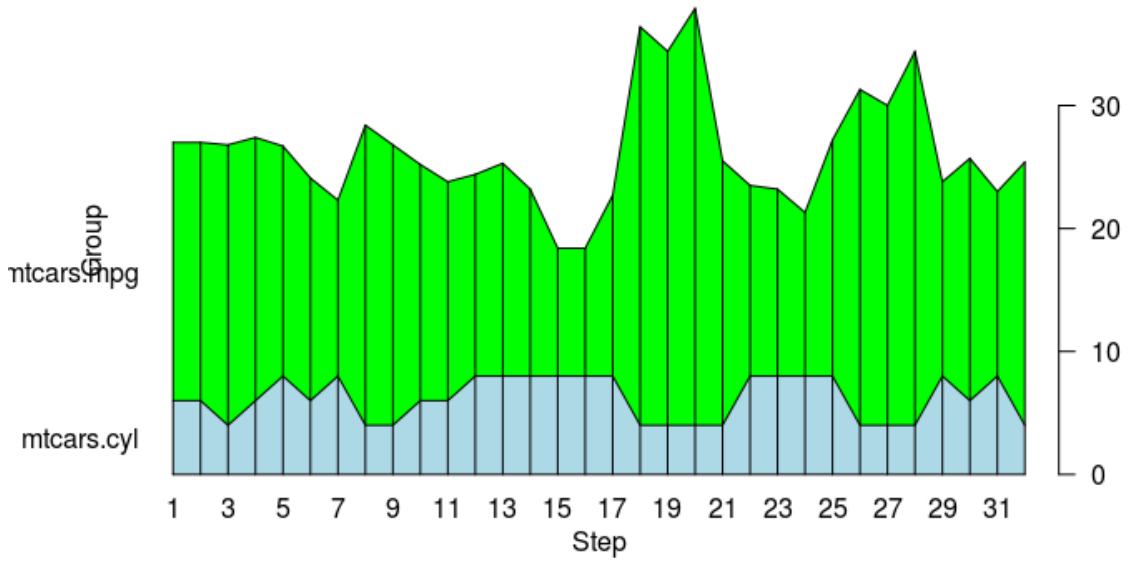

Figura 4.5.2.3 Comparación de Millas/Galón y número de cilindros de los 32 automóviles en el paquete mtcars Programada en R con Graphics.

Conforme se van haciendo más complicados los gráficos, menos paquetes son capaces de programarlos, para este ejemplo sólo se utilizan dos paquetes/librerías tanto en R cómo en Python para programar el patrón, como se puede apreciar en la tabla 4.5.2.3.

Lenguaje	Librería utilizada	líneas de código	librería externas	Liga a repositorio
R	Graphics	3	0	https://github.com/cimat/data-visualization-patterns/wiki/A23-Stacked-Area-Chart#code-example-with-graphics
R	Ggplot	7	1	https://github.com/cimat/data-visualization-patterns/wiki/A23-Stacked-Area-Chart#code-example-with-ggplot2
Python	Plotly	4	14	https://github.com/cimat/data-visualization-patterns/wiki/A23-Stacked-Area-Chart#code-example-with-matplotlib-and-plotly
Python	Seaborn	15	4	https://github.com/cimat/data-visualization-patterns/wiki/A23-Stacked-Area-Chart#code-example-with-seaborn

Tabla 4.5.2.3 Paquetes y librerías utilizadas para desarrollo de gráfico de área apilada

4.5.3 Cantidades Discretas

Las cantidades discretas a menudo suelen ser representadas a partir de valores numéricos, esto permite que al momento de graficar estos valores pueden ser divididos, haciendo más sencilla la forma de visualización, en este meta patrón existen patrones muy comunes como son las gráficas barra, multibarra e incluso matrices.

4.5.3.1 Grafico de barra simple

El gráfico de barras o gráfica de columnas es uno de los gráficos más comúnmente utilizados por su simplicidad y sencillez al comprender. Los gráficos de barras muestran un conjunto de datos separado por barras proporcionales a los datos analizados.

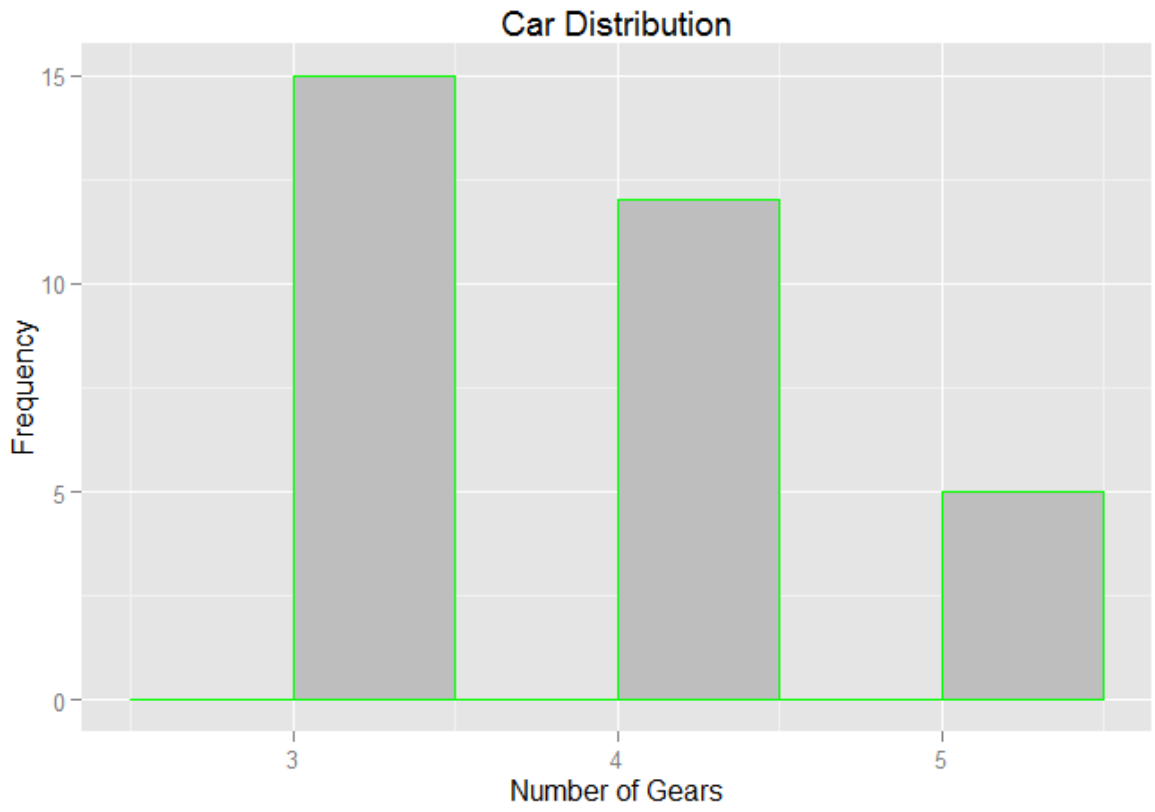

Figura 4.5.3.1 Distribución de autos con el número de velocidades según la frecuencia en los 32 automóviles en el paquete mtcars Programada en R con Lattice.

Puesto que el gráfico de barras es uno de los gráficos más comunes, gran parte de las librerías/paquetes desarrolladas para la visualización de datos son capaces de implementarlas. Cabe mencionar que algunos paquetes con gráficos complicados no son capaces de implementarlos ya que están desarrollados para gráficos específicos. Dada la sencillez del tipo de gráfico y su habitualidad, no es compleja su programación y frecuentemente se hace con pocas líneas de código.

Lenguaje	Librería utilizada	líneas de código	librería externas	Liga a repositorio
R	Graphics	2	0	https://github.com/cimat/data-visualization-patterns/blob/master/display-patterns/Discrete%20Quantities/A31Simple_Bar_ChartR.md#graphics
R	Lattice	3	1	https://github.com/cimat/data-visualization-patterns/blob/master/display-patterns/Discrete%20Quantities/A31Simple_Bar_ChartR.md#ggplot2
R	ggplot2	3	1	https://github.com/cimat/data-visualization-patterns/blob/master/display-patterns/Discrete%20Quantities/A31Simple_Bar_ChartR.md#ggplot2
Python	Matplotlib	11	3	https://github.com/cimat/data-visualization-patterns/wiki/A22-Multiset-Line-Chart#code-example-with-matplotlibalization-patterns/wiki/A21-Simple-Line-Chart#code-example-with-matplotlib
Python	Seaborn	14	5	https://github.com/cimat/data-visualization-patterns/wiki/A22-Multiset-Line-Chart#code-example-with-seaborn
Python	Pyqtgraph	17	4	https://github.com/cimat/data-visualization-patterns/wiki/A22-Multiset-Line-Chart#code-example-with-pyqtgraph

Tabla 4.5.3.1. Tabla de librerías de gráficos de barra en R y Python

4.5.3.2 Gráfico de multibarras

Este tipo de diagrama permite a los usuarios inspeccionar y comparar varios conjuntos de datos cuantitativos mientras ocupa significativamente menos espacio de visualización que un conjunto de múltiples gráficos de barras convencionales (Behrens, 2008).

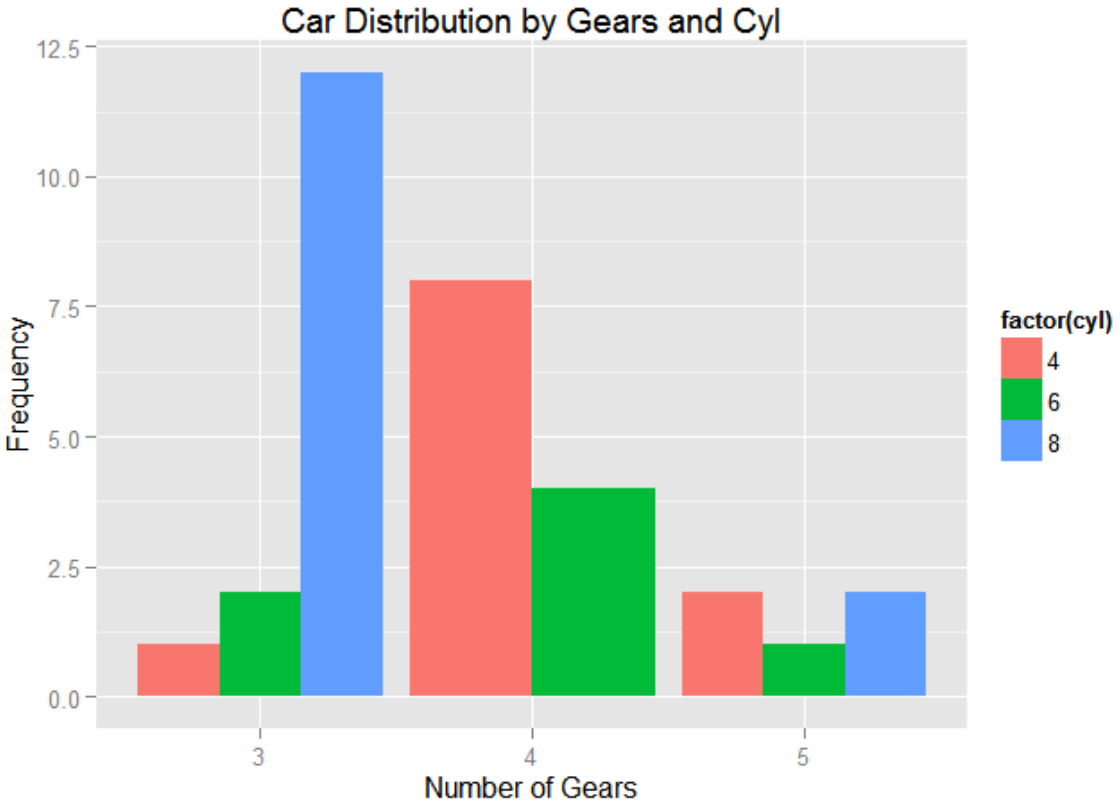

Figura 4.5.3.2 Distribución de autos con el número de velocidades y cilindros según la frecuencia en los 32 automóviles en el paquete mtcars Programada en R con ggplot2

Tomando en cuenta que al igual que los gráficos de barra, los gráficos multibarra son gráficos comúnmente utilizados para la visualización de datos, por el mismo motivo no es tan complicada su codificación en algunos lenguajes, ya que existen muchos paquetes y librerías que permiten hacer el desarrollo.

En la tabla 4.5.3.2 se puede apreciar las líneas de código y paquetes/librerías externas utilizadas para programar gráficos de multibarra tanto en Python como en R.

Lenguaje	Librería utilizada	líneas de código	librería externas	Liga a repositorio
R	Graphics	3	0	https://github.com/cimat/data-visualization-patterns/wiki/A32-Multiset-Bar-Chart#graphics
R	Lattice	4	1	https://github.com/cimat/data-visualization-patterns/wiki/A32-Multiset-Bar-Chart#lattice
R	ggplot2	3	1	https://github.com/cimat/data-visualization-patterns/wiki/A32-Multiset-Bar-Chart#ggplot2
Python	Matplotlib	15	3	https://github.com/cimat/data-visualization-patterns/wiki/A32-Multiset-Bar-Chart#matplotlib
Python	Seaborn	9	3	https://github.com/cimat/data-visualization-patterns/wiki/A32-Multiset-Bar-Chart#seaborn
Python	Pyqtgraph	21	5	https://github.com/cimat/data-visualization-patterns/wiki/A32-Multiset-Bar-Chart#pyqtgraph

Tabla 4.5.3.2. Tabla de librerías de gráficos de multibarra en R y Python

4.5.3.3 Matriz de puntos

La matriz de puntos es una forma de representación unidimensional para datos cuantitativos discretos. En lugar de fusionar los valores discretos en una representación continua, tal como una barra o un gráfico circular, retiene la capacidad contable de los datos empleando una serie de elementos representados. Por lo tanto, este patrón de representación sólo es útil cuando se aplica una proporción de escala apropiada (Behrens, 2008).

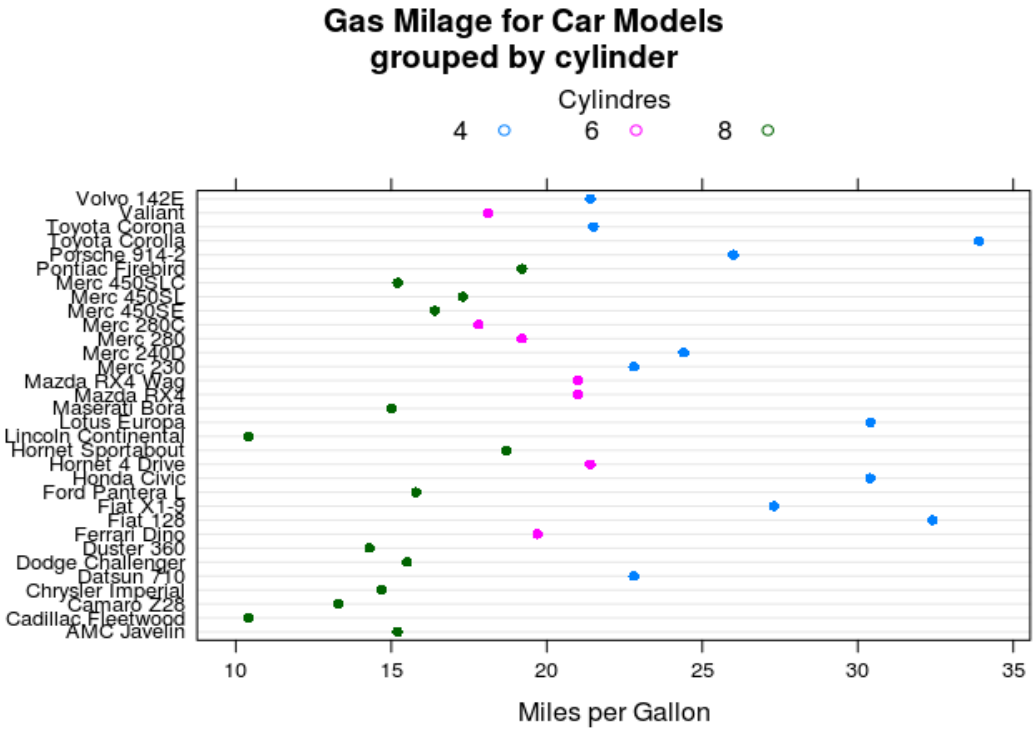

Figura 4.5.3.3 Distancia en millas por modelos de auto agrupadas por cilindros según la frecuencia en los 32 automóviles en el paquete mtcars Programada en R con Lattice

En comparación con otros patrones de visualización, el patrón de matriz de puntos tiene diferencias más marcadas en la codificación entre R y Python, como se puede observar en la tabla 4.5.3.3. Los códigos desarrollados en Python son hasta tres veces más extensos que los programados en R y las librerías externas que se utilizan van de cero a uno en R y de cinco librerías externas para los códigos en Python.

Lenguaje	Librería utilizada	líneas de código	librería externas	Liga a repositorio
R	Graphics	10	0	https://github.com/cimat/data-visualization-patterns/wiki/A33-Dot-Matrix#graphics
R	Lattice	2	1	https://github.com/cimat/data-visualization-patterns/wiki/A33-Dot-Matrix#lattice
R	ggplot2	3	1	https://github.com/cimat/data-visualization-patterns/wiki/A33-Dot-Matrix#ggplot2
Python	Matplotlib	38	5	https://github.com/cimat/data-visualization-patterns/wiki/A33-Dot-Matrix#matplotlib
Python	Seaborn	39	5	https://github.com/cimat/data-visualization-patterns/wiki/A33-Dot-Matrix#seaborn
Python	Pyqtgraph	49	5	https://github.com/cimat/data-visualization-patterns/wiki/A33-Dot-Matrix#pyqtgraph

Tabla 4.5.3.3. Tabla de librerías de gráficos de multibarra en R y Python

4.5.3.4 Gráfico de barras de apilado

En un gráfico de barras apilado, cada barra representa un conjunto completo de datos cuantitativos. Una barra se separa entonces en segmentos de modo que cada segmento representa un solo elemento del conjunto correspondiente. Los gráficos de barras apilados son, por tanto, una representación alternativa para los datos cuantitativos con múltiples conjuntos de datos (Behrens, 2008).

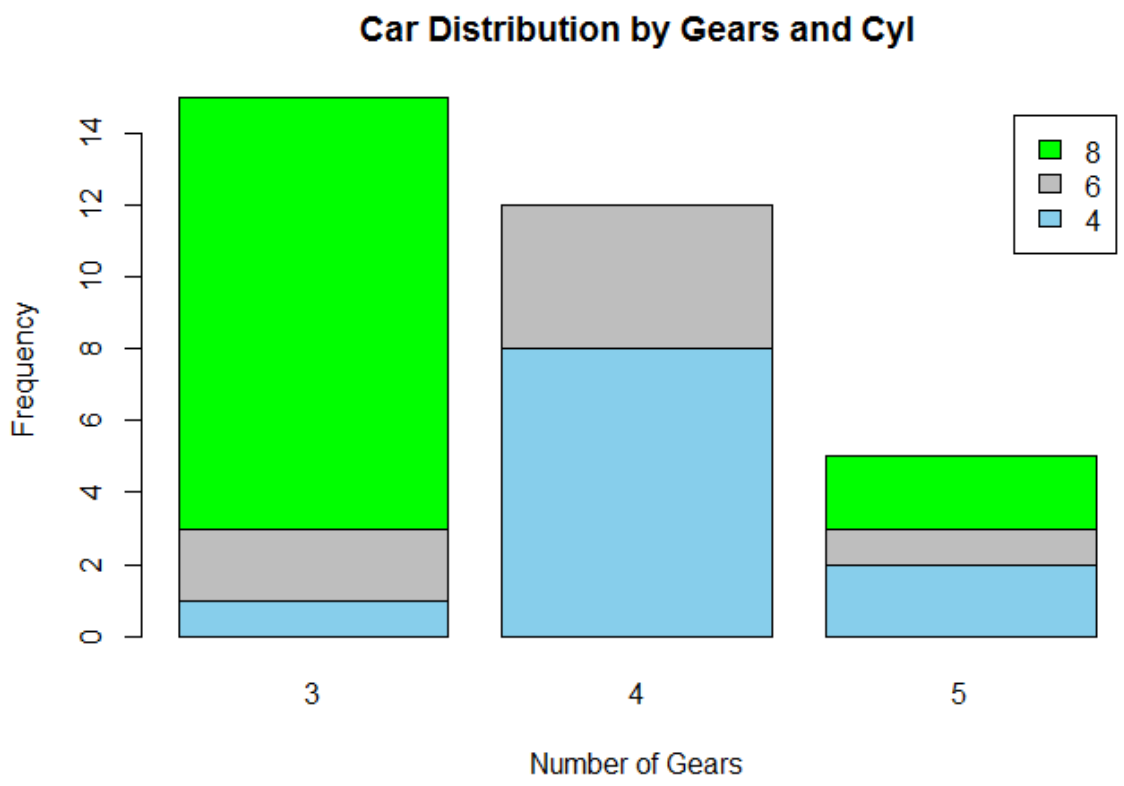

Figura 4.5.3.4 Distribución de automóviles por número de velocidades y cilindros según la frecuencia en los 32 automóviles en el paquete mtcars Programada en R con Graphics.

Los gráficos de barras apiladas tienen la capacidad de mostrar el contenido de varios sets de datos en una solo gráfica, por su sencillez de programar y su facilidad de comprensión es que los gráficos de barras apiladas son tan comunes dentro del campo de la ciencia de datos. Programar este patrón en R es muy sencillo y con unas cuantas líneas de código se pueden tener gráficos completos y sencillos de comprender.

Lenguaje	Librería utilizada	líneas de código	librería externas	Liga a repositorio
R	Graphics	4	0	https://github.com/cimat/data-visualization-patterns/wiki/A34-Stacked-Bar-Chart#graphics
R	Lattice	4	1	https://github.com/cimat/data-visualization-patterns/wiki/A34-Stacked-Bar-Chart#lattice
R	ggplot2	3	1	https://github.com/cimat/data-visualization-patterns/wiki/A34-Stacked-Bar-Chart#ggplot2
Python	Matplotlib	16	4	https://github.com/cimat/data-visualization-patterns/wiki/A34-Stacked-Bar-Chart#matplotlib
Python	Seaborn	16	4	https://github.com/cimat/data-visualization-patterns/wiki/A34-Stacked-Bar-Chart#seaborn
Python	Pyqtgraph	22	6	https://github.com/cimat/data-visualization-patterns/wiki/A34-Stacked-Bar-Chart#pyqtgraph

Tabla 4.5.3.4. Tabla de librerías de gráficos de barras apiladas en R y Python

Aparentemente la diferencia en líneas de código es bastante entre Python y R puesto que para la programación de los patrones en Python es necesario detallar bien los aspectos visuales del patrón, tales como tamaños, colores, textos y distribuciones, que estos en algunos paquetes para R están implícitos.

4.5.3.5 Gráfico de barras isométricas

Las gráficas de barras isométricas utilizan la proyección en perspectiva de elementos visuales para mostrar varios conjuntos de datos cuantitativos dentro del mismo diagrama que se superponen en un gráfico bidimensional convencional. De hecho, representan un estilo alternativo de gráficos de barras multiconjunto en lugar de romper los diferentes conjuntos y colocar grupos de barras de la misma variable uno al lado del otro, permiten al diseñador mantener la estructura original de cada sub-cuadro (Behrens, 2008).

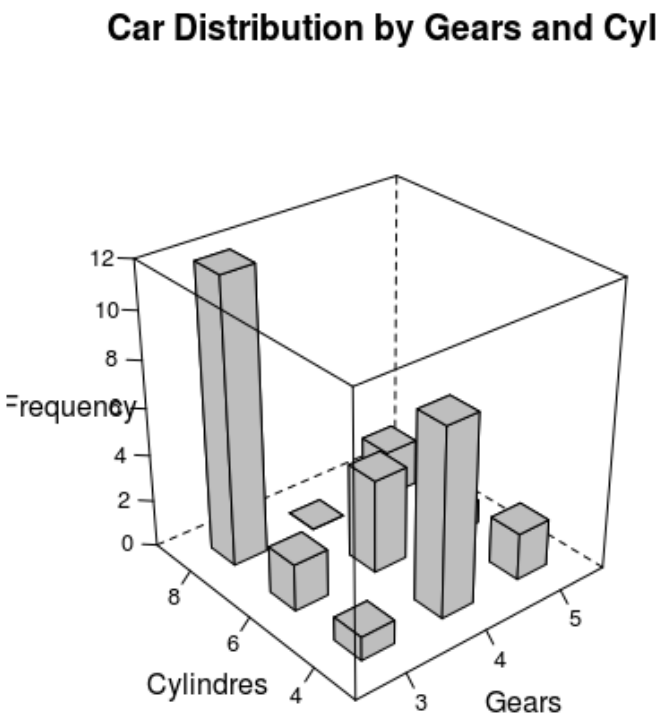

Figura 4.5.3.5 Distribución de automóviles por número de velocidades y cilindros en un gráfico de barras isométrico según la frecuencia en los 32 automóviles en el paquete mtcars Programada en R con Lattice.

El gráfico de barras isométricas a pesar que es sencillo de entender, también para algunos científicos de datos no es un gráfico que se utilizará para hacer visualización de grandes cantidades de datos, puesto que entre más sets de datos tenga integrado puede tender a mostrar “datos chatarra” o se vuelve más complicado de interpretar.

Para este patrón dentro de las librerías utilizadas en Python, como en los paquetes en R, solo una librería y un paquete respectivamente son útiles para desarrollar el gráfico de barras isométricas, pero las diferencia en las líneas de código y uso de librerías externas siguen siendo muy marcadas entre los dos lenguajes de programación.

Lenguaje	Librería utilizada	líneas de código	librería externas	Liga a repositorio
R	Lattice	8	0	https://github.com/cimat/data-visualization-patterns/wiki/A35-Isometric-Bar-Chart#lattice
Python	Matplotlib	21	4	https://github.com/cimat/data-visualization-patterns/wiki/A35-Isometric-Bar-Chart#matplotlib

Tabla 4.5.3.5. Tabla de librerías de gráficos de barras isométricas en R y Python

4.5.3.6 Gráfico de lapsos

Se utilizan gráficos de intervalo para mostrar varios conjuntos de datos, cada conjunto deberá contener un valor mínimo y uno máximo estos valores extremos son de interés para el observador. Es decir, la tabla de datos contiene dos columnas variables, aplicando la misma escala y unidad. Estos conjuntos de datos también pueden contener más de dos valores. Sin embargo, esos valores "intermedios" se perderán como materia mínima y máxima para este tipo de representación (Behrens, 2008).

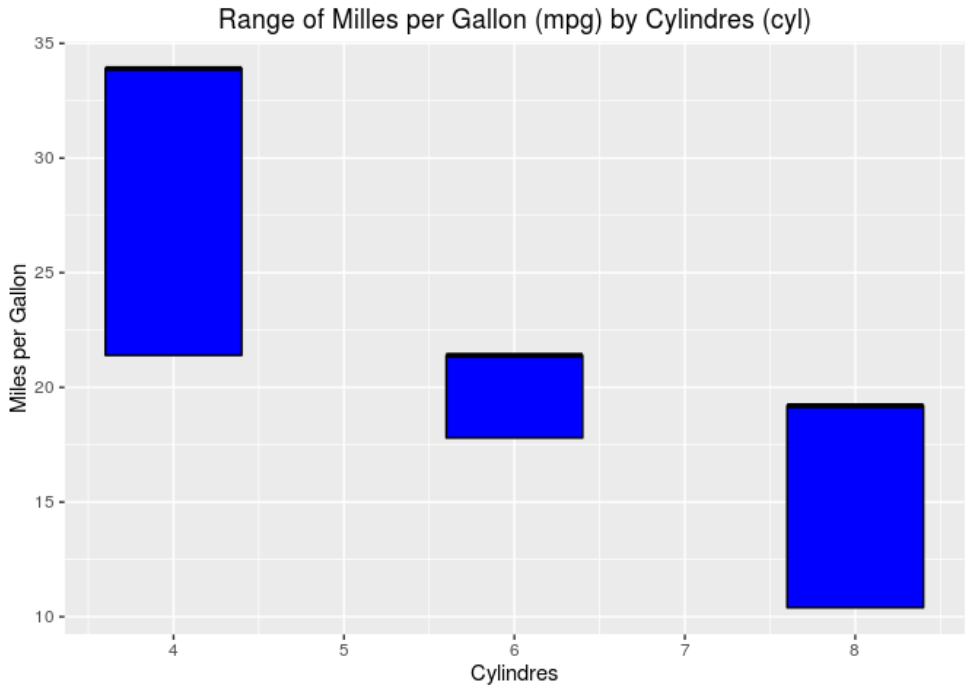

Figura 4.5.3.6. Gráfico de lapsos que muestra el rango de millas por galón y cilindros Programada en R con Ggplot2.

Para este patrón de visualización de datos, en Python solo se pudo desarrollar con dos de las librerías que se decidieron utilizar para este trabajo, en cambio para el desarrollo del patrón en R se utilizaron tres paquetes, para este modelo en cambio con los anteriores la diferencia en los tamaños de código no es tan marcada como se muestra en la tabla 4.5.3.6.

Lenguaje	Librería utilizada	líneas de código	librería externas	Liga a repositorio
R	Graphics	12	0	https://github.com/cimat/data-visualization-patterns/wiki/A36-Span-Chart#graphics
R	Lattice	18	1	https://github.com/cimat/data-visualization-patterns/wiki/A36-Span-Chart#lattice
R	ggplot2	10	1	https://github.com/cimat/data-visualization-patterns/wiki/A36-Span-Chart#ggplot2
Python	Matplotlib	17	4	https://github.com/cimat/data-visualization-patterns/wiki/A36-Span-Chart#matplotlib
Python	Seaborn	16	4	https://github.com/cimat/data-visualization-patterns/wiki/A36-Span-Chart#seaborn

Tabla 4.5.3.6. Tabla de librerías de gráficos de barras apiladas en R y Python

4.5.4 Patrones de proporciones

Los patrones de proporciones son capaces de demostrar proporciones porcentuales de los tipos de datos dentro de un conjunto en el caso de las gráficas de pastel y de un multiconjunto de datos en el caso de las gráficas de anillo. Los gráficos de proporciones son de los gráficos más comúnmente utilizados no solo dentro de la ciencia de datos, también en la industria al realizar cualquier tipo de reportes dada su sencillez y facilidad de lectura y comprensión.

4.5.4.1 Gráfico de pastel simple

Un gráfico pastel es un objeto circular dividido en múltiples segmentos polares. Muestra la distribución de varios valores que pertenecen al mismo conjunto de datos. El círculo completo representa la magnitud total de este conjunto de datos, igual al 100 por ciento, mientras que cada segmento representa la magnitud de una variable particular. El área de segmento, la longitud del arco y el ángulo de arco de cada segmento son proporcionales al valor que representa el segmento. Los segmentos de un gráfico circular generalmente se etiquetan con números porcentuales en lugar de valores totales (Behrens, 2008).

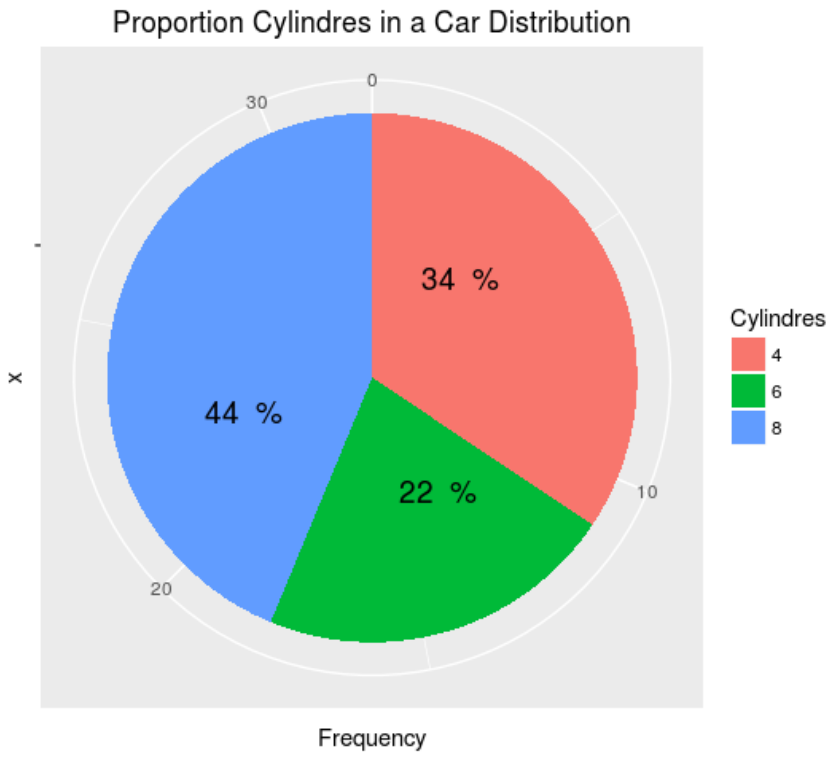

Figura 4.5.4.1 Gráfico de lapsos que muestra la proporción de cilindros en distribución de autos Programada en R con Ggplot2.

Como se puede apreciar en la tabla 4.5.4.1. A pesar que el gráfico de pastel es comúnmente utilizado, cuando se programa no suele ser tan sencillo como los anteriores y tanto en Python como en R se llevan número de líneas de código muy cercanas en algunas de las librerías y paquetes utilizados, para este modelo también se utilizaron más líneas de código y librerías externas en Python que en R, pero la programación en lattice R no estuvo tan lejana en cuanto a líneas de código que los desarrollos en Python, incluso fue más grande el código de lattice que en Seaborn y Matplotlib de Python.

Lenguaje	Librería utilizada	líneas de código	librería externas	Liga a repositorio
R	Graphics	6	0	https://github.com/cimat/data-visualization-patterns/wiki/A41-Simple-Pie-Chart#graphics
R	Lattice	40	1	https://github.com/cimat/data-visualization-patterns/wiki/A41-Simple-Pie-Chart#lattice
R	ggplot2	9	1	https://github.com/cimat/data-visualization-patterns/wiki/A41-Simple-Pie-Chart#ggplot2
Python	Matplotlib	11	3	https://github.com/cimat/data-visualization-patterns/wiki/A41-Simple-Pie-Chart#matplotlib
Python	Seaborn	13	4	https://github.com/cimat/data-visualization-patterns/wiki/A41-Simple-Pie-Chart#seaborn
Python	Pyqtgraph	45	5	https://github.com/cimat/data-visualization-patterns/wiki/A41-Simple-Pie-Chart#pyqtgraph

Tabla 4.5.4.1. Tabla de paquetes y librerías de gráficos de pastel en R y Python

4.5.4.2 Gráfico de anillo

Los gráficos de anillo son una extensión del gráfico de pastel convencional. Permiten la visualización de varios conjuntos de datos de la misma configuración, o aquellos con datos relacionados entre sí, dentro de un gráfico, haciendo que los diferentes conjuntos y sus características sean inmediatamente comparables (Behrens, 2008).

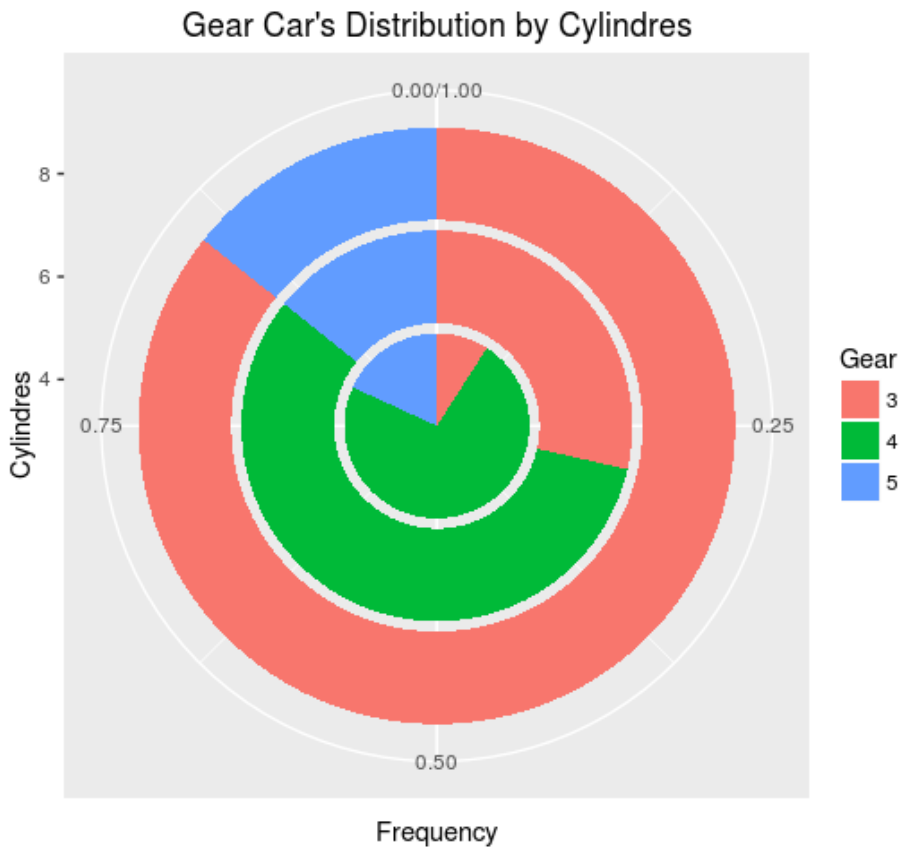

Figura 4.5.4.2 Gráfico de anillos muestra la distribución de velocidades por cilindros por automóvil desarrollada en R con Ggplot

Para el desarrollo de este modelo tanto en R como en Python sólo fue posible llevarlo a cabo con un paquete y una librería y la diferencia entre el tamaño de líneas de código de los programas y las librerías externas son realmente marcadas.

Lenguaje	Librería utilizada	líneas de código	librería externas	Liga a repositorio
R	Ggplot	8	1	https://github.com/cimat/data-visualization-patterns/wiki/A42-Ring-Chart#ggplot2
Python	Matplotlib	42	4	https://github.com/cimat/data-visualization-patterns/wiki/A42-Ring-Chart#matplotlib

Tabla 4.5.4.2. Tabla de paquetes y librerías de gráfico de anillo en R y Python

4.5.5 Patrones de flujo

Los patrones de flujo no se centran en las propiedades cuantitativas de elementos de datos individuales, sino en la naturaleza secuencial de la estructura de datos como un todo. No se enfatiza en la cuestión de la totalidad estática de una condición es de interés primario, sino en la cuestión de cómo esta condición va evolucionando.

4.5.5.1 Gráfico Sankey

El Gráfico Sankey es un gráfico que contiene entradas y salidas, que representan un flujo de datos, respectivamente. En tales diagramas, es fácil ver representaciones de los esfuerzos de entrada y salida.

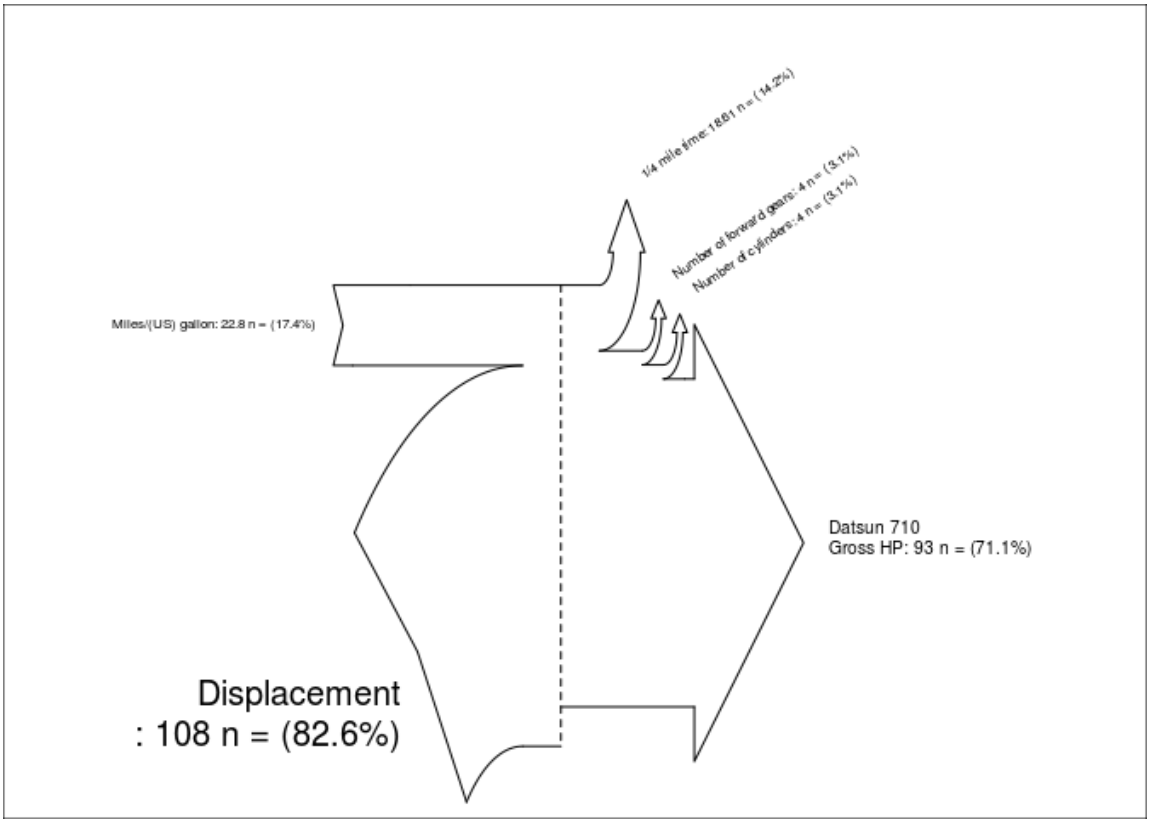

Figura 4.5.5.1 Gráfico Sankey muestra la entrada de millas por galón y el desplazamiento y las salidas respectivas en Datsun 710 desarrollado en R con Graphics

Para el desarrollo de este patrón sólo es posible con una librería en Python y una en R con las q se ha estado trabajando, también se programó en R con otro paquete externo desarrollado por google llamado googlevis y que cumple con los requerimientos establecidos para el uso de paquetes en este trabajo.

Lenguaje	Librería utilizada	líneas de código	librería externas	Liga a repositorio
R	Graphics	19	0	https://github.com/cimat/data-visualization-patterns/wiki/A51-Sankey-Diagram#code-example-with-graphics
R	Googlevis	16	1	https://github.com/cimat/data-visualization-patterns/wiki/A51-Sankey-Diagram#code-example-with-googlevis
Python	Matplotlib	21	4	https://github.com/cimat/data-visualization-patterns/wiki/A51-Sankey-Diagram#code-example-with-matplotlib

Tabla 4.5.5.1. Tabla de paquetes y librerías de gráfico Sankey en R y Python

4.5.5.2 Diagrama de hilo de arcos

Los "hilos" se usan principalmente en sistemas de comunicación en línea tales como correo electrónico para describir un grupo de mensajes que se relacionan entre sí. Cuando una persona crea un mensaje al que responden una o más personas, todos estos mensajes forman un hilo que se relacionan entre sí con respecto a un tema común. Por lo tanto, un hilo es una estructura estrictamente ordinal, lo que significa que cada elemento tiene lugar fijo en un orden causal.

Para este tipo de diagramas no se encontró ninguna librería en Python para su desarrollo, tampoco se encontró una en R de las que se han venido utilizando, pero sí se encontró otro paquete externo llamado Arcdiagram que cumple con los requerimientos establecidos para el uso de paquetes para este trabajo.

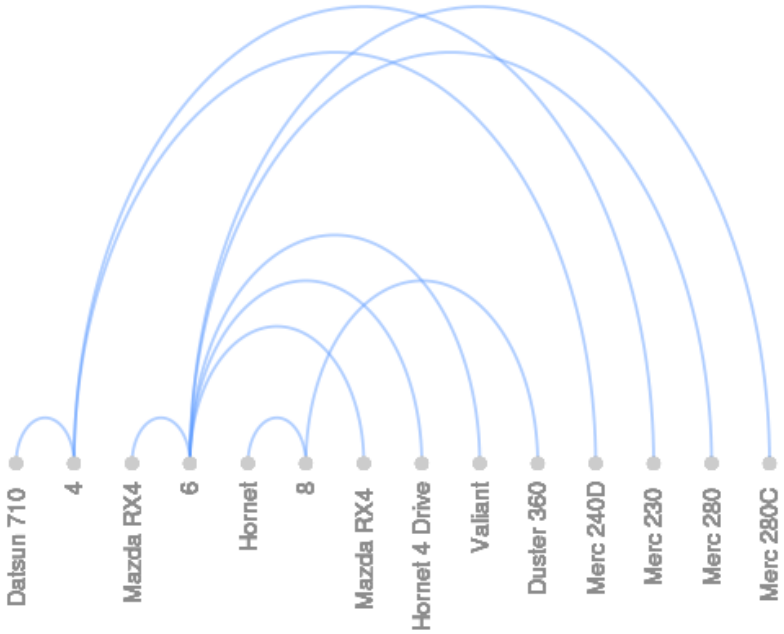

Figura 4.5.5.2. diagrama de hilo de arco que muestra la relación entre vehículos y el número de cilindros desarrollado en R con Arcdiagram.

Para el Script desarrollado en R con Arcdiagram para la visualización del modelo se utilizó una librería externa (Arcdiagram) y se necesitaron sólo siete líneas de código para el desarrollo.

4.5.6 Patrones de Jerarquía

Los patrones jerárquicos son esquemas de orden de hereditarios donde se generan conexiones entre nodos con ciertas relaciones parentales, pero también con propiedades individuales.

Para el alcance de este trabajo de tesis se llegó solo hasta el patrón de diagramas de árbol, pero también dentro de los patrones de jerarquía está el mapa de árbol el cual no fue desarrollado.

4.5.6.1 Diagrama de árbol

Un diagrama de árbol es una representación gráfica de una estructura estrictamente jerárquica, que describe un conjunto de elementos y sus relaciones entre sí. Todos los elementos de una estructura de este tipo tienen relaciones familiares con otros elementos, clasificándolos en un nivel superior, inferior o igual en la jerarquía del conjunto. Un elemento en particular (el elemento raíz) no tiene ningún elemento superior o igual (Behrens, 2008).

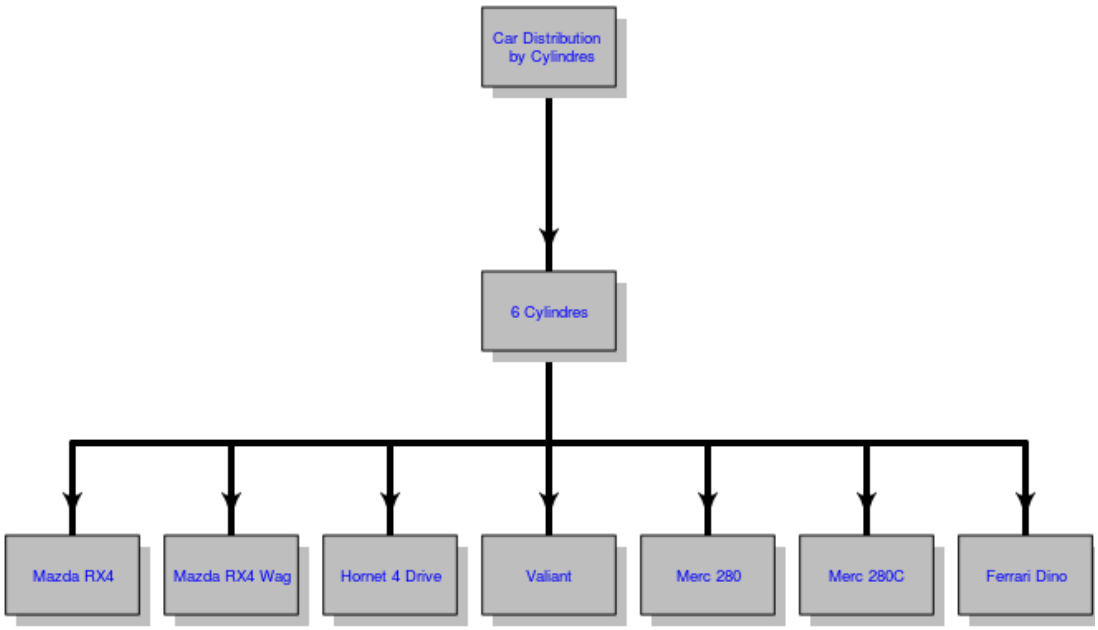

Figura 4.5.6.1 Diagrama de árbol de la distribución de automóviles con seis cilindros desarrollado en R con Diagram

El desarrollo en R solo se realizó con un paquete externo llamado diagram, pero en python se logró desarrollar con dos librerías como se muestra en la Tabla 4.5.6.1. Aunque hay más librerías capaces de soportar el modelo, cuando es desarrollado en R se utilizan menos líneas de código y librerías externas.

Lenguaje	Librería utilizada	líneas de código	librería externas	Liga a repositorio
R	Graphics	12	1	https://github.com/cimat/data-visualization-patterns/wiki/A61-Tree-Diagram#diagram
Python	Graphvis	18	2	https://github.com/cimat/data-visualization-patterns/wiki/A61-Tree-Diagram#graphviz
Python	Matplotlib	50	7	https://github.com/cimat/data-visualization-patterns/wiki/A61-Tree-Diagram#matplotlib

Tabla 4.5.6.1. Tabla de paquetes y librerías de gráfico de árbol en R y Python

Capítulo 5

Conclusiones y trabajos futuros

En este capítulo se muestran las conclusiones obtenidas a partir del trabajo desarrollado obtenidas de la hipótesis propuesta, también se presentan los trabajos futuros que podrían realizarse en base de este trabajo de tesis.

5.1 Conclusiones

A partir del análisis de resultados del experimento realizado, que consistió en la programación de patrones de visualización de datos en R propuestos por Behrens y la comparación de los mismos pero desarrollados en Python se obtuvieron las conclusiones a continuación mostradas. Es imprescindible mencionar que para el desarrollo de este trabajo se tomaron solo los patrones de despliegue hasta el patrón de jerarquía llamado diagrama de árbol, programando un total de dieseis de los patrones de despliegue propuestos.

Las gráficas obtenidas de la programación de los patrones de despliegue hasta donde se dio el alcance, fueron en superioridad desarrolladas en los dos lenguajes de programación y por lo menos en su mayoría por una o más librerías/paquetes. Sólo el patrón de diagramas de hilo de arcos no fue capaz de ser desarrollado en Python, pero fue desarrollado en R no con uno de los paquetes que desde el inicio se estaban utilizando y si con otro que cumplía con los requerimientos del uso de paquetes. Todos los patrones fueron implementados a partir del conjunto de datos mtcars, el cual es un set de datos que está implícito en lenguaje de programación R y se puede utilizar también en Python.

De la validación y el análisis de resultados del experimento realizado, se tomaron las conclusiones del lenguaje, de la programación y la calidad de los programas y de los tipos de gráficos generados, tomando en cuenta factores como el set de datos utilizado y el alcance de los patrones desarrollados.

5.1.1 Del lenguaje de programación

Puesto que R es un lenguaje de programación creado para el análisis de datos, su sintaxis es sencilla de comprender y con sets de datos pequeños como mtcars es rápido en el análisis de los mismos, es un lenguaje de código abierto y casi tan útil como usar una hoja de cálculo, con la diferencia que es mucho más potente y programable, sin embargo Python aunque no es un lenguaje cien por ciento desarrollado para la visualización de datos y aunque su sintaxis no sea tan sencilla como la de R también el análisis de datos es rápido en sets de datos como mtcars y actualmente es uno de los lenguajes más utilizados para análisis y visualización, por lo tanto para este trabajo como lenguaje de programación son muy parejos tanto Python como R puesto que el desarrollo de los programas no implican grandes conjuntos de datos.

5.1.2 De la programación y la calidad de los programas

A partir de las métricas de calidad de software utilizadas para este trabajo a conseguidas de la selección en AHP, para la medición de la calidad de los programas fue el total de las líneas de código y el número de las librerías externas para el desarrollo. Tomando en cuenta lo anterior y el análisis de los resultados la calidad de los programas desarrollados en R es mucho mejor a la calidad de los programas desarrollados en Python puesto que el 93% de los programas realizados en R sus líneas de código fueron menores a las de Python y el 73% de los programas en Python tienen el doble o más líneas de código que los programados en R.

En cuanto a las librerías/paquetes externos utilizados, los programas desarrollados en R el 65% usa solo una librería externa, el 35% no utiliza librerías externas, en cambio los programas desarrollados en Python el 99% de ellos utiliza dos o más librerías externas para la implementación del modelo. En cuanto a lo que se refiere a la calidad de software y tomando en cuenta los datos anteriores se llega a la conclusión que los programas desarrollados en R según las métricas de calidad seleccionadas, tienen mejor calidad que los programas en Python para la visualización de datos.

5.1.2 De los gráficos generados

En cuanto a los gráficos generados, los dos lenguajes generan gráficos muy similares como se muestra en la figura 5.2.1.1 desarrollada con R y la 5.2.1.2 que es desarrollada en Python. De igual forma para los dos lenguajes son amplias las opciones de personalización de los gráficos en los dos lenguajes, para esto la mayoría de las librerías en Python traen opciones más amplias de personalización.

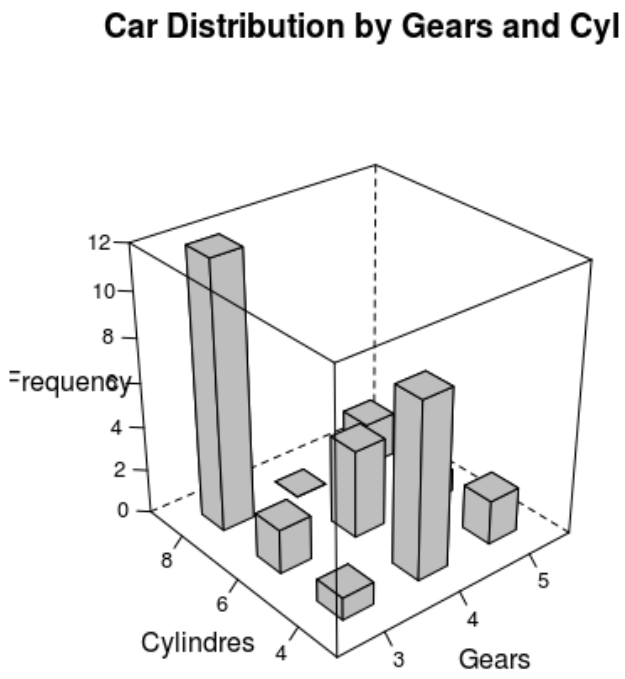

Figura 5.1.2.1Distribución de autos por cilindros desarrollados en R con Lattice

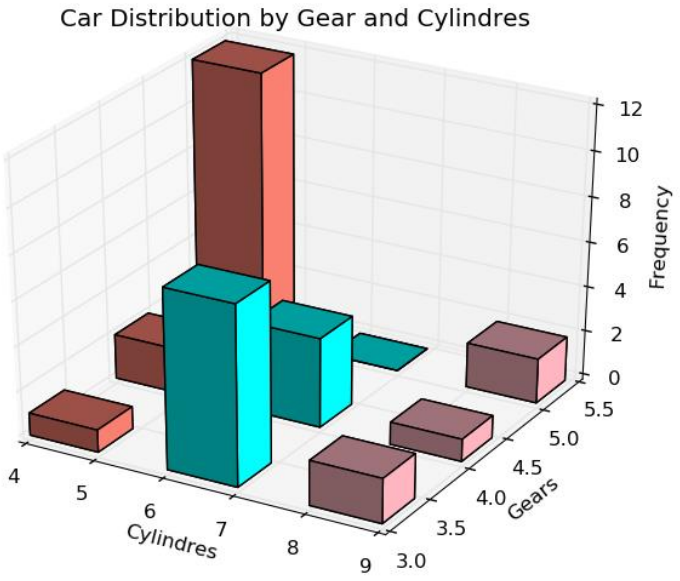

Figura 5.1.2.2 Distribución de autos por cilindros desarrollados en Python con Matplotlib

5.2 Trabajos futuros

En base a la investigación y los resultados obtenidos de este trabajo de tesis se proponen los siguientes trabajos futuros que podrían ser una aportación dentro de la ciencia de datos.

- Desarrollo de los patrones faltantes propuestos por Behrens tanto en R como en Python.
- Aportación de los nuevos patrones en la wiki, de ahí la importancia de programarlos en los dos lenguajes, si bien la programación de patrones puede resultar más sencilla en R que en Python, también Python es uno de los lenguajes más utilizados dentro de la ciencia de datos y la existencia de los patrones desarrollados en Python sería también un gran aporte.
- Desarrollo de una aplicación con información de los patrones y en donde se pueda sugerir que tipo de patrón es el mejor para utilizar al analizar cierto tipo de datos, también la aplicación podría aceptar un set de datos y en base a ellos generar gráficos adecuados.
- Programar los patrones en otros lenguajes que también están siendo utilizados en el área de la ciencia de datos.

Bibliografía y Anexos

Bibliografía

1. Behrens, Ch. (2008). *The Form of Facts and Figures Design Patterns for Interactive Information Visualization.* (Master's Thesis). Potsdam University of Applied Sciences. Potsdam.
2. Guerrero, A. L (2015). Uso de apps en México, oportunidad para pymes. Consultado 5 de Febrero del 2017 en: http://conacytprensa.mx/index.php/centros-conacyt/3943-uso-de-apps-en-mexico-oportunidad-para-pymes-estudio-nota
3. Friedman, V (2008). Data Visualization and Infographics. Consultado el 6 de Febrero del 2017 en: https://www.smashingmagazine.com/2008/01/monday-inspiration-data-visualization-and-infographics/
4. Cat, J. (2017). Epistemology , Aesthetics and Pragmatics of Scienti fi c and Other Images : Visualization , Representation and Reasoning. https://doi.org/10.1007/978-3-319-47190-7
5. Behrens, Ch. (2008). The Form of Facts and Figures Design Patterns for Interactive Information Visualization. (Master's Thesis). Potsdam University of Applied Sciences. Potsdam.
6. KDnuggets (2017). What programming/statistics languages you used for an analytics / data mining / data science work in 2014?. consultado el 7 de febrero del 2017 en: http://www.kdnuggets.com/polls/2014/languages-analytics-data-mining-data-science
7. Sthephen H. Kan. (1999), Complexity Metrics and Models ,Metrics and Models in Software Quality Engineering (pp. 4), Massachusetts, United States of America, Addison Wesley Longman, Inc.
8. Actual, S., Gobierno, E., & Gobierno, E. (2014). Tecnologías de la Información Tecnologías de la Información La Nueva Visión del Sector de las TI La Nueva Visión del Sector de las TI Compañías establecidas en México, (2), 2–3.
9. Efficiency, P. (2013). How to Deliver Resilient , Secure , Efficient , and Easily Changed IT Systems in Line with CISQ Recommendations, 1–13.
10. Saaty, T. L. (2008). Relative measurement and its generalization in decision making why pairwise comparisons are central in mathematics for the measurement

of intangible factors the analytic hierarchy/network process. *Revista de La Real Academia de Ciencias Exactas, Fisicas Y Naturales. Serie A. Matematicas*, *102*(2), 251–318. https://doi.org/10.1007/BF03191825.

11. Sthephen H. Kan. (1999), Complexity Metrics and Models ,Metrics and Models in Software Quality Engineering (pp. 254), Massachusetts, United States of America, Addison Wesley Longman, Inc.
12. Al Danial. 2006-2015. cloc. Consultado el 27 de octubre de 2016 en http://cloc.sourceforge.net/
13. R foundation 2016, Consultado el 29 de octubre de 2016 en https://www.r-project.org/
14. Boef, S. De, Ondercin, H. L., Elman, C., Collier, D., & Henry, E. (2002). *The Political Methodologist*. *The Political Methodologist* (Vol. 11).
15. Hadley Wickham (2015), What is a Package?,Ann Spencer and Marie BeauGureau (ed), R Packages organize, test, document and Share your Code(pp. 5), Sebastopol, California, United States of America, O'Reilly Media Inc.
16. Hadley Wickham (2013). Ggplot2. consultado el 27 de octubre de 2016 en http://ggplot2.org/
17. Murrell, P. (2005) R Graphics. consultado el 27 de octubre de 2016 en https://stat.ethz.ch/R-manual/R-devel/library/graphics/html/graphics-package.html
18. Graphics, T. T., Sarkar, A. D., & Sarkar, M. D. (2016). Package " lattice ."
19. Gesmann, M. (2016). Introduction to googleVis, 1–40.
20. Gaston Sanchez (2010), Introduction to the R package arcdiagram, 1,9
21. Soetaert, K., Cash, J., & Springer, F. M. (2015). Package " diagram ."
22. Package, T., Visualization, T. T., & Tennekes, A. M. (2016). Package " treemap ."

Anexos

1. Librerías seleccionadas para el lenguaje de programación R y Python: https://docs.google.com/spreadsheets/d/1xcyRa6fG5uvYYa4FQLtY-1y9JNoVtxSJSbUuE2AeCFI/edit#gid=0