Recopilación y visualización de datos de fuentes heterogéneas sobre crímenes en Montevideo

Proyecto de Grado
Ingeniería en Computación

Autores

Federico Balarini federico.balarini1995@gmail.com
Mateo Suburú suburu2894@gmail.com

Tutora

Libertad Tansini libertad@fing.edu.uy

Resumen

La inseguridad es un tema candente en el Uruguay actual. Es un tópico recurrente para la sociedad, los medios, la política, entre otros aspectos del día a día. Sin embargo, en un mundo en dónde la tecnología ocupa cada vez un lugar más importante, los uruguayos no cuentan con un sitio dónde informarse de forma fácil, rápida, amigable y confiable acerca de lo que está ocurriendo respecto a la seguridad en nuestro país. Dada esta situación, es que los medios cobran aún mayor relevancia en la percepción de la sociedad en cuánto a este aspecto, pudiendo moldear la opinión pública de así quererlo.

Considerando la realidad descripta, se reconoce la necesidad de una plataforma que haga posible a cualquier ciudadano manejar información relevante respecto a criminalidad en Uruguay, y este trabajo intenta ser un puntapié inicial en este sentido. Es así que se construyó un sistema que realiza una recopilación de datos sobre crímenes en Montevideo de fuentes diversas como El Observador, Teledoce, El País, Montevideo COMM y CityCop. Por otro lado, también se incluyen en el sistema datos oficiales, que si bien son públicos ya que su acceso es derecho de los ciudadanos, no son fácilmente consumibles o procesables.

Los datos anteriormente descriptos son procesados con el objetivo de lograr su presentación de forma unificada, amigable y de fácil acceso por cualquiera. Para ello se construyeron un mapa y un dashboard de estadísticas, de forma tal que el entendimiento de las grandes cantidades de datos manejada se haga posible. Los dos tipos de datos distintos en conjunto con una serie de filtros que se encuentran disponibles permiten, además, que el usuario haga comparativas y evaluaciones propias según sus intereses.

Índice

1. Introducción 6
   - 1.1. Motivación . . . 6
   - 1.2. Objetivos . . . 6
   - 1.3. Organización del documento . . . 7

2. Antecedentes y conceptos básicos 8
   - 2.1. Extracción de información de sitios web . . . 8
     - 2.1.1. Web Scraping . . . 8
     - 2.1.2. Reconocimiento de entidades nombradas en un texto . . . 9
   - 2.2. ElasticSearch . . . 12
   - 2.3. Sistemas de información geográfica . . . 13
     - 2.3.1. Sistemas de referencia . . . 13
     - 2.3.2. Bases de datos geográficas . . . 13
     - 2.3.3. Servidores y visualizadores de información geográfica . . . 14
   - 2.4. Sistemas relacionados . . . 15
     - 2.4.1. Datos de crímenes en Montevideo . . . 15
     - 2.4.2. Sistemas relacionados a nivel global . . . 17

3. Fuentes de datos 21
   - 3.1. Fuentes . . . 21
     - 3.1.1. Observatorio Nacional de Violencia y Criminalidad . . . 22
     - 3.1.2. Crimeometer . . . 22
     - 3.1.3. Portales Web de Noticias . . . 22
   - 3.2. Estructura de los datos . . . 23

4. Diseño e implementación 25
   - 4.1. Arquitectura . . . 25
   - 4.2. Servidor . . . 28
     - 4.2.1. Fetcher . . . 29
     - 4.2.2. Classifier . . . 37
     - 4.2.3. Service Layer . . . 42
   - 4.3. Cliente . . . 45
     - 4.3.1. Mapa . . . 47
     - 4.3.2. Estadísticas . . . 51
   - 4.4. Interfaz para administración . . . 53
   - 4.5. Infraestructura . . . 56
     - 4.5.1. Cliente . . . 56
     - 4.5.2. Servidor . . . 57

5. Validación y experimentos 59
5.1. Cantidad de artículos y crímenes en el sistema 59
5.2. Pruebas unitarias en muestreo de noticias 63
5.2.1. Reconocimiento de categorías 64
5.2.2. Reconocimiento de barrio 65
5.2.3. Reconocimiento de ubicación 67
5.2.4. Falsos negativos 69
5.3. Análisis de los datos 70

6. Gestión del proyecto 74

7. Conclusiones y trabajo futuro 79
7.1. Conclusiones 79
7.2. Trabajo futuro 80

8. Referencias 82

A. Documentación del sistema 87

B. Desgloce de pruebas unitarias 93

1. Introducción

1.1. Motivación

En los últimos años, la inseguridad en Uruguay ha sido un tema que ha estado muy presente para toda la población. Frecuentemente, se ven noticias en diarios y televisión sobre crímenes ocurridos en distintos puntos del país. Asimismo, han aparecido plataformas y aplicaciones que proveen información a la población sobre estos crímenes brindando también la posibilidad de reportarlos. Por otra parte, el Ministerio del Interior hace públicas las memorias anuales [1] para brindar transparencia sobre los datos obtenidos en seccionales y comisarías.

A partir de estas fuentes, tanto las oficiales como las no oficiales, la población tiene un gran volumen de datos y estadísticas para observar y analizar. Si bien esto ayuda a que las personas estén informadas, también tiene una desventaja; tener tantos datos, dificulta obtener una visión global de lo que ocurre a nivel país o departamental. Además, no es fácil para el usuario la interacción con las fuentes ya que implica recorrer los distintos diarios o, en el caso del Ministerio del Interior, archivos en distintos formatos.

Considerando el gran número y heterogeneidad de fuentes disponibles, poder comparar datos y estadísticas de estas resulta sumamente interesante por su amplia variedad y diferencias. Sin embargo, es una tarea muy complicada, no solo por la forma de interactuar con las fuentes, sino que también por no tener una presentación unificada de estos datos. Esto se detalla en las secciones 4.2.1 y 4.2.2 respectivamente.

1.2. Objetivos

El objetivo principal de este proyecto es diseñar e implementar un prototipo para la visualización de datos de crímenes en Montevideo. Este debe unificar los datos de las distintas fuentes además de ser amigable y fácil para los usuarios. El objetivo implica relevar las fuentes, tanto oficiales como no oficiales, candidatas a ser usadas por el prototipo por su organización, calidad y confianza.

También se pretende ofrecer la posibilidad de comparar los datos de diferentes fuentes, en particular fuentes oficiales con no oficiales e integrarlo al prototipo. El particular objetivo de esta comparación es poder analizar con mayor claridad los datos relevados y obtener estadísticas y/o patrones en ellos.

Finalmente se cuenta con dos objetivos transversales a los ya mencionados. El primero de ellos es que tanto el prototipo como la información sea de acceso público para poder dar visibilidad y transparencia a toda la población. Por otro lado, se pretende construir un corpus con información relevada de portales y otras fuentes de información relativas a criminalidad que pueda ser de valor en otras iniciativas.

1.3. Organización del documento

Este informe se encuentra dividido en 6 capítulos, incluido el actual. En el Capítulo 2 se realiza una recapitulación de algunos de los conceptos necesarios para comprender el trabajo realizado, así como también se brinda un pantallazo de algunos sistemas que presentan características similares al que se construyó. Por su parte, en el Capítulo 3 se brinda información relativa a las distintas fuentes de datos utilizadas en el proyecto. Luego, en el Capítulo 4, se presenta la solución y diseño elaborados, los detalles de cada uno de los componentes del sistema implementado y su puesta en producción. En el Capítulo 5 se realiza una evaluación de la calidad de los resultados obtenidos a través de una serie de muestreos y experimentos sobre los datos. Además, se presenta un pequeño conjunto de conclusiones que se desprenden de los mismos. En el Capítulo 6 se expone la forma en la cual se desarrolló el trabajo para lograr los objetivos propuestos. Por último, en el Capítulo 7, se comentan conclusiones generales del trabajo realizado y se proponen posibles enfoques de trabajo a futuro sobre el problema atacado y el sistema desarrollado.

Por otro lado, se tienen dos anexos que poseen información complementaria al trabajo elaborado. El Anexo A contiene información referente al sistema, incluyendo breve documentación de los servicios que expone el servidor. El Anexo B, por su parte, presenta los detalles de los experimentos realizados, expuestos en el Capítulo 5, especificando uno a uno los muestreos y evaluaciones realizadas.

2. Antecedentes y conceptos básicos

En este capítulo se presentan los conceptos teóricos más importantes para lograr una comprensión cabal de lo realizado y expuesto en este trabajo. Estos abarcan aspectos de todas las disciplinas puestas en práctica:

- Para la obtención de los datos, su análisis y extracción de la información se introducen los conceptos de Web Scraping y reconocimiento de entidades nombradas en un texto.
- Para el almacenamiento y recuperación se presentan las herramientas utilizadas más importantes: ElasticSearch y Postgis [2].
- Para la presentación de los datos se detallan los visualizadores de información geográfica, de forma de tener un contexto más amplio de los desafíos y oportunidades existentes en este aspecto.

2.1. Extracción de información de sitios web

En el proceso de obtención de información de las fuentes de noticias seleccionadas se utilizaron tanto técnicas de Web Scraping 2.1.1 como de reconocimiento de entidades nombradas 2.1.2 para la extracción. Es por ello que aquí se presentan brevemente estos conceptos, de modo que el lector tenga una referencia concreta a su respecto.

2.1.1. Web Scraping

Web Scraping es el término utilizado para agrupar diversas técnicas de recolección de información de la web, y es especialmente útil cuando la información no se encuentra accesible en otros formatos [3]; en particular cuando la forma de obtener los datos no incluye interactuar con una API (Application Programming Interface [4]) o a una persona haciéndolo de forma manual [5]. En la sección 2.4.1 se exponen las razones por las cuales es necesario aplicar Web Scraping para obtener la información pertinente para este trabajo.

Típicamente el proceso se divide en dos:

- Obtener el recurso de internet. Puede estar en formatos como HTML, XML, JSON, o incluso recursos multimedia como audio, imágenes o vídeos.
- Extraer la información deseada del recurso.

Dependiendo del tipo de recurso, hay diversas formas en las cuales se puede realizar la extracción de su contenido. El caso más típico es el análisis sintáctico del HTML, dónde utilizando el árbol de elementos y selectores CSS se realizan las búsquedas y podados del árbol, para así obtener únicamente aquellas partes del recurso que resultan de interés.

Las aplicaciones web, por su parte, pueden aplicar ciertas medidas para dificultar la extracción de la información. Algunas de ellas son: el análisis de los cabezales HTTP de los pedidos, intentando deducir si el mismo se trata de un bot o no; chequeo de la reputación de la IP origen del pedido; análisis de patrones como puede ser un ratio muy alto de pedidos; y otras herramientas como captchas, cookies, etc. [6].

2.1.2. Reconocimiento de entidades nombradas en un texto

Una vez obtenido el contenido de los recursos, es decir el documento HTML en crudo, se procede a realizar la extracción de la información del mismo. La Extracción de Información (IE por sus siglas en inglés), es la rama del procesamiento del lenguaje natural (NLP) que se encarga de ello.

Existen 3 tipos de textos de los cuales puede ser útil extraer información [7]:

- Texto estructurado: documentos altamente estructurados, como pueden ser bases de datos o archivos JSON.
- Texto semi-estructurado: texto que está presentado y formateado de una manera estructurada dentro de un contexto específico. Por ejemplo datos económicos o relacionados a la medicina.
- Texto libre: texto sin una estructura definida, como pueden ser noticias, cuentos, etc. Es el tipo de texto del cual resulta más compleja la extracción de información.

Dentro del área de la IE, existe lo que se conoce como reconocimiento de entidades nombradas (NER). Por entidades nombradas se entienden ubicaciones, lugares, personas, países, categorías, compañía, o cualquier objeto particular presente en un texto que se quiera identificar e individualizar. Esta tarea consta de 2 fases [7]. Una primera fase donde se identifican todas las entidades nombradas del texto, y una segunda en la cual se agrupan y categorizan dichas entidades.

Los sistemas que realizan este tipo de tareas pueden hacerlo utilizando métodos variados: Basados en diccionario y/o léxico, Machine Learning, o

Basados en reglas. En particular, los sistemas que utilizan métodos de Machine Learning se subdividen según la técnica utilizada. Los mismos pueden ser de aprendizaje supervisado, semi-supervisado, no supervisado, o híbrido. La Figura 1 ilustra dicha categorización.

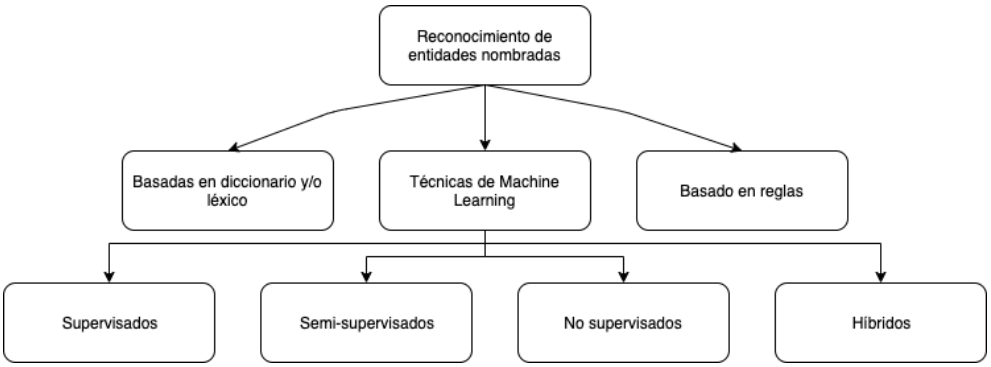

Figura 1: *Categorías sistemas NER*

Técnicas basadas en diccionario y/o léxico

Los enfoques basados en el léxico utilizan sets preconstruidos con fuentes de conocimiento externas al texto a analizar. Se intenta lograr una correspondencia entre los sets y el texto a analizar, para así lograr la identificación y clasificación de las entidades.

Técnicas basadas en reglas

Estas técnicas construyen reglas con las cuales se realiza la detección de las entidades nombradas. Las reglas pueden construirse de forma manual o automática. Este tipo de enfoque, haciendo uso de reglas construidas manualmente, fue el más utilizado en el pasado, pero en la actualidad han cobrado mayor popularidad las técnicas de la siguiente categoría.

Métodos de Machine Learning

Este tipo de enfoque es el más utilizado por los sistemas más modernos. En general, requieren un set de datos de entrenamiento muy grande para lograr buenos resultados, que constan de textos anotados de forma que los parámetros que se utilizan puedan ajustarse lo mejor posible.

Aquí se engloban varios enfoques [8]:

- Hidden Markov Models (HMMs)
- Conditional Random Fields (CRFs)

- Support Vector Machines (SVMs)
- Modelos de máxima entropía

No es el objetivo de este trabajo ahondar en cada una de estas técnicas, pero cabe mencionarlas ya que, como se ve en la sección 7, son posibles mejoras al sistema.

Herramientas existentes

Existen diversas herramientas para realizar la extracción de entidades con nombre de un texto:

- Spacy [9]
- NLTK [10]
- MonkeyLearn [11]

Algunas de ellas pueden ser utilizadas en el contexto de un proyecto Python como Spacy y NLTK, y otros son servicios en la nube que se pueden consumir sin importar el ambiente del sistema.

Caso particular: reconocimiento de direcciones

Dentro del desafío de la extracción de información, y particularmente de entidades nombradas, se encuentra el reconocimiento de direcciones en un texto no estructurado, que presenta un conjunto de características que complejizan la tarea.

En primer lugar, las direcciones constan de distintos atributos como pueden ser calle, número de puerta, esquinas, barrio, ciudad, país, etc. En la Figura 2 se puede ver el formato de una dirección uruguaya completa, en idioma español. La ventaja de tener este formato de dirección es que se puede explotar para reconocer y obtener la información de una forma más precisa. Por otro lado, para identificar un lugar es necesario lograr reconocer múltiples atributos para que lo obtenido tenga sentido.

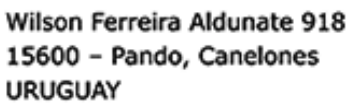

Figura 2: *Formato dirección en idioma español*

Al tratarse de texto "libre", no siempre se tiene una dirección completa como la de la Figura 2 ya que la misma depende del contexto en que el texto estaba

destinado a ser publicado. En el caso concreto de nuestro trabajo, al tratar con noticias policiales en medios formales, cuando hay información relativa a la dirección de un hecho no se la da en un formato fijo como el anterior. Además se pueden indicar direcciones con referencias a puntos de interés de la ciudad en cuestión. Dado el contexto, lo más usual es encontrarse con los nombres de las calles y el barrio, por lo que se atacaron específicamente estos dos atributos.

En el artículo *Extraction of Address Data from Unstructured Text using Free Knowledge Resources* [12] se estudia el problema general de las direcciones, y también particularmente se plantean formas de atacar el problema de la identificación del nombre de calles. Los nombres de las calles no siguen ningún patrón común, y los mismos pueden ser nombres de personas o países, fechas, o incluso formar una numeración dentro de la localidad a la cual pertenece la calle. La solución parcial al problema utilizada para este trabajo consta de un diccionario de calles y utiliza el contexto de las palabras en el texto. En el Capítulo 4 se planta con mayor detalle.

2.2. ElasticSearch

Es un motor de búsqueda y análisis Open Source basado en arquitectura REST(REpresentational State Transfer) [13][14]. Es de uso libre y gratuito, además de compatible con los sistemas operativos más populares. Posiblemente su característica más diferencial sea que es muy escalable y funciona muy bien de forma distribuida, aportando una gran performance. Estas características, sumado a ser un proyecto Open Source, lo convierten en el motor de búsqueda más utilizado en la actualidad.

ElasticSearch define un lenguaje llamado Query DSL [15], utilizado para consultar y operar con el motor de búsqueda mediante una API. Dicho lenguaje está fuertemente basado en JSON (JavaScript Object Notation), lo que lo hace simple de utilizar.

Asimismo, los datos son almacenados en un formato de la forma de JSON, dónde los documentos constan de pares clave-valor. Los objetos se insertan al motor de búsqueda mediante una operación POST con el objeto a insertar, mientras que para recuperarlos se realiza una operación GET; aspecto que denota estar basado en la arquitectura anteriormente mencionada.

En el contexto de este trabajo se manejan grandes cantidades de datos, por lo que ElasticSearch resultó de gran utilidad para realizar búsquedas sin caer en problemas de performance. En el Capítulo 4 se estudia de qué manera

se utilizó la herramienta y qué posibilidades concretas brindó con mayor detalle.

2.3. Sistemas de información geográfica

El sistema construido como parte de este trabajo se categoriza como sistemas de información geográfica, ya que manipula, almacena y presenta este tipo de información. Por ello, resulta importante indagar en algunos tópicos relacionados a este tipo de sistema, que estuvieron presentes durante todo el proceso llevado a cabo y que en el Capítulo 4 se analiza de qué forma participan.

2.3.1. Sistemas de referencia

Un sistema de referencia es un sistema mediante el cual se puede ubicar geográficamente una entidad. Cada sistema de referencia define una proyección distinta de las coordenadas en una esfera (o elipsoide como la Tierra) sobre un plano o mapa, y se identifica mediante un número denominado Spatial Reference System Identifier (SRID). Estos números son de interés al trabajar con múltiples entidades geográficas en un mapa, como pueden ser polígonos y puntos, para lograr un resultado integrado que sea consistente. [16][17]

Para la realización de este trabajo se utilizó WGS 84 [18], un sistema de referencia que utiliza la Tierra como centro, considerando su forma, tamaño, y otras medidas de interés, para lograr una localización muy precisa de las entidades. Fue elaborado por la National Geospatial-Intelligence Agency de Estados Unidos en el año 1984, y es el sistema utilizado en el Global Positioning System (GPS).

2.3.2. Bases de datos geográficas

Una base de datos geográfica es una base de datos que permite que se guarden y consulten objetos espaciales en ella. El Simple Feature Standard (SFS) [19] es un estándar ISO que especifica un modelo mediante el cual se pueden representar geometrías en sistemas de información geográfica. Los objetos espaciales son entidades del mundo real representadas, para lo cual es necesario definir una tipología con la que se puedan construir dichas entidades [20]. Para ello se definen los siguientes objetos espaciales elementales en el SFS:

- Punto: Es el tipo más básico, ya que a partir de él se construyen los

otros. Es representado por dos o tres atributos, a partir de los cuales se construyen las coordenadas del punto en 2 o 3 dimensiones respectivamente.

- Línea: Es un set ordenado de puntos (nodos). A su vez, una línea puede ser un conjunto de múltiples líneas rectas distintas, que juntas por ejemplo forman un río, calle, o línea de ómnibus. La división de qué forma parte de la misma entidad y qué no es arbitraria.
- Polígono: Es un set ordenado y cerrado de líneas, donde el primer y último punto de la primer y última línea respectivamente deben ser el mismo.

Cabe mencionar que de los anteriores se derivan a su vez otros objetos.

Es particularmente interesante analizar las relaciones entre las distintas entidades, para lo cual se definen operaciones en el SFS que permiten operar con los objetos. En el contexto de este trabajo, por ejemplo, es de interés saber qué delitos (punto) fueron cometidos dentro de cada uno de los barrios (polígono).

Postgis [2] es una extensión para PostgreSQL [21], y es un ejemplo de base de datos que conforma con el estándar SFS. Se eligió como la base de datos a utilizar en este proyecto por ser de uso libre, con integración muy sencilla en el servicio en la nube en el que despliega el sistema, y con la cual se cuenta con experiencia previa. En el Capítulo 4 se ve de qué manera se utiliza.

2.3.3. Servidores y visualizadores de información geográfica

Los servidores de información geográfica son los responsables de proveer información a los visualizadores. Los mismos pueden implementar una serie de estándares distintos, según los cuales brindan distintas posibilidades al cliente que los utilice y retornan información geográfica en distintos formatos. A modo informativo, se mencionan algunos ejemplos de servidores:

- WMS (Web Map Service) [22]
- WMTS (Web Map Tile Service) [23]
- WFS (Web Feature Service) [24]
- CSW (Catalog Service Web) [25]
- WCS (Web Coverage Service) [26]

Algunos de los servicios de mapas más populares y potentes de la actualidad son Google Maps [27] y Mapbox [28], cada uno de ellos brindando múltiples servicios.

Por su parte, los visualizadores son la capa que se encarga de la interacción con el cliente, permitiendo realizar consultas geográficas sobre la información geográfica. Algunos ejemplos de visualizadores son:

- Google Maps
- Bing Maps [29]
- Open Street Map [30]

2.4. Sistemas relacionados

2.4.1. Datos de crímenes en Montevideo

Existen diversas formas de informarse acerca de la criminalidad en Montevideo, pero pocas de ellas están pensadas de forma tal que los datos sean fácilmente accesibles por un sistema informático. Además interesa la posibilidad de las plataformas de brindar una visión global, con la posibilidad de estudiar distintos segmentos dentro de un fenómeno complejo como este. A continuación se introducen algunas de ellas.

Observatorio Fundapro [31]

Es una organización del Partido Colorado que maneja cifras de delitos propias, y que no necesariamente coinciden con los datos oficiales. Si bien el Observatorio Fundapro está activo hoy en día, su portal tiene datos hasta el año 2012. Brinda a los usuarios la posibilidad de realizar reportes de delitos por la página, y un mapa con datos de distintos tipos de crímenes obtenidos de reportes y noticias de la época.

Figura 3: *Portal Observatorio Fundapro*

CityCop [32] y Crimeometer [33]

CityCop es una reconocida aplicación móvil para reporte y consulta de delitos. Su aplicación web ofrece además un mapa con los reportes de usuarios en distintas ciudades.

Recientemente el equipo de CityCop desarrolló una nueva plataforma llamada Crimeometer, que ofrece una API con los datos recabados por CityCop y además, para algunas ciudades, datos oficiales. Lamentablemente los datos oficiales no están disponibles en Montevideo, ya que no se cuenta con una infraestructura por parte del Estado que lo haga posible.

En la realización de este trabajo existió un contacto directo y prolongado con el equipo de Crimeometer, a quienes desde aquí se agradece, por proporcionar ideas y compartir experiencias previas propias en proyectos similares. Con su ayuda, y como se ve más en detalle en el Capítulo 4, se realizó la integración de los reportes de CityCop en el sistema desarrollado como una fuente alternativa. Además, en una reunión que tomó lugar en sus oficinas, se recomendaron algunas tecnologías que no habían sido

consideradas a priori y resultaron de gran utilidad.

Noticias

Como es de público conocimiento, es el medio por el cual la población general se informa de todos los temas relacionados a los delitos. Brindan información detallada de una selección limitada de hechos. Si bien existen múltiples portales informativos en los cuales se publican noticias de este tipo, se eligieron Teledoce [34], El Observador [35], El País [36] y Montevideo COMM [37] por tener una sección dedicada a asuntos policiales, facilitando en gran medida la realización de la investigación. De no existir la categorización previa, habría un gran trabajo de clasificación a realizar por el sistema que podría constituir un trabajo en sí mismo, y por lo tanto fue dejado por fuera del alcance. Además los portales anteriormente mencionados cuentan con renombre y buena reputación, de forma que pueden ser tomados como fuentes de noticias confiables.

Observatorio Nacional de Violencia y Criminalidad [38]

Pertenece a la División de Estadísticas y Análisis Estratégico del Ministerio del Interior, y es el medio oficial por donde se publican datos y estadísticas referentes a la criminalidad a nivel nacional. En particular, ofrece un reporte anual y comparativo con el año anterior de Hurtos, Rapiñas y Homicidios.

Además, plantea que uno de los objetivos es la modernización de los sistemas de información referentes a este tipo de estadísticas, pero desafortunadamente no se logró acceder a ninguna herramienta que haga posible una integración a nivel de sistemas con los datos oficiales. Otra forma de obtener datos oficiales es por medio de un Pedido de Información Pública, pero tampoco existió mayor respuesta al solicitar información por este medio.

2.4.2. Sistemas relacionados a nivel global

A nivel global hay múltiples sistemas que brindan detalles de información oficial en sistemas de información que resultan amigables para la población. Una gran ventaja que tienen estos sistemas es que los datos se actualizan permanentemente. En general, cada uno de ellos sirvió como inspiración para distintos aspectos en la implementación del sistema desarrollado como parte de este trabajo. Algunos ejemplos de este tipo de sistema son:

- CrimeMapping [39]: sacando provecho de la posibilidad que brinda el gobierno norteamericano, este sistema muestra los datos directamente de las distintas agencias policiales de los Estados Unidos. Presenta un

mapa amigable con la posibilidad de utilizar diversos filtros, y la posibilidad de generar un reporte de los datos.

- ADT Crime Map [40]: también en Estados Unidos, esta herramienta utiliza los mismos datos pero construye con ellos un mapa de calor.
- Crime UK [41]: ofrece información de datos oficiales en la ciudad de Londres. Su interfaz es muy amigable, y brinda la posibilidad de filtrar según la zona del mapa que se desee.

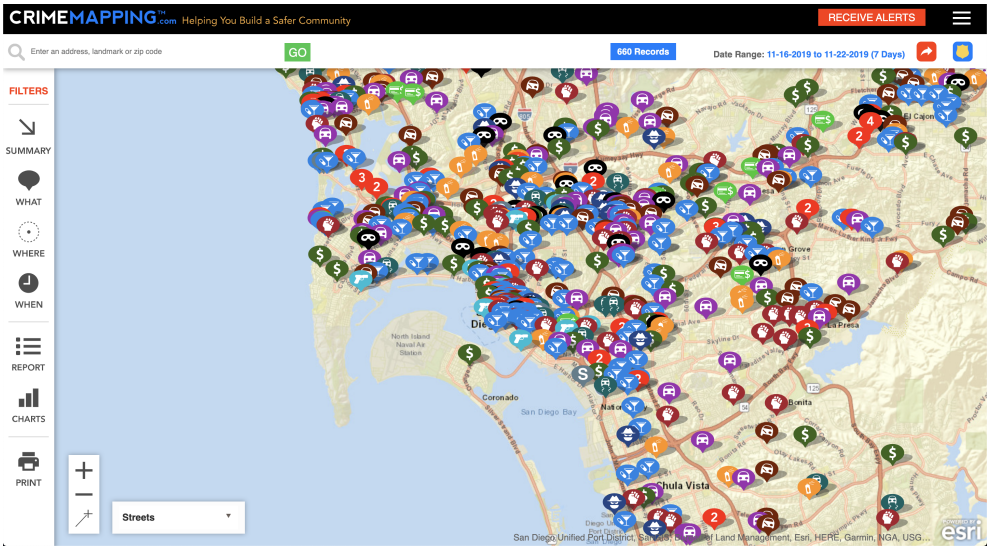

Figura 4: *CrimeMapping*

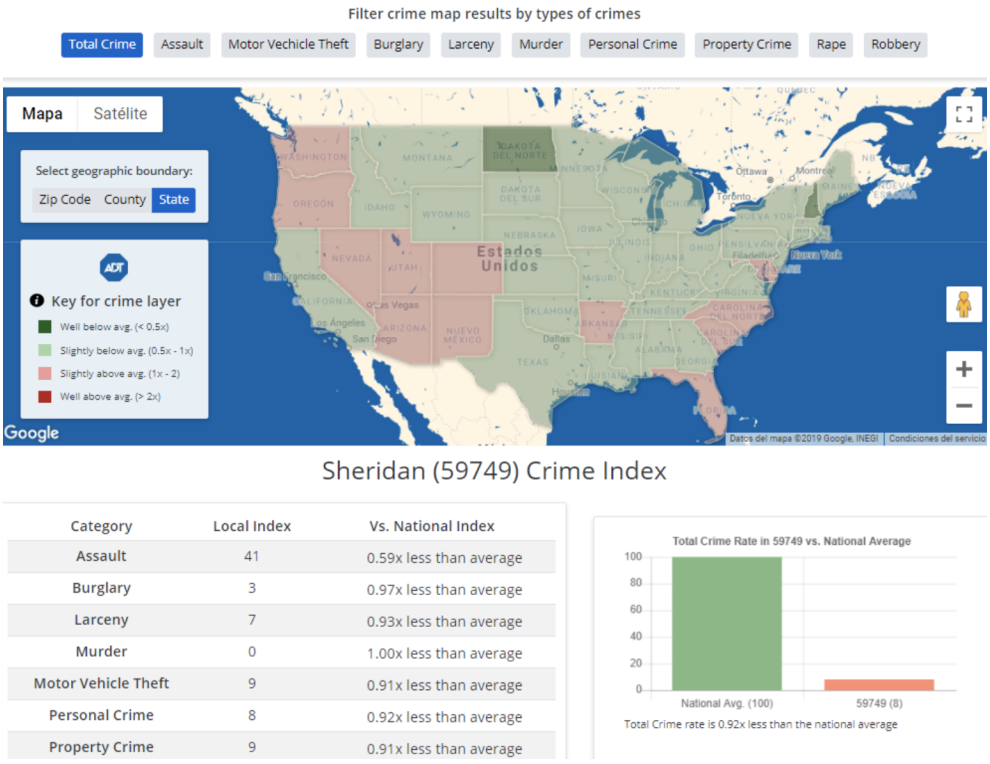

Figura 5: *ADT Crime Map*

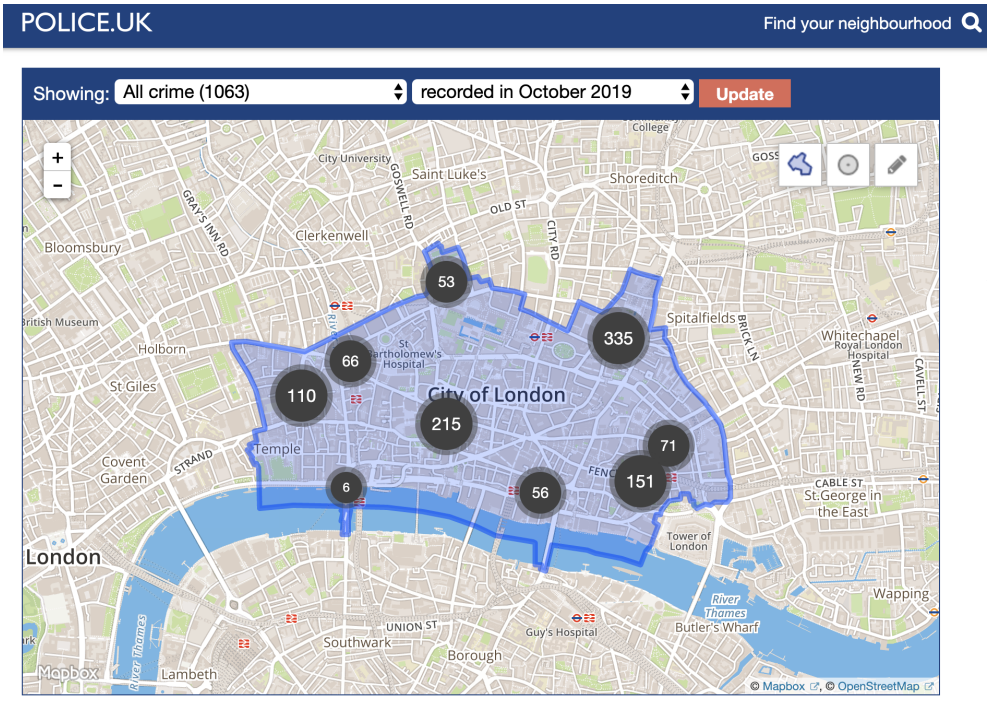

Figura 6: *Crime UK*

3. Fuentes de datos

Para comprender de qué manera se obtienen y procesan los datos que se muestran en el sistema, es de suma importancia conocer las diversas fuentes de datos utilizadas. En la sección 2.4.1 se hace una primera mención a las mismas, por lo que en el presente capítulo se recapitula y se ahonda en algunas características de las fuentes que son relevantes en el contexto del proyecto.

Por otro lado, se presentan las estructuras de datos más importantes del sistema: los *artículos* y los *crímenes*, y se explica qué representan en función de la información obtenida de las fuentes.

3.1. Fuentes

Como se observa en la Figura 7, se hace una primera clasificación de las fuentes entre aquellas consideradas oficiales, y aquellas que provienen de otro tipo de información no oficial.

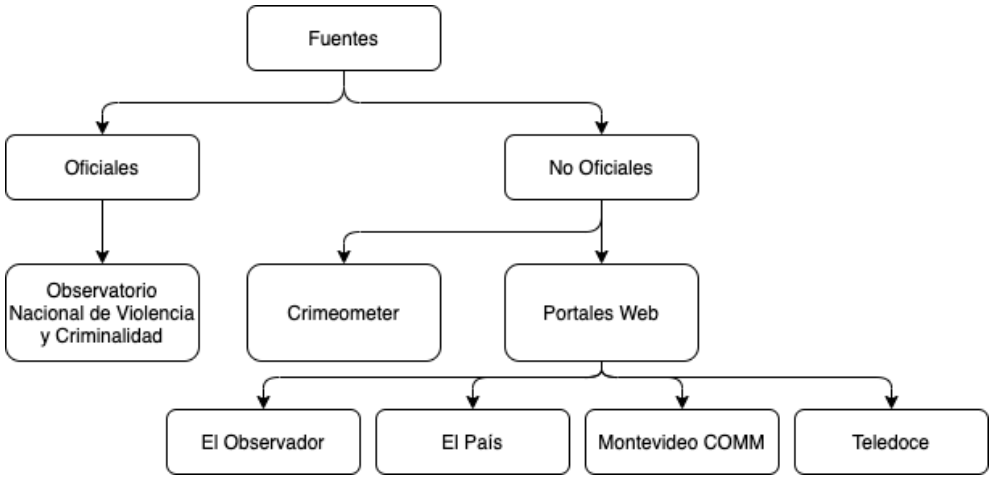

Figura 7: *Clasificación de fuentes*

Dicha clasificación es utilizada para permitir comparaciones entre las dos categorías de datos presentadas. De esta forma es posible estudiar aspectos como el impacto relativo de los distintos tipos de crímenes en las noticias, y cómo las características de los mismos influyen en su difusión.

3.1.1. Observatorio Nacional de Violencia y Criminalidad

La fuente oficial utilizada es el Observatorio Nacional de Violencia y Criminalidad, que como se explica en la sección 2.4.1 pertenece al Ministerio del Interior. Ofrece reportes anuales de hurtos, rapiñas y homicidios en formato PDF, lo que implica que para extraer los datos de los mismos es necesario implementar algún proceso de extracción, transformación y carga (ETL). En la sección 4.2.1 se presentan detalles sobre estos reportes y el proceso de extracción implementado.

Por otro lado, el Ministerio del Interior cuenta con un portal de noticias que puede ser utilizada como fuente de información oficial [42]. Sin embargo, estas noticias son generalmente enfocadas a actualizaciones o resoluciones de casos anteriores, y no tratan necesariamente sobre los crímenes en sí mismos, por lo que no fueron utilizadas como fuente de datos en este proyecto.

3.1.2. Crimeometer

Dentro de las fuentes no oficiales, la fuente de datos más predominante del sistema es Crimeometer. Estos datos tienen origen en reportes realizados de forma manual por usuarios de la aplicación CityCop, para los cuales detallan un conjunto de datos de valor para el análisis que se realiza en este proyecto.

Crimeometer es un conjunto de puntos de acceso construido para exponer distintos tipos de datos relativos a crímenes, donde una de las posibilidades es consultar dichos reportes originados en CityCop. Además, por medio de integraciones con seccionales o entidades estatales, brindan acceso a datos oficiales en algunas ciudades, principalmente de Estados Unidos. Esta opción no se encuentra disponible en Uruguay.

3.1.3. Portales Web de Noticias

Los portales de noticias son la segunda y última fuente de datos no oficiales utilizada. En particular, el sistema utiliza datos provenientes de los portales web de los siguientes diarios:

- El Observador
- El País
- Montevideo COMM
- Teledoce

Al momento de la elección de estos portales frente a otros se consideraron múltiples aspectos. Una razón fundamental es que se valora que brindan una sección dedicada únicamente a noticias de asuntos policiales, lo cual simplificó la labor de extraer los artículos que interesaban para este sistema. Asimismo, si bien muchas de ellas publican novedades de Uruguay y todo el mundo, tienen un gran porcentaje de noticias de Montevideo, el departamento en el cual se centra este proyecto.

Otra característica que se valoró al momento de seleccionar los portales web fue el historial de noticias disponible en cada uno de ellos. Considerando el tiempo acotado de elaboración del proyecto, se buscó contar con el mayor conjunto de datos anteriores a la implementación y extracción, ya que es posible acumular los datos a partir del despliegue del sistema en adelante.

3.2. Estructura de los datos

Conocidas las fuentes, se presenta ahora el diseño de los datos. En este proyecto se maneja información que es obtenida directamente de cada una de las fuentes, y también información que es fruto del procesamiento de la anterior. Es así que al primer tipo de información sin procesar se le denomina *artículo* en el contexto de este trabajo, mientras que una vez procesada dicha información se obtiene lo que se denomina *crimen* que es lo que se muestra en última instancia a los usuarios finales.

Los artículos cuentan con los siguientes campos:

- Fuente: cada artículo guarda una referencia a la fuente de la cual fue obtenido.
- URL: la url de la cuál se obtuvo el artículo, de forma de poder consultar el origen en caso de requerirlo. Para la información obtenida de Crimeometer se guarda un número identificador del incidente, mientras que para los datos oficiales se deja vacío.
- Título: el título de la noticia, si corresponde.
- Texto: el cuerpo de la noticia, si corresponde.
- Raw: campo que contiene el artículo en el formato de origen para la fuente correspondiente, cómo por ejemplo HTML.
- Fecha de creación: marca de tiempo en que se creó el artículo en el sistema.

- Fecha de publicación: marca de tiempo correspondiente a la fecha de publicación de la noticia en la fuente de origen.

Por otro lado, luego de procesar la información anterior, se obtienen los crímenes. Este modelo fue diseñado pensando en qué datos es posible obtener de la información relevada de cada artículo. Esto es, no intentar identificar un dato para un crimen si ese dato no se puede extraer de la información que se tiene en el artículo correspondiente. Además se consideró qué datos son de interés para la población y que debían poder ser visualizados. Es así que los crímenes consisten de la siguiente información:

- Artículo: referencia al artículo a partir del cual se obtuvo el crimen.
- Categorías: referencia a las categorías a las cuales se asocia el crimen. En el contexto de este proyecto las categorías consideradas fueron 9 e incluyen Hurto, Rapiña, Homicidio, Violación, Violencia Doméstica, Femicidio, Secuestro, Drogas y Accidente. Un crimen puede tener más de una categoría asociada. El conjunto de categorías posibles fue definido manualmente.
- Calles: las calles en las cuales ocurrió el hecho, que se utilizan a su vez para obtener la ubicación del mismo.
- Ubicación: el punto en el cual sucedió el crimen.
- Barrio: referencia al barrio en el cual sucedió el crimen. El conjunto de barrios viene dado por la capa de barrios pública de la IMM. Haciendo uso de la misma, se persisten con el nombre y el polígono que lo define.
- Fecha: marca de tiempo en la que ocurrió el crimen.

Guardar la noticia permite procesarlas más de una vez, de forma que es sencillo realizar cambios al módulo de clasificación detallado en 4.2.2 y obtener un nuevo set de crímenes ajustados a los parámetros actualizados.

4. Diseño e implementación

En esta sección se expone todo lo relativo al diseño e implementación del sistema propuesto. En primer lugar, se explica la arquitectura del mismo, presentada en la sección 4.1, para luego entrar en detalle en cada uno de los componentes que se desprenden de esa arquitectura, esto sucede en las siguientes secciones (4.3, 4.2 y 4.4).

4.1. Arquitectura

La arquitectura utilizada para este sistema fue Cliente/Servidor [43]. En una arquitectura de este tipo, hay dos capas lógicas claramente definidas: el cliente y el servidor.

El servidor es el encargado de proveer respuestas a las distintas solicitudes realizadas por los clientes. También, está a cargo de la gestión de los recursos, mientras que el cliente, provee una interfaz al usuario para que este pueda interactuar con el sistema.

Los clientes son el componente lógico activo ya que son los que realizan las solicitudes al servidor, el cual se considera pasivo dado que espera esas peticiones para realizar alguna acción.

Esta arquitectura permite contar con un ambiente heterogéneo en cuanto a hardware y software. Tanto el sistema operativo como el hardware elegido para el servidor no tiene que ser necesariamente el mismo que el elegido para los clientes.

Uno de los ejemplos mas importantes es el de la World Wide Web [44] en donde se tienen servidores web los cuales atienden a miles de millones de clientes web.

La motivación para utilizar este tipo de arquitectura viene dada por las distintas ventajas que presenta, las cuales son beneficiosas para la implementación de este caso de estudio.

A continuación se listan las ventajas más relevantes:

- Centralización: al ser el servidor el que gestiona los recursos de la aplicación, se tiene total control sobre los mismos de manera de no permitir que un cliente no autorizado pueda acceder a ellos.
- Escalabilidad horizontal: implica añadir más nodos al sistema (clientes y/o servidores). En el caso de añadir mas servidores, se potencia el

rendimiento del sistema aunque se hace cada vez más complicada la gestión del mismo.

- Escalabilidad vertical: cualquier nodo del sistema puede crecer y ser más potente por separado.
- Modularidad: facilita las modificaciones de nodos sin que el sistema se vea afectado.

Para este caso de estudio, las ventajas que brinda el modelo elegido son esenciales para el sistema construido. A su vez, como todo modelo de arquitectura, este también presenta las siguientes desventajas:

- Congestión del tráfico: al tener muchos clientes utilizando el mismo servidor, es posible que este último se vea afectado así como también la performance de todo el sistema.
- Tolerancia a fallas: si un servidor falla, las peticiones realizadas al mismo no van a poder ser atendidas.
- Costo elevado: de lado del servidor el costo es elevado ya que requiere de mucho hardware para que este sea potente. Además, el costo de mantenimiento del mismo también lo es.

Dentro de la arquitectura Cliente/Servidor hay dos formas posibles de modelar el sistema en cuanto a la cantidad de capas físicas: arquitectura de dos o tres capas. La gran diferencia radica en si el servidor es capaz de responder a la solicitud con los recursos y datos que dispone localmente o necesita de otro para poder hacerlo.

En este caso, se optó por una arquitectura de tres capas físicas definidas de la siguiente manera:

- Cliente
- Servidor
- DBMS y Base de datos

Por último, se utilizó uno de los modelos de arquitectura en tres capas físicas definidos por Gartner Group [45]. Esta es una empresa líder en investigación y consultoría de alcance mundial. Fundada en 1979, provee a las organizaciones con herramientas y métodos que ayudan a que puedan cumplir su misión como organización.

En la siguiente figura se pueden observar los modelos de arquitectura en tres capas físicas propuestos por Gartner Group:

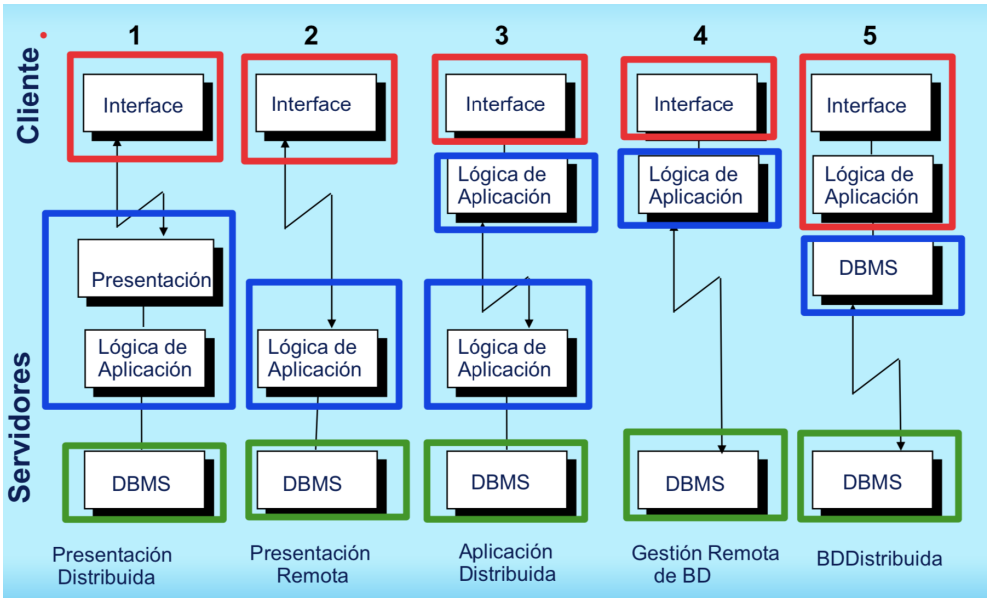

Figura 8: *Modelos de arquitectura en tres capas del Gartner Group*

Como se puede observar, estos modelos de arquitectura definen la responsabilidad y labor de las capas cliente y servidor. Se basan en la idea de que hay tres claras funciones que tiene que cumplir una aplicación: presentarle al usuario los datos solicitados, manejar la lógica de la misma y administrar los datos.

Se optó por el modelo número tres ya que es el que mejor se adapta a las necesidades y requerimientos de este sistema. En este modelo, la lógica de la aplicación se divide en dos partes. El cliente no solo se encarga de la interfaz con el usuario sino que también de parte de la lógica de la aplicación. En cuanto al servidor, este se encarga de la comunicación con el DBMS para leer y modificar datos de la base de datos. Además, de parte de la lógica de la aplicación como por ejemplo, todo lo que respecta a autenticación y autorización de usuarios.

En la siguiente figura se puede observar la arquitectura del sistema:

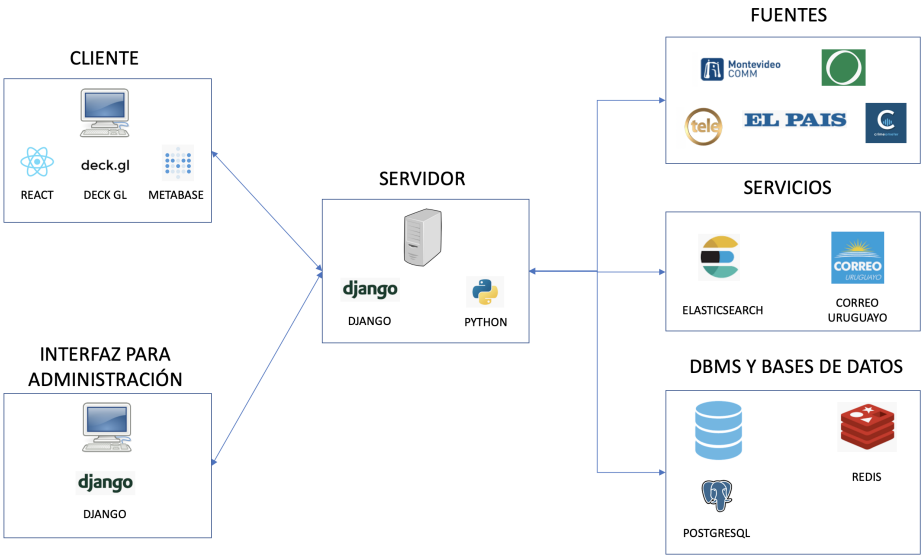

Figura 9: *Arquitectura del sistema*

4.2. Servidor

El servidor se implementó utilizando Django [46], un framework para el desarrollo de aplicaciones web que utiliza Python [47] como lenguaje de programación. Django permite a los desarrolladores avanzar rápidamente en el desarrollo de la aplicación no solo por la forma en que está diseñado sino también porque, permite incluir paquetes y de esta forma, solucionar problemas que se dan comúnmente como pueden ser: autenticación, administración del contenido, site maps, entre otros.

En cuanto a la estructura de este proyecto, se crean distintas aplicaciones para los diferentes componentes que presenta este sistema. Esta es una gran ventaja en la forma de Django en cuanto a como se estructuran los proyectos, ya que se tiene una aplicación distinta para cada componente. Muchas veces a estas aplicaciones se las ve como paquetes con responsabilidades disjuntas los cuales se unen para formar una única aplicación web. En este caso, se crean tres módulos independientes:

- *Fetcher:* responsable de obtener los datos de las distintas fuentes seleccionadas, manejando las particularidades de cada una de ellas.
- *Classifier:* se encarga de procesar y clasificar los datos obtenidos en el módulo anterior, siguiendo una serie de pasos y reglas definidas.
- *Service Layer:* por último, se tiene una API REST que provee los

distintos servicios que son utilizados por los clientes web.

A continuación se detalla cada uno de estos componentes.

4.2.1. Fetcher

Tomando en consideración los datos detallados en el Capítulo 3, se construyó un módulo cuyo objetivo es obtener la información correspondiente para su posterior procesamiento. Al momento de extraer dicha información cada una de las fuentes tiene particularidades propias que se presentan a continuación.

Extracción de datos de portales web

Para extraer los datos de las noticias provenientes de portales web se hizo uso de técnicas de Web Scraping. Fue necesario el uso de esta técnica debido a que es la única forma de automatizar el proceso, dado que los distintas portales web no proveen una API de donde poder obtener los datos.

Esta técnica tiene dos pasos bien definidos:

1. Obtener el artículo, en formato HTML, de la fuente.
2. Extraer la información deseada del artículo.

Para el primer paso, es necesario contar con un cliente HTTP desde el lado de la aplicación. Un cliente de este tipo permite realizar consultas HTTP a un servidor y obtener respuesta del mismo. En este caso se uso Requests [48], un cliente HTTP construido especialmente para Python. Tiene la particular característica de ser sumamente simple de utilizar ya que no hay necesidad de agregar manualmente los parámetros de la consulta en la URL, ni tampoco de codificar el formulario en caso de ser de tipo POST.

En lo que respecta al segundo paso, es imprescindible hacer uso de una herramienta que permita manipular el HTML obtenido por el cliente HTTP. En este caso, se uso la herramienta llamada Beautiful Soup [49]. Esta es una librería de Python que permite extraer datos de un archivo HTML o XML haciendo uso del árbol de elementos y selectores CSS para buscar elementos específicos que sean de interés.

Por último, luego de obtener los datos que se creen pertinentes de un artículo, se persiste el artículo en la base de datos del sistema realizando previamente ciertas validaciones.

De esta manera, quedan definidos los tres pasos que se ejecutan para obtener los datos de estas fuentes: los dos pasos definidos por la técnica de

scraping y la persistencia de esos datos con sus validaciones previas respectivas. Estos tres pasos son los mismos para todos los portales, excepto por ciertas diferencias. Una de ellas radica en cuál es la URL utilizada por el cliente HTTP para obtener los artículos de cada una de ellas. Otra radica en la forma de extraer los datos del HTML dado que cada fuente define su propio árbol de elementos y esto implica utilizar diferentes selectores.

En un principio se desarrolló una clase para cada uno de los portales web, pero esto implicaba tener código duplicado dado que la serie de pasos es la misma para todos. Por esta razón, se hizo uso del patrón de diseño Template Method [50]. Este sugiere definir un algoritmo (serie de pasos) en donde cada paso corresponde a un método en una súper clase, siendo estos abstractos para ser definidos en las distintas subclases. Por lo tanto, la súper clase tiene la implementación común a todas las posibles variantes del algoritmo mientras que cada subclase define los pasos que son específicos a cada tipo.

En este caso, existe una súper clase llamada BaseFetcher y una subclase para cada portal. Cada subclase define su propia URL a donde hacer la consulta para obtener los artículos en la web y los distintos selectores para extraer los datos pertinentes del HTML del artículo, como el título y la fecha, entre otros.

Este patrón no solo hace el código más mantenible sino que también simplifica la tarea de agregar nuevas fuentes, siendo únicamente necesario definir los métodos abstractos y no todo el algoritmo. En la siguiente figura se puede observar un diagrama que muestra la estructura mencionada:

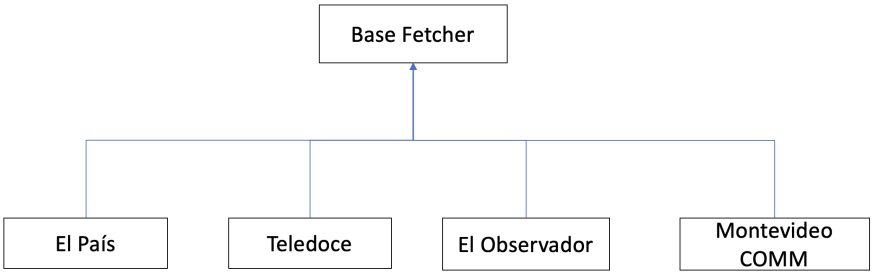

Figura 10: *Estructura del patrón de diseño, Template Method, aplicado en este sistema*

Como ya se mencionó previamente, en una visión macro, se tienen los siguientes tres pasos: obtener el HTML de los artículos, extraer los datos

pertinentes y persistir el artículo. Se desarrolla a continuación cómo cada uno de ellos está implementado en la súper clase.

Los portales web cuentan en su sección de asuntos policiales con un listado de noticias, pero para obtener los detalles de las mismas es necesario ingresar a cada noticia en particular para lo cual se provee un link. Por lo tanto, para obtener el artículo en su totalidad, primeramente se hace uso del Web Scraping para obtener la URL de todos los artículos a partir del listado. Además, como cada sección está paginada [51], se paralelizan las consultas para cada página haciendo uso de una librería de Python llamada Asynchronous I/O [52].

Luego de obtenidas todas las URLs de los artículos, y haciendo uso nuevamente de la misma librería, se obtienen los artículos con toda su información. Para cada uno de ellos se obtiene el HTML completo y luego, utilizando los distintos selectores definidos, se extrae la información pertinente detallada en el Capítulo 3. El paso restante es entonces la persistencia en la base de datos del sistema.

Extracción de datos de Crimeometer

Si bien Crimeometer es una fuente no oficial al igual que los portales web, la forma en que se extraen los artículos es totalmente diferente. Se utiliza una API privada a la cuál el equipo de Crimeometer nos brindó acceso en su primer versión, que es la única versión construida hasta el momento. Esta cuenta con tres puntos de acceso:

- Raw Data: en este se usa el FBI-NIBRS [53], un estándar implementado por el FBI para mejorar la calidad de los datos recabados, para brindar información acerca de los incidentes de los Estados Unidos.
- Stats: provee información estadística de los incidentes.
- Crowdsourced Raw Data: permite acceder a una gran cantidad de datos provenientes de reportes de incidentes en todo el mundo. La información es conseguida de la aplicación de CityCop.

En este sistema se hace uso únicamente del punto de acceso Crowdsourced Raw Data, ya que provee información de todo el mundo y por lo tanto también de Montevideo.

En un comienzo, se definen ciertos parámetros que la API requiere para poder ser utilizada. Estos parámetros son latitud, longitud y distancia que son usados para filtrar la zona geográfica de dónde se quieren datos. En este caso, se ingresan coordenadas de un punto céntrico de Montevideo y un

distancia de 23 kilómetros de forma de abarcar toda la ciudad. Además la API soporta filtros por fecha de inicio y de fin, que fueron ajustados de forma de obtener todos los datos disponibles desde el lanzamiento de la aplicación CityCop en el Uruguay.

Con esos parámetros definidos, se utiliza la API para obtener los datos de interés. Se realizan consultas para cada semestre por separado para limitar el tamaño de las respuestas, y para todos los incidentes obtenidos, se persiste un artículo indicando que la fuente es Crimeometer. Vale destacar que se utiliza el mismo modelo de artículo que para los artículos creados a partir de noticias de portales web, ya que de esta manera, se pueden manipular artículos de distintas fuentes indistintamente. Esto es de gran utilidad y simplifica, luego, el procesamiento de estos artículos.

Los artículos generados a partir de datos de Crimeometer no tienen la misma información que los generados a partir de las noticias de los portales web. Al ser datos que se introducen desde la aplicación de CityCop y no artículos de noticias, datos como el título y cuerpo del artículo no tienen sentido y por ende se dejan vacíos.

Extracción de datos de fuentes oficiales

Como se mencionó en la sección 2.4.1, el Observatorio Nacional de Violencia y Criminalidad es el medio oficial por donde se publican datos y estadísticas referentes a la criminalidad a nivel nacional en el Uruguay. Los reportes que ofrece son anuales y comparativos con el año anterior, conteniendo información de Hurtos, Rapiñas y Homicidios.

En primer lugar, cabe destacar que los datos extraídos de esta fuente se utilizaron de forma diferente a los demás. Los reportes del Observatorio brindan datos de crímenes por lo cual no se generan artículos a partir de los mismos. Al ser estadísticas de crímenes ya procesados no es necesario clasificarlos como se hace con los datos de fuentes no oficiales.

La tarea de extracción y carga de estos datos no fue simple. Una de las grandes razones es que no existe un sistema de parte del Observatorio el cual integrar. Asimismo, los datos que se precisan no están explícitamente en los documentos de acceso público, sino que se hizo un procesamiento manual de los mismos cómo se detalla en esta sección.

Para poder comenzar a desarrollar la extracción de datos de esta fuente oficial, es de suma importancia presentar el formato de los reportes anuales.

Se tiene un reporte de homicidios y otro de rapiñas y hurtos para cada año.

En cada uno de ellos se observan gráficas y tablas comparativas con el año anterior, no solo de Montevideo sino también a nivel nacional. Las que interesan para este sistema son las de crímenes consumados en un cierto periodo de tiempo. En la siguiente figura se expone un ejemplo de este tipo:

GRÁFICO 2

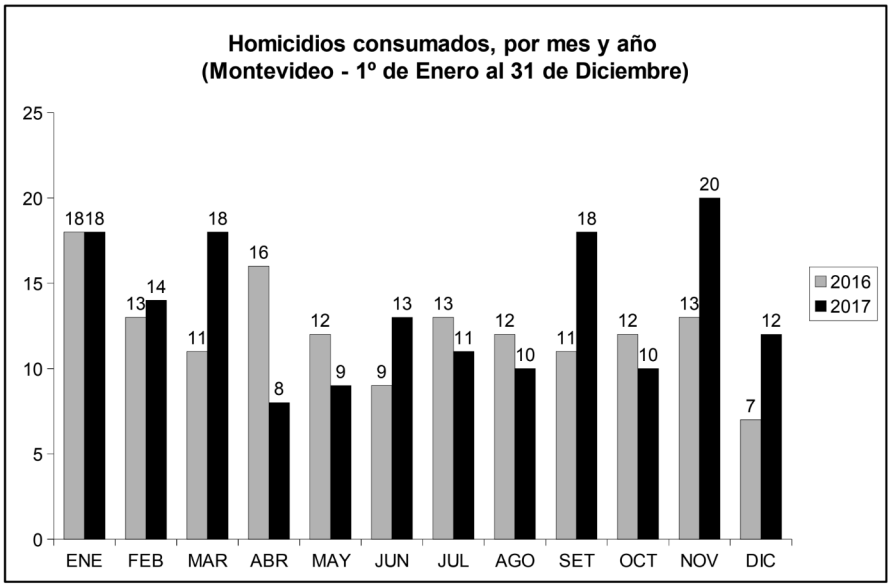

Figura 11: *Análisis comparativo de homicidios consumados entre el 2016 y 2017*

Si bien esta información es útil, no cuenta con información acerca de la ubicación que es imprescindible para poder cumplir con los objetivos del sistema. Sabiendo que los crímenes consumados son recabados por las 25 seccionales que hay en Montevideo, una posible solución es poder conocer, de alguna manera, la seccional de cada crimen. Teniendo esa información, se podrían asociar coordenadas a los crímenes y poder ser visualizados por la población. Esta no es información que se encuentre explícitamente en los reportes del Observatorio pero si se cuenta con otro dato, la cantidad de crímenes consumados por seccional a lo largo de un año.

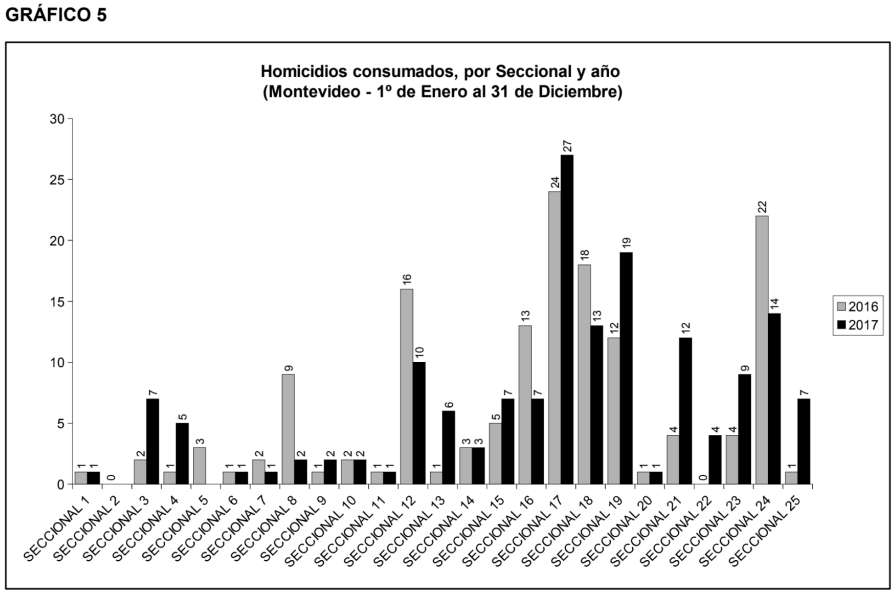

Figura 12: *Homicidios consumados por seccional en el 2016 y 2017*

Como se puede observar en la Figura 12, se cuenta con la cantidad de crímenes (en este caso homicidios) por seccional que ocurrieron en un período de un año (2016 y 2017). De forma de tener mayor granularidad y poder visualizar datos de crímenes recabados por seccional, dado no solo un año sino también un mes, se utilizaron ambas gráficas realizando una transformación de los datos para generar un nuevo documento.

En primer lugar, se decidió qué formato se iba a utilizar para el nuevo documento en donde se guardaron los nuevos datos generados por esta transformación. Esta decisión fue tomada en base a que tenía que ser viable manipular el mismo en el servidor para poder cargar esos nuevos datos en la base de datos. Por lo tanto, se decidió que el formato a utilizar fuese CSV [54].

En la siguiente figura se muestra un fragmento de uno de ellos:

Seccional	Enero			Febrero			Marzo		
	Homicidios	Rapiñas	Hurtos	Homicidios	Rapiñas	Hurtos	Homicidios	Rapiñas	Hurtos
1	0	11	110	0	5	84	1	23	99
2	0	18	262	0	15	287	0	13	259
3	0	22	154	0	19	164	0	21	188
4	0	49	239	0	33	221	1	48	291
5	1	51	332	0	52	314	0	58	344
6	0	31	145	0	16	125	0	38	173
7	0	50	151	0	54	119	0	69	197
8	1	132	236	1	89	242	0	117	261
9	0	45	386	0	55	413	0	80	433
10	1	34	338	0	46	256	0	60	350
11	1	84	293	0	73	212	0	70	220
12	3	103	255	2	85	229	1	140	287
13	0	111	275	0	151	232	1	145	319
14	0	69	262	0	107	184	0	103	196
15	0	160	303	0	163	270	0	165	313
16	2	160	211	2	167	222	2	187	244
17	2	266	200	2	232	177	1	216	197
18	2	146	180	2	131	189	2	138	225
19	2	185	304	2	212	299	1	232	398
20	0	11	23	0	6	19	0	11	22
21	2	110	156	1	90	171	1	130	159
22	0	39	94	1	33	74	0	59	96
23	1	88	94	1	86	86	1	76	94
24	3	176	183	2	146	175	2	163	196
25	1	100	94	1	114	108	1	82	105
	22	2251	5280	17	2180	4872	15	2444	5666

Figura 13: *Fragmento de un CSV con los datos del Observatorio transformados*

Como se puede observar, se tiene la cantidad de crímenes para cada tipo recabados en cada seccional en un mes dado. Además, es importante mencionar que se tiene un CSV para cada año en el cual se haya generado un reporte en el Observatorio.

La transformación realizada es la siguiente:

Dado un tipo de crimen,

1. Se calcula el porcentaje de crímenes que se cometieron en cada seccional sobre el total de crímenes del año. Esto se hace utilizando los datos de la gráfica de la Figura 12. Para obtener el total de crímenes del año, se suman los crímenes recabados en cada seccional para ese año.
2. A partir de ese porcentaje calculado, y asumiendo que se distribuye equitativamente en cada mes del año, se obtiene la cantidad en unidad de crímenes para cada seccional en ese mes.

Ejemplificando:

Contando con la siguiente información del año 2016 (tomada de los reportes):

- En la seccional 17 se registraron 24 homicidios.
- El total de homicidios es 147.
- En Enero de ese año, se cometieron 18 homicidios.

Entonces:

1. Como el total es 147, en Enero ocurrieron un 12.2 % de los homicidios.
2. Por lo tanto en la seccional 17 en Enero se registraron 3 crímenes en esa seccional (24 * 0.122).

Realizando esta transformación para cada mes y año, se tienen CSVs con los datos tal y como se muestra en la Figura 13.

Vale destacar que esta no termina siendo información exacta en cuanto al número de crímenes registrados en cada seccional en una fecha dada, ya que claramente considerar que la distribución de los crímenes en los distintos meses del año es igual para todas las seccionales no es correcto. Sin embargo, teniendo en cuenta la información con la que se cuenta, se cree que esta es la mejor forma de procesar estos datos para luego ser presentados a la población. Se cuenta con consistencia tanto en la cantidad de crímenes en cada seccional en un año, como también en la cantidad de crímenes en un mes en el conjunto de todas las seccionales.

Es importante notar que estos datos son diferentes a los que se extraen en las fuentes no oficiales (portales web y Crimeometer). En estas fuentes, se obtienen artículos que necesariamente se deben clasificar para entenderlos como crímenes, mientras que en este caso planteado, ya se tienen crímenes registrados.

Por último, luego de la transformación mencionada, se cargan esos datos procesados en la base de datos del sistema. Para ello, se crea un programa escrito en Python que forma parte del código del sistema. Este interpreta los distintos CSVs generados y los carga. Para cada crimen registrado, crea un nuevo crimen asociándole mes, año y tipo registrado. Además, en lo que respecta a la ubicación del mismo, se le asocian las mismas coordenadas que la de la seccional donde fue registrado, así como también el barrio a la cual pertenece dicha seccional.

En la sección 3 se detalla cómo están modelados los crímenes en este sistema.

El proceso de carga de estos reportes es manual tanto para generar nuevos documentos con datos procesados como para cargarlos en formato CSV en la base de datos.

4.2.2. Classifier

En Fetcher, los artículos son persistidos junto con múltiples atributos que son de utilidad para poder identificar si la noticia en cuestión trata de un crimen. En este componente, se hace uso de los mismos y mediante técnicas, herramientas y servicios externos se obtiene la información necesaria para poder procesar las noticias y llevar a cabo la identificación anterior.

Considerando el modelo de crimen detallado en el Capítulo 3, se implementó el proceso de clasificación. En el mismo, se hace distinción entre artículos de Crimeometer y de noticias de portales web ya que el proceso de identificación de alguno de los atributos es diferente.

Clasificación de artículos de noticias de portales web

A continuación se detalla el proceso de clasificación de artículos de noticias de portales web.

En primer lugar, se identifican las categorías del crimen. Para ello, se hizo uso de la herramienta Spacy para extraer entidades nombradas de un texto. Spacy es una librería open source para el procesamiento de lenguaje natural en Python. Tiene un amplio conjunto de funcionalidades, entre ellas se destacan las utilizadas en este proyecto:

- **Tokenization**: permite tener el texto segmentado en palabras, signos de puntuación, etcétera.
- **Lemmatization**: permite manipular las bases léxicas de las palabras. Por ejemplo: la base léxica de *hurto* es *hurtar*.

En el caso de las categorías, se especifican patrones (reglas) para cada una de ellas de forma de identificar si un token dentro del texto corresponde a una categoría. Teniendo en cuenta que es posible manipular la base léxica de cada token, estos patrones definen cuales de éstas están relacionadas con la categoría correspondiente. Por ejemplo: para la categoría *Hurto*, los patrones definidos son *{'LEMMA': 'hurtar'}* y *{'LEMMA': 'robar'}* . Esto quiere decir que si hay un token dentro del texto para el cual su base léxica es hurtar o robar, entonces el módulo indica que el artículo se trata de un hurto.

Cabe mencionar que estos patrones fueron definidos manualmente en base a un previo análisis del contenido de los artículos, reconociendo cuales eran las bases léxicas que mejor identifican a cada categoría, y que se fueron ajustando iterativamente. Además estos patrones se han adaptado a lo largo del proyecto a medida que se añadieron nuevas fuentes.

El proceso de clasificación de categorías toma los patrones de cada una de las categorías y los agrega al Matcher [55] de Spacy. Este permite encontrar coincidencias entre tokens del texto y ciertos patrones indicados, en este caso, los patrones de las categorías. Luego de que el Matcher analiza el texto, la salida son todos los tokens del texto para los cuales hubo coincidencias, es decir, las categorías reconocidas para ese artículo. En caso de no haber coincidencias, el proceso de clasificación finaliza sin éxito y el artículo no es considerado como un crimen.

En una segunda etapa, se identifica el barrio del crimen. Antes de desarrollar las distintos pasos de esta etapa, es importante mencionar que alguno de los barrios no tienen un único nombre debido a que la capa de barrios realiza agrupaciones de hasta 2 barrios distintos, y por lo tanto hay que tenerlo en cuenta dentro del proceso de identificación.

El reconocimiento del barrio tiene dos pasos bien definidos. En un primer paso, se intenta encontrar los nombres de los barrios dentro del texto. En caso de éxito, se retorna ese barrio como el reconocido. Si no se encuentra ninguno, se intenta buscar barrios los cuales sus nombres estén en el texto pero no estén escritos de la misma manera. Para ello, se encuentran las secciones del texto que son candidatas a especificar una ubicación como pueden ser las secciones que contengan las palabras: *en*, *barrio* o *barrios*. De esta manera, se acotan significativamente las secciones de texto a analizar mejorando significativamente el tiempo de ejecución.

Una vez obtenidas estas secciones, se intenta reconocer los barrios dentro de ellas. Para ello, se hace uso del ratio de Levenshtein que permite obtener la similaridad entre dos porciones de texto. Este oscila entre cero y uno, donde uno indica que los textos son iguales. Se incluye la librería de Python llamada Levenshtein [56] en la cual se provee el calculo de dicho ratio.

Además, se define un umbral de aceptación para cada nombre de barrio a partir del cual una sección candidata se considera como una coincidencia y por lo tanto, como potencial barrio. Este umbral también oscila entre cero y uno, decrementando a medida que el nombre del barrio está compuesto por un mayor número de palabras.

Por lo tanto, para cada nombre de barrio, se calcula el ratio de similaridad de Levenshtein y si es mayor que el umbral de aceptación definido para ese nombre, se reconoce como potencial barrio y es almacenado en memoria. Cuando este proceso finaliza para todos los barrios del sistema, los potenciales barrios se ordenan de menor a mayor cantidad de palabras que componen al nombre. Por último, se identifica al barrio de ese artículo

como al potencial candidato cuyo nombre es el compuesto por mayor cantidad de palabras.

Si bien los umbrales de aceptación tienden a uno a medida que la cantidad de palabras que componen al nombre del barrio disminuyen, se considera que el candidato con mayor largo es menos probable que sea un falso positivo.

En caso de que no se reconozca un barrio, el proceso de clasificación puede finalizar considerando que el artículo no es un crimen. Esto ocurre si además de no reconocerse el barrio, se encuentra dentro del texto alguno de los nombres de los departamentos fuera de Montevideo.

Luego de finalizado el proceso de identificación del barrio, se pasa a reconocer la ubicación del crimen. En una primera instancia, se buscan las calles dentro del artículo. Para este reconocimiento, la implementación es muy similar a la segunda etapa del reconocimiento de barrio. Se buscan las secciones del texto que potencialmente pueden indicar una calle, estas son: ***en***, ***calle*** o ***calles***. Luego, se utiliza Levenshtein para identificar la similaridad. En este caso, el umbral de aceptación es siempre 0.9. A diferencia que en el reconocimiento de barrio, se almacenan todas las calles candidatas que superaron el umbral como calles del crimen. Otra diferencia es que las calles que se buscan en los textos no están almacenadas en la base de datos del sistema como si están los barrio. Las calles se extraen de un archivo externo que contiene todas las calles de Montevideo.

Una vez reconocidas las calles, se intentan obtener las coordenadas dónde ocurrió el crimen. Esta tarea es llamada Geocoding [57]. Dado un texto como calles o nombre de un lugar, se retornan las coordenadas que corresponden. En este sistema, se utiliza un servicio de la API del Correo Uruguayo para realizar esta tarea [58].

La solución planteada para obtener la ubicación es consultar a la API del Correo Uruguayo enviando dos calles como parámetro. Se asume que ambas dos se cruzan y por ello, se añade la palabra ***esq*** entre ellas. La respuesta puede ser vacía ya sea porque las calles no se cruzan o bien, algunas de las calles reconocidas no es una calle realmente. Una vez se obtiene una respuesta se verifica que si el crimen cuenta con un barrio asociado, entonces las coordenadas estén contenidos dentro del mismo. En caso que la respuesta sea vacía o inválida, se intenta con un nuevo par de calles. En el momento en que la API retorne un par latitud/longitud válido, se consideran esas como las coordenadas dando por finalizado este paso.

Al estar utilizando un servicio externo, esta etapa de reconocimiento se realiza asincrónicamente haciendo uso de Celery [59] y Redis [60]. De esta manera

se logra realizar el procesamiento en procesos en segundo plano y con la capacidad de recuperarse de errores que pueden surgir si la API utilizada no está accesible en un momento dado.

Por último, se obtiene la fecha en la cual ocurrió el crimen. Se considera que esta fecha coincide con la fecha de publicación del artículo. Si bien esto puede no ser cierto en algunos casos, se cree que para efectos de este sistema hacer esta suposición es suficiente. Una posible mejora a este proceso es utilizar la misma idea que para reconocer el barrio y las calles. Se seleccionan secciones en el texto que sean candidatas a indicar una fecha. La diferencia en este caso es que los artículos nunca indican explícitamente la fecha en que el crimen fue cometido pero si aparecen frases como las siguientes: *la semana pasada* o *hace un mes*. Obteniendo estas frases, se podrían utilizar junto con la fecha de publicación para determinar la fecha exacta del crimen.

Clasificación de artículos de Crimeometer a crímenes

En lo que respecta a artículos de Crimeometer, la clasificación fue significativamente más sencilla dado que hay datos que son brindados por su API directamente.

En cuanto a las categorías, cada artículo de Crimeometer tiene un tipo de crimen asociado brindado por la API, el cual es parte del FBI-NIBRS standard. Por lo tanto, puede que no sea una de las categorías utilizadas en este sistema y listadas en el Capítulo 3. Por ello, se definieron qué categorías del estándar se asocian con las del sistema. No todas las categorías del sistema son parte del estándar y hay otras que tienen más de una correspondiente en el estándar. Por ejemplo, dentro del estándar está definido el tipo *Homicide Offenses* que se corresponde en el sistema con *Homicidio*. Si se observa ahora la categoría *Hurto*, se reconoce dentro del estándar como cualquiera de los tipos *Motor Vehicle Theft*, *Burglary/Breaking & Entering* o *Larceny/Theft Offenses*. En caso de no poder identificar ninguna categoría el artículo que se esta clasificando no se considera útil para el sistema y por lo tanto la clasificación finaliza, sin crear un nuevo crimen.

Para este tipo de artículos no es necesario identificar las calles en dónde fue cometido el crímen ya que las coordenadas son un dato que provee Crimeometer. Por lo tanto, los crímenes generados tienen ubicación pero no tienen calles, que es suficiente para las necesidades del sistema.

El siguiente paso es identificar el barrio donde el crimen fue cometido. Para ello, se utiliza la ubicación del crimen que es brindada por la API y el polígono de cada barrio. Se busca cuál es el polígono que contiene esa ubicación. Esto es

posible realizarlo gracias al manejador Postgis que cuenta con una estructura para realizar consultas geográficas de este tipo [61]. En caso de que ningún barrio contenga esas coordenadas, se considera que el crimen ocurrió fuera de los límites del departamento de Montevideo y no es considerado como parte de este sistema.

Por último, se determina la fecha en la cual ocurrió el crimen. Este dato, al igual que la ubicación, es parte de los datos enviados por la API y por lo tanto no es necesario identificarlo. Suponiendo que esta fecha no fue enviada en el momento en el que se consultaron los datos, el crimen no es persistido porque si bien se tienen todos los datos anteriores, la fecha es imprescindible para la utilidad de la plataforma.

Problemas detectados

Dado el proceso de clasificación detallado anteriormente, se encontraron ciertos problemas.

En primer lugar, en cuanto a la clasificación de artículos de noticias de portales web, se detectó que por la forma en la cual se obtiene la ubicación (coordenadas) de los crímenes puede haber ciertas incoherencias. Dado que se generan todas las permutaciones de dos entidades reconocidas como calles dentro del artículo para luego obtener las coordenadas, puede que se intenten conseguir coordenadas para dos calles que se cruzan pero que no son las que definen dónde fue cometido el crimen. Por lo tanto, si bien el reconocimiento de calles puede haber sido exitoso, la obtención de la ubicación es errónea. Más adelante en la sección 5.2 se valida y experimenta con más detalle este proceso de obtención de la ubicación sobre una muestra de artículos.

Otro de los inconvenientes detectados es la duplicación de crímenes. Al no haber un control de unicidad de un crimen entre las distintas fuentes no oficiales, es posible que uno de estos sea reconocido en más de una fuente y por lo tanto, el mismo esté duplicado en los datos de este sistema. Este problema no es trivial y por lo tanto, no es sencillo de solucionar. Si se agrega un control de unicidad a la hora de persistir un crimen, este debe definir ciertas reglas sobre los datos para identificar si ya existe ese crimen. Dados dos crímenes, un posible conjunto de reglas para determinar si se trata del mismo crimen puede ser:

- Ambos tienen la misma ubicación.
- Ambos ocurrieron en la misma fecha.
- Ambos tienen asociadas las mismas categorías.

Claramente, este conjunto está sujeto a que los tres datos utilizados se estén reconociendo con exactitud. En caso de que esté ocurriendo, igualmente pueden encontrarse falsos positivos ya que puede haber dos crímenes que cumplen esas tres reglas pero son diferentes y por lo tanto, no están duplicados. Es por esto que se decidió dejar este problema por fuera del alcance de este proyecto.

4.2.3. Service Layer

El tercer componente de este servidor es una API que permite acceder a determinadas funciones del servidor. El principal objetivo de una API es permitir que dos o más aplicaciones se comuniquen entre sí. En este sistema, es el componente que brinda varios puntos de acceso al servidor desde el cliente, para que este último haga la presentación de los datos recabados.

Además, la API implementada sigue el estándar REST. En él, todo dato se debe manejar como un recurso donde cada recurso debe tener un identificador único. La comunicación entre los distintos componentes se realiza mediante el protocolo HTTP y algunos de sus métodos como GET, PUT, POST y DELETE.

Este estándar especifica un conjunto de principios a seguir:

- **Cliente-servidor**: al seguir una arquitectura del tipo cliente-servidor, se aumenta la escalabilidad y portabilidad del sistema.
- **Sin estado**: las consultas que realiza el cliente al servidor son independientes unas de otras. Es decir, cada consulta debe incluir toda la información necesaria para ser interpretada de forma correcta sin necesidad de consultar por el contexto actual.
- **Almacenable en caché**: este principio requiere que las respuestas del servidor estén claramente identificadas como almacenables en caché o no, y de esta forma, el cliente puede luego reutilizar los datos consultados.
- **Interfaz uniforme**: REST genera una interfaz uniforme definiendo cómo se identifican los recursos, cómo se manipulan los mismos, cómo generar mensajes auto descriptivos y por último, utilizando hipermedia [62] como el motor del estado de la aplicación.
- **Sistema en capas**: en un sistema de este tipo, cada componente tiene su particular responsabilidad sin tener mayor detalle de cómo están

implementadas las capas con las cuales está interactuando.

- **Implementación bajo demanda**: indica que REST permite extender el funcionamiento del cliente bajando o ejecutando código en forma de scripts o applets [63]. Cabe mencionar que este principio es opcional.

Para implementar una API REST en este sistema se incluyó la herramienta llamada Django REST [64]. Esta herramienta provee un framework para crear APIs REST en Django. Tiene la ventaja de que se integra dentro de cualquiera proyecto ya creado en Django como si fuese cualquier otra librería o herramienta. Luego de integrarla, se crea un componente nuevo dentro de la aplicación que incluye todas las funcionalidades para la generación de APIs REST que brinda la librería.

Antes de pasar a detallar qué puntos de acceso provee la API implementada, es de interés mencionar un servicio externo que utiliza para poder llevar a cabo la tarea de proveer los datos necesarios al cliente web.

Dicho servicio es Elasticsearch, el motor de búsqueda y análisis de datos más utilizado en la actualidad. Como se mencionó en la sección 2.2, se debe almacenar los datos que se quieren consultar dentro del mismo. El formato estándar tanto para almacenar datos como para consultarlos es JSON.

La principal función de Elasticsearch en este sistema es poder mejorar los tiempos de respuesta de las consultas de la aplicación web, ya que se cuenta con grandes volúmenes de datos. Además, simplifica significativamente la implementación del buscador de crímenes basado en texto libre dado que provee consultas específicas sobre los datos basadas en un texto.

La integración de este servicio generó algunos desafíos para los que se detallan los más relevantes:

1. Manipular los datos mediante el uso del Query DSL.
2. Mantener la consistencia de los datos almacenados en la base de datos y en Elasticsearch.
3. Administrar el espacio de almacenamiento disponible.

Si bien la documentación de la herramienta es amplia y tiene todo lo necesario para poder ser utilizada, no es sencillo al inicio hacer uso del Query DSL especificado. La principal razón de ello es que no es un DSL similar a los que ya hay en el mercado y por lo tanto, requiere de una etapa de aprendizaje previa. Además, Elasticsearch almacena los datos en formato JSON llamando a la unidad de almacenamiento, documento. Esto hace aun más complicada la

etapa inicial de aprendizaje dado que no es la forma tradicional de manipular datos.

En lo que respecta al inconveniente número dos, en este sistema es de interés mantener la consistencia de los datos entre la base de datos y Elasticsearch. Esto no es trivial ya que cada vez que un crimen se añade o elimina de la base de datos, hay que replicar esa acción en los datos que hay almacenados en el motor de búsqueda. Para ello se simplificó la tarea de sincronizar los datos entre ambos almacenamientos, agregando una acción en la interfaz del administrador que se detalla en la sección 4.4.

Por otro lado, en un ambiente de desarrollo el espacio de almacenamiento disponible en la herramienta es tan grande como soporte la computadora utilizada. Sin embargo, cuando se libera el sistema es cuando aparecen problemas de este tipo. El servicio de hosting en la nube que se utiliza para liberar el servidor, detallado en la sección 4.5.2, integra Elasticsearch mediante Bonsai Elasticsearch [65].

El plan gratuito de este servicio permite almacenar únicamente 10.000 documentos. Como ya se mencionó, el documento es la unidad de almacenamiento en Elasticsearch. Un documento representa un recurso específico del sistema, como puede ser un crimen, un barrio o una categoría. Originalmente, se deseaba almacenar cada entidad relacionada a un crimen con todos sus datos. Por ejemplo, si se quería persistir una categoría relacionada a un crimen dado, se almacenaba dentro del mismo el documento correspondiente a esa categoría. Esto implica tener un documento con documentos anidados. Por ello, la cantidad se incrementó considerablemente superando el límite provisto por Bonsai, una vez que se intentaron almacenar todos los crímenes.

La solución a este impedimento fue no almacenar los documentos relacionados a los crímenes de manera completa, es decir, con todos sus datos asociados, de modo de no anidar documentos. En otras palabras, se hizo una desnormalización de los datos. Continuando con el ejemplo de las categorías, en lugar de almacenar un documento para cada categoría, se almacena un arreglo con los identificadores de las mismas. De esta manera, se reduce el número de documentos en Elasticsearch.

Retomando la implementación de la capa de servicios implementada, se detallan los puntos de acceso que provee:

- **Cantidad de crímenes por barrio según filtros dados**: para cada uno de los barrios, retorna la cantidad de crímenes total así como también la cantidad de estos para cada tipo aplicando los filtros solicitados.

- **Datos completos de crímenes según filtros dados**: retorna los datos completos de todos los crímenes filtrando los que correspondan.
- **Búsqueda de datos completos de crímenes según un texto dado**: Dado un texto libre, se retornan los crímenes que contengan en su ubicación (calles), título, contenido o URL del artículo relacionado, ese texto.
- **Cantidad de crímenes por seccional según filtros dados**: para cada seccional de Montevideo y aplicando los filtros dados, retorna el total de crímenes relevados. También, expone la cantidad de crímenes para cada tipo de crimen dentro de cada seccional.
- **Valores medios de la cantidad de crímenes por barrio según filtros dados**: retorna valores medios de cantidad de crímenes por barrio según filtros dados que son usados en el mapa de calor de la aplicación web para definir la escala de colores a utilizar. Haciendo uso de estos valores, se tiene una distribución de calores acorde y proporcional a los datos que se están visualizando según los filtros aplicados.

Los filtros soportados en ciertos puntos de acceso son:

- Categorías de los crímenes.
- Rango de fechas en el cual ocurrieron los crímenes.
- Fuentes de donde fueron extraídos los crímenes.

Estos servicios son accedidos por la aplicación web para obtener los datos necesarios para cumplir con los requerimientos. El formato elegido tanto para retornar los recursos en las respuestas como para recibir los parámetros en las consultas es JSON. Este es sencillo de manipular tanto para el servidor como para el cliente. El servidor utiliza serializadores [66], otra de las herramientas provistas por Django REST. Estos simplifican la serialización de los recursos, permitiendo enviar en la respuesta únicamente los datos deseados de modo de no dar información innecesaria a los clientes y así mejorar no solo la performance del servidor sino que también la seguridad del mismo, no exponiendo lógica o datos de la cual los clientes no deben tener conocimiento.

4.3. Cliente

El cliente es una aplicación web desarrollada en React JS [67], una librería de JavaScript [68] enfocada en el desarrollo de interfaces de usuario. Su principal ventaja es que permite desarrollar aplicaciones web de mejor calidad, donde el código está más ordenado y es más simple de entender. Otra de las virtudes

es que las interfaces se asocian con los datos que tiene siendo reactivas en caso de que algún dato se modifique.

Previo a la implementación, se realizó una etapa de diseño de la aplicación web. Fue sumamente importante para poder tener en claro el producto final al cual se quería llegar y además, para asegurarse que los objetivos del sistema se iban a cumplir. Para plasmar este diseño se eligió Figma [69]. Es una interfaz de diseño muy sencilla de utilizar en donde se pueden diseñar tanto aplicaciones web como de cualquier otra índole.

Sacando los detalles, lo mas sustancial del diseño final logrado es que cuenta con dos componentes:

- Mapa que permita tener una visión macro y micro de crímenes ocurridos en Montevideo, permitiendo filtrar por distintos parámetros de forma de acotar el set de datos.
- Estadísticas de crímenes en donde poder analizar los datos obtenidos.

Antes de pasar a explicar cómo se implementó cada uno de los componentes y los retos que cada uno presentó, es pertinente mencionar las distintas tecnologías utilizadas.

En lo que respecta al mapa, se utilizó una librería complementaria llamada Deck GL [70]. Fue creada por la empresa Uber [71] para el análisis y visualización de grandes set de datos. Las ventajas frente a otras librerías y, por las cuales se decidió hacer uso de está, son:

- Muy buena performance al momento de renderizar grandes set de datos. Esto es posible mediante el uso de buenas prácticas en la GPU.
- Excelente integración con React JS.

Cabe mencionar que Deck GL fue una de las herramientas sugeridas por el equipo de CityCop. En la sección 4.3.1 se brinda mayor información sobre cómo se utilizó en este sistema.

En lo que respecta a las estadísticas, se hizo uso de Metabase [72]. Es una herramienta de de BI (Business Intelligent) que permite explorar, analizar y visualizar datos de manera simple e intuitiva. Es posible generar reportes de todo tipo y con distintos componentes como por ejemplo gráficas, tablas, entre otros. Permite integrar reportes en aplicaciones web muy fácilmente mediante el uso de IFrames [73]. En este sistema, esta herramienta da acceso a estadísticas de los crímenes, tanto generales como comparativas (por ejemplo, entre fuentes), las cuales se presentan con mayor profundidad en la sección 4.3.2.

4.3.1. Mapa

El mapa es el componente principal de la capa de presentación del sistema, ya que es lo primero que enfrenta el usuario al ingresar. Como se mencionó anteriormente, utiliza la librería Deck GL que brinda la posibilidad de renderizar y visualizar capas de datos, en nuestro caso, sobre un mapa. Para ello dispone de una serie de capas predefinidas [74], y que en conjunto con la capacidad de actualizarse de forma rápida y eficiente, y de integrarse con facilidad a un mapa, demostró ser muy valioso para la realización de este sistema. Esta construido utilizando WebGL [75], que es una API Javascript para el renderizado de gráficos tanto en 2D como en 3D, con el objetivo de ser performante en la Web.

Por su parte, el mapa que se utiliza es Mapbox GL, plataforma open source para el renderizado de mapas y análisis de información geográfica. Más precisamente, la librería utilizada fue React MapGL [76], que es un wrapper de Mapbox GL que facilita su integración e interacción con DeckGL, de forma que las capas se dibujen y actualicen de forma consistente con el mapa.

Cuenta con dos modos de visualización distintos. El primero está dado por la vista de los barrios de Montevideo, mientras que el segundo está orientado a mostrar puntos por cada crimen. Cada uno de los modos de visualización utilizan una capa distinta provista por DeckGL, que se muestran o esconden según el nivel de zoom del usuario en el mapa. Esto permite mejorar la performance, ya que al tener una de las capas siempre escondida cuando la otra es visible, el número de operaciones se reduce. Por otro lado, hay una serie de filtros que se utilizan para que el usuario pueda ver el subconjunto de datos que sean de su interés, y aplican a ambos modos de visualización. A continuación se describe cada uno de ellos.

Capa de barrios

La capa de barrios es la capa que se muestra por defecto. Es una GeoJsonLayer de DeckGL, que como su nombre indica, renderiza un archivo en formato geojson. Para el caso de este sistema, se cargó un archivo en formato GeoJson con los mismos polígonos utilizados en el clasificador (mencionados en 4.2.2, de forma que se tiene una correspondencia 1-1 entre los barrios de los crímenes y los barrios de la capa de presentación. Gracias a esta correspondencia se vuelve muy sencillo presentar los datos, ya que se utiliza el número identificador de los barrios para todas las operaciones. Una característica de la capa mencionada anteriormente es que expone una API para manejar el color, la elevación, y otros atributos de cada una de las

entidades presentes en el archivo cargado; permitiendo así la generación de un mapa de calor 3D por barrio, como se observa en la Figura 14.

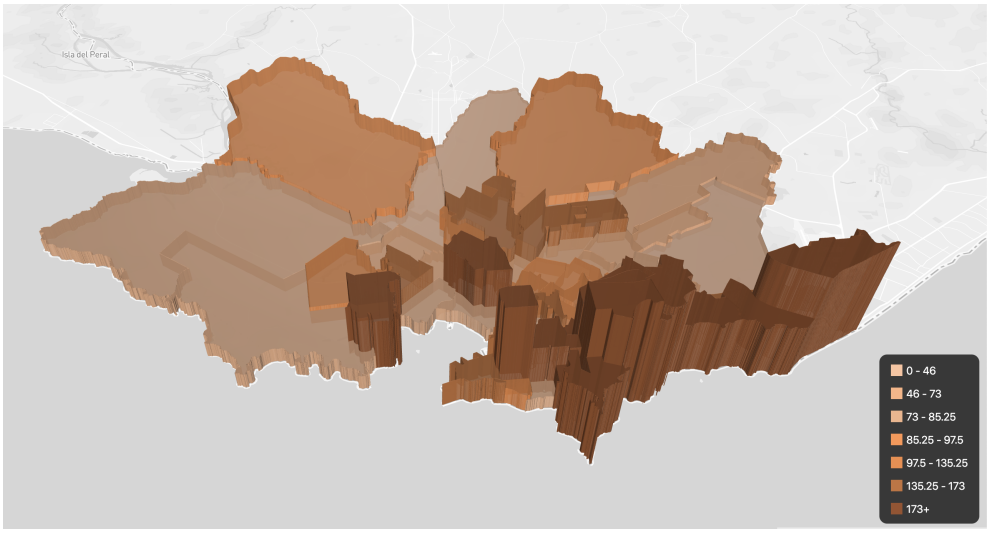

Figura 14: *Capa de barrios del mapa*

Para obtener la información y mostrarla en esta capa, se realizan una serie de operaciones. Al iniciar la aplicación (y cada vez que se modifican los filtros activos), se hacen 2 solicitudes en paralelo al servidor. La primera obtiene la cantidad de crímenes por barrio y por tipo, que es la información principal según la cual se calculan los atributos de la capa a renderizar. Por su parte, la segunda obtiene los rangos que delimitan los colores del mapa de calor y se pueden visualizar en el cuadro de referencia de la esquina inferior derecha en la Figura 14. A ambas solicitudes se le agregan como parámetros los filtros aplicados, que al inicial el sistema son vacíos.

Una vez terminadas ambas solicitudes al servidor, se actualiza una variable que es observada por la capa de DeckGL, de forma que al verse modificada, la capa es renderizada nuevamente con la nueva información. Esta actualización refresca los colores de los polígonos, sus bordes, y su elevación.

Además, se agregó un tooltip que se despliega al hacer hover sobre cada uno de los barrios. El mismo contiene la cantidad de crímenes para el barrio según los filtros, y un detalle para cada una de las categorías para las cuales hay crímenes en el barrio y que además no fueron filtradas.

Capa de crímenes individuales

Por su parte, la capa de crímenes individuales se muestra al acercar el zoom

del mapa. La característica más importante es que realiza un clusterizado de los marcadores de crímenes, agrupándolos según su ubicación geográfica y desarmando los grupos según se modifica el nivel de zoom. Para lograr este comportamiento, se creo una capa específica heredando de la clase CompositeLayer y utilizando IconLayer, de forma que se logran manejar distintos métodos del ciclo de vida del componente de forma manual y redibujar la capa de puntos que brinda IconLayer cuando corresponda. Además, se utilizó la librería [77] para la construcción de los clusters.

Los datos a mostrar en esta capa, al igual que para la de barrios, se obtienen al iniciar la aplicación por primera vez y cada vez que se modifican los filtros. Consta de una llamada al servidor que obtiene las coordenadas y datos de los crímenes de forma individual.

En este caso también se construyó un tooltip, que detalla las principales características de los crímenes y da la posibilidad de abrir la noticia que la generó en caso que corresponda.

Filtros

Como se mencionó anteriormente, existen una serie de filtros que al aplicarse desencadenan las actualizaciones de las capas anteriores. Los filtros disponibles son:

- Fecha: Permite seleccionar un rango de fechas a considerar. Por defecto se considera todo el rango de fechas disponible.
- Tipo: Permite seleccionar las categorías a considerar. Por defecto no hay ninguna categoría seleccionada, que resulta en considerar todas ellas.
- Fuente: Aplica solamente a los datos obtenidos de las noticias. Permite seleccionar datos provenientes de qué fuentes interesa visualizar. Al igual que en el caso anterior, por defecto no hay ninguna fuente seleccionada y por ende todas son tenidas en cuenta en la información que se despliega en el mapa.

Figura 15: *Filtros*

Por otro lado, existe un interruptor para elegir si los datos a desplegar son oficiales o provenientes de noticias, que es la opción por defecto. Al elegir los datos oficiales, los filtros descriptos anteriormente que no tienen sentido por el contexto, son deshabilitados. Estos son:

- Fuente: todos los datos son provenientes del Ministerio del Interior, y por ende no hay filtro aplicable.
- Tipo: se dejan habilitadas solamente 3 categorías - Hurtos, Rapiñas y Homicidios - ya que son para las cuales se cuenta con información oficial.

Es importante remarcar que la capa de crímenes individuales, cuando el filtro

de crímenes oficiales está activo, muestra los crímenes en la seccional asociada a los mismos y no en el lugar del hecho. Esto se debe a una limitación en los datos oficiales, que no cuentan con información suficiente para realizar la geolocalización a nivel individual de los crímenes.

4.3.2. Estadísticas

Este componente pretende cumplir con el objetivo de poder realizar un análisis comparativo entre los datos relevados de las fuentes oficiales y no oficiales. Además, al ser tan sencillo explorar los datos relevados con la herramienta utilizada, se creyó valioso exponer algunos otros datos que pueden ser de interés para la población.

Como ya se mencionó, Metabase es la herramienta elegida para implementar este componente. En esta, se generan las distintas gráficas y tablas para luego introducirlas en un dashboard. Luego, puede ser accedido públicamente mediante una URL. Por lo tanto, la manera en la cual se integra el mismo con el sistema es mediante el uso de los IFrames. Esta estructura HTML permite insertar una página web dentro de otra tan solo teniendo la URL de la web que se quiere anidar. Esto además de ser muy sencillo de realizar tiene otra ventaja, los cambios que se realizan en el dashboard de Metabase son reflejados instantáneamente en la aplicación web principal.

Dentro de la herramienta es posible consultar cualquiera de las tablas de la base de datos de tres maneras diferentes:

1. Consultas pre-armadas por la herramienta sobre las tablas.
2. Consultas armadas mediante el uso del editor que provee la herramienta. Esta opción permite realizar consultas más avanzadas que la anterior. Es posible realizar joins, agrupaciones y decidir parte de la visualización de los datos.
3. Consultas en SQL nativo escritas por el usuario.

En este sistema, se utilizaron consultas del tipo número dos ya que el editor que brinda la herramienta es suficiente para obtener los datos deseados.

Luego, los resultados de las consultas pueden visualizarse de distintas maneras:

- Gráficas
- Tablas
- Mapas

Por último, los resultados de distintas consultas pueden agruparse en un mismo dashboard y de esa manera visualizar todos los datos en conjunto. Además, se pueden agregar filtros interactivos dentro de los mismos.

En la siguiente figura se puede observar parte del menú de estadísticas dentro de la aplicación web principal:

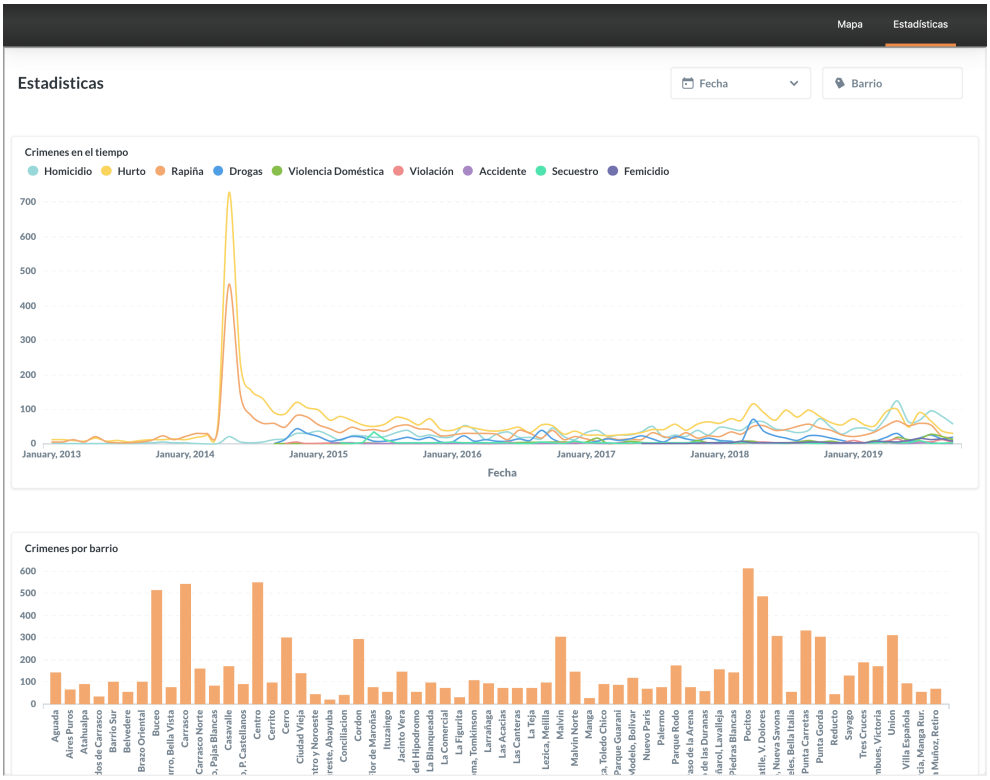

Figura 16: *Primera parte del menú de estadísticas de la aplicación web*

En esta se pueden observar dos gráficas que exponen los datos relevados de las fuentes. La primera de ellas muestra la evolución de los crímenes en el tiempo para cada tipo de crimen mientras que, la siguiente, enseña la cantidad de crímenes por barrio de Montevideo.

Por otro lado, se comparan los datos recabados entre las fuentes oficiales y las no oficiales. Este análisis comparativo se puede observar en la siguiente figura:

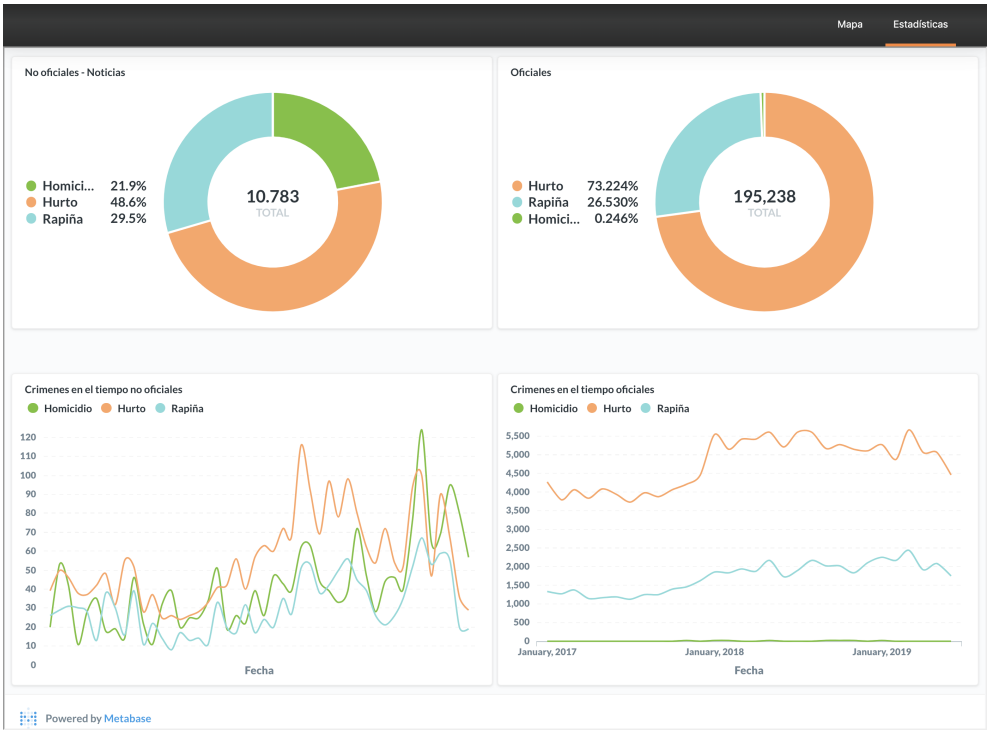

Figura 17: *Segunda parte del menú de estadísticas de la aplicación web*

En la parte superior, se comparan los porcentajes de homicidios, hurtos y rapiñas recabados mientras que, en la segunda comparación, se observa como se comportan estos tipos de crímenes a lo largo del tiempo.

La herramienta ha mostrado tener buenos tiempos de respuesta a la hora de realizar estas consultas. En caso de tener problemas de performance, es posible agregar una configuración para que los resultados de las consultas se almacenen en caché. Esto es una gran ventaja al momento de decidir qué herramienta usar dado que el volumen de datos en este tipo de sistemas tiende a crecer significativamente a lo largo del tiempo.

4.4. Interfaz para administración

Una interfaz para administración es una interfaz que permite a usuarios administradores poder crear, consultar, actualizar y borrar datos de su aplicación. Es de mucha ayuda para este tipo de usuarios por varias razones. La primera es que al no estar capacitados técnicamente en el rubro, no es posible para ellos acceder a los datos persistidos en una base de datos. Si estuviesen técnicamente preparados, puede que no conozcan como

es la estructura y el modelado del sistema y por lo tanto, les es muy complicado manipular datos. Por último, se pueden automatizar tareas de administración que usualmente llevan tiempo o requieren de una serie de pasos difícil de memorizar.

Para implementar una interfaz de este tipo en este sistema, se hizo uso de Django Admin [78]. Esta viene incluida dentro del framework sin necesidad de realizar ningún tipo de configuración, la única tarea a llevar a cabo es registrar los modelos que se quieren manipular en esta interfaz. Si se desea crear, consultar, modificar y eliminar crímenes por ejemplo, basta con especificar esto dentro de un archivo particular. Por otro lado, también trae configurado el manejo de usuarios administradores, su autenticación y autorización, lo cual redujo significativamente los tiempos de implementación de la misma.

Habiendo iniciado sesión en el sitio, se observa una interfaz como la que muestra la siguiente figura:

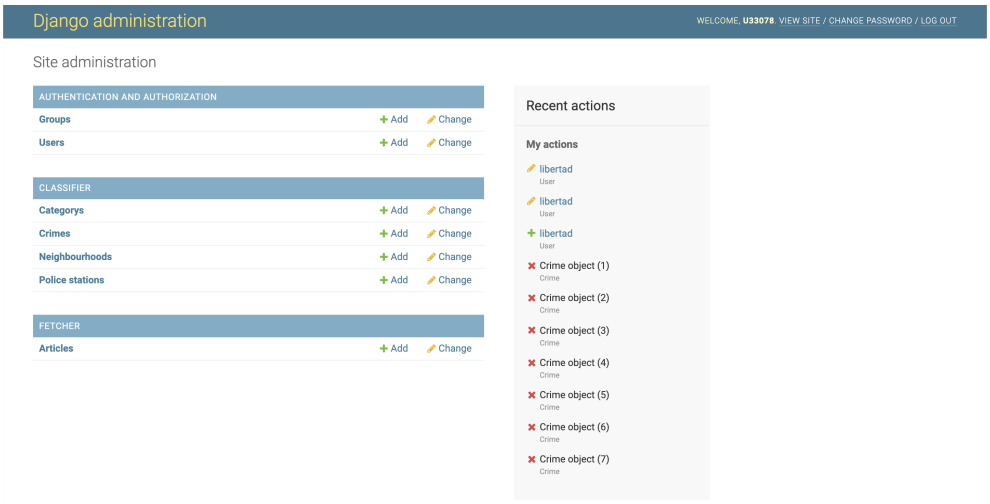

Figura 18: *Interfaz para administración del sistema*

En la Figura 18 se detallan claramente los modelos registrados además de los usuarios administradores ingresados al sistema. Dentro de cada modelo, se pueden ver los distintos registrados persistidos en la base de datos, siendo posible reducir la cantidad observada haciendo uso de distintos filtros. Esto se muestra en la siguiente figura:

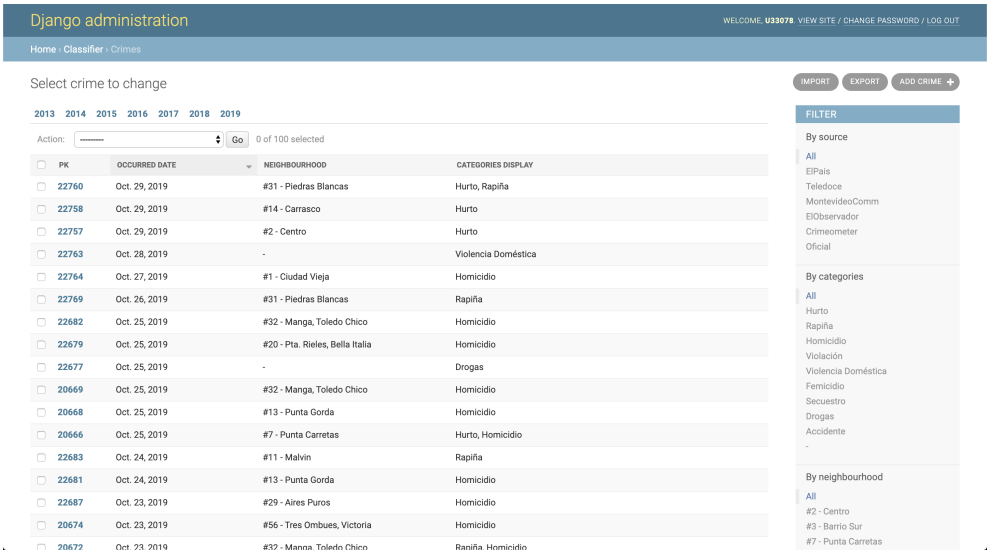

Figura 19: *Registro de crímenes dentro de la interfaz de administración*

Asimismo, como cualquier interfaz de administración permite crear, consultar, actualizar y borrar un crimen en particular.

Por otra parte, como ya se mencionó, se pueden automatizar tareas. Para esto, se deben implementar dichas tareas como acciones de cada modelo. En este sistema, se implementó más de una tarea de forma de simplificar algunos procesos. Éstas son:

- **Extraer artículos dada una fuente**: esta tarea permite obtener los artículos de una fuente seleccionada. Los fuentes posibles son: El País, El Observador, Teledoce, Montevideo COMM y Crimeometer.
- **Clasificar artículos**: es posible seleccionar varios artículos para luego ejecutar esta acción y generar los crímenes correspondientes.
- **Indexar crímenes en Elasticsearch**: automatiza el proceso de mantener la consistencia entre los crímenes que hay en la base de datos y en Elasticsearch. Esta tarea indexa todos los que hay en la base de datos sin importar si ya estaban indexados anteriormente, en cuyo caso el motor de Elasticsearch no duplica sino que simplemente actualiza el registro si algún atributo fue modificado.

Vale destacar que estas tres tareas implementadas se ejecutan asincrónicamente utilizando Celery. Dado que pueden tardar mucho tiempo en culminar y afectar la performance de la aplicación, cada vez que se quiere llevar a cabo una de estas acciones desde la interfaz se crea una tarea

asincrónica a ser ejecutada por Celery. Esto permite que la interfaz obtenga una respuesta rápida sin tener que esperar a que la acción finalice.

Otra cualidad importante de la interfaz de administración es que provee la habilidad de exportar los datos de cualquiera de los modelos disponibles, visibles en la figura 18. De esta manera es posible generar y descargar archivos en múltiples formatos que contienen toda la información del sistema relativa a un modelo para realizar respaldos o para reutilizar la información de forma sencilla en otros sistemas. Para que el respaldo tenga sentido, se provee también de la funcionalidad de importar datos al sistema. Algunos de los formatos de archivos disponibles son CSV, XLSX y JSON.

4.5. Infraestructura

En esta sección se presenta la infraestructura y herramientas utilizadas, tanto para el servidor como para el cliente, para poder hacer este sistema de acceso público. Sin realizar esta tarea, parte de los objetivos de este proyecto que son dar visibilidad y transparencia a la población de la información relevada, no se cumplen y por lo tanto el proyecto quedaría inconcluso.

Al momento de decidir de qué manera hacerlo, surgen diferentes opciones. En primer lugar, se puede elegir entre tener una infraestructura propia o en la nube. En este caso, se decidió manejar la infraestructura en la nube por dos razones. Primero, los costos que implica tener una infraestructura propia son muy elevados y la otra razón es que el tiempo que lleva implantar una infraestructura propia es mucho mayor al de hacerlo en la nube.

4.5.1. Cliente

Para el cliente web se utiliza Netlify [79], un servicio de web hosting en la nube. Hoy en día está siendo muy utilizado por brindar ciertas ventajas sobre otros servicios de este tipo, algunas de estas son:

- Plan gratuito para cuentas individuales.
- Provee una plataforma en línea para administrar la aplicación muy sencilla e intuitiva.
- La aplicación web manejada por medio de este servicio cuenta automáticamente con HTTPS.
- En caso de encontrar un error en la última versión liberada, es muy sencillo regresar a una versión anterior.

- Es sencillo de integrar con Github, servicio donde está el código de la aplicación web de este proyecto.

4.5.2. Servidor

Al momento de decidir la infraestructura en la nube a utilizar para el servidor, se plantea otra incógnita. Se tienen tres modelos de servicios en la nube, estos son:

- IaaS [80]: en estos servicios, el proveedor brinda infraestructura virtualizada la cual se usa del modo que se desee teniendo total control sobre la misma. Es similar a tener una infraestructura propia con la diferencia de que el encargado del hardware en este caso es el proveedor del servicio. Por lo tanto, este realiza toda tarea de mantenimiento y administración sobre los distintos equipos. Un servicio muy utilizado hoy en día es AWS (Amazon Web Services) [81]
- PaaS [82]: en este modelo, el que contrata el servicio únicamente se enfoca en el desarrollo de la aplicación y en ciertas configuraciones de la misma. Los proveedores exponen plataformas en línea para brindarle a los desarrolladores la posibilidad de administrar la aplicación. Algunos ejemplos son Heroku [83], Oracle [84] y Google App Engine [85].
- SaaS [86]: estos servicios brindan soluciones completas a las cuales los usuarios pueden acceder mediante la web. En estos casos, el que contrató el servicio no tiene de que preocuparse. Servicios de correo electrónico y de streaming de videos son ejemplos de este tipo de servicios.

En este caso, se decidió utilizar Heroku, un PaaS que brinda herramientas para desarrollar, liberar y administrar una aplicación sin necesidad de tener que lidiar con todos los posibles problemas que se encuentran a la hora de implantar una infraestructura.

Además de tener las ventajas inherentes de ser un PaaS, este servicio tiene algunas otras que lo caracterizan:

- Es muy simple y ágil configurar un nuevo proyecto dentro de la plataforma.
- Soporta una amplia gama de lenguajes de programación.
- Posibilidad de integrar varios servicios, algunos de ellos construidos por el equipo de desarrolladores de Heroku y muchos otros externos.

- Plan gratuito para aplicación pequeñas.

En lo que respecta a este proyecto, se utiliza Heroku para liberar y administrar el servidor Django. Como se mencionó en la sección, este incluye los componentes de fetcher, classifier y una API, así como también la Interfaz para administración.

En cuanto a los servicios y herramientas que son posibles integrar con la aplicación, se utilizan los siguientes:

- Heroku Postgres [87]: servicio que provee la base de datos PostgreSQL.
- Autobus [88]: realiza automáticamente respaldos de la base de datos.
- Bonsai ElasticSearch [65]: servicio que integra Elasticsearch a la aplicación.
- Heroku Redis [89]: este servicio integra Redis a la aplicación.
- Logentries [90]: servicio de logs que brinda la posibilidad de consultar y administrar los distintos logs de la aplicación. Este permite configurar alertas de forma de notificar a los desarrolladores en caso de una falla. Además, es posible agregar etiquetas sobre los logs registrados de forma de identificarlos fácilmente.
- Rollbar [91]: servicio de administración y seguimiento de errores para aplicaciones.

5. Validación y experimentos

5.1. Cantidad de artículos y crímenes en el sistema

En una visión global del sistema, se tiene registro de 219.002 *artículos*, dónde por artículo se entienden tanto noticias propiamente dichas como reportes de Crimeometer y datos oficiales. Dentro de ellos, una inmensa mayoría son de datos oficiales, lo que resulta lógico considerando que en los mismos se ve la totalidad de denuncias realizadas en el territorio de Montevideo. En la Figura 20 se ven los totales para cada uno de los tipos de datos que se manejan.

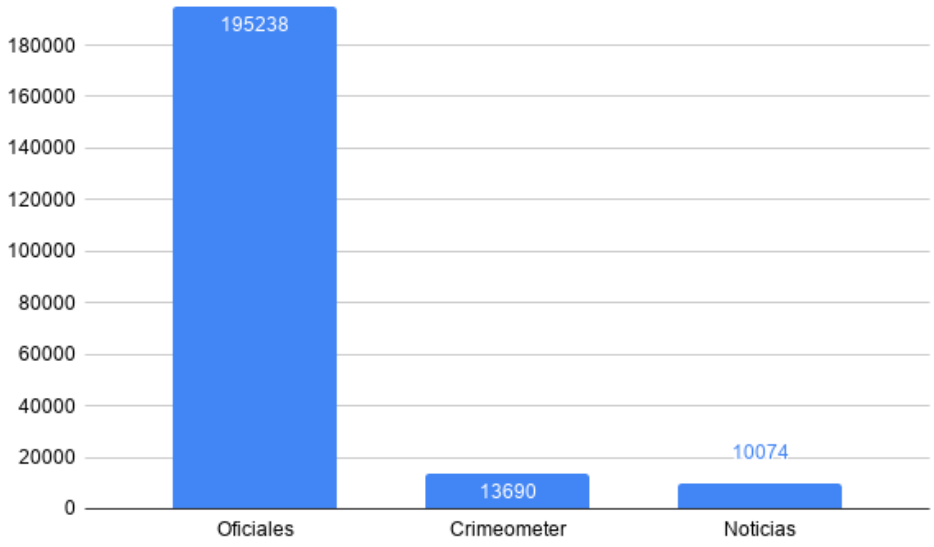

Figura 20: *Total de artículos*

A continuación se indaga en los detalles y pormenores de estos valores.

Datos oficiales

Se cuenta con datos oficiales para los años 2017, 2018, y el primer semestre del año 2019. Estos datos fueron extraídos de las estadísticas del Observatorio Nacional Sobre Violencia y Criminalidad [38], y cuenta con datos de Hurtos, Rapiñas y Homicidios. Como se observa en la Figura 21, por más que el rango de meses cargados al sistema no es muy extenso, hay una clara tendencia al aumento; con uno especialmente importante a finales del 2017 y principios del 2018.

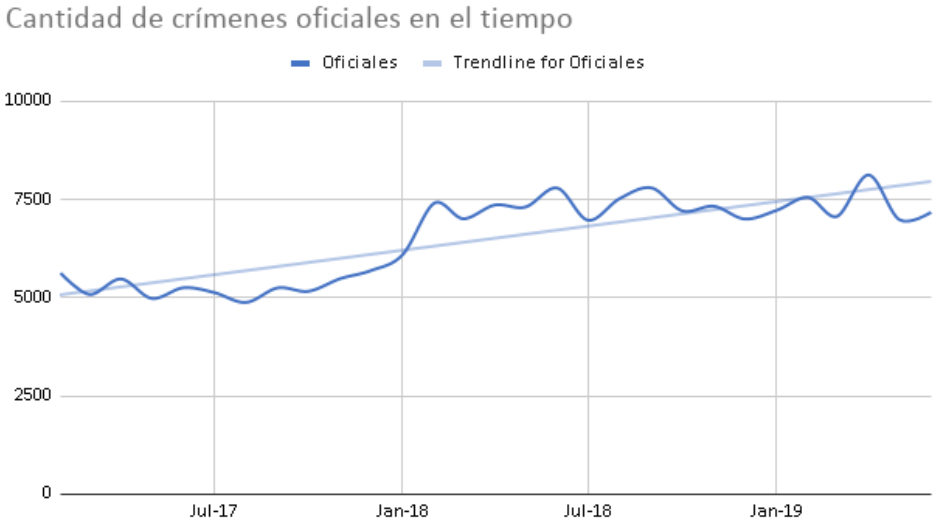

Figura 21: *Cantidad de crímenes oficiales en el tiempo*

Estos datos se agrupan por tipo de crimen y por seccional. Considerando la primer categorización, los Hurtos son los delitos más preponderantes. Al considerar las seccionales, la #19 de La Teja, la #9 de La Blanqueada y la #15 de La Unión son las que tienen una mayor cantidad de crímenes. Vale destacar que para realmente comprender la importancia relativa de la criminalidad en cada barrio, sería importante analizar la cantidad de crímenes según la cantidad de gente que allí vive.

Datos extra oficiales

Dentro de lo que se denominan datos *extra oficiales*, están aquellos provenientes de alguna de las fuentes de noticias que se utilizaron en este trabajo, y también aquellos provenientes de los reportes realizados por usuarios de la plataforma CityCop y obtenidos mediantes la API de Crimeometer.

Los primeros tienen la particularidad de que al obtenerlos, ya tienen la información procesada, como por ejemplo el tipo y la ubicación. Por lo tanto, el 100 % de los reportes obtenidos desde la API de Crimeometer podrían ser mapeados a un crimen. Sin embargo, algunas categorías existentes en dicha plataforma no son compatibles con las categorías de crimen definidas para este trabajo, y por ende fueron descartados algunos reportes. Además, no todos los reportes eran en la ciudad de Montevideo, por lo que aquellos que no están contenidos en alguno de los barrios

tampoco fueron tenidos en cuenta. En la Figura 22 se ve la cantidad de reportes y crímenes para estos datos, siendo la diferencia entre ambos valores la cantidad de reportes descartados por uno de los dos conceptos descriptos anteriormente.

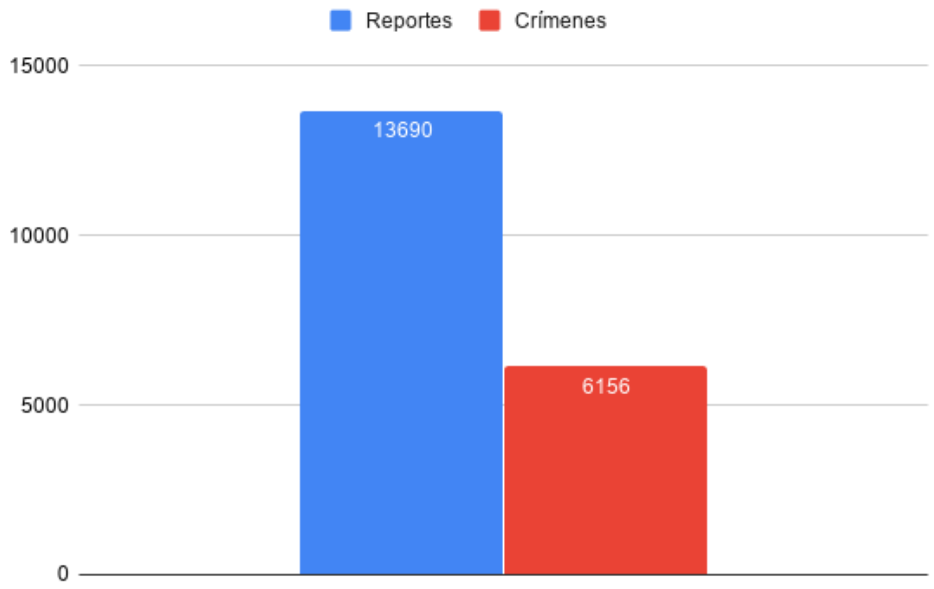

Figura 22: *Cantidad de reportes y crímenes provenientes de Crimeometer*

Con respecto a los datos obtenidos desde fuentes de noticias, hay una gran diferencia entre la cantidad de artículos que se pudo obtener de Teledoce con respecto al resto de las fuentes utilizadas, como se observa en la Figura 23. Esta diferencia está dada mayoritariamente porque es el portal de noticias que tiene accesible un mayor historial de noticias.

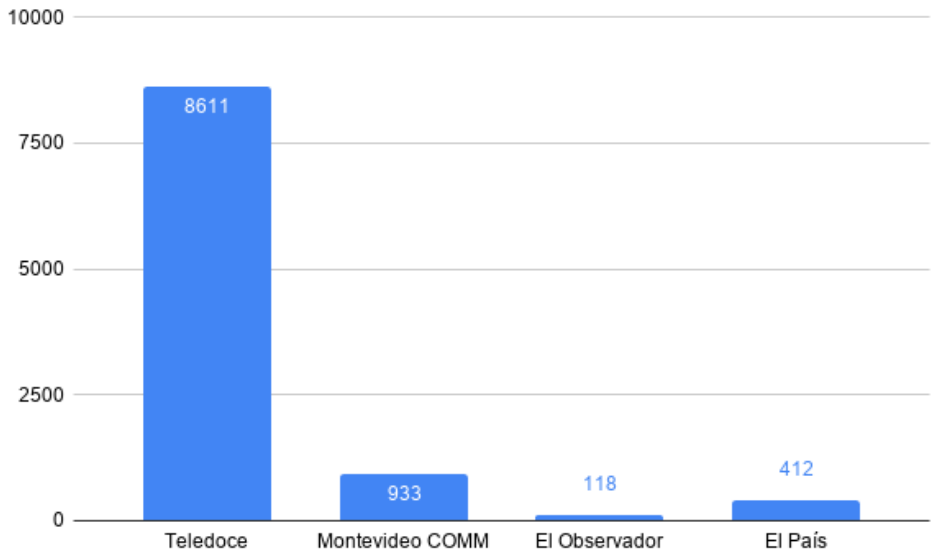

Figura 23: *Cantidad de artículos por fuente de noticias*

Para poder evaluar si existe alguna tendencia en cuanto a la cantidad de noticias relacionadas a criminalidad en Montevideo, es menester hacerlo dividiendo los datos por cada una de las fuentes. Observando la Figura 24, resulta complejo formar conclusiones para Montevideo COMM y El País dado el breve rango de tiempo en el que están presentes, y tampoco para El Observador ya que son escazas. La mejor fuente para poder sacar conclusiones en este sentido es Teledoce, y coincide con la tendencia que se observó anteriormente también con los datos oficiales. La cantidad de noticias relativas a criminalidad crece de forma constante y bastante acelerada durante los últimos años, lo que se condice con la creciente importancia que tiene el tema en la sociedad.

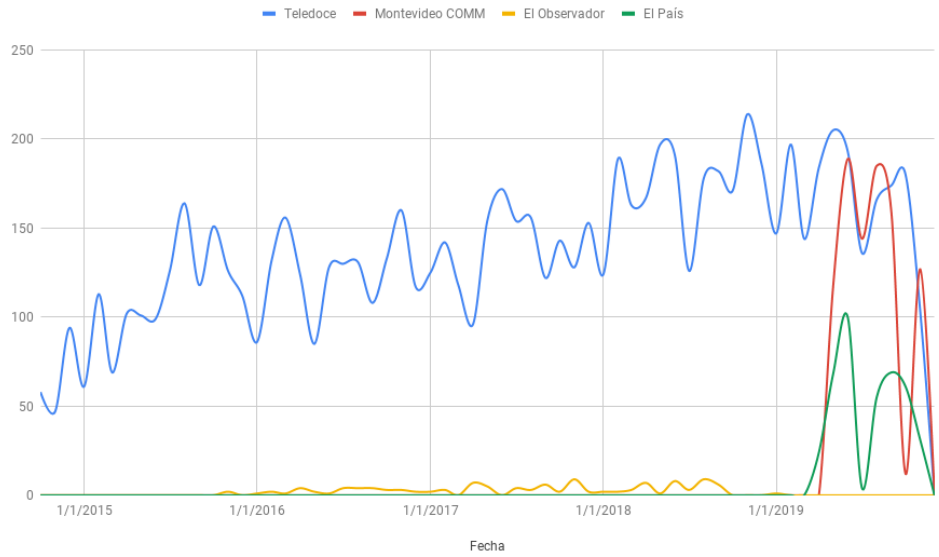

Figura 24: *Cantidad de artículos por fuente de noticias en el tiempo*

Es interesante destacar que fue posible clasificar el 46.33 % del total de las noticias, dejando un total de 4668 crímenes correspondientes a este tipo de fuente.

5.2. Pruebas unitarias en muestreo de noticias

En esta sección se analiza el comportamiento del módulo de clasificación descripto en la sección 4.2.2, con el objetivo de evaluar la precisión de los datos extraídos desde las fuentes de noticias. En todos los casos se realiza el asesoramiento de si la noticia realmente corresponde a un delito, y si así fuera, si la información asociada a la misma es precisa.

En particular, interesa investigar el reconocimiento de categorías, barrios y ubicación de los crímenes. Para ello, se realizó un muestreo lo más representativo posible del corpus existente, pero a su vez considerando todas las fuentes por separado para poder discernir posibles diferencias en el comportamiento para cada una de ellas. De esta forma es que se tomaron las siguientes cantidades de crímenes a modo de muestra:

- 50 de un total de 4032 para Teledoce.
- 25 de un total de 399 para Montevideo COMM.
- 5 de un total de 35 para El Observador.

- 15 de un total de 202 para El País.

Cada uno de estos crímenes es analizado manualmente y comparado con los datos que el clasificador arrojó en cada una de los puntos anteriormente mencionados. Se observan además los resultados separados según la fuente, para luego realizar una pequeña evaluación del desempeño y posibles mejoras.

Por otro lado, se evalúa la presencia de falsos negativos al momento de clasificar las noticias, para lo cual también se realizó una selección de 40 noticias que no fueron clasificadas, de forma de poder identificar si en algunos casos las noticias sí tenían suficiente información como para ser tenidas en cuenta.

A modo informativo se menciona que las distintas muestras se tomaron utilizando la librería Random de Python - y en particular la función choices del mismo [92].

5.2.1. Reconocimiento de categorías

El reconocimiento de categorías, como se explicó en la sección 4.2.2, es obligatorio. Por lo tanto, los crímenes pueden tener la categoría correctamente reconocida o no, pero no ocurren situaciones en que crímenes no tengan una categoría cuando sí debería.

Teledoce

De los 50 crímenes seleccionados al azar, 47 reconocieron la categoría a la que pertenecen correctamente. Esto representa un 94 % de precisión.

Montevideo COMM

Para el caso de Montevideo COMM se observa que 22 de los 25 (88 %) crímenes tienen una categoría que refleja correctamente el delito descripto en la noticia.

El Observador

Solamente 1 de las 5 muestras fue identificada como correcta, situación que se debe a que el resto de las noticias no eran directamente describiendo un crimen.

El País

13 de las 15 muestras presentaron un reconocimiento de categorías preciso para el caso del diario El País, que implica un 86.6 % de acierto. Una

característica que se notó al analizar los dos casos de no reconocimiento es que en ambos se trata de noticias que no refieren directamente a un crimen, sino que son noticias derivadas de crímenes y por ende no deberían formar parte del corpus final en primer lugar.

Conclusiones

Como es notorio en los números expuestos anteriormente, se cuenta con una precisión muy alta en este rubro con 83 de 95 muestras correctas, que representa un 87 % y que puede también implicar la existencia de un número de falsos negativos importante, hecho que se estudia en la sección 5.2.4. La fuente que se separa de la norma es El Observador, donde se descubrió que las noticias de la muestra no estaban enfocadas de la forma esperada.

Considerando que en el contexto de este trabajo resultaba importante tener cierta confiabilidad en esta área, ya que son un componente escencial en la mayor parte de las estadísticas y datos presentados, se considera que el módulo de clasificación se comporto de acuerdo a lo esperado. La mejora más escencial que surge aquí es la necesidad de introducir el reconocimiento de artículos que no describen exactamente un hecho delictivo, sino que brindan estadísticas o información adyacente a los mismos. Sin embargo, al ser El Observador la fuente menos predominante con diferencia y ver que el resto de las fuentes brindan una confiabilidad mayor, se minimiza el impacto que la ausencia de esta funcionalidad pueda causar.

5.2.2. Reconocimiento de barrio

Para el reconocimiento de barrios se tuvo en cuenta sólamente un barrio por noticia. Esto implica que algunos crímenes que por distintas circunstancias podrían pertenecer a más de un barrio, son considerados sólamente en uno. Además, agrega algunos tipos de falsos positivos, ya que el barrio reconocido puede estar mencionado en el texto del artículo pero no ser el lugar dónde ocurrió el crimen.

Es así que quedan configuradas las siguientes situaciones en las cuales el reconocimiento de barrio es considerada errónea:

- Barrio no reconocido pero la información sí está presente en la noticia.
- Barrio reconocido pero dicho barrio no está asociado de ninguna manera a la noticia.
- Barrio reconocido correctamente en la noticia, pero no es representativo

del lugar del crimen.

A nivel global, 3333 crímenes tienen un barrio asociado (71.5 %); se realiza a continuación un evaluación de estos números.

Teledoce

Fueron reconocidos correctamente los barrios de 33 de los 50 crímenes de la muestra, de dónde los restantes se explican de las siguientes formas:

- 13 de estos fueron falsos positivos.
- Los restantes 4 fueron falsos negativos, dónde había información suficiente para asignar una categoría pero no se hizo.

Montevideo COMM

Sólamente 12 de los 25 barrios se reconocieron correctamete. La gran mayoría de los errores se debieron a noticias que no eran en Montevideo, pero no se las descartó debidamente.

El Observador

Cómo ya se mencionó para esta fuente, debido a que las noticias no están directamente relacionadas a hechos delictivos, los reconocimientos de barrios logrados (en este caso del 100 %) no son representativos de un delito, sino de otra información no relevante. Es por esta situación que sólamente 1 de las 5 muestras cuenta con un reconocimiento del barrio correcto.

El País

Un 60 % (9 de 15) barrios lograron ser identificados con precisión, con la particularidad que el problema identificado para Montevideo COMM se repite también aquí.

Conclusiones

A nivel general se puede decir que el módulo de clasificación arrojó resultados correctos para esta clasificación en 55 de las 95 muestras totales, que no es un número muy alentador.

Para el enfoque de este trabajo tienen mayor relevancia las muestras que arrojaron un resultado no vacío en el reconocimiento del barrio, ya que son las que se desplieguen a los usuarios finales. Tomando entonces el subconjunto de muestras que sí cuentan con un barrio asociado, el clasificador se comporta correctamente un 50.7 % de las veces, con 35 verdaderos positivos y 34 falsos positivos.

Algo que se notó al realizar el chequeo manual es que los errores tienden a concentrarse en 3 barrios: Centro, Prado/Nueva Savona y Peñarol/Lavalleja. Esta información es importante, ya que implica que los datos allí desplegados al considerar únicamente fuentes de los portales de noticias es poco representativa de la realidad. Por otro lado, también da indicios de que en el resto de los barrios la información sí es más representativa, ya que están bastante más diluidos.

Del análisis anterior se desprende que esta sea muy probablemente el punto más debil del clasificador, con dos potenciales puntos de mejora principales:

- Mejorar el reconocimiento de noticias que no refieren a situaciones dentro de la ciudad de Montevideo, de forma que no sean consideradas en ningún punto del problema.
- Dotar al módulo de la habilidad de discernir, cuando en una noticia se menciona más de un barrio, cuál es la que hace referencia al lugar del crimen en concreto.

5.2.3. Reconocimiento de ubicación

El reconocimiento de la ubicación tiene la particularidad de que si bien es más compleja de obtener - y de obtener correctamente - que los criterios anteriores, presenta un menor riesgo ya que los resultados que arroja se utilizaron pura y exclusivamente en la parte de presentación visual en el mapa. Todos los números presentados en cuanto a estadísticas tanto en gráficas cómo en el mapa de calor no tomaron en cuenta los datos a este nivel de granularidad.

Del total de crímenes, a 617 de ellos se les logró asignar coordenadas, que representa un 13.2 % del total. Además resulta interesante que 228 de ellos (36.9 %) tienen información tanto del barrio como coordenadas. Este dato es importante ya que, como se explicó en la sección 4.2.2, se realiza un chequeo donde las coordenadas deben sentido en relación al barrio del crimen, y por ende cuentan con un grado de certeza mayor al resto.

Para el estudio de esta sección, los siguientes casos se consideran reconocimientos erróneos:

- Ubicación reconocida pero la misma no coincide con los datos expuestos en el artículo.
- Ubicación no reconocida y había datos suficientes para hacerlo.

Tomando en cuenta lo anteriormente expuesto, se analiza para cada una de las fuentes cuál fue el comportamiento obtenido.

Teledoce

De las 50 muestras, 33 de ellas fueron identificadas cómo habiendo reconocido correctamente la ubicación. Dentro de los crímenes para los cuáles el clasificador obtuvo coordenadas, que fueron 7 en la muestra, se reconocen dos situaciones distintas:

- 4 de los 7 tienen también un barrio asociado, y todos ellos presentaban una ubicación correcta.
- De los restantes 3, sólamente 1 presenta coordenadas incorrectas.

Esto deja además 12 falsos negativos, dónde la ubicación pudo haber sido reconocida pero no lo fue.

Montevideo COMM

Se reconoció ubicación para 3 de las 25 muestras, de las cuales 1 cuenta con un barrio asociado y se reconoció correctamente, mientras que las restantes 2 fueron errores. Por otro lado, en 12 de las muestras había información suficiente para ubicar el delito pero no se logró.

El Observador

4 de las 5 muestras fueron valoradas como reconocimientos positivos, ya que no poseen una ubicación que era el comportamiento esperado. El restante sí tiene un valor de ubicación pero que no es representativo de un crimen real.

El País

Únicamente 2 de las 15 muestras presentan una ubicación, logrando un 50 % de precisión. El acierto se corresponde con una muestra para la cual se logró reconocer también el barrio, mientras que el error se trató de un crimen que no fue en la ciudad de Montevideo.

Conclusiones

Se concluye que el reconocimiento de ubicación cuando también se logró reconocer un barrio es muy bueno, acertando en 6 de las 7 muestras con éstas características (85 %), lo que tiene sentido considerando que se utiliza también la información del barrio para obtener la ubicación en estos casos. De alguna manera se podría decir que la mayor mejora a realizar para este tipo de reconocimiento es mejorar el reconocimiento de barrios.

Además se observa que hay una cantidad importante de noticias en las cuales se podían haber reconocido coordenadas, pero el módulo de clasificación no logró hacerlo. Esto puede deberse tanto a problemas del módulo mismo, como a problemas del servicio externo utilizado para geocoding.

5.2.4. Falsos negativos

En esta sección se analiza, por medio de la misma técnica de muestreo utilizada en secciones anteriores, la presencia de artículos de noticias que pudieron haber sido clasificados y tenidos en cuenta para los dístintos análisis disponibles en la plataforma construida, pero sin embargo fueron descartados por el clasificador. Es así que se realiza una selección de 40 artículos, y se define de forma manual si los mismos se corresponden con un crimen y además cuentan con suficiente detalle para haber sido clasificados.

De las 40 muestras tomadas, 30 de ellas fueron evaluadas como correctamente descartadas del corpus utilizado en los estudios de este trabajo (75 %). Dentro de estas 30, las razones por las cuales una noticia no evalúa como crimen varía entre crímenes que ocurrieron fuera del área de Montevideo, y noticias que no refieren directamente un crimen, sino a datos sobre los mismos o sus involucrados.

Por otro lado, las restantes 10 noticias sí tienen información suficiente para haber sido categorizadas con algún tipo de crimen del sistema, pero no lo fueron. Además, en todos los casos tenían información acerca del barrio, e incluso algunos de ellos de las calles del hecho.

Analizando en mayor profundidad, se observa que gran parte de los falsos negativos refieren a usos linguísticos que no fueron considerados al construir el clasificador. Esto ocurre por la forma en que se deciden las categorías de los crímenes, que como se describió en la sección 4.2.2, no utiliza un modelo entrenado para realizar la clasificación lo que deja potencialmente casos afuera que pudieron haber sido considerados.

Sin embargo, se realizó un buen trabajo descartando algunos crímenes por fuera de la ciudad de Montevideo que no eran relevantes para nuestro estudio. Si bien en ciertos casos, como se observó en 5.2.2, el módulo de clasificación no logra detectar esta situación, en otros sí y por ende es algo que se puede ajustar y mejorar. En la sección 7 se explica de qué manera se puede trabajar en estos ajustes como parte de trabajo futuro, que no entró dentro de los márgenes propuestos de este proyecto.

5.3. Análisis de los datos

Tanto el apartado de estadísticas del sistema como el mapa permiten al usuario indagar la información que más le interesa por medio del uso de los filtros disponibles. Sin embargo, interesa destacar información que expone el sistema a alto nivel. Aparte de poder validar que ciertos datos macro tienen sentido en el contexto que se maneja, se presentan algunas breves conclusiones sobre el caso de estudio del trabajo, es decir, la criminalidad en Montevideo en los últimos años.

Observando la gráfica inferior de la Figura 16, se puede ver que el barrio con mayor cantidad de crímenes en las noticias es Pocitos. Este valor se puede entender analizando el mapa y aplicando algunos filtros. En la Figura 25 se muestra que los barrios costeros de Montevideo tienen altos niveles de Hurtos y Rapiñas en comparación con el resto, según los portales de noticias. Al ser estos dos tipos de crimen los más abundantes, se entiende cómo Pocitos pudo terminar en el primer puesto en el gráfico anteriormente mencionado.

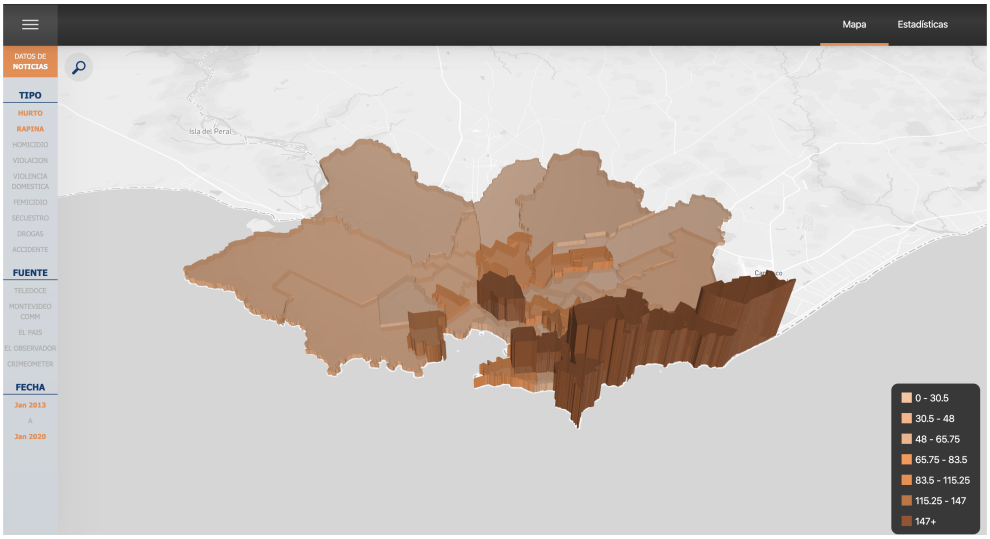

Figura 25: *Mapa de Hurtos y Rapiñas según las noticias.*

Al considerar también los Homicidios, se nota que la tendencia es opuesta a la vista para los anteriores tipos de delito, como se observa en 26. En este caso, barrios como Cerro, Casavalle y Tres Ombúes son los que sobresalen del resto - en el análisis se excluyeron los datos del barrio Centro por lo expuesto en la sección 5.2.2. Sin embargo se observa que Pocitos también cuenta con un número elevado de Homicidios.

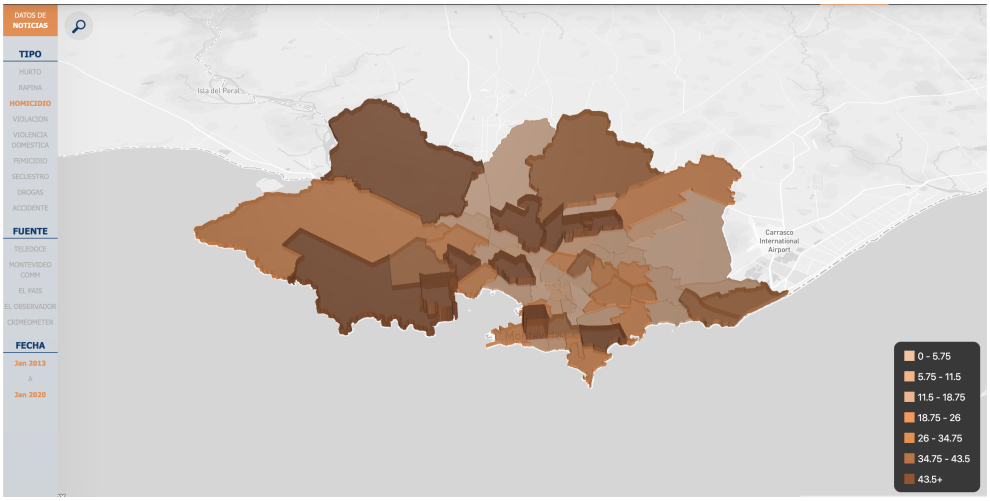

Figura 26: *Mapa de Homicidios según las noticias.*

Resulta interesante observar cómo estos datos extraídos de portales de noticias puedan corresponderse a valores oficiales. Es así que aplicando la misma búsqueda para datos del Ministerio del Interior se obtiene lo expuesto en las figuras 27 y 28.

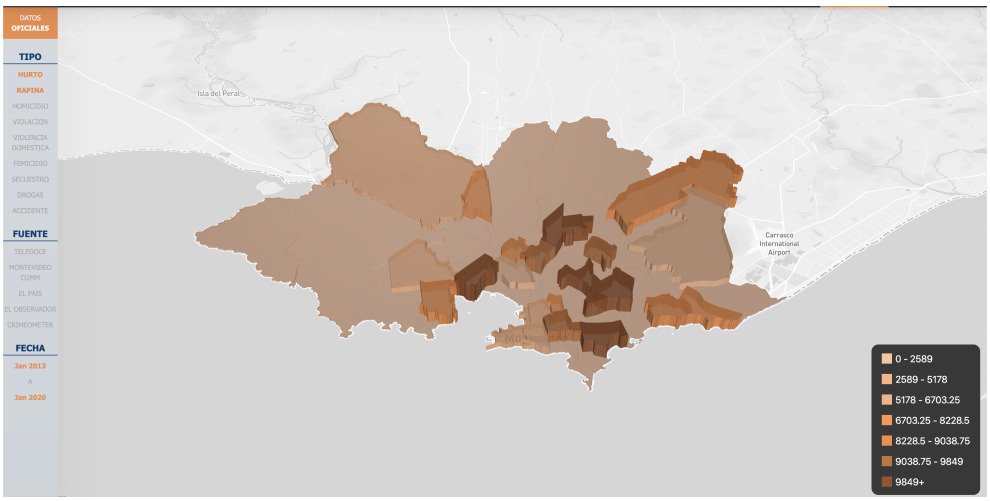

Figura 27: *Mapa de Hurtos y Rapiñas según datos oficiales.*

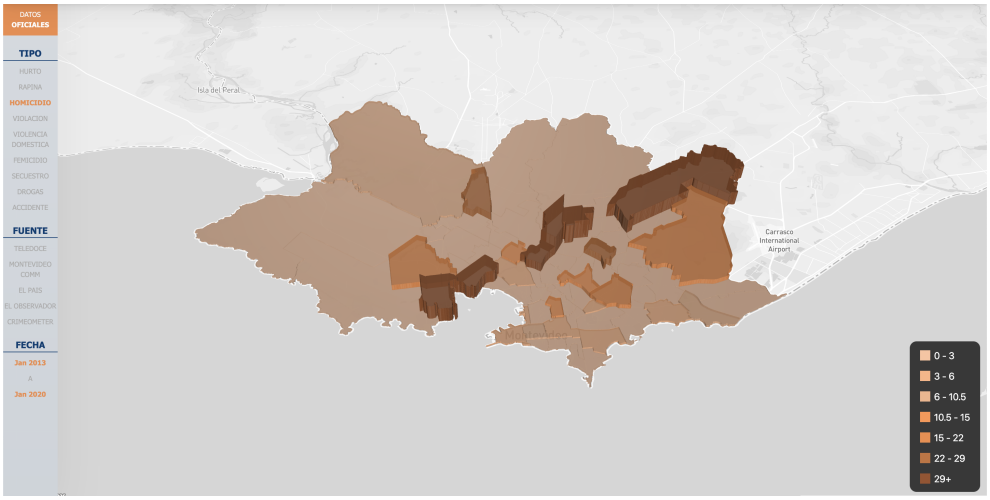

Figura 28: *Mapa de Homicidios según datos oficiales.*

De lo anteriormente expuesto se llega a la conclusión de que si bien barrios costeros aparecen mucho más preponderantes al considerar hurtos y rapiñas que al considerar homicidios, otros barrios que son los que más homicidios tienen también tienen altos índices de estos otros dos tipos de delito. Se puede decir entonces que los hurtos y rapiñas en barrios dónde los homicidios tienen un gran peso, son considerablemente dejados de lado por los medios. Esto puede tener sentido, explicándolo desde el punto de vista de que se hace foco a los crímenes de mayor gravedad para cada uno de los barrios en los portales de noticias. Además, se puede decir que los homicidios en barrios costeros cómo Pocitos tienen potencialmente más repercusión en las noticias que aquellos ocurridos en otros barrios, considerando el importante número de crímenes de este tipo que tiene según las noticias que no se corresponde en lo absoluto con los datos oficiales.

A continuación se exponen algunas otras conclusiones genéricas que se obtienen de 16 y 17:

- Sólamente en años recientes se empieza a tener conciencia acerca del problema existente en la sociedad con los femicidios, ya que los primeros datos relativos a este tipo de crimen se comienzan a ver a mediados del año 2017; con un fuerte incremento en las noticias en el año 2019, lo que ayuda a visibilizarlo.
- La cantidad de hurtos y rapiñas ha crecido rápidamente en los últimos dos años, y su presencia en los medios de noticias creció de igual manera.
- La cantidad de homicidios tuvo un fuerte aumento en el año 2018 - de

hecho, fue un año récord en este aspecto -, pero la tendencia para el año 2019 es a la baja con respecto a ese año. En las noticias su crecimiento ha sido muy importante, aspecto que se corresponde con lo que se entiende es una preocupación general de la sociedad actual en nuestro país. Más específicamente en 2017 hubo 161 homicidios en Montevideo (80 en el primer semestre), contra 223 en 2018 (114 en el primer semestre) y 97 en el primer semestre de 2019. De continuar la tendencia, 2019 va para un año de unos 200 homicidios, lo que representa un número muy elevado. Lo anteriormente expuesto es quizás la razón por la cual los homicidios son el tipo de crimen más tratado en los medios de noticias de nuestro país en el año 2019, a diferencia de años anteriores dónde los hurtos y las rapiñas llenaban los titulares.

6. Gestión del proyecto

En un proyecto de esta índole, la gestión del mismo es sumamente importante. No solo garantiza que los objetivos planteados se cumplan sino que también deja en claro, a lo largo de la ejecución del proyecto, cual es el avance que se ha obtenido en cierto momento dado. Otra de las consecuencias es tener un proceso ordenado en el cual se tiene seguridad de que cada persona esta realizando la tarea adecuada en el momento adecuado.

Teniendo en cuenta las ventajas que brinda la gestión para la correcta realización de un proyecto, se hizo fuerte incapíe en la misma desde sus comienzos.

Una de las herramientas utilizada fue Trello [93] que permite administrar muy facilmente las distintas tareas a realizar. Es una aplicación web y móvil creada en 2011 con el objetivo de simplificar algunos problemas en la planificación y gestión de proyectos. Seis años mas tarde, en 2017, la empresa Atlassian [94] la adquiere y con ello Trello obtiene mayor reputación y reconocimiento. Hoy en día es utilizada por equipos en empresas como National Geographic y Google.

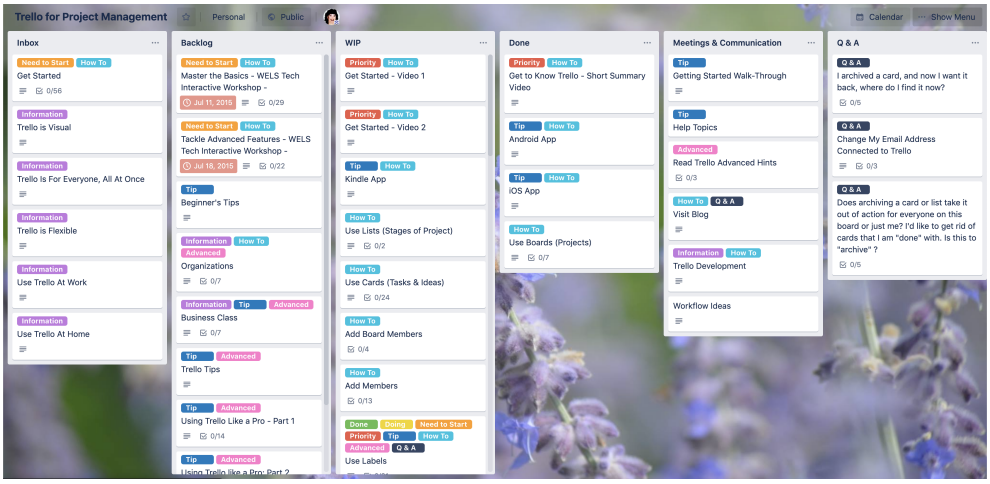

Figura 29: *Ejemplo de board en Trello*

La Figura 29 muestra un ejemplo de uso de esta herramienta para un proyecto específico. Se puede observar claramente la estrucutra de esta herramienta, con los diferentes componentes que presenta como son tableros, listas y tarjetas, entre otros. Cada tarjeta es una tarea a realizar con un título, una descripción detallada y etiquetas si es necesario. Las

etiquetas simplifican la clasificacion de tareas. Una forma de agrupar tareas es usando listas de tareas, éstas son las columnas que se observan en la Figura 29.

Este proyecto fue gestionado utilizando cinco columnas o listas:

- Improvements: posibles mejoras.
- Backlog: tareas a realizar.
- Doing: tareas que se están realizando.
- Waiting for Review: tareas que esperan ser revisadas.
- Done: tareas terminadas.

Si bien el proceso de revisar las tareas podría ser innecesario, se cree que es un elemento que aumenta la calidad del proyecto y producto final. Cada integrante del equipo siempre revisada la tarea realizada por el otro integrante. Esto evita posibles errores en el código fuente así como también genera un espacio de discusión, muy productivo para el equipo, en donde poder discutir distantas soluciones para mejorar la calidad del producto. Estas revisiones se hacían en Github [95], la plataforma en donde está el repositorio con el código fuente del producto.

Con el fin de poder distinguir el tipo de tarea, se utilizaron etiquetas. Algunas de ellas son:

- API: tareas relacionadas al servidor.
- Web: tareas relacionadas a la aplicación web.
- Infrastructure: tareas relacionadas a la infraestructura en donde se publica el producto.
- Documentation: tareas relacionadas a la documentación.

En la siguiente Figura se puede observar el board de Trello de este proyecto ya sobre el final de la implementación:

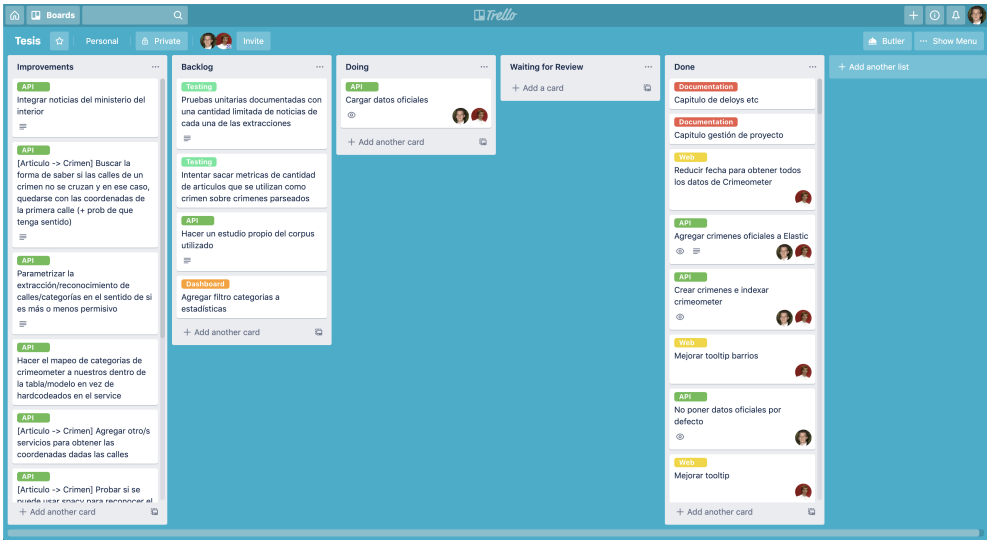

Figura 30: *Board de Trello del proyecto en la etapa final*

Parte de la gestión y planificación del proyecto ocurría también en las reuniones periódicas que se tenían con la tutora. En ellas, se profundizaba en el trabajo realizado en el último tiempo y para ello se utilizaba Trello de forma de poder tener mayor visibilidad. Luego, se planificaban las siguientes tareas. A medida que se desarrollaban las tareas, uno de los integrantes del equipo tomaba notas sobre puntos importantes discutidos o cosas a tener en cuenta. Estas notas quedaban guardadas en Google Drive [96] de forma de poder ser vistas luego por cualquiera de los integrantes y tomar acciones. Estas acciones podían ser cambiar la descripción de una tarea en Trello ya que los requerimientos cambiaron, agregar una nueva tarjeta dado que surgió una nueva tarea, o agregar una tarjeta para realizar un cambio sobre una tarea ya realizada, entre otras.

Por otro lado, para planificar las distintas etapas del proyecto se hizo uso de los Diagramas de Gantt [97]. Estos sirven para programar tareas en un largo período de tiempo. Una de sus grandes ventajas es poder visualizar de manera sencilla y realizar un seguimiento de las mismas.

A continuación se presenta el Diagrama de Gantt para las tareas planificadas:

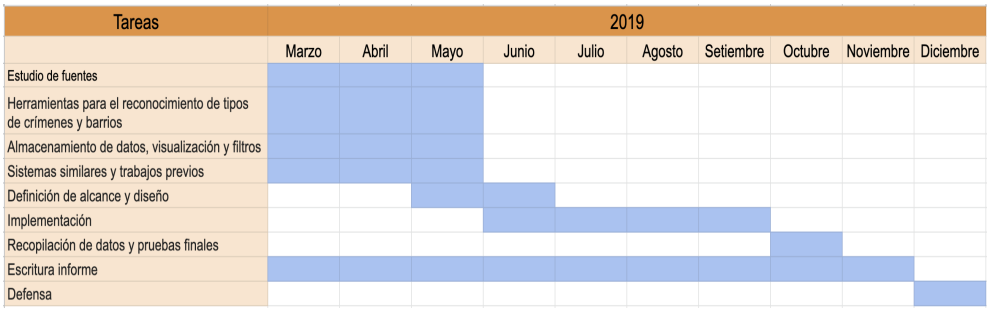

Figura 31: *Diagrama de Gantt para las tareas planificadas*

Por otra parte, el siguiente diagrama muestra cómo se ejecutaron las tareas realmente:

Figura 32: *Diagrama de Gantt de cómo se ejecutaron las tareas realmente*

Como se puede observar en las figuras 31 y 32, hubo diferencias en cuanto a tiempos lo que implica que algunas de las tareas no se realizaron de acuerdo a lo planificado.

En primer lugar, vale destacar que la tarea de analizar herramientas para el reconocimiento de tipos de crímenes y barrios no se realizó de acuerdo a lo planificado dado que fue necesario un período de tiempo más largo para poder analizar mayor cantidad de opciones y elegir la correcta antes de continuar.

En cuanto a la acción de almacenamiento de datos, visualización y filtros también hubo una demora la cual afectó la definición del alcance y diseño. Esto ocurrió porque luego de investigar distintos sistemas similares en todo el mundo, tomó más tiempo decidir cual era la mejor forma de diseñar la aplicación web.

Claramente, éstas dos demoras afectaron las etapas siguiente como la implementación, la escritura del informe y la defensa del proyecto.

En lo que respecta a la tarea de recopilación de datos y pruebas finales, hubo un atraso por dos razones. La primera está relacionada con que no es una tarea sencilla y se subestimó. La otra es que era necesario tener la implementación del componente de clasificación terminado para poder realizar las pruebas finales.

7. Conclusiones y trabajo futuro

7.1. Conclusiones

El objetivo principal de este proyecto, como se ha explicado a lo largo del mismo y particularmente en el Capítulo 2, es aumentar la visibilidad de los datos relativos a la criminalidad en nuestro país, a través del diseño e implementación de un prototipo con dichas características. Se considera que el sistema construido no sólo cumple con el objetivo inicial planteado, sino que considerando ciertas limitantes planteadas en el alcance, lo hace de buena manera.

Como parte del proyecto se construyó una base de datos de tamaño considerable con información recabada de fuentes heterogéneas. Se cuenta con información procesada y sin procesar, lo que permite junto a la capacidad de exportar dicha información desde la interfaz de administración que sea utilizada en futuras iniciativas de índoles diversas. Asimismo, considerando la arquitectura del sistema construido, la inclusión de nuevas fuentes de datos es sencilla.

Además se elaboró un sistema capaz de mostrar datos de críminalidad agregados de una forma intuitiva, que brinda la posibilidad de analizarlos de diversas formas a gusto del usuario. Particularmente considerando los datos oficiales del Ministerio del Interior, se logró una mejora en cuanto a su lectura y procesamiento. Además, se deja una plataforma en la que es relativamente sencillo realizar nuevas cargas de datos de forma de que los datos puedan mantenerse actualizados si se toma en cuenta la fuente en la cuál están basados.

En cuanto a la recopilación de datos de fuentes y la posibilidad de verlos en perspectiva con los datos oficiales, creemos que se construyó una base sólida dónde es posible obtener información de todas las fuentes y procesarla de una manera automatizada y realizable por cualquiera con los accesos pertinentes. Si bien la calidad de estos datos tiene mucho lugar a mejora y sobre todo en cuánto a la identificación de la ubicación de los crímenes, cómo se detalló en el Capítulo 5, los resultados son suficientemente confiables para realizar análisis en múltiples áreas. Además, el proyecto fue pensado desde el comienzo con foco en la modularidad y extensibilidad, haciendo sencilla la mejora o incluso el reemplazo de algunos de los componentes que forman parte del proceso.

Por otro lado, fue un proyecto rico desde el punto de vista de conocer hasta qué punto esta información está realmente al alcance de cualquiera. Se

conocieron otros grupos de personas con intereses relacionados, como el equipo de CityCop, que han intentado de igual forma a la que se intentó como parte este proyecto lograr accesos relativamente más sencillos a los datos oficiales, sin éxito. Es entonces que el prototipo elaborado aquí puede servir cómo puntapié inicial, de forma de evidenciar el valor de ofrecer de forma publica y con transparencia este tipo de información.

7.2. Trabajo futuro

A continuación se describen posibles mejoras al sistema desarrollado que fueron dejadas fuera del alcance de este proyecto. Las mismas incluyen tanto extensiones a ciertos módulos del mismo, cómo posibles nuevas funcionalidades.

Cómo punto principal a mejorar está el módulo de clasificación de los datos procedentes de medios periodísticos escritos. El mismo presenta un buen índice de reconocimiento de categorías, pero con el costo de muchos falsos negativos. Asimismo este módulo presentó problemas para identificar cuando noticias no hacían referencia a hechos dentro del territorio de interés. Se considera que los puntos débiles anteriores se pueden trabajar más, posiblemente a través del entrenamiento de modelos específicos para el caso de uso en cuestión.

Otro aspecto que, como se mencionó en la sección 4.2.2, fue dejado fuera del alcance es el reconocimiento de crímenes duplicados. El estudio e implementación de la solución al mismo brinda la posibilidad de entender qué características de un crímen hace que un mismo hecho sea referenciado múltiples veces en los portales de noticias, cualidad que hoy en día no está presente en el sistema.

Por otro lado se encuentra el reconocimiento de la ubicación para crímenes particulares. Los resultados no fueron excelentes, pero sí esperados. Cómo se explicó en el Capítulo 2, la detección de nombres de calles es un problema aún abierto del procesamiento del lenguaje natural. Existen modelos potentes entrenados en este sentido para el idioma inglés, y quizás esta sea otra posible punta a explorar como parte de trabajos futuros para el idioma español. Aparte del reconocimiento de calles, además, puede ser pertinente lograr la identificación de puntos de interés de la ciudad y generar ubicaciones a partir de ellos.

Este proyecto, además, fue acotado a la ciudad de Montevideo, pero a modo de extensión se podrían integrar nuevas fuentes de noticias y permitir las noticias en un territorio menos acotado, o incluso sin ninguna acotación.

Cómo último punto a destacar como una extensión posible, y que sería de gran valor, es la integración en tiempo real con algún sistema del Ministerio del Interior o las diversas seccionales. Esto permitiría no sólo tener acceso instantáneo a los últimos datos, sino que también haría posible la eliminación de la carga manual de datos oficiales que hay que realizar en la actualidad; sustituyéndola por algo mucho más confiable y automático.

8. Referencias

[1] *Memorias Anuales*. URL: https://minterior.gub.uy/index.php/transparencia/5727-memorias-anuales.

[2] PostGIS. *Spatial and Geographic Objects for PostgreSQL*. URL: https://postgis.net/.

[3] Katharine Jarmul y Richard Lawson. *Python web scraping: fetching data from the web*. Packt Publishing, 2017.

[4] *API*. URL: https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces.

[5] Ryan Mitchell. «Building Scrapers». En: *Web Scraping with Python Collecting Data from the Modern Web*. 1.ª ed. O'Reilly Media, 2015. ISBN: 9781491985564.

[6] Bo Zhao. «Web Scraping». En: mayo de 2017, págs. 1-3. ISBN: 978-3-319-32001-4.

[7] Muawia Elsadig, Ali Ahmed y M. Himmat. «Information Extraction methods and extraction techniques in the chemical document's contents: Survey». En: *ARPN Journal of Engineering and Applied Sciences* 10 (feb. de 2015), págs. 1068-1073.

[8] Venkat N. Gudivada. «Chapter 12 - Natural Language Core Tasks and Applications». En: *Computational Analysis and Understanding of Natural Languages: Principles, Methods and Applications*. Ed. por Venkat N. Gudivada y C.R. Rao. Vol. 38. Handbook of Statistics. Elsevier, 2018, págs. 403-428. URL: http://www.sciencedirect.com/science/article/pii/S0169716118300257.

[9] *spaCy · Industrial-strength Natural Language Processing in Python*. URL: https://spacy.io/.

[10] *Natural Language Toolkit*. URL: https://www.nltk.org/.

[11] MonkeyLearn. *Text Analysis*. URL: https://monkeylearn.com/.

[12] Sebastian Schmidt y col. «Extraction of Address Data from Unstructured Text using Free Knowledge Resources». En: sep. de 2013.

[13] Elastic. *Elasticsearch*. URL: https://www.elastic.co/products/elasticsearch.

[14] *REST API*. URL: https://restfulapi.net/.

[15] Elastic. *Query DSL*. URL: https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl.html.

[16] *Coordinate systems, map projections, and geographic (datum) transformations*. URL: http://resources.esri.com/help/9.3/arcgisengine/dotnet/89b720a5-7339-44b0-8b58-0f5bf2843393.htm.

[17] *Taller de Sistemas de Información Geográficos Empresariales - Facultad de Ingeniería - UDELAR*. URL: https://eva.fing.edu.uy/pluginfile.php/159225/mod_resource/content/2/Presentaci%C3%B3n%202019.pdf.
[18] *World Geodetic System 1984 (WGS84)*. URL: https://confluence.qps.nl/qinsy/8.0/en/world-geodetic-system-1984-wgs84-29855173.html.
[19] *Simple Feature Access - Part 2: SQL Option*. URL: https://www.opengeospatial.org/standards/sfs.
[20] Guo Renzhong. «Spatial Objects and Spatial Relationships». En: *Geo-spatial Information Science* 1.1 (1998), págs. 38-42.
[21] *The World's Most Advanced Open Source Relational Database*. URL: https://postgresql.org/.
[22] *Web Map Service*. URL: https://www.opengeospatial.org/standards/wms.
[23] *Web Map Tile Service*. URL: https://www.opengeospatial.org/standards/wmts.
[24] *Web Feature Service*. URL: https://www.opengeospatial.org/standards/wfs.
[25] *Catalog Service Web*. URL: https://www.opengeospatial.org/standards/cat.
[26] *Web Coverage Service*. URL: https://www.opengeospatial.org/standards/wcs.
[27] *Google Maps*. URL: https://www.google.com.uy/maps.
[28] *Mapbox*. URL: https://www.mapbox.com/.
[29] *Bing Maps*. URL: https://www.bing.com/maps.
[30] *Open Street Map*. URL: https://www.openstreetmap.org/#map=7/-32.565/-55.758.
[31] *Observatorio Fundapro*. URL: http://seguridad.observatoriofundapro.com/.
[32] CityCop Inc. *Outsmarting Crime. Together.* URL: https://www.citycop.org/.
[33] CrimeoMeter. *Worldwide Crime Data API*. URL: https://www.crimeometer.com/.
[34] *Teledoce Policiales*. URL: https://www.teledoce.com/categoria/telemundo/policiales.
[35] *El Observador Policiales*. URL: https://www.elobservador.com.uy/tag/policiales.
[36] *El País Policiales*. URL: https://www.elpais.com.uy/noticias/policiales.

[37] *MontevideoComm Policiales*. URL: https://www.montevideo.com.uy/categoria/Policiales-1672.
[38] Ministerio del Interior. *Observatorio Nacional Sobre Violencia y Criminalidad*. URL: https://www.minterior.gub.uy/observatorio/.
[39] *CrimeMapping.com - helping you build a safer community*. URL: https://www.crimemapping.com/.
[40] ADT. *Interactive Crime Heat Maps*. URL: https://www.adt.com/crime.
[41] Crime map. *Crime map for Community Policing, City of London Police - Police.uk*. URL: https://www.police.uk/city-of-london/cp/crime/.
[42] Ministerio del Interior. *Noticias*. URL: https://www.minterior.gub.uy/index.php/unicom/noticias.
[43] *Arquitectura Cliente/Servidor*. URL: https://www.ecured.cu/Arquitectura_Cliente_Servidor.
[44] *World Wide Web*. URL: https://webfoundation.org/.
[45] *Gartner Group*. URL: https://www.gartner.com/en.
[46] *Django*. URL: https://www.djangoproject.com/.
[47] *Python*. URL: https://www.python.org/.
[48] *Requests*. URL: https://requests.readthedocs.io/en/master/.
[49] *Beautiful Soup*. URL: https://www.crummy.com/software/BeautifulSoup/bs4/doc/.
[50] *Template Method*. URL: https://en.wikipedia.org/wiki/Template_method_pattern.
[51] *Paginación*. URL: https://www.digitalstudio.pe/blog/paginacion-en-sitios-web/.
[52] *Asyncio*. URL: https://docs.python.org/3/library/asyncio.html.
[53] *FBI-NIBRS*. URL: https://www.fbi.gov/services/cjis/ucr/nibrs.
[54] *CSV*. URL: https://en.wikipedia.org/wiki/Comma-separated_values.
[55] *Matcher Spacy*. URL: https://spacy.io/api/matcher.
[56] *Python Levenshtein*. URL: https://pypi.org/project/python-Levenshtein/.
[57] *Geocoding*. URL: https://en.wikipedia.org/wiki/Geocoding.

[58] *Geocoding Correo Uruguayo*. URL: https://www.correo.com.uy/servicios-web/-/asset_publisher/KOZQzJts06ds/content/busqueda-de-direcciones.
[59] *Celery*. URL: http://www.celeryproject.org/.
[60] *Redis*. URL: https://redis.io/.
[61] *PSQL contains*. URL: https://postgis.net/docs/ST_Contains.html.
[62] *Hypermedia*. URL: https://en.wikipedia.org/wiki/Hypermedia.
[63] *Applets*. URL: https://es.wikipedia.org/wiki/Applet.
[64] *Django REST*. URL: https://www.django-rest-framework.org/.
[65] *Bonsai Elasticsearch*. URL: https://bonsai.io/.
[66] *Django REST Serializers*. URL: https://www.django-rest-framework.org/api-guide/serializers/.
[67] *React*. URL: https://es.reactjs.org/.
[68] *JavaScript*. URL: https://www.javascript.com/.
[69] *Figma*. URL: https://www.figma.com/.
[70] *DECK.GL*. URL: https://deck.gl/#/.
[71] *Uber*. URL: https://www.uber.com/global/en/sign-in/.
[72] *Metabase*. URL: https://www.metabase.com/.
[73] *IFrame*. URL: https://es.wikipedia.org/wiki/Iframe.
[74] *DECK.GL Layers*. URL: https://deck.gl/#/documentation/deckgl-api-reference/layers/overview.
[75] *WebGL*. URL: https://developer.mozilla.org/en-US/docs/Web/API/WebGL_API.
[76] *React MapGL*. URL: https://github.com/uber/react-map-glx.
[77] *Supercluster*. URL: https://github.com/mapbox/supercluster.
[78] *Django Admin*. URL: https://docs.djangoproject.com/en/3.0/ref/contrib/admin/.
[79] *Netlify*. URL: https://www.netlify.com/.
[80] *Infrastructure as a Service - IaaS*. URL: https://searchcloudcomputing.techtarget.com/definition/Infrastructure-as-a-Service-IaaS.
[81] *Amazon Web Services*. URL: https://aws.amazon.com/.
[82] *Platform as a Service - PaaS*. URL: https://searchcloudcomputing.techtarget.com/definition/Platform-as-a-Service-PaaS.
[83] *Heroku*. URL: https://www.heroku.com/.

[84] *Oracle*. URL: https://www.oracle.com/index.html.
[85] *Google App Engine*. URL: https://cloud.google.com/appengine/.
[86] *Software as a Service - SaaS*. URL: https://es.wikipedia.org/wiki/Software_como_servicio.
[87] *Heroku Postgres*. URL: https://devcenter.heroku.com/articles/heroku-postgresql.
[88] *Autobus Heroku*. URL: https://devcenter.heroku.com/articles/autobus.
[89] *Heroku Redis*. URL: https://devcenter.heroku.com/articles/heroku-redis.
[90] *Logentries*. URL: https://logentries.com/.
[91] *Rollbar*. URL: https://rollbar.com/.
[92] *Random - Python*. URL: https://docs.python.org/3/library/random.html#random.choices.
[93] *Trello*. URL: https://trello.com/.
[94] *Atlassian*. URL: https://www.atlassian.com/.
[95] *Github*. URL: https://github.com/.
[96] *Google Drive*. URL: https://www.google.com/intl/en_ALL/drive/.
[97] *Diagrama de Gantt*. URL: https://obsbusiness.school/int/blog-project-management/diagramas-de-gantt/que-es-un-diagrama-de-gantt-y-para-que-sirve.

A. Documentación del sistema

En este anexo se documentan los servicios que expone el servidor, junto con ejecuciones de ejemplo. Como se vio en la sección 4.2.3, se exponen 5 servicios para dar soporte a la aplicación cliente. Se estudia uno a uno cómo están definidos y cómo son utilizados.

CrimesPerNeighbourhoods

```
  API endpoint that returns the amount of crimes
  per neighbourhood.

  path         -> /api/neighbourhoods/crimes_count
  query_params -> sources
                  categories
                  start_date
                  end_date
```

A continuación se adjunta un ejemplo que visibiliza de qué forma se debe declarar cada uno de los parámetros, y su respuesta en formato json a modo de ejemplo.

```
  api/neighbourhoods/crimes_count?
sources=TD,MC,EP&
categories=1,2&
start_date=2013-01-09&
end_date=2020-01-07

  [
    {
        "pk": 1,
        "name": "Ciudad Vieja",
        "total_crimes": 38,
        "categories": [
            {
                "pk": 1,
                "name": "Hurto",
                "crimes_count": 30
            },
            {
                "pk": 2,
                "name": "Rapiña",
                "crimes_count": 17
```

```
            },
            {
                "pk": 3,
                "name": "Homicidio",
                "crimes_count": 0
            },
            {
                "pk": 4,
                "name": "Violación",
                "crimes_count": 0
            },
            {
                "pk": 5,
                "name": "Violencia Doméstica",
                "crimes_count": 0
            },
            {
                "pk": 6,
                "name": "Femicidio",
                "crimes_count": 0
            },
            {
                "pk": 7,
                "name": "Secuestro",
                "crimes_count": 0
            },
            {
                "pk": 8,
                "name": "Drogas",
                "crimes_count": 0
            }
        ]
    },
    {
      "pk": 2,
      "name": "Centro",
      "total_crimes": 223,
      ....
    }
]
```

El mismo procedimiento es seguido para el resto de los servicios.

Crime

```
  API endpoint that allows crimes to be viewed.

  path         -> /api/crimes
  query_params -> sources
                  categories
                  start_date
                  end_date
```

Ejemplo:

```
  api/crimes?
sources=TD,MC,EP&
categories=1,2&
start_date=2013-01-09&
end_date=2020-01-07

  [
    {
        "pk": "21369",
        "occurred_date": "2015-10-30",
        "source": "TD",
        "streets": [
            "cuareim",
            "galicia",
            "yi"
        ],
        "location": {
            "lat": -34.90020396664815,
            "lon": -56.19082077813756
        },
        "categories": [
            "Rapiña"
        ],
        "url": "https://www.teledoce.com/...."
    },
    {
        "pk": "20690",
        ...
    },
```

```
  ...
]
```

SearchCrimes

```
API endpoint that allows crimes to be searched.

path         -> /api/crimes/search
query_params -> q - String query to be searched
```

Ejemplo:

```
  api/crimes/search?
q=bomba

[
  {
      "pk": "21215",
      "occurred_date": "2015-10-05",
      "source": "TD",
      "streets": [
          "los helechos",
          "plutarco",
          "buceo"
      ],
      "location": {
          "lat": -34.89190721373397,
          "lon": -56.1310692208722
      },
      "categories": [
          "Homicidio"
      ],
      "url": "https://www.teledoce.com/..."
  },
  {
      "pk": "13901",
      ...
  },
  ...
]
```

ListPoliceStationsCrimes

```
API endpoint that allows police stations' crimes
```

```
  to be viewed.

  path         -> /api/police_stations/crimes_count
  query_params -> categories
                  start_date
                  end_date

  Note: categories can be 1 (Hurto), 2 (Rapiña)
  and/or 3 (Homicidio).
```

Ejemplo:

```
  api/police_stations/crimes_count?
categories=2,3&
start_date=2010-08-26&
end_date=2019-08-31
```

```
  [
    {
        "pk": 1,
        "location": {
            "lat": -34.907411,
            "lon": -56.21088
        },
        "source": "OF",
        "total_crimes": 325,
        "categories": [
            {
                "pk": 1,
                "name": "Hurto",
                "crimes_count": 0
            },
            {
                "pk": 2,
                "name": "Rapiña",
                "crimes_count": 320
            },
            {
                "pk": 3,
                "name": "Homicidio",
                "crimes_count": 5
            }
```

```
        ]
    },
    {
        "pk": 2,
        ...
    },
    ...
]
```

CrimesMedian

```
API endpoint that returns the median values
of the amount of crimes per neighbourhood.

path         -> /api/neighbourhoods/crimes_median
query_params -> sources
                categories
                start_date
                end_date

It returns:
    lowest  -> the lowest value.
    lower   -> the median between the lowest and the
               middle values.
    median  -> the median between the lower and the
               upper values.
    upper   -> the median between the middle and the
               highest values.
    highest  -> the highest value.

Reference: https://docs.python.org/3/library/
statistics.html#statistics.median
```

Ejemplo:

```
api/neighbourhoods/crimes_median?
sources=TD,MC,EP&
categories=1,2&
start_date=2013-01-09&
end_date=2020-01-07

{
  "lowest": 0,
```

```
  "lower": 13,
  "median": 26,
  "upper": 42,
  "highest": 223
}
```


B. Desgloce de pruebas unitarias

El objetivo de este anexo es detallar cada una de las pruebas ejecutadas sobre el corpus para sacar métricas en cuanto a su comportamiento y resultados en el Capitulo 5.

Teledoce

A continuación se enumeran los artículos considerados en el muestreo aleatorio de crímenes para la fuente Teledoce.

1. https://www.teledoce.com/telemundo/policiales/detuvieron-a-4-hombres-e-incautaron-3-pistolas-en-el-40-semanas/
2. https://www.teledoce.com/telemundo/policiales/procesaron-sin-prision-a-un-hincha-de-nacional-que-tenia-banderas-de-penarol-por-un-delito-de-receptacion/
3. https://www.teledoce.com/telemundo/policiales/grupo-de-menores-asalto-a-un-hombre-en-jose-belloni/
4. https://www.teledoce.com/telemundo/policiales/investigan-el-asalto-y-robo-de-un-arma-a-una-mujer-policia-que-reviste-en-la-seccional-17/
5. https://www.teledoce.com/telemundo/policiales/policia-busca-a-un-joven-de-22-anos-que-se-fugo-de-la-carcel-de-campanero/
6. https://www.teledoce.com/telemundo/policiales/aun-no-hay-detenidos-por-el-homicidio-del-quiosquero-de-manga/
7. https://www.teledoce.com/telemundo/policiales/diez-personas-mojadas-encapuchadas-con-guantes-y-botas-llegaron-a-carmelo-desde-buenos-aires-por-el-rio-en-busca-de-un-empresario/
8. https://www.teledoce.com/telemundo/policiales/multitudinaria-marcha-de-vecinos-de-young-por-el-caso-del-comerciante-asesinado/
9. https://www.teledoce.com/telemundo/nacionales/las-similitudes-entre-los-homicidios-de-cristina-jones-y-janice-schroll/

10. https://www.teledoce.com/telemundo/policiales/policia-cientifica-hallo-el-adn-de-dos-hombres-bajo-las-unas-de-daniela-perez/

11. https://www.teledoce.com/telemundo/policiales/un-delincuente-que-estaba-robando-una-garrafa-baleo-a-un-policia-que-intento-detenerlo/

12. https://www.teledoce.com/telemundo/policiales/fuerte-cruce-entre-los-padres-del-nino-asesinado-en-pinamar-norte-y-los-del-asesino/

13. https://www.teledoce.com/telemundo/policiales/la-policia-continua-la-busqueda-del-doble-homicida-de-quebracho-con-helicopteros-y-controles/

14. https://www.teledoce.com/telemundo/policiales/el-policia-asesinado-en-quebracho-estaba-terminando-el-liceo-y-tenia-planes-de-casarse-en-abril-de-este-ano/

15. https://www.teledoce.com/telemundo/policiales/una-mujer-de-36-anos-funcionaria-del-ministerio-del-interior-fue-procesada-por-intentar-matar-a-la-pareja-de-su-madre/

16. https://www.teledoce.com/telemundo/policiales/caratula-del-caso-del-taxista-cambiara-a-partir-de-su-fallecimiento/

17. https://www.teledoce.com/telemundo/policiales/hombre-que-asesino-a-su-esposa-a-punaladas-fue-formalizado-por-180-dias-tras-recuperarse-de-un-intento-de-suicidio/

18. https://www.teledoce.com/telemundo/policiales/hombre-que-asesino-a-su-esposa-a-punaladas-fue-formalizado-por-180-dias-tras-recuperarse-de-un-intento-de-suicidio/

19. https://www.teledoce.com/telemundo/policiales/intento-de-robo-a-un-banco-desato-espectacular-persecucion-que-no-pudo-capturar-a-los-delincuentes/

20. https://www.teledoce.com/telemundo/policiales/delincuentes-coparon-una-vivienda-en-progreso-quedate-quieta-porque-te-metemos-un-cano/

21. https://www.teledoce.com/telemundo/policiales/hablo-el-taxista-que-ayudo-a-detener-al-menor-asaltante-del-buceo-lo-unico-que-dijo-fue-que-lo-dejara-ir-que-no-tenia-nada-que-ver/

22. https://www.teledoce.com/telemundo/policiales/joven-declara-ante-la-justicia-tras-admitir-ante-la-policia-que-asesino-a-su-pareja/

23. https://www.teledoce.com/telemundo/policiales/robaron-un-millon-de-pesos-en-un-local-de-cobranza-en-parque-guarani/

24. https://www.teledoce.com/telemundo/policiales/fue-procesado-por-homicidio-especialmente-agravado-el-hombre-que-asesino-a-su-pareja-de-un-golpe-en-la-cabeza/

25. https://www.teledoce.com/telemundo/policiales/policia-busca-al-principal-sospechoso-del-doble-homicidio-ocurrido-en-solymar/

26. https://www.teledoce.com/telemundo/policiales/un-paraguayo-de-49-anos-fue-enviado-a-prision-por-ingresar-219-kilos-de-marihuana-al-pais/

27. https://www.teledoce.com/telemundo/policiales/ministerio-del-interior-difundio-imagen-de-dominicano-profugo/

28. https://www.teledoce.com/telemundo/policiales/guardia-de-seguridad-quedo-herido-de-bala-en-un-asalto-a-un-supermercado/

29. https://www.teledoce.com/telemundo/policiales/un-hombre-de-61-anos-con-antecedentes-penales-fue-asesinado-este-miercoles-en-una-vivienda-del-cordon/

30. https://www.teledoce.com/telemundo/policiales/tres-personas-resultaron-procesadas-con-prision-tras-varios-allanamientos-en-carrasco-norte-vinculados-al-narcotrafico/

31. https://www.teledoce.com/telemundo/policiales/el-hombre-que-aparecio-calcinado-en-un-auto-en-el-parque-roosevelt-fue-identificado-como-un-expolicia-de-57-anos/

32. https://www.teledoce.com/telemundo/policiales/policia-incauto-300-kilos-de-cafeina-y-diez-de-cocaina-en-dos-allanamientos/

33. https://www.teledoce.com/telemundo/policiales/fue-procesado-el-menor-que-dias-atras-habia-sido-indagado-por-el-homicidio-de-axel-britos/

34. https://www.teledoce.com/telemundo/policiales/homicida-del-omnibus-no-aparecen-pistas/

35. https://www.teledoce.com/telemundo/policiales/el-delincuente-muerto-en-la-comercial-tenia-antecedentes-por-hurto-el-delincuente-herido-tiene-antecedentes-por-rapina-y-homicidio/

36. https://www.teledoce.com/telemundo/policiales/un-comerciante-fue-asesinado-en-toledo/

37. https://www.teledoce.com/telemundo/policiales/la-policia-investiga-los-asesinatos-de-cuatro-personas/

38. https://www.teledoce.com/telemundo/policiales/a-seis-anos-de-la-bomba-que-mato-a-miriam-mazzeo-familia-y-companeros-reclamaron-respuestas/

39. https://www.teledoce.com/telemundo/policiales/el-sector-frentista-casa-grande-plantea-cambios-en-la-politica-de-seguridad-con-una-policia-mas-insertada-en-la-sociedad/

40. https://www.teledoce.com/telemundo/policiales/fue-detenido-un-joven-acusado-de-tres-homicidios-recientes-en-el-40-semanas/

41. https://www.teledoce.com/telemundo/policiales/un-hombre-se-entrego-a-la-justicia-tras-asesinar-a-un-joven-de-24-anos-que-presuntamente-le-robo-el-casco-de-su-moto/

42. https://www.teledoce.com/telemundo/policiales/este-lunes-dos-conductores-chocaron-contra-distintas-sucursales-de-bancos-en-el-centro-de-montevideo/

43. https://www.teledoce.com/telemundo/policiales/bordaberry-pide-que-se-investiguen-homicidios-aunque-se-traten-de-ajustes-de-cuentas/

44. https://www.teledoce.com/telemundo/policiales/segun-bonomi-asesino-de-joven-hincha-de-nacional-esta-detenido/

45. https://www.teledoce.com/telemundo/policiales/se-entregaron-tres-personas-que-cometieron-hurtos-durante-los-festejos-de-penarol/

46. https://www.teledoce.com/telemundo/policiales/un-hombre-se-encuentra-delicado-tras-haber-sido-baleado-durante-un-tiroteo-en-cerro-norte/

47. https://www.teledoce.com/telemundo/policiales/fue-enviado-a-prision-preventiva-por-180-dias-el-hombre-que-intento-asesinar-a-su-exesposa/

48. https://www.teledoce.com/telemundo/policiales/tres-delincuentes-armados-con-una-pistola-de-aire-comprimido-intentaron-asaltar-una-panaderia-en-nuevo-paris/

49. https://www.teledoce.com/telemundo/policiales/son-estas-personas-que-encuentran-una-senora-que-esta-sola-le-brindan-carino-compania-ayuda-ganan-su-confianza-y-despues-se-endulzan-con-las-cosas-y-se-quieren-quedar-con-todo/

50. https://www.teledoce.com/telemundo/policiales/cuatro-menores-declaran-ante-la-justicia-por-los-robos-y-la-trifulca-fuera-de-un-local-bailable-en-plaza-mateo/

El primer paso para ejecutar las pruebas fue recuperar los resultados arrojados por el módulo de clasificación. Dichos resultados fueron los siguientes:

	Categoría	Barrio	Ubicación
1	Hurto Rapiña	-	LAT: -34.84338033863112 LON: -56.18701498184579
2	Hurto	-	LAT: -34.89819588643322 LON: -56.17325741359667
3	Hurto	Peñarol Lavalleja	-
4	Hurto	Unión	-
5	Hurto	Peñarol Lavalleja	-
6	Homicidio	Manga Toledo Chico	-
7	Secuestro	-	-
8	Rapiña Homicidio	-	-
9	Homicidio	Lezica Melilla	-
10	Homicidio	-	-
11	Hurto Violencia Doméstica	Pta. Rieles Bella Italia	-
12	Homicidio	-	-
13	Homicidio	Lezica Melilla	-
14	Hurto Homicidio	Villa Muñoz Retiro	-
15	Hurto Homicidio	Prado Nueva Savona	-
16	Hurto Homicidio	Prado Nueva Savona	-

	Categoría	Barrio	Ubicación
17	Homicidio	Manga Toledo Chico	-
18	Hurto Rapiña	Malvin	LAT: -34.895854997048076 LON: -56.09556147344587
19	Hurto	Piedras Blancas	-
20	Hurto	-	-
21	Hurto	Buceo	-
22	Homicidio	Prado Nueva Savona	-
23	Hurto	Unión	-
24	Homicidio Violencia Doméstica	-	-
25	Homicidio	-	-
26	Hurto	Prado Nueva Savona	-
27	Secuestro	-	-
28	Hurto	Cordon	-
29	Homicidio	Cordon	-
30	Drogas	Carrasco Norte	-
31	Homicidio	Parque Rodo	-
32	Hurto	-	-
33	Homicidio	-	-
34	Homicidio	Centro	-
35	Hurto Rapiña Homicidio	La Comercial	LAT: -34.886440377491546 LON: -56.17106483474256
36	Hurto Rapiña Homicidio	-	-
37	Homicidio	Cerro	-
38	Homicidio	Buceo	LAT: -34.89190721373397 LON: -56.1310692208722
39	Rapiña Homicidio Violencia Doméstica	-	-
40	Hurto Homicidio	Villa Muñoz Retiro	-
41	Hurto Homicidio	-	LAT: -34.8855238123847 LON: -56.24709220189088
42	Accidente	Centro	-

	Categoría	Barrio	Ubicación
43	Homicidio	-	-
44	Homicidio	Peñarol Lavalleja	-
45	Hurto	-	-
46	Rapiña	Cerro	-
47	Homicidio Femicidio	-	-
48	Hurto	Nuevo París	LAT: -34.8450141862861 LON: -56.24036415869904
49	Homicidio	Unión	-
50	Rapiña	Parque Rodo	-

Una vez obtenida la información de la tabla anterior, se procedió a realizar la evaluación manual:

	Categoría	Barrio	Ubicación
1	OK	X	OK
2	OK	OK	X
3	OK	OK	OK
4	OK	OK	OK
5	OK	X	OK
6	OK	OK	X
7	OK	X	X
8	OK	OK	OK
9	OK	X	OK
10	OK	OK	OK
11	OK	OK	X
12	OK	OK	OK
13	OK	X	OK
14	OK	X	OK
15	OK	X	OK
16	OK	X	X
17	OK	OK	X
18	OK	OK	OK
19	OK	X	X
20	OK	OK	OK
21	OK	OK	OK
22	OK	X	OK
23	OK	OK	X
24	OK	OK	OK

	Categoría	Barrio	Ubicación
25	OK	OK	OK
26	X	X	X
27	OK	OK	OK
28	OK	OK	X
29	OK	OK	X
30	OK	OK	OK
31	OK	X	OK
32	X	X	OK
33	OK	OK	OK
34	OK	OK	OK
35	OK	OK	OK
36	OK	X	OK
37	OK	OK	OK
38	OK	OK	OK
39	OK	OK	OK
40	OK	X	OK
41	OK	OK	OK
42	OK	OK	X
43	OK	OK	OK
44	OK	X	OK
45	OK	OK	OK
46	X	OK	OK
47	OK	OK	OK
48	OK	OK	OK
49	OK	X	OK
50	OK	OK	OK

Este mismo procedimiento fue realizado para todas las fuentes por igual, por lo que para las restantes simplemente se adjuntará la información relevante.

Montevideo COMM

1. https://www.montevideo.com.uy/Noticias/Dos-personas-detenidas-durante-censo-en-Los-Palomares-de-Cerro-Norte-uc716698
2. https://www.montevideo.com.uy/Noticias/Investigan-presunto-secuestro-de-un-hombre-tras-una-denuncia-de-su-pareja-uc725575
3. https://www.montevideo.com.uy/Noticias/Ladron-murio-tras-caer-en-huida-Se-investiga-responsabilidad-de-comerciante-armado-uc724475

4. https://www.montevideo.com.uy/Noticias/Detuvieron-a-un-profugo-de-la-Justicia-argentina-que-tenia-pedido-de-captura-nivel-Rojo–uc719592

5. https://www.montevideo.com.uy/Noticias/Detuvieron-al-sospechoso-del-homicidio-de-un-profesor-de-ingles-en-San-Carlos-uc719830

6. https://www.montevideo.com.uy/Noticias/Murio-el-delincuente-que-habia-sido-herido-por-un-policia-este-domingo-uc724839

7. https://www.montevideo.com.uy/Noticias/Formalizaron-al-hombre-que-golpeo-y-robo-a-un-peon-rural-en-Artigas-uc718906

8. https://www.montevideo.com.uy/Noticias/Un-hombre-murio-apunalado-en-Avenida-Lezica-se-presume-que-hubo-una-discusion-por-drogas-uc717280

9. https://www.montevideo.com.uy/Noticias/Dos-hombres-condenados-por-abuso-sexual-contra-un-joven-de-18-anos-uc720185

10. https://www.montevideo.com.uy/Noticias/Rapineros-agarraron-a-patadas-a-una-mujer-con-problemas-de-movilidad-en-Piedras-Blancas-uc733859

11. https://www.montevideo.com.uy/Noticias/Rapinaron-hotel-de-alta-rotatividad-y-golpearon-a-recepcionista-en-Brazo-Oriental-uc716144

12. https://www.montevideo.com.uy/Noticias/Policia-cree-que-posiblemente-los-432-kilos-de-cocaina-vengan-de-Bolivia–uc717512

13. https://www.montevideo.com.uy/Noticias/Tiroteo-en-Flor-de-Maronas-se-maneja-la-hipotesis-de-un-enfrentamiento-entre-bandas-uc721417

14. https://www.montevideo.com.uy/Noticias/-A-vos-que-sos-la-policia–rapinaron-y-golpearon-a-funcionaria-policial-uc721664

15. https://www.montevideo.com.uy/Noticias/Balearon-a-un-joven-en-El-Monarca-y-antes-de-morir-dijo-el-nombre-del-presunto-homicida-uc722048

16. https://www.montevideo.com.uy/Noticias/San-Carlos-golpeo-a-un-peaton-para-robarle-el-celular-perdio-la-cedula-y-lo-atraparon-uc728446

17. https://www.montevideo.com.uy/Noticias/Delincuentes-armados-fueron-detenidos-en-intento-de-rapina-a-local-de-cobranzas-uc731671

18. https://www.montevideo.com.uy/Noticias/Videovigilancia-tres-jovenes-fueron-atrapados-tras-rapinar-a-una-pareja-en-Buceo-uc727378

19. https://www.montevideo.com.uy/Noticias/Piriapolis-hombre-de-25-anos-asesinado-en-posible-ajuste-de-cuentas-uc732536

20. https://www.montevideo.com.uy/Noticias/Delincuentes-rociaron-local-de-pagos-con-combustible-y-robaron-a-clientes-uc725293

21. https://www.montevideo.com.uy/Noticias/La-foto-de-Rocco-Morabito-bajando-de-un-taxi-La-clave-para-dar-con-el-restaurante-uc724035

22. https://www.montevideo.com.uy/Noticias/Tercer-caso-de-arresto-ciudadano-en-una-semana-esta-vez-ocurrio-en-Las-Piedras-uc725996

23. https://www.montevideo.com.uy/Noticias/Joven-de-20-anos-fue-encontrado-sin-vida-en-una-cuneta-en-el-barrio-Casabo-uc722171

24. https://www.montevideo.com.uy/Noticias/Fiscal-Diaz-celebro-avance-en-caso-Lola–Hoy-hay-un-poco-menos-de-impunidad-en-este-pais–uc719132

25. https://www.montevideo.com.uy/Noticias/La-Policia-encontro-camperas-policiales-en-casa-de-trabajador-de-la-bebida-detenido-uc721031

	Categoría	Barrio	Ubicación
1	Homicidio Drogas	Casavalle	-
2	Secuestro	Carrasco	-
3	Hurto	Centro	-
4	Homicidio	-	LAT: -34.7822696353972 LON: -56.19514781161123
5	Homicidio	Cerro	-
6	Hurto Rapiña	Maroñas Parque Guaraní	-
7	Hurto Rapiña	Unión	-
8	Hurto Rapiña Homicidio Drogas	Lezica Melilla	-
9	Violencia Doméstica	Cerro	-

	Categoría	Barrio	Ubicación
10	Rapiña	Piedras Blancas	-
11	Hurto Rapiña	Brazo Oriental	-
12	Drogas	Sayago	-
13	Homicidio	Flor de maroñas	
14	Hurto Rapiña	-	-
15	Homicidio	Carrasco	-
16	Hurto Rapiña	Ituzaingo	-
17	Rapiña	Centro	-
18	Hurto Rapiña	Buceo	-
19	Homicidio	-	LAT: -34.9096375602059 LON: -56.20480162612375
20	Hurto Rapiña	Castro P. Castellanos	-
21	Hurto	Punta Carretas	-
23	Secuestro	Casabo Pajas Blancas	LAT: -34.885893750815406 LON: -56.27689970502447
24	Rapiña Homicidio	-	-
25	Hurto	-	-

	Categoría	Barrio	Ubicación
1	OK	X	OK
2	OK	X	X
3	OK	X	OK
4	OK	X	X
5	OK	X	X
6	OK	OK	OK
7	OK	X	OK
8	OK	OK	OK
9	OK	X	OK
10	OK	OK	X
11	OK	OK	X
12	OK	X	OK
13	OK	OK	X

	Categoría	Barrio	Ubicación
14	OK	OK	OK
15	OK	OK	OK
16	OK	X	OK
17	OK	X	X
18	OK	OK	X
19	OK	X	X
20	OK	X	X
21	X	OK	OK
22	OK	X	OK
23	X	OK	OK
24	X	OK	OK
25	OK	OK	OK

El Observador

1. https://www.elobservador.com.uy/nota/en-seis-anos-se-dispararon-en-un-300-los-ataques-a-policias-2017101500
2. https://www.elobservador.com.uy/nota/vigilancia-colectiva-crece-la-demanda-y-oferta-de-servicios-de-seguridad-privada-2018861100
3. https://www.elobservador.com.uy/nota/asesinatos-multiples-que-encendieron-la-cronica-roja-2016623500
4. https://www.elobservador.com.uy/nota/5-fugas-espectaculares-de-carceles-que-superan-a-las-peliculas-policiales-20187313500
5. https://www.elobservador.com.uy/nota/policia-encontro-muerto-a-un-matrimonio-desaparecido-en-flores-201761916510

	Categoría	Barrio	Ubicación
1	Violencia Doméstica Drogas	Casavalle	-
2	Hurto Rapiña	Carrasco	-
3	Hurto Rapiña Homicidio Violación	Barrio Sur	-
4	Hurto	Capurro Bella Vista	LAT: -34.87291328443434 LON: -56.2003109809616
5	Homicidio	Unión	-

	Categoría	Barrio	Ubicación
1	X	X	OK
2	X	X	OK
3	X	X	OK
4	X	X	X
5	OK	OK	OK

El País

1. https://www.elpais.com.uy/informacion/policiales/desarticularon-red-criminal-personas-kilos-sustancia-vegetal-vehiculos.html
2. https://www.elpais.com.uy/informacion/policiales/recluso-quito-vida-ex-penal-libertad.html
3. https://www.elpais.com.uy/informacion/policiales/vecinos-malvin-nuevo-denuncian-aumento-robos.html
4. https://www.elpais.com.uy/informacion/policiales/policia-encontro-celular-micaela-limite-san-jose-florida.html
5. https://www.elpais.com.uy/informacion/policiales/policia-defiende-rapina-asaltante-recibio-tres-tiros.html
6. https://www.elpais.com.uy/informacion/policiales/homicidio-plena-via-publica-minas.html
7. https://www.elpais.com.uy/informacion/policiales/disminuyeron-homicidios-hurtos-aumentaron-rapinas.html
8. https://www.elpais.com.uy/informacion/policiales/delincuentes-abatidos-policia-rapinas-diferentes.html
9. https://www.elpais.com.uy/informacion/policiales/fuerte-operativo-casavalle-termino-detencion-monica-lider-chingas.html
10. https://www.elpais.com.uy/informacion/policiales/mujer-anos-murio-recibir-disparos-flor-maronas.html
11. https://www.elpais.com.uy/informacion/policiales/hallaron-restos-oseos-casa-cristina-jones-desaparecida-junio.html
12. https://www.elpais.com.uy/informacion/policiales/rapineros-son-abatidos-policias-bus-seguro.html
13. https://www.elpais.com.uy/informacion/policiales/mes-micaela-buscan-silencio-imputado.html

14. https://www.elpais.com.uy/informacion/policiales/asaltan-lujosa-chacra-maldonado.html

15. https://www.elpais.com.uy/informacion/policiales/femicidio-florida-hombre-dijo-policia-asesino-esposa-macetazo.html

	Categoría	Barrio	Ubicación
1	Drogas	Peñarol Lavalleja	-
2	Hurto Violación	-	-
3	Hurto	Malvín	-
4	Homicidio	Capurro Bella Vista	-
5	Rapiña	Paso de las Duranas	-
6	Homicidio Violencia Doméstica	-	LAT: -34.913109246111055 LON: -56.178935310498815
7	Rapiña Homicidio	-	-
8	Rapiña Homicidio	Piedras Blancas	-
9	Homicidio	Casavalle	-
10	Homicidio	Flor de Maroñas	-
11	Homicidio	Centro	-
12	Rapiña Homicidio	Paso de la Arena	LAT: -34.831826373569335 LON: -56.25527704256597
13	Homicidio	Centro	-
14	Hurto	Malvín Norte	-
15	Homicidio Femicidio	Manga Toledo Chico	-

	Categoría	Barrio	Ubicación
1	OK	X	OK
2	X	OK	OK
3	OK	OK	X
4	OK	X	OK
5	OK	OK	X
6	OK	OK	X
7	OK	OK	OK
8	OK	OK	X
9	X	OK	OK

	Categoría	Barrio	Ubicación
10	OK	OK	X
11	OK	X	OK
12	OK	OK	OK
13	OK	X	OK
14	OK	X	OK
15	OK	X	OK